GUM v.1.0.0

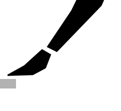
The GIMP User Manual

The Complete Guide to GIMP

Karin Kylander & Olof S Kylander

GUM

The Gimp User Manual version 1.0.0



Karin Kylander & Olof S Kylander

legalities

Legalities

The Gimp user manual may be reproduced and distributed, subject to the following conditions:

Copyright © 1997 1998 by Karin Kylander

Copyright © 1998 by Olof S Kylander

E-mail: karin@frozenriver.ale.se (summer 98 karin@frozenriver.com)

The Gimp User Manual is an open document; you may reproduce it under the terms of the Graphic Documentation Project Copying Licence (aka GDPL) as published by Frozenriver.

This document is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the Graphic Documentation Project Copying License for more details.

GRAPHIC DOCUMENTATION PROJECT COPYING LICENSE

The following copyright license applies to all works by the Graphic Documentation Project. Please read the license carefully---it is similar to the GNU General Public License, but there are several conditions in it that differ from what you may be used to.

The Graphic Documentation Project manuals may be reproduced and distributed in whole, subject to the following conditions:

All Graphic Documentation Project manuals are copyrighted by their respective authors. THEY ARE NOT IN THE PUBLIC DOMAIN.

- The copyright notice above and this permission notice must be preserved complete.
- All work done under the Graphic Documentation Project Copying License must be available in source code for anyone who wants to obtain it. The source code for a work means the preferred form of the work for making modifications to it. For a manual this means the the preferred format for the program that the author has used to create the work in.
- Any translation or derivative work of Graphic Documentation Project manuals that is published in any form must be approved by the author before distribution. All alterations of the original manuscript like editing to prepare for printing, making modifications to images, adding or removing text etc. is considered derivative work.
- Small portions may be reproduced as illustrations for reviews or quotes in other works without this permission notice if proper citation is given.
- If you have permission to make a translation or derivative work of a manual, then the preferred source code of the work must be made public under the terms of the Graphic Documentation Project Copying License.

This license is based on the Linux Documentation Project Copying License

Printing guidelines

Whoever wishes to print this manual for commercial purposes it is free to do so, as long as it is made in accordance with the GDPL licence. However, if you would like to distribute a translation or derivative work based on a GDP manual, you must obtain permission from the author before doing so.

As the authors of this manual we naturally want such a publication to be a high quality product. We therefore encourage interested parties to contact us before printing. If you do publish the Gimp User Manual, we would appreciate if you would send us a copy, so we can review it at our website www.frozenriver.com. If you don’t, we can’t encourage anyone to buy it.

If someone wants to cooperate with Frozenriver to make a printed version, then this version will be given Official status as the certified GUM publication which is supported and approved by the authors. This will be mentioned at our website and also stated on the cover. The official version will also include a color calibration kit.

You may ask yourself why this issue is so important? The reason for our concern is that GUM is not an ordinary user manual for some shell utility, mail reader etc. but for an advanced image manipulation program. The manual contains more than 400 informative images which are necessary for grasping the context. There are for example certain chapters, where crucial images would be rendered meaningless in b/w or in a low resolution printing. These facts alone make it very important that a printed edition should raise, not lower the standard of the on-line version. Another reason is that we at Frozenriver wrote this manual to provide all Gimp users with free information, but also to show that our work stands for high quality, and we do not wish to be associated with a product that might damage that impression.

You may also bear in mind that we do not consider the PS, PDP, FM or HTML versions to be ready for commercial printing. The Gimp User Manual will need further editing and proofing before it's ready for commercial printing.

We hope that you can respect our wishes.

overview

Legalities i

Preface xxxvii

Frozenriver and the authors of GUM xxxix

Contributions xli

Changelog and TODO xliii

How to read GUM xlv

Part I About Gimp 1

1 What is Gimp? 3

2 Default short cuts and dynamic keybinding 9

Gallery 17

3 Gimp around in 80 minutes 19

Part II Gimp installation 41

4 Obtaining and Installing Gimp 43

Part III Basic functions 51

5 Files and Preferences 53

6 Selection tools 75

7 Paint Tools 89

8 Edit & View 101

9 Transform Tools 107

10 Text and fonts 115

12 Brushes, Gradients and Patterns 119

Part IV Color knowlage 127

13 Color Models 129

14 Pre-press and color in Gimp 135

Part V Extend Gimp 155

15 Image Menu 157

16 Selection menu 181

17 Modes 187

18 Layers 199

20 Channels, Floating Selections and Duotones 211

Part VI Filters 221

21 An introduction to filters 223

22 Animation filters 227

23 Artistic filters 231

24 Blur filters 241

25 Color filters 245

26 Combine filters 257

27 Cryptographic filters 263

28 Distort filters 267

29 Edge detect filters 283

30 Enhance filters 287

31 Generic filters 293

32 Glass effect filters 301

33 Light effect filters 305

34 Map filters 321

35 Misc. Filters 337

36 Noise filters 341

37 Render filters 345

Part VII Animations 379

38 Advanced animation 381

Part VIII Script-Fu 393

39 Script-Fu; description and function 395

40 Writing a Script Fu 405

41 Mike Terry's school of Script-Fu 419

Part IX Adv installations 439

42 How to get fonts to Gimp 441

43 Compiling plug-ins 451

Appendix 461

A Gimp start flags and rc-files 463

B Gimp man pages 477

C Commercial Gimp support 483

D SIOD: Scheme reference appendix 485

E Links and References 517

Index 523

contents

Legalities i
  Graphic Documentation Project Copying License i
    Printing guidelines ii

Preface xxxvii

Frozenriver and the authors of GUM xxxix
  Authors xxxix
    Karin Kylander xxxix
    Olof S Kylander xxxix
  Frozenriver xl

Contributions xli
  Contributions xli
    Gimp contributions xli
    GUM contributions xlii

Changelog and TODO xliii
  Changelog xliii
    1.0.0 xliii
    0.9 xliii

- 0.8 xliv
- 0.7 xliv
- 0.6 xliv
- 0.5 xliv

TODO xliv

How to read GUM xlv

an introduction xlv
- Preface xlv
- Part one xlv
- Gallery xlvi
- Part two xlvi
- Part three xlvi
- Part four xlvi
- Part five xlvi
- Part six xlvi
- Part seven xlvii
- Part eight xlvii
- Part nine xlvii
- Appendixes xlvii

Conventions xlvii

Part I About Gimp I

Chapter 1 What is Gimp? 3

”About The Gimp 4
- Features and Capabilities 4
- Authors 4
- What we can say about Gimp 5

Gimp History 5
- 0.54 5
- 0.60 6
- 0.99 7
- 1.0 8

The future of Gimp 8

Chapter 2 Default short cuts and dynamic keybinding 9
- Dynamic Key bindings 10
  - Conventions 10
- Default short cuts in Gimp. 11

Gallery 17

Chapter 3 Gimp around in 80 minutes 19
- Creating Image Objects 20
  - A Twisted personality 21
    - Variations - gold and water spirits 21
  - Creating a simple pen and an ink stain 22
    - The technicolor ink spot 22
  - Making grooved text 23
    - Organic patterns 23
- Handling Glass, Water and Reflections 24
  - The Wet look 25
    - Adding water 25
    - Making rain 25
    - Displacing along a curve 25
  - How to empty a bottle of wine 26
    - Cloning away unwanted parts 26
    - Making a dark and bright layer 26
    - Recreating missing parts 26
  - Glass Distortion 27
    - Rectifying banding 27
  - Reflections 27
- Transforming a Photograph to a Drawing 28
  - Manage without artistic plugins 29
    - Instant cartoon pictures 29
    - Making a pencil drawing 30
    - From pencil to ink 30
    - Digital crayons 31
- Light, Motion and Texture transformation 32
  - An Electric Horseman 33
    - Using a Cow to make a Leopard of a Horse... 33
    - Making things glow 34

      - Adding motion 35
      - Adding scenery 35
  - Making a Montage 36
      - The Background 37
      - The Vignette 37
      - Adding Noise 38
      - Making an element stand out in a composition 38
      - Adding depth to text layers 39

Part II Gimp installation 41

Chapter 4 Obtaining and Installing Gimp 43

- How to Install Gimp Personal files 44
    - What are all these files good for? 45
- Obtaining Gimp 46
    - Installing a Source Distribution 46
    - Installing a Binary Distribution. 48
    - Installing Extra Packages to Extend Gimp 49

Part III Basic functions 51

Chapter 5 Files and Preferences 53

- The File Menu 54
- Creating Images 55
- Guash 55
    - Opening Files 57
        - Opening Postscript and PDF Files 58
- Saving Images 58
    - Supported File Formats 59
    - Save dialogs 62
        - GBR 62
        - GIcon 62
        - GIF 63
        - jPEG 64
        - PAT 64

- PNG 65
- PNM 65
- PostScript & EPS 65
- SunRas 66
- TGA 66
- TIFF 67
- Xpm 67

Mailing Images 67

Printing Images 68
  - Supported printers 68
  - Settings 68

Gimp Preferences 69
  - Display 69
  - Interface 70
  - Environment 70
  - Directories 71
    - Suggestions 71

Misc. features & extensions 71
  - Tip of the day 71
  - DB Browser 71
  - PDB Help 72
  - Gimptcl Consolio 72
  - Screen Shot 72
  - Waterselect 72
  - Webbrowser 73

Chapter 6 Selection tools 75

The Basic controls 76

Toggle 76
  - Selection control 76
  - 77
  - 77
  - 77

Moving selections 78
  - Once more 78
    - Mask 78

Guides 78

Rectangular and ellipse selection tools 79
  - Short cuts 79

Short Repetition: 80
Selection options 81
Feather 81
Antialiasing 81
The Free-hand selection tool 82
Options 82
Fuzzy select 82
Options 82
The Bezier selection tool 83
Use beziers as a simple drawing tool 83
Control points 84
Plan ahead 84
Modifying control points 84
Move 84
Sharp corners 85
The final touch 85
Options 85
Intelligent Scissors 85
Activation and short cuts 86
Options 86
Tips 86

Chapter 7 Paint Tools 89
The Color Picker 90
Color info 90
Options 90
Palettes 91
Adding colors 91
The Bucket Fill 92
Options 92
The Blend Tool or Gradient Fill 92
Options 92
Gradient Types 93
The Pencil and Paintbrush 96
Options 97
The Eraser tool 97
Hints 97
The Airbrush tool 98

    - Options 98
- The Clone Tool 98
    - Options 98
    - Retouching hints 98
- The Convolver 99
- Brush Selection 99
- The Foreground/Background Icon 100
- The color dialog 100

Chapter 8 Edit & View 101

- Cut, Copy, and Paste 102
- Paste Into 102
- Cut, Copy and Paste Named 102
        - How to 102
- Clear, Fill and Stroke 103
    - Clear 103
    - Fill 103
    - Stroke 104
- Undo and Redo 104
- Copy visible 104
- Zoom 104
- Guides and Rulers 105
- Toggle and snap 105
- Window info, New view and Shrink wrap 105
    - New view 105
    - Shrink wrap 106
    - Window info 106

Chapter 9 Transform Tools 107

- The Move Tool 108
    - Moving floating selections 108
    - Moving the whole image or a single layer 108
    - Moving empty selections 109
        - Hints 109
- Magnifying glass or Zoom tool 109
    - Options 109
- The Crop Tool 110

    - The dialog 110
- The Transform tool 110
    - Rotate 111
    - Scale 111
    - Shearing 112
    - Perspective 112
    - Options 113
- The Flip tool 113

Chapter 10 Text and fonts 115

- The Text tool 116
    - Border matters 117

Chapter 11 Brushes, Gradients, Palettes and Patterns 119

- Brushes 120
    - The dialog 120
    - Making a brush 120
- Patterns 121
    - Make a pattern 121
- Palettes 121
    - Create a palette from an image 122
        - Hints 122
- Gradients 122
    - The Gradient editor 122
        - How to use the gradient editor 123
        - The popup menu 124
        - Endpoints 124
        - Segments 124
        - Blend 125
        - Coloring 125
        - Flips 125
        - Replicate 125
        - Save, Save as, and POV gradient format 125

Part IV Color knowlage 127

Chapter 12 Color Models 129

- Color models 130
- RGB 130
- CMYK 130
- INDEXED 131
- HSV 132
  - Hue 132
  - Saturation 132
  - Value 132
- NCS 133
- Spot color 133
- Grayscale and Line Art 134
- Complementary or Inverted colors 134

Chapter 13 Pre-press and color in Gimp 135

- What is prepress? 136
- Printing from Gimp 136
  - File formats for printing 136
- Resolution and image size 137
- Preparing for the press 138
  - dpi, lpi, ppi and Scanning resolution 138
    - lpi& dpi 138
    - ppi & Scan resolution 139
    - Read more 139
- At the print shop 139
  - Color 139
    - Make it simple 139
    - Spot color 140
  - How to transfer images to the print shop 140
    - Removable drives 140
    - Internet and BBS 140
    - Email 141
    - Filesystem format 141
- Scanning under UNIX/Linux 141
  - Sane 141
  - Other scanner programs 141
  - Commercial scanner programs 142
- Calibration 142

- Gamma Calibration 143
- Black level and white level adjustment 143
- Color calibration 145
  - Plain 145
  - CMS 145
  - Frozenriver’s plain color calibration 145
  - Poor man’s color calibration 147
  - Why don’t the colors look like they should, even if I have calibrated the system? 147

In-depth information 148
- Color 148
- Color management 148
  - RGB and CMYK 148
  - Gamut 148
  - CMS 148
  - Profiles 149
- Resolution 149
  - Lpi, dpi and screen frequencies 149
  - Halftone dots 150
  - Why doesn’t my inkjet look like halftone screens? 151
  - FM Screening 151

Tables 153
- Lpi table 153
- Printer table 153
- Image table 153
- Shades of gray 154
- Screening matrix geometry 154

Part V Extend Gimp 155

Chapter 14 Image Menu 157

Colors 158
- Equalize 158
- Invert 159
- Posterize 159
- Threshold 160
  - Lineart 160
- Color Balance 161

  - Brightness-Contrast 162
  - Hue-Saturation 162
    - Hints 163
  - Curves 164
    - Workflow 164
  - Levels 166
    - Levels on Grayscale 166
    - Levels on RGB or Alpha 167
    - Example: Making Carved Text with Levels 167
  - Desaturate 168
  - Auto-Stretch HSV 169
  - Contrast Auto-Stretch 169
  - Normalize 170
- Channel Ops 170
  - Duplicate 170
  - Offset 170
    - Tips 170
  - Compose and Decompose 172
    - RGB decomposing 172
    - HSV decomposing 173
    - CMYK & CMY decomposing 174
    - Alpha decomposing 175
- RGB, Grayscale, Indexed 175
  - Indexed options 175
- Resize and Scale 176
  - Resize 176
  - Scale 177
- Histogram 177
- Alpha 178
  - Holes: 178
  - Threshold Alpha 178
- Save palette 178
- Transforms 178
  - Autocrop 178
  - Image 179
  - Layer 179
  - Rotate 179
  - Zealous Crop 179

Chapter 15 Selection menu 181

- Toggle 182
- Invert 182
- Select all or none 182
- Float 182
- Feather, Sharpen and Border 182
  - Feather 182
  - Sharpen 183
  - Border 183
- Grow and Shrink 183
- Save to channel 183
- Select by color 184
  - Representation 184
  - Modes 184
  - Options 185
  - 185

Chapter 16 Modes 187

- What is modes 188
  - Normal Mode 188
  - Dissolve Mode 189
    - Multiply Mode 189
    - Screen Mode 189
  - Overlay Mode 190
  - Difference Mode 190
  - Addition Mode 191
  - Subtraction Mode 191
  - Darken only 191
  - Lighten only 191
  - Hue 192
  - Saturation 192
  - Color 192
  - Value 193
  - Behind 193
- Comparing pictures in different modes 193
  - Why are there so few colors in Screen, Addition, and Lighten only? 195
  - What is the difference between Screen, Addition and Lighten Only? 195
  - What is the difference between Multiply and Darken Only? 196

- What is the difference between Color and Hue? 197

Chapter 17 Layers 199

- Introduction 200
- Adding layers to your image 200
  - The dialog 200
- Layers 200
- Layer operations 201
    - New layer 201
    - The active layer 201
    - Symbols and explanations 201
    - Naming 202
- Raise Layer and Lower Layer 202
- Duplicate Layer 202
- Delete Layer 202
- Scale Layer 202
- Resize Layer 203
- Add Layer Mask 203
  - Description 203
  - How to 204
    - With gradients 204
    - Green and Red Mask display 204
- Apply Layer Mask 204
- Anchor layer 204
- Merge Visible Layers 205
- Flatten Image 205
- Alpha to Selection 205
- Mask to Selection 206
- Add Alpha Channel 206
- Align Visible Layers 206
  - Horizontal style 206
  - Horizontal base 206
  - Vertical style 207
  - Vertical base 207
  - Collect 207
  - Fill 208
  - Snap to grid 208
  - Misc. 208

- Import layers 209
- Adjust layers 209
- Move layer 210

Chapter 18 Channels, Floating Selections and Duotones 211

- Channels 212
- RGB Channels 212
- Alpha Channels 212
  - What are Alpha Channels? 212
  - Storing selections 213
  - Editing Alpha Channels 213
- Using Channels for creating Spot Color Separation 214
  - Adding color to a Channel 214
  - Spot Colors 215
  - Duotones 215
  - Checklist 216
  - How to create a duotone 217
- Floating Selections 218
  - What is a floating selection? 218
    - Anchoring a floating selection 218
    - Moving objects in a floating selection layer 219
    - Tips on working with floating selections 219

Part VI Filters 221

Chapter 19 An introduction to filters 223

- Plug-ins 224
  - The main categories 225

Chapter 20 Animation filters 227

- Animation Playback 228
- Animation optimize 228
- Animation unoptimize 228
- Filter all Layers 229
- How to create a Gif animation 229

Specifying the delay of each frame 229
Combining frames 229
Replacing frames 229

Chapter 21 Artistic filters 231
Apply Canvas 232
Cubism 232
Tip 232
Mosaic 233
Tip 233
Oilify 234
Tip 234
Van Gogh (LIC) 234
Map image 234
Blur 235
Texture 235
Creating a nice burlap cloth texture: 236
Warp 236
Main Options 236
Secondary Options 237
Other Options 238

Chapter 22 Blur filters 241
Blur 242
Gaussian Blur (IIR) 242
Gaussian Blur (RLE): 242
Motion Blur 243
Linear 243
Radial 243
Zoom 243
Pixelize 244
Variable Blur 244

Color filters 245
Alien Map 246
Color exchange 247
Colorify 247

- Color Map Rotation 248
  - Main window 248
  - Range 248
  - Example 248
  - The Misc. window 249
- Filter Pack 250
  - Advanced options 250
- Gradient Map 251
- Hot 252
- Max RGB 252
- Quantize 253
- Scatter HSV 253
- Semi-Flatten 253
  - Semi transparency in Web images 253
- Smooth Palette 254
- Value Invert 255

Chapter 24 Combine filters 257

- Depth merge 258
  - An example 258
- Film 260
  - How to 260
- Fuse 260
  - Parameters 261

Chapter 25 Cryptographic filters 263

- Digital Signature 264
- Encrypt/Decrypt: 264
- Stegano 265

Chapter 26 Distort filters 267

- Blinds 268
- Curtain 268
- Emboss 269
- Engrave 270
- IWarp 270
  - Parameters 271

- Page curl 273
- Polar Coords 273
  - . Examples 274
- Ripple 275
- Shift 275
- Twist 276
  - Functions Effects 276
  - Parameter settings 277
- Value Propagate 278
  - Tips 278
- Waves 279
- Whirl and Pinch 280

Chapter 27 Edge detect filters 283

- Introduction 284
- Edge 284
- Laplace 284
- Sobel 285

Chapter 28 Enhance filters 287

- Deinterlace 288
- Despeckle 288
  - How to use it 289
- Destripe 289
- NL Filter 290
- Sharpen 291

Chapter 29 Generic filters 293

- Convolution Matrix 294
  - Examples 294
- MathMap 296
  - Variables 296
  - Built-in Functions 297
- Universal filter 299
- User Filter (Adobe Photoshop Filter Factory) 300

Chapter 30 Glass effect filters 301

- Apply lens 302
- Conical Anamorphose & Central-Reflection 302
  - Settings 302
- Glass Tile 303
- Refract 303
  - Parameters 303

Chapter 31 Light effect filters 305

- FlareFX 306
- Gflare 306
  - Main window 306
  - The Gflare editor 307
    - Glow settings 308
    - Rays 309
    - Second Flares 310
  - Back to the main window 311
- Light Effects 312
  - The main interface 312
    - Options 312
  - Light 313
    - Type of Light 313
    - Light color 313
    - Position for point light: 313
  - Materials 314
    - Intensity: 314
    - Reflection: 314
  - Bumpmapping 314
  - Environment mapping 315
  - A simple tutorial 315
- Sparkle 318
    - 318
    - 318
  - Parameters 318
- Super Nova 319

Chapter 32 Map filters 321

- Bump map 322

    - Usage 322
- Coordinate Map 323
- Displace 323
    - Description 324
    - Calculations 324
    - The user interface 324
    - Examples 325
        - Example 1; basic displacing 325
        - Example 2; displacing in two directions 326
        - Example 3 and 4; spreading and curving 326
    - Tips & Tricks 327
    - What's the difference between Black, Smear and Wrap? 327
    - More calculations 328
- Fractal Trace 329
- Illusion 329
- Make Seamless 330
- Map Object 330
    - Main interface 331
        - Options 331
    - Light 332
    - Materials 333
        - Intensity 333
        - Reflection 333
    - Orientation 334
- Paper Tile 335
- Small tiles 335
- Tile 336

Chapter 33 Misc. Filters 337

- Magic Eye 338
    - Example and parameters 338
- Stereogram 339
    - How to: 339
    - Parameters 340
- Video 340

Chapter 34 Noise filters 341

- Noisify 342

- Randomize 342
  - Randomization types 342
    - Blurring 342
    - Hurling 342
    - Picking 343
    - Slurring 343
- Spread 343

Chapter 35 Render filters 345

- CML explorer 346
  - General Settings 346
    - Functions 346
    - Composition and Arrangement 346
    - The Slide Bars 347
  - Advanced settings 347
    - Other options 348
    - Misc. options 348
- Checkerboard 348
  - Example 349
- Diffraction patterns 349
- Figures 350
- Flame 350
  - Main interface 350
    - Render and Camera parameters 350
    - Colormap controls 351
  - The Edit dialog 351
- Gfig 352
  - User interface 352
  - Preview area 352
    - Lines 353
    - Circle 353
    - Ellipse 353
    - Curve 353
    - Poly(gon) 353
    - Star 353
    - Spiral 354
    - Bezier 354
    - Move 354
    - MvPNT 354
    - Copy 354

        - Delete 354
        - Misc. 354
    - Settings 355
        - Objects 355
        - Command bar 355
        - Grid 355
    - The Tab folders 356
        - Paint 356
        - Brush 357
        - Select 358
        - Options 358
    - Example 359
- Grid 360
- Ifs Compose 360
        - Usage 360
        - Settings 361
    - How to use Ifs Compose 361
        - Making a leaf or a branch: 361
        - 363
        - 363
    - Main options 363
    - Render Options: 364
- Fractal explorer 365
    - Parameters 365
            - Min and Xmax 365
            - Ymin and Ymax: 365
            - ITER 366
            - CX and CY 366
            - Load, Reset and Save 366
            - Fractal type; 366
    - Colors 366
        - Color Function 366
            - Color Density 367
            - Color mode 367
    - Gradients 367
    - Fractals 367
- L-system 368
    - A Simple Example 368
    - Graphic Representation 369
    - Using the L-Systems Plug-in 371
    - Advanced Features 372

            - Using Braces 372
            - The Purpose of Last Rules 372
            - Effective Use of Special Rules 373
            - Where to Look for More Information 373
            - Examples 373
    - Maze 374
    - Plasma 374
    - Qbist 375
    - Sinus 376
        - Functions 376
            - X/Y scale 376
            - Complexity and Random Seed 377
            - Force tiling 377
            - Ideal/Distorted 377
        - The Color tab folder 377
        - The Blend tab folder 377
    - Solid Noise 378

Part VII   Animations 379

Chapter 36   Advanced animation or how to use AnimFrames 381

- Basic concept 382
- How to create an animation with AnimFrames 382
    - Making a frame 382
    - How to navigate our frames 382
- Your first animation 383
    - Moving along 383
- Moving beyond the basics 384
    - How far can AnimFrames take me? 384
    - The Move Path tool 384
        - Source select 385
            - The SourceImage/Layer 385
            - Mode 385
            - Handle 385
            - Step mode 385
        - The preview window 385
            - Move path preview and control points 385
        - Control slides 386

      - The Frame Slidebars 386
      - The Layerstack 387
  - The AnimFrame menu 387
    - Undo and preview? 387
    - Frames LayerDel 387
    - Frames Convert and Exchange 387
    - Frames Flatten and Frames to Image 387
    - Frame Duplicate 388
- Animation gallery/tutorial 388

Part VIII Script-Fu 393

Chapter 37 Script-Fu; description and function 395

- Script Fu? 396
- Installing Script-Fu:s 396
- Don’t and Do:s 396
- Different kinds of Script Fu:s 397
  - Stand alone scripts 397
    - Patterns 397
    - Web page themes 397
    - Logos 398
    - Make Buttons 398
    - Utils 399
    - Misc. 399
    - Make Brush 399
  - Image dependent scripts 400
    - Decor 400
    - Modify 401
    - Animators 401
    - Stencil Ops 401
    - Alchemy 402
      - Unsharp Mask 402
    - Shadow 403
      - Drop Shadow 403
      - Perspective Shadow 403
    - Render 404
    - Utils 404
    - Selection 404

Chapter 38 Writing a Script Fu 405

- Introduction 406
- Expressions 406
- Functions 406
  - car, cdr and friends (*) 407
  - Local variables (*) 408
- The Gimp PDB 408
- Registering the script with Script-Fu 409
- A commented script 410
  - Hanging a script in the image menu 411
- Painting areas with selections 413
- Loops 414
- Floating selections 414
  - Hello World - writing text in an image 415
  - Copying a selection 416

Chapter 39 Mike Terry's black belt school of Script-Fu 419

- The road to Script-Fu Mastery 420
  - Course outline 420
    - Meet your instructor 420
    - Audience 420
- Lesson 1: Getting acquainted with Scheme 421
    - Let's start Scheme'ing 421
    - Watch out for extra parens. 422
- Lesson 2: Of Variables and Functions... 423
    - Variables 423
    - Declaring global variables with "set!" 423
    - Declaring local variables with "let" 423
    - White space 424
    - Assigning a new value to a variable 424
    - Functions 424
- Lesson 3: '(Lists Lists and More Lists) 425
    - Defining a list 425
    - Concatenating variables to lists 426
    - Accessing values in a list 426
    - car 426
    - cdr 426
    - Accessing other elements a list 427
- Lesson 4: Your First Script-Fu Script 427

    - Creating a text box script 427
  - Getting started 427
    - Editing and storing your scripts 427
    - The bare essentials 428
    - Naming conventions 428
    - Registering the function 428
    - Steps for registering the script 430
    - The required parameters 430
    - Registering the script's parameters 431
- Lesson 5: Giving our script some guts 432
  - Creating a new image 432
    - Adding a new layer to the image 433
    - Adding the text 434
    - Clearing the "dirty" flag 435
    - Enabling and Disabling undo 435
  - Lesson 6: Extending the Text Box Script 436
    - The game plan 436
    - Modifying the parameters and the registration function 436
    - Adding the new code 437

Part IX Adv installations 439

Chapter 40 How to get fonts to Gimp 441

- How fonts work in Gimp 442
  - Scalable fonts 442
  - Where are the fonts and font PATH 442
- Installing fonts 443
  - Type 1 fonts installation & the type1inst program 443
    - Preparing for installation 443
    - Copying the fonts to the fonts dir 443
    - Font management 443
    - Running type1inst 443
    - Loading the fonts into X 444
  - Installing type 1 fonts by hand 444
    - The font file 444
  - The font field in the font file 444
    - Extracting data 445
- Tables 446

Foundery table 446
Weight table 447
Slant table 447
Set Width table 448
Additional style table 448

Chapter 41 Compiling plug-ins 451
What is a plug-in? 452
Compile? 452
What way to go when you want to compile 452
How to obtain and install the source code 452
Unpacking the source code 453
Compiling the code 453
Finding out how to compile the plug-in 453
Using GCC to compile the plug-in straight off 454
A first try 454
Libraries 455
Another try 455
Include file 455
The -L flag and how to find out which libs to link 455
The -I flag 456
What to do when there are several source files 456
How to create a Makefile and how to use it 457
A Makefile example 458
Variables 458
Configure a way to automate the building process 459

Appendix 461

Appendix A Gimp start flags and rc-files 463
Gimp command line switches aka flags (options) 464
Batch mode and ”no-interface” 465
More options 465
Initiations files aka rc files 467
gimprc and ~/.gimp/gimprc 467
menurc 475

    pluginrc 475
    gtkrc 475
  Installing a new Gimp 476

Appendix B Gimp man pages 477
  Gimp man page 478
  Gimp tool man page 481

Appendix C Commercial Gimp support 483
  WilberWorks 484
  Frozenriver 484

Appendix D SIOD: Scheme in One Defune, reference appendix 485
  Reference Section for built-in procedures 486

Appendix E Links and References 517
  Links 518
    Web 518
    Mail 518
    IRC (dev chat) channel #Gimp 519
    FTP 519
    Commercial Support 521
    Books 521

Index 523

part P

Preface

- *FROZENRIVER AND THE AUTHORS*
- *CONTRIBUTIONS*
- *CHANGELOG & TODO*
- *HOW TO READ GUM*

preface

a

Frozenriver and the authors of GUM

AUTHORS

KARIN KYLANDER

Karin is a computer designer and illustrator, but also an architect with a Master's degree in Architecture at the Chalmers' University of Technology. She has been working with graphic designs and art since 1985. Computer aided design entered her scene in the late 80:ies. In the beginning, she worked in a Mac and Windows environment with programs like Photoshop, Pagemaker, Corel Draw and Illustrator etc. In 1996 she entered the Unix arena, and in 1997 she started to use Gimp for image manipulation and graphic design. She is now using Gimp on a daily basis. Most of her work has been concerned with different publications, posters and exhibition displays. If you want to take a look some of her work, please check out the Gimp Around chapter. All artistic images in this manual are produced by her (there are some marked exceptions). She is now moving over to Web design and will for example create the web site for Gyve; which is a free drawing program and a counterpart to Gimp.

OLOF S KYLANDER

Olof is a Unix/Network system administrator. He received his formal computer education at the Chalmers University of Technology. He has been into computers since the early 80:ies. Unix caught is attention in 1993, and he has since been configurating various Unix systems as well as Networks. He is currently working for the Unix/Network consulting company Sigma-nbit in

Gothenburg, and is at the moment occupied by configuring Solaris servers for Ericson. He also has a knowledge of various other systems such as Mac, Windows, NT, Citrix, Novell etc. His speciality is thin clients and Xwindow configurations as well as Internet technologies. He is the author of the technical parts of this manual as well as most of the plug-ins.

FROZENRIVER

Frozenriver, which is Karin's company, deals with digital design in various fields as well as with training courses in digital image manipulation. The speciality is different kinds of informational material such as technical documentation/reports, brochures, magazines and exhibitions as well as webdesign. Frozenriver can provide the entire range from a full advertisement concept to a leaflet at the local mall. Frozenriver can also provide support for Gimp users (and also other image manipulating programs such as Photoshop) in the form of training courses and advice by email.

By hiring Frozenriver for designing your advertisement, web or assisting at your educational program, you will ensure that we can continue developing and maintaining GUM and other free manuals for free design programs in Unix environments (e.g the Gyve user manual GYUM).

If you want to contact Frozenriver, please visit our website at www.frozenriver.com or mail us at karin@frozenriver.ale.se (from summer -98 karin@frozenriver.com).

Or even better, contact us directly:

`Frozenriver`
`Karin Kylander`
`N.Dragspelsg 12`
`S-421 43 V.FRÖLUNDA`
`SWEDEN`
`Phone: +46 (0)31 47 43 56`
`Fax: +46 (0)31 49 48 33`

preface

b

Contributions

CONTRIBUTIONS

GIMP CONTRIBUTIONS

First we want to thank all the developers of the Gimp:

Spencer Kimball, Peter Mattis, Federico Mena-Quintero, Xach Beane, Adrian Linkns, Miguel de Icaza, Tom Bech, Sven Neuman, Albert Cahalan, Adam D. Moss, Torsten Martinsen, Tristan Tarrant, Andreas Beck, David Mosberger, Gordon Matzigkeit, Peter Kirchgessner, Eric L. Hernes, Francisco Bustamante, Thorsten Schnier, Jochen Friedrich, Tim Newsome, Christoph Hoegl, Xavier Bouchoux, Owen Taylor, Andy Thomas, Ray Lehtiniemi, Marcelo de G Malheiros, Miles O'Neal, Chris Laas, Daniel Risacher, Gerd Knorr, Michel Taylor, Ole Steinfatt, Michael Sweet, Eiichi Takamori, Tracy Scott, Gordon Matzigkeit, Andrew Kieschnick, Alexander Schulz, Thomas Noel, Robert L.Cross, Kevin Turner, Sean Cier, Nick Lamb, Kim-Minh Kaplan, Matthias Cramer, Lauri Alanko, Tim Newsome, Bucky LaDieu, Scott Goehring, Morten Eriksen, Raphaël Quinet, Daniel Skarda, Daniel Dunbar, Jens Ch. Restemeier, Marc Lehmann, Scott Draves, Alessandro Baldoni, Michael Schubart, Dan Risache, Josh MacDonald, Eduardo Perez, Daniel Cotting, Nathan Summers, John Beales, Marc Bless, John Breen, Brent Burton, Jim Geuther, Pavel Grinfeld, Matthias Greim, Jan Hubicka, Shuji Narazaki, Stephen Norris, Tim Rowley, Christoph Hoegl, Wolfgang Hofer and all of you we have forgotten (please write to us karingimp@frozenriver.ale.se later frozenriver.com)

GUM CONTRIBUTIONS

We also want to give credit to everybody coming with suggestions, tips, constructive criticism, contributions etc..

- Dov Grobgeld author of Writing a Script Fu,
- Mike Terry author of ”black belt school of Script-Fu”
- John Sigerson pdf file format,
- Aristidi Yannick French translation
- Yasuhiro Shirasaki leader of the Japanese translation team
- Mark Probst Documentation
- Peter Uray Documentation
- Petri Alanko Documentation
- Ole Steinfatt Documentation
- Michal Gomulinski Documentation
- George J Carret Documentation
- Eric Galluzzo and Christopher Macgowan proof reading
- Nicholas Lamb tip about selections,
- Michael Kaiser correction layers,
- Cristoph Hogeld contrib correction
- Marco Schmidt
- Adrian Links
- Adam D. Moss tips about animation filters and psd
- Tom Bech tips & lesson about light effects and map objetcs
- Nathan Carl Summers tips & lesson about iscissors
- Wolfgang Hofer tips about animframe
- and all of you we have forgot (please write to us karingimp@frozenriver.ale.se, later frozenriver.com)

preface

C

Changelog and TODO

CHANGELOG

1.0.0

- This release is first stable GUM version.
- A better layout
- Devided in to several parts
- New chapters, Gallery, Prepress and colors, Anim Frames i.e gap, Mike Terrys script fu totorial, Fonts install, Compile plug-ins
- Several new Appendix and preface ”chapters”
- An index
- A TOC
- A TOC at a glance
- All the filters are now devided in to seperate chapters
- Chages in all chapters minor as well as major (to mutch to doc in the change log)
- Etc.... it’s so mutch
- All plug-ins (except script and database) that we could get hold of as of 19/4 is documeted.

0.9

- All plug-ins up to Gimp standard Dec 17 1997 have been documented plus some non-standard.

- All chapters improved, proof-read and corrected
- Update on Gimp core behavior
- Better examples and many new images

0.8

- Not released publicly

0.7

- Plug-ins chapter is now separated into several chapters
- Many new images and examples in the Image menu chapter
- Many new plug-ins described
- Minor correction in all chapters
- A raw TOC

0.6

- GPL license
- New Chapters: What is Gimp, Image menu, Selection menu, Edit menu, Legals. and Writing a Script Fu (thanks to Dov Grobgeld)
- Many new images in nearly all chapters.
- Major and Minor correction is in all chapters
- More plug-ins described.

0.5

- first public release

TODO

- Describe SANE interface and other Extensions.
- Write some sort of WOW book for Gimp
- Get some desent sleep
- Fix errors
- Apply more of Eric Galluzzo and Christopher Macgowan proofs
- Update all kinds of stuf on a regular basis

preface

d

How to read GUM

AN INTRODUCTION

GUM is the complete Gimp user manual. GUM is the most comprehensive source of information available, and covers nearly all aspects of Gimp. It's a user manual, so it will not cover how to write plug-ins to Gimp, however, some basic scripting tutorials have been included. All images in this manual have been created or manipulated with Gimp exclusively, no other software has been used..

GUM is divided into several parts. If you are an experienced graphics artist, you can read the first parts quickly in order to pick up the main differences between Gimp and the programs that you are used to working with.

GUM also covers features which aren't part of the standard Gimp distribution. These features may be found in the unstable distribution or at the plug-in registry. We have covered all available Gimp features up to the 23/3 -98, with the exception of how to use pressure sensitive drawing tablets, Gimp perl scripting extensions, Dumpwindow, xmorph (a Gimpified version of the xmorph program) and HaruspexX (a SQL Gimp extension).

Preface

- About Frozenriver and the authors
- Contributions
- Changelog
- How to read GUM

Part one

- What is Gimp is; a brief ***history*** of Gimp and Gtk
- Gimp's default ***short cuts*** (accelerator keys) and how to reassign them

Gallery

- This is a gallery showing what you can achieve with the powerful resources of Gimp. It also gives an insight into ***advanced image manipulation***.

Part two

- How to ***get*** and ***install*** Gimp for your system. ***Troubleshooting***.

Part three

- What ***file format*** Gimp supports and how to use them. How to ***open*** and ***save*** files in Gimp
- Personal ***adjustments***
- How to use the different ***Paint tools***.
- How you can use the different ***Edit*** functions in Gimp
- ***Transformation*** functions in Gimp
- How to work with ***Text*** in Gimp
- How to use the ***Gradient editor*** as well as information about ***Brushes***, ***Palettes*** and ***Patterns***.

Part four

- A general discussion about ***color models***. To understand how different ***modes*** work in Gimp, you need the information in this chapter.
- How to prepare your Gimp image for ***prepress***
- ***Color calibration*** discussion and a simple color calibration of your system

Part five

- In depth discussion about the ***image menu***, which includes ***color, brightness, curves and other image adjustments***. It also covers ***image conversions*** like RGB to Indexed as well as ***image transformations***.
- How to use different ***selection*** methods
- How ***modes*** work in Gimp
- In depth discussion on how to use ***layers*** - the key factor to advanced image manipulation
- ***Channels*****;** what they are and how to use them

Part six

- This part is about the ***filter plug-ins*** available for Gimp, a glimpse of what it's all about
- Different ***color exchange*** filters
- How to use the ***lighting effects*** in Gimp
- ***Render*** fantastic patterns and images in Gimp
- Etc.

Part seven

- ***Animations*** in Gimp or how ***AnimFrames*** can make it easy to create advanced web animations.

Part eight

- Discussion about the ***Script-Fu:s*** that come with Gimp
- Two different angles/tutorials on how to ***write Script-fu:s*** and how they can help you automate Gimp tasks.

Part nine

- How ***fonts*** works in Gimp and Xwindow, how to install more fonts.
- How to ***compile*** plug-ins, ***make your own make file*** and how to use the ***configure script***

Appendixes

- ***Man*** pages in the Gimp distribution
- ***Initiation*** file descriptions as well as description of ***command*** line flags
- ***SIOD*** reference for those who write Script-Fu
- ***Links*** and ***books*** that can be useful

CONVENTIONS

You'll find four different typing styles besides the normal text in GUM.

We use bold italics to call out very important things and warnings:

Don't do this!

Less important things are emphasized with regular italics:

Well, *layers* are quite important, so you better learn how to handle them

For important items or issues we use bold text:

choose an appropriate **font**

For things that you have to execute, file names, commands and code we use Courier instead of Times:

`copy` the `gimprc` file to the new location

part I

About Gimp

- ABOUT GIMP AND GIMP HISTORY
- SHORT CUTS AND DYNAMIC KEY BINDINGS

chapter

1

What is Gimp?

A quick description of Gimp.

According to the ”About the Gimp” page at www.gimp.org We quote the following:

”ABOUT THE GIMP

GIMP is an acronym for GNU Image Manipulation Program. It is a freely distributed piece of software suitable for such tasks as photo retouching, image composition and image authoring.

It is an extremely capable piece of software with many capabilities. It can be used as a simple paint program, a expert quality photo retouching program, an on-line batch processing system, a mass production image renderer, a image format converter, etc.

GIMP is extremely expandable and extensible. It is designed to be augmented with plug-ins and extensions to do just about anything. The advanced scripting interface allows everything from the simplest task to the most complex image manipulation procedures to be easily scripted.

FEATURES AND CAPABILITIES

This is only a very quickly thrown together list of GIMP features. This is only the tip of the iceberg.

- Full suite of painting tools including Brush, Pencil, Airbrush, Clone, etc.
- Tile based memory management so image size is limited only by available disk space
- Sub-pixel Sampling for all paint tools for high quality anti-aliasing
- Full alpha channel support
- Layers and channels
- A Procedural Database for calling internal GIMP functions from external programs as in Script-fu
- Advanced scripting capabilities
- Multiple Undo/Redo (limited only by diskspace)
- Transformation tools including rotate, scale, shear and flip
- File formats supported include gif, jpg, png, xpm, tiff, tga, mpeg, ps, pdf, pcx, bmp, and many others.
- Load, display, convert, save to many file formats.
- Selection tools including rectangle, ellipse, free, fuzzy, bezier and intelligent
- Plug-ins which allow for the easy addition of new file formats and new effect filters.
- More Features here...

AUTHORS

The GIMP was written by Peter Mattis and Spencer Kimball. Many, many other developers have contributed plug-ins. And thousands have provided support and testing.

GIMP releases are currently being orchestrated by Manish Singh.”

End quote

WHAT WE CAN SAY ABOUT GIMP

First we want to congratulate Peter Mattis and Spencer Kimball and all of the other developers of this lovely program. What is said at ”About the Gimp” is only the tip of the iceberg. Gimp is capable of everything from advanced image manipulation to basic drawing. Many of its features are inspired by Photoshop and other image manipulation programs.

Karin, who is an architect and designer and a former Photoshop user in both MAC and Windows environment, can only say this:

Compared to Photoshop, Gimp has it all, and even more if you don’t buy third party plug-ins. Most of the features in Gimp are more flexible and powerful when you get to know them.The great thing is that Gimp supports psd fileformat and Filter Factory afs files, so you can easily switch from Photoshop to Gimp. Simply, it’s a hack of a program and it’s comes loaded with a sack of plug-ins. So GO AND GET IT!! you will not be disappointed, and well, it’s not wrong that it is free...

Karin Kylander & Olof S Kylander

GIMP HISTORY

0.54

We want to quote Peter Mattis and Spencer Kimball who are the original creators of Gimp:

”The GIMP arose from the ashes of a hideously crafted cs164 (compilers) class project. The setting: early morning. We were both weary from lack of sleep and the terrible strain of programming a compiler in LISP. The limits of our patience had long been exceeded, and yet still the dam held.

And then it happened. Common LISP messily dumped core when it could not allocate the 17 MB it needed to generate a parser for a simple grammar using ”jyack”. An unbelieving moment passed, there was one shared look of disgust, and then our project was vapor. We had to write something...ANYTHING...useful. Something in C. Something that did not rely on nested lists to represent a bitmap. Thus, the GIMP was born.

Like the phoenix, glorious, new life sprung out of the burnt remnants of LISP and jyacc. Ideas went flying, decisions were made, the GIMP began to take form.

An image manipulation program was the consensus. A program which would at the very least lessen the necessity of using commercial software under ”Windoze” or on the ”Macintoy”. A program that would provide the features missing from the other X painting and imaging tools. A program that would help maintain the long tradition of excellent and free UNIX applications.

Six months later, we've reached an early BETA stage. We want to release now to start working on compatibility issues and cross platform stability. Also, we feel now that the program is actually usable and would like to see other interested programmers developing plug-ins and various file format support."

end quote.

0.54 was released in February 1996, and had a major impact as the first truly professional free image manipulation program. This was the first free program that could compete with the big commercial image manipulation programs.

0.54 featured:

- Supports 8, 15, 16 and 24 bit color
- Ordered and Floyd-Steinberg dithering for 8 bit displays
- View images as rgb color, grayscale or indexed color
- Simultaneously edit multiple images
- Zoom and pan in real-time
- GIF, JPEG, PNG, TIFF and XPM support
- Selection tools including rectangle, ellipse, free, fuzzy, bezier and intelligent scissors
- Transformation tools including rotate, scale, shear and flip
- Painting tools including bucket, brush, airbrush, clone, convolve, blend and text
- Effects filters (such as blur, edge detect)
- Channel & color operations (such as add, composite, decompose)
- Plug-ins which allow for the easy addition of new file formats and new effect filters
- Multiple undo/redo (note this is called new feature in Photoshop 5)

Even if 0.54 was a beta software, it was so stable that you could use it for daily work. However, one of the major drawbacks of 0.54 was that the toolkit (the code that the slidebars, menus, file dialogs etc.). was built on Motif, which is a commercial toolkit. This was a big drawback for systems like Linux, because you had to buy Motif if you wanted to use the faster, dynamically linked Gimp. Many developers were also students running Linux, who could probably not afford to buy Motif.

0.60

When 0.60 was released in July 96, it had been under S&P (Spencer & Peter) development for four months. One of the main programming advantages was the new toolkit called Gtk (Gimp toolkit), which removed the Motif problem. For the graphic artist, 0.60 was full with new features like:

- Basic Layers

- Improved painting tools (sub-pixel sampling, brush spacing)
- A better airbrush
- Cloning between all image types
- A pattern selection dialog, and a clone tool making it possible to clone from the active pattern
- Paint modes
- Border and Feather selection commands
- Select by color
- Better palette handling
- etc..

0.60 was only a developer's release, and was not intended for a widespread audience. It served as a workbench for 0.99 and the final 1.0 version., so functions and enhancement could be tested and dropped or changed. You can look at 0.6 as the alpha version of 0.99.

0.99

In February 97 0.99.X entered the scene. Together with other developers, S&P had made several changes to Gimp and added even more features. The main difference was the new API and the PDB which made it possible to write scripts; Script-Fu's (or macros) could now automate things that you would normally do by hand. 0.99 used a new form of tilebased memory handling which made it possible to load huge images into Gimp (loading a 100MB image to Gimp is no problem). 0.99 also introduced a new native Gimp file format called XCF.

The new API made it really easy to write extensions and plug-ins for Gimp. Several new plug-ins and extensions emerged to make Gimp even more useful (e.g SANE that enables direct scan in to Gimp). At the time we're writing this, Gimp has more than 150 plug-ins, covering everything from file formats to fractal tracers.

In the summer of -97, Gimp had reached version 0.99.10, and S&P had to drop most of their support since they had graduated and got jobs. However, the other developers of Gimp continued under the orchestration of Federico Mena to make Gimp ready for primetime. Gtk was separated from Gimp in September -97. Gtk had been recognized as an excellent toolkit to build other applications with, and other developers had started to use Gtk to build their applications.

Gimp goes into feature freeze in October 97, this means that no new features will be added to the Gimp core libraries and program. GUM version 0.5 is also released early in October -97. The developing work continues to make Gimp stable and ready for version 1.0.

1.0

Gimp version 1.0 is released 19 May 1998 Finally, Gimp is considered stable enough for a worldwide announcement.

THE FUTURE OF GIMP

Gimp will naturally continue to evolve. The future is bright, and we will see new versions of Gimp with new features and functions. The naming convention of Gimp will be the same as for Linux, meaning that the stable version will be called 1.0.X and the development version will be called 1.1.X. This makes it possible for normal users to grab the stable version and use it for prime time job, while the developers can work on a bleeding edge version without introducing new bugs into the stable version. If you want to, you can always download the development version and test it to check out the new features and give the developers feedback about bugs and enhancements, but be aware, it will be unstable, so don’t use it for your daily work and don’t flood the developers with bug reports (of unnecessary character).

chapter

2

Default short cuts and dynamic keybinding

Gimp has a ton of accelerator keys which can help you to quicker and easier image manipulation. Gimp also has a fantastic feature in the possibility to reassign short cuts keys on the fly.

DYNAMIC KEY BINDINGS

Gimp has a very nice way of dealing with **short cuts** or **hotkeys**. If you don't like a default short cut or if your favorite command hasn't been assigned to a default hotkey, then just bring up the menu holding the command, for example `<Image>/File/Preference` and press an alternative key, like `Ctrl-U`. The preference dialog will now pop up when you press `Ctrl-U`. (remember not to have caps lock on). If Ctrl-U was occupied by e.g `<Image>/ File /Print` before you did the reassignment, then this short cut has been erased. All of this happened on the fly. You will actually see the reassignment take place and there is no need to restart Gimp. *(in some other operating systems you'll get messages like ""$$" has discovered that the mouse cursor has changed position, please restart "$$" so it can be updated")*

This can also be done by editing your **personal menu rc file** in your personal gimp directory. This makes it possible to share your short cuts with your friends. If you are a former Photoshop user, you can for example re-bind your keys so Gimp will use the same hotkeys as Photoshop.

As authors of this book we suggest that you stick to the default short cuts. Otherwise it may be difficult to follow our instructions *(you may run in to some problems when we write Ctrl-N to get a new image, and you quit Gimp because you have reassigned the short cut).*

If you want to remove your personal short cuts just type `rm .gimp/menurc` in an xterm shell.

The next pages display a **Quick Reference** to the default keybindings in Gimp. It's a good idea to print this page, and keep it beside your computer.

CONVENTIONS

- The left column lists options available for a certain tool (clicking different mouse buttons and/or pressing hotkeys). The right column describes the result.
- The different tools are listed in the left column, and the hotkeys for calling up each tool are listed in the right column.
- `{1}` represents left mouse button
- `{2}` represents the middle mouse button
- `{3}` represents the right mouse button
- `Shift + {1} - Shift + Shift + Ctrl` means the following:
- First press `Shift`, then press the `left mouse button.` Release `Shift` then press `Shift` again. Press `Ctrl` and drag the `mouse` while holding `Shift` and `Ctrl`. When you are finished, first release the mouse button, then release `Shift` and `Ctrl`.

DEFAULT SHORT CUTS IN GIMP.

Selection tools	
{1}	Draw selection
{1} + {3} - {1}	Cancel selection
On existing selections	
Shift + {1} - Shift	Add to selection
Ctrl + {1} - Ctrl	Subtract from selection
Shift + Ctrl + {1} - Shift - Ctrl	Union of selection
Alt + {1}	Move selection
Rectangular and ellipse	**Short cut:R/E**
{1} + Shift	Circle/square only
{1} + Ctrl	Start from centre
{1} + Shift + Ctrl	Start circle/square from centre
Shift + {1} - Shift + Shift	Add circle/square to selection
Shift + {1} - Shift + Ctrl	Add rect/ellipse to selection starting from centre
Shift + {1} - Shift + Shift + Ctrl	Add circle/square to selection starting from centre
Ctrl + {1} - Ctrl + Shift	Subtract circle/ square from selection
Ctrl + {1} - Ctrl + Ctrl	Subtract ellipse/ square from selection starting from centre
Ctrl + {1} - Ctrl + Shift +Ctrl	Subtract circle/ square from selection starting from centre
Shift + Ctrl + {1} - Shift - Ctrl + Shift	Circle/square union of selection
Shift + Ctrl + {1} - Shift - Ctrl + Ctrl	Ellipse/rectangle union of selection starting from centre
Shift + Ctrl + {1} - Shift - Ctrl + Shift + Ctrl	Circle/square union of selection starting from centre
Bezier	B
{1} (inside bezier)	Convert to selection
{1}	Move both control handles
Shift + {1}	Move one control handle
Ctrl + {1}	Move control point
Remaining selection tools	
Fuzzy select	Z
Free	F
Intelligent scissors	I

.

Move tool	M
{1}	Move current layer
{1} + Shift	Move current even if 100% transparent
Ctrl + [arrow key]	Move layer 1 pixel
Shift + [arrow key]	Move layer 25 pixels
Alt + [arrow key]	Move selection 1 pixel
Alt + Shift + [arrow key]	Move selection 25 pixels

Magnify Tool	Shift + M
{1} or =	Zoom in
Shift + {1} or -	Zoom out
{2}	Pan image

Crop tool	Shift + C
{1}	Make crop selection
{1} + {3} -{1}	Cancel crop
{1} inside crop	Perform crop

Text tool	T
{1}	Set top left of text

Color picker tool	O
{1}	Get color

Transform tool	Shift + T
Rotation mode	
{1}	Rotate with 1° increments
Ctrl + {1}	Rotate with 15˚ increments
Scaling mode	
{1}	Free scaling
Shift + {1}	Scale in X direction only
Ctrl + {1}	Scale in Y direction only
Shift + Ctrl + {1}	Scale with fixed ratio
Shearing mode	
{1}	Free shearing
Perspective mode	
{1}	Move point
All	
{1} + {3} - {3}	Preview

.

Bucket fill tool	Shift + M
With no selection	
{1}	Fill with FG color
Shift + {1}	Fill with BG color
With selection	
{1}	Fill selection with FG color
Shift + {1}	Fill selection with BG color

Colorpicker tool	O
{1}	Set active color with cursor
X	Swap BG FG color
D	Set default colors

Blend tool	L
{1} + drag - {1}	Set start => end of gradient
{1} + {2} - {1}	Cancel gradient

Paint tools	
Blend tool	L
Pencil tool	Shift + P
Airbrush tool	A
{1}	Paint
Alt + {1}	Quick draw
Shift +/- {1} + distance +/- {1}	Line draw

Eraser tool	Shift + E
{1}	Set to BG and Clear
Alt + {1}	Quick erase
Shift +/- {1} + distance +/- {1}	Line erase

Clone tool	C
{1}	Clone
Ctrl + {1}	Set clone source point
Alt + {1}	Quick clone
Shift +/- {1} + distance +/-! {1}	Line clone

Convolver Tool	V
{1}	Blur/Sharpen with brush

..

File	
New	Ctrl + N
Open	Ctrl + O
Close	Ctrl + W
Quit	Ctrl + Q

Dialogs	
Brushes	Shift + Ctrl + B
Patterns	Shift + Ctrl + P
Palette	Ctrl + P
Gradient editor	Ctrl + G
Tool options	Shift + Ctrl + T
Layers & Channels	Ctrl + L

Edit	
Cut	Ctrl + X
Copy	Ctrl + C
Paste	Ctrl + V
Clear	Ctrl + K
Fill	Ctrl + .
Undo	Ctrl + Z
Redo	Ctrl + R
Cut Named	Shift + Ctrl + X
Copy Named	Shift + Ctrl + C
Paste Named	Shift + Ctrl + V

Filters	
Repeat Last	Alt + F
Re-show Last	Shift + Alt + F

Select	
Toggle	Ctrl + T
Invert	Ctrl + I
All	Ctrl + A
None	Shift + Ctrl + A
Float	Shift + Ctrl + L
Sharpen	Shift + Ctrl + H
Feather	Shift + Ctrl + F

View	
Zoom in	=
Zoom out	-
Zoom 1:1	1
Window info	Shift + Ctrl + I
Toggle Rulers	Shift + Ctrl + R
Toggle Guides	Shift + Ctrl + T
Shrink Wrap	Ctrl + E

Image	
Channel Ops Dupli	Ctrl + D
Channel Ops Offset	Shift + Ctrl + O

Layers	
Dialog	Ctrl + L
Raise Layer	Ctrl + F
Lower Layer	Ctrl + B
Merge Visible Layers	Ctrl + M

Default short cuts in Gimp.

Tools	
Rect Select	R
Ellipse Select	E
Fuzzy Select	Z
Bezier Select	B
Intelligent Scissors	I
Move	M
Magnify	Shift + M
Crop	Shift + C
Transform	Shift + T
Flip	Shift + F
Text	T
Color Picker	O
Bucket Fill	Shift + B
Blend	L
Paintbrush	P
Pencil	Shift + P
Eraser	Shift + E
Airbrush	A
Clone	C
Convolve	V

Layers & Channels dialog	
Layers ops menu	
New layer	Ctrl + N
Raise layer	Ctrl + F
Lower layer	Ctrl + B
Duplicate layer	Ctrl + C
Delete layer	Ctrl + X
Scale layer	Ctrl + S
Resize layer	Ctrl + R
Merge visible layers	Ctrl + M
{1}	Select layer (visible and anchor)
Shift + {1}	View current layer only
Alt + {1} Green layer mask	View layer mask (or remove green mask)
Alt + {1} Green layer mask	View layer mask (or remove green mask)
Alt + {1} Green layer mask	View layer mask (or remove green mask)
Channels ops menu	
New channel	Ctrl + N
Raise channel	Ctrl + F
Lower channel	Ctrl + B
Duplicate channel	Ctrl + C
Delete channel	Ctrl + X
Channel to selection	Ctrl + S

.

Gradient Editor dialog: Ops menu	
Left endpoint color	L
Load from Left neighbor's right endpoint	Ctrl + L
Load from Right endpoint	Alt + L
Load from FG color	Ctrl +
Right endpoint color	R
Load from Right neighbor's right endpoint	Ctrl + R
Load from Left endpoint	Alt + R
Load from FG color	Alt + F
Segments	
Split segment at midpoint	S
Split segment uniformly	U
Delete segment	D
Re-centre segment's midpoint	C
Re-distribute handles in segment	Ctrl + C

Selection operands	
Flip segments	F
Replicate segment	M
Blend endpoints' colors	B
Blend endpoints' opacity	Ctrl + B

part

G

Gallery

- GIMP AROUND IN 80 MINUTES

chapter 3

Gimp around in 80 minutes

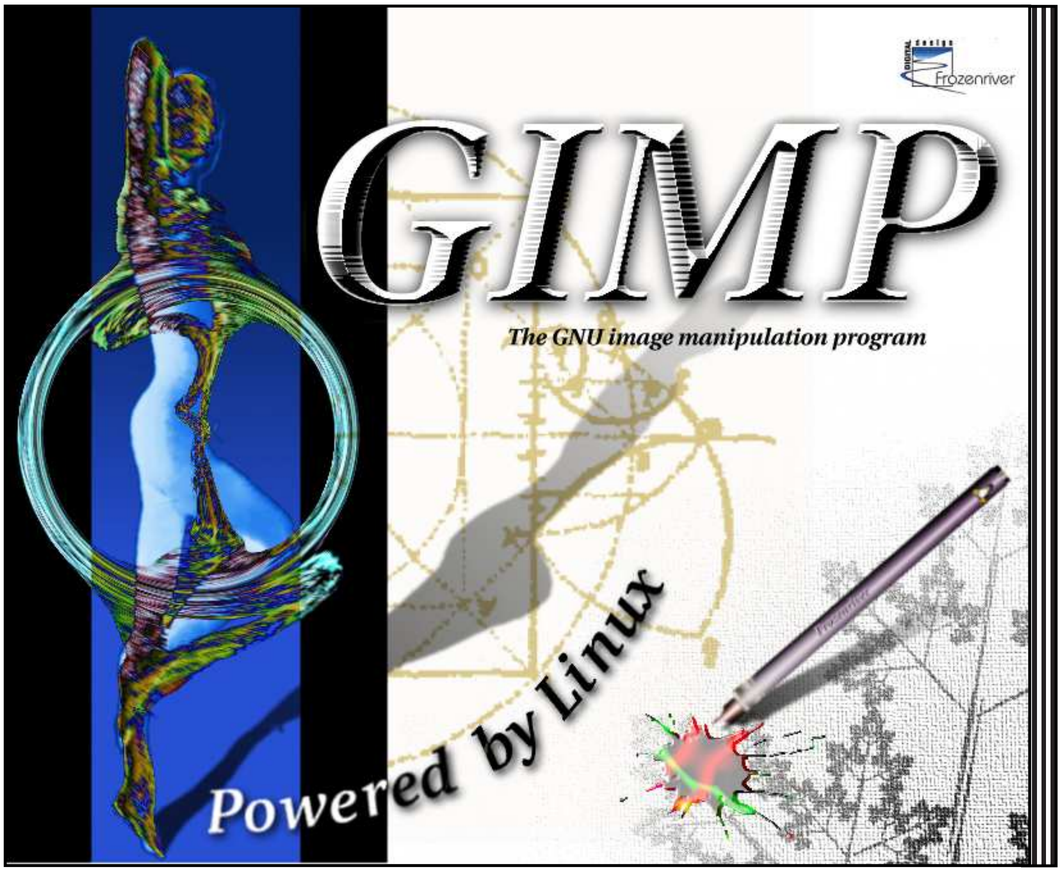
Don't underestimate the power of Gimp...

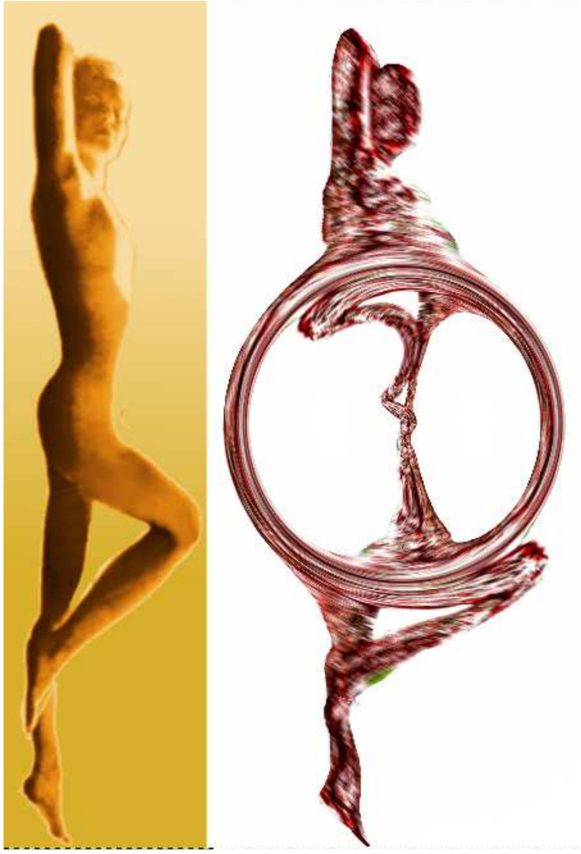
This manual describes the many functions, plug-ins and options Gimp has to offer, but it doesn't describe how to create great digital art or designs. There are probably as many Gimp tricks and tips as there are Gimp-users, and even if we wanted to, we couldn't include them all in this book. We can't teach you how to be an artist, but we've included a few examples here that will hopefully inspire you to new ideas and help you on the way to getting the best out of Gimp. This is more a gallery than a tutorial, and the object of this chapter is not to give detailed instructions, but rather to demonstrate the great versatility and power of Gimp to beginners, and maybe give an insight on new ways of using Gimp to more experienced users.

so, let's unleash the power of Gimp and Unix!

CREATING IMAGE OBJECTS

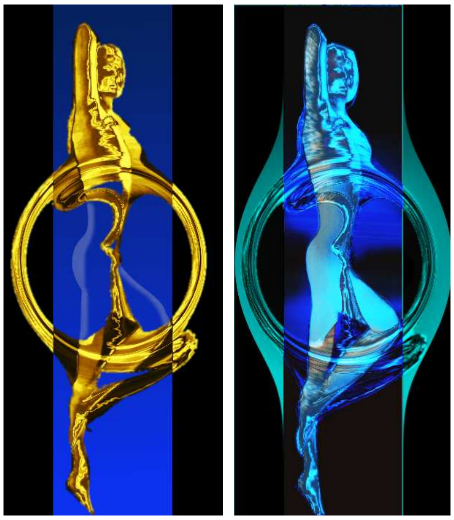
You don’t always need to import a photo, drawing or 3D-image of an object. There are many ways of creating astonishingly convincing image objects directly in Gimp.


Frozenriver
GIMP
The GNU image manipulation program
Powered by Linux



A TWISTED PERSONALITY

To create the *emblem statue* I started out with a b/w photo. The background was isolated with the **bezier** tool, removed and replaced with a **gradient** fill. The image was rotated to vertical position, and tinted yellow with **Image/Hue-Saturation**. I reopened the selection made for changing the background, inverted it and used it again to isolate the figure, which was copied to a white layer. The figure was further adapted to supply an interesting surface for the **Distorts/Twist** plugin to work on. Finally, the yellow layer was applied in **Difference Mode** on top of the twisted layer.

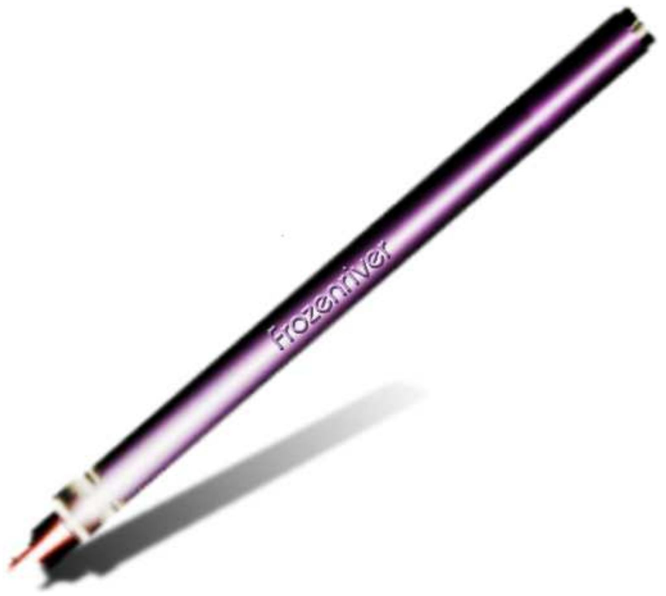


Variations - gold and water spirits

The *gold emblem* was created by running the **Gradient Map** filter on the twisted figure (using the custom Golden gradient). The pale outline of the non-twisted parts was painted with low opacity **airbrush** and blurred within a sharp-edged selection. The *blue ghost emblem* was made by three copies of the original yellow-tint image. The middle copy was twisted, trimmed and set to **Saturation Mode**, and the top layer was set in **Difference Mode**. To accentuate the water- or ghostlike appearance, the original twisted shape was added to a layer, desaturated and set to **Overlay**. The side parts needed to be more visible so they were pasted separately in **Multiply Mode**. Finally a pale fluorescent shape of the figure was added to enhance the shape of the woman inside the waterwheel.

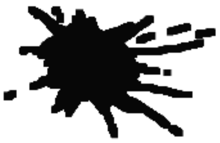
CREATING A SIMPLE PEN AND AN INK STAIN

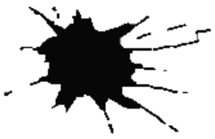
The *pen* was made by filling selection shapes with different gradients. In this case, I used **Bilinear FG to BG** with medium opacity, and also a number of **FG to transparent** gradients on several cylinder shaped selections. The metal pen tip was adjusted with **Brightness-Contrast** to achieve the metallic look. The pen shadow, as well as the emblem statue shadow, was made with the **Perspective Shadow Script-Fu**, cleared from color and filled with a FG to transparent linear gradient.

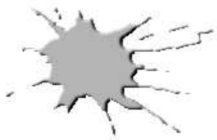


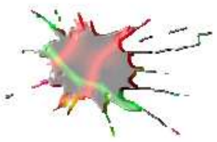
The technicolor ink spot

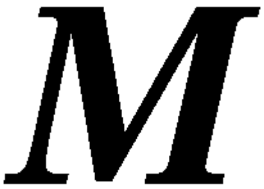
To make the *multi-color splash*, I started by drawing a simple black sun-shape on a white background, using a medium pencil tip. I then applied the **Distort/Value Propagate** plugin, choosing **more white**. The result was **blurred** and **bumpmapped** and the background was cut away. A copy of the splash was pasted to a transparent layer, filled with a colorful pattern and set to **Lighten Only Mode**.





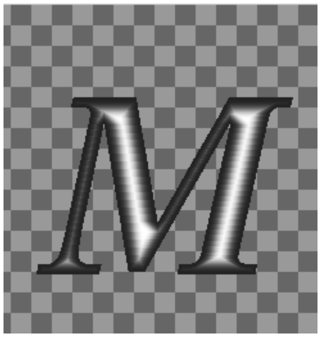






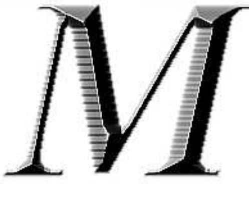
MAKING GROOVED TEXT

The carved *form pressed steel* text was generated by using a deliberately **jagged** (not antialiased) font. The text was filled with a black/white **shapeburst** gradient, the tonal range was adjusted with **Image/Levels** and the whole thing was **bumpmapped**. Final touch with **Image/Brightness-Contrast**. A gray glow was added to emphasise the shape of the letters, by copying the text layer, filling it with gray and applying **Gaussian blur** (with Keep Trans. unchecked).



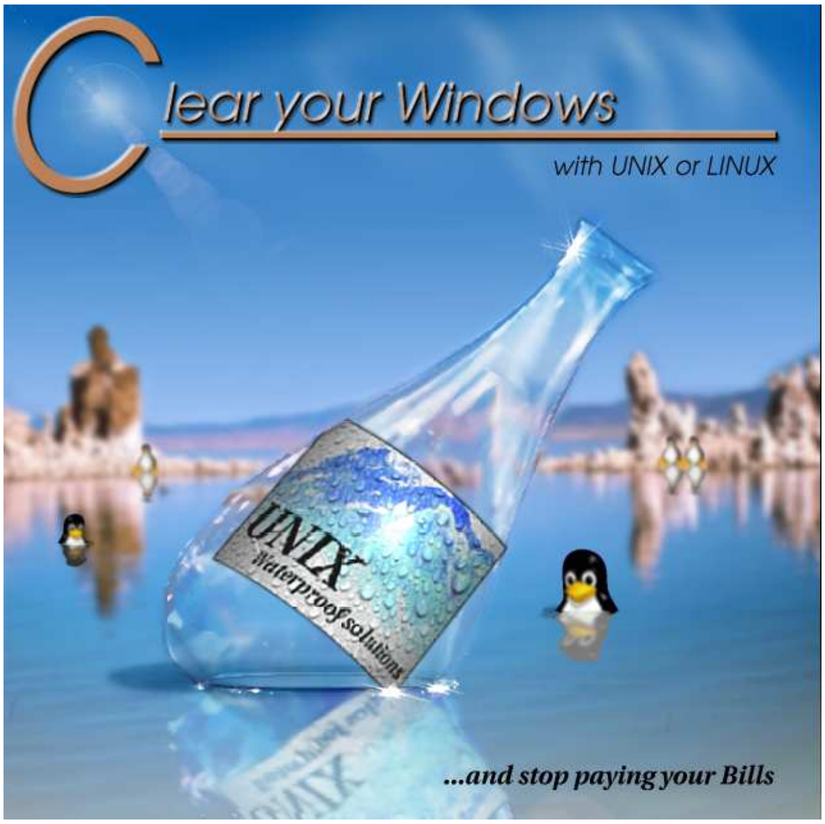
Organic patterns

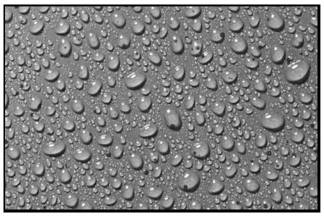
The *leaf pattern* was created in **Render/IfsCompose**, and **Artistic/Apply canvas** was added. The image was blended with the background with the help of a **Layer Mask** and a radial gradient.

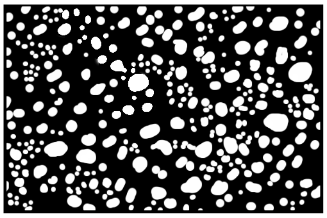


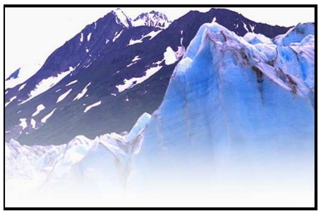
HANDLING GLASS, WATER AND REFLECTIONS

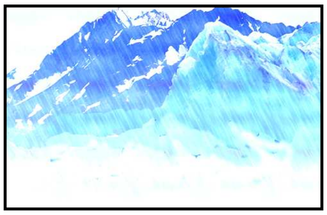
Glass and water effects are usually one of the hardest things one can try to set about, but with a little help from Gimp, you're halfway there.

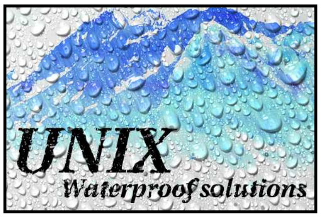

Clear your Windows
with UNIX or LINUX
UNIX
Waterproof solutions
...and stop paying your Bills

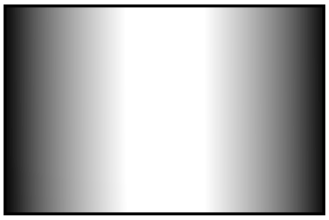


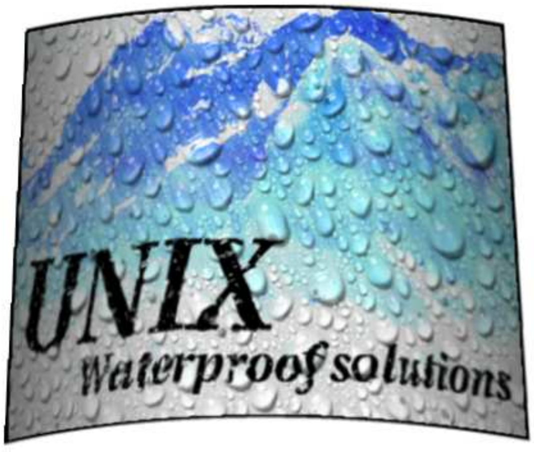












THE WET LOOK

Adding water

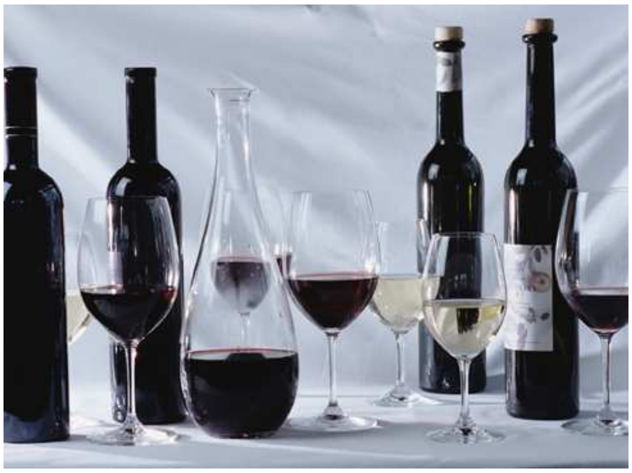
For the *Wet Unix label* I used a photo of waterdrops on a solid blue background. I **desaturated** it and made a duplicate to create a **highlight** and a **shadow** layer. This was achieved by adjusting the tonal range with **Image/Levels** as described in page x. To create the illusion of water, I needed to displace the background where the drops were, so I opened a new **Channel** and painted a mask for the drops. The channel selection was loaded on a black layer which I called Displace layer, and the drop shapes were filled with a b/w **shapeburst** gradient..

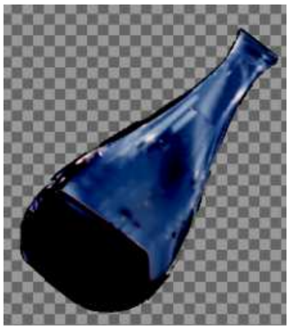
Making rain

I used a nice, clean photo of a mountain top as background image, and added a little fog with a **FG to Transparent** gradient. To stylize and add some wetness to the image I created a rain layer. This layer was filled with the custom **Rain pattern**, darkened somewhat and set in **Addition Mode**. A text layer was also added. Then I ran the **displace filter** on the text and the mountain background, using the Displace layer with the drops.

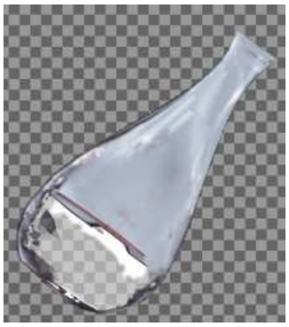
Displacing along a curve

To make the label fit the bottle shape, I made a new **displacement map** with the gradient editor (dark to the left and right and bright in the middle to make a round displacement). After displacing the label, the displacement map was used to add a metallic sheen to the label by setting it in **Overlay Mode**. The final adjustments were made with the **Transform/Perspective** tool.

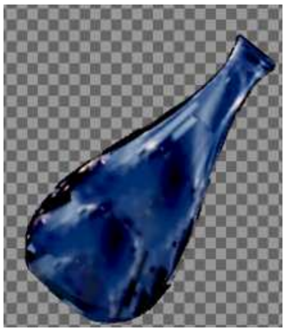




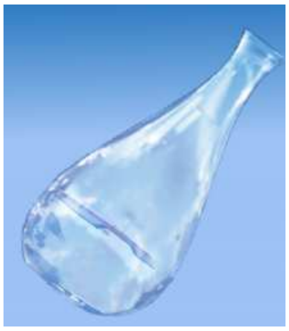
Highlight layer



Shadow layer



Recreated bottle



Result

HOW TO EMPTY A BOTTLE OF WINE

Cloning away unwanted parts

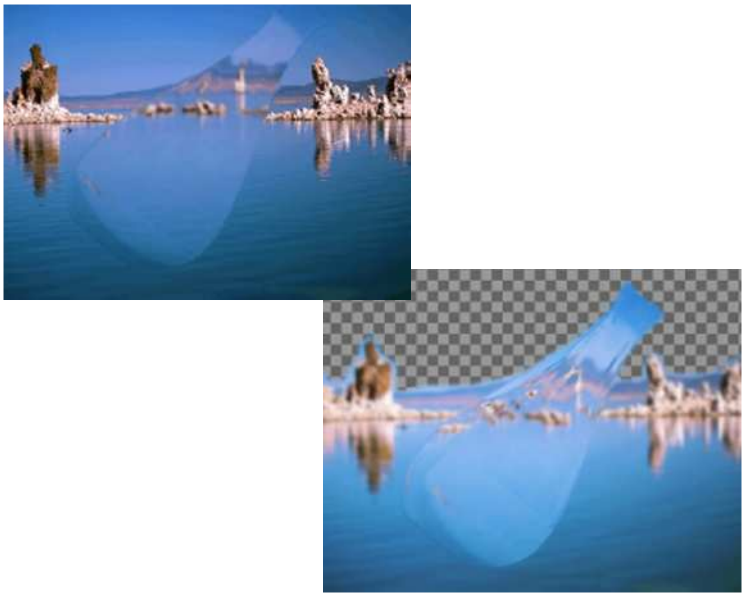
The *bottle* was made from a photo of a decanter filled with dark red wine. This was a bit troublesome because I wanted the bottle to be empty, or at least filled with some transparent liquid. The wine glass which was visible behind the bottle was easily **cloned** away, but the rest was left alone, most of it would be covered by the label anyway.

Making a dark and bright layer

The bottle was cut out, rotated and pasted to a transparent layer. A **highlight** and **shadow** bottle was created with **Levels**, (but not desaturated) and the highlight bottle was allowed to keep a quite large range of shades, otherwise the reflections in the glass would look too hard and unnatural.

Recreating missing parts

To cover the flat and dull looking part of the bottle (where the wine was) the bottom part was recreated by **cloning** different parts of the highlight bottle and set this layer to **Screen** Mode. The shadow bottle was set to **Multiply**.

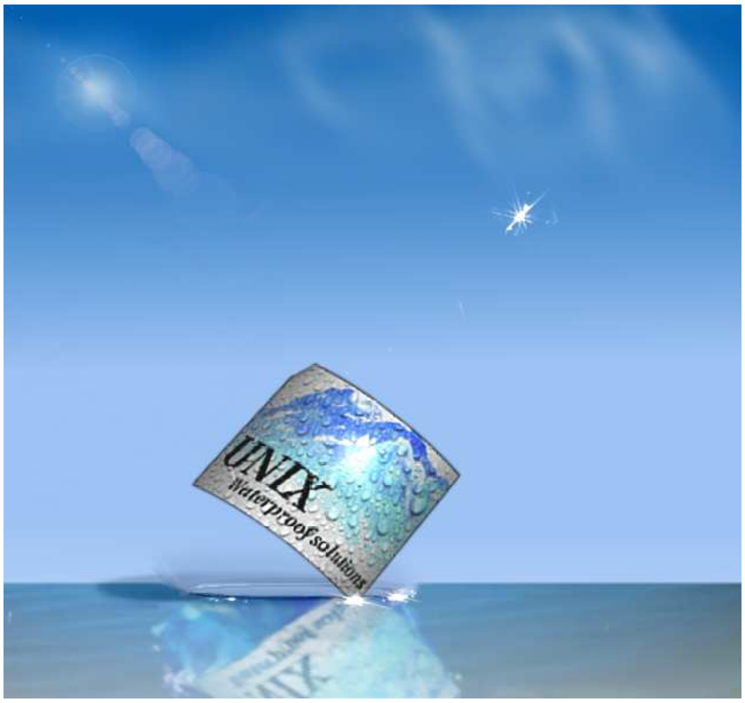


GLASS DISTORTION

The background, a photo of a lake, was blurred to create the illusion of distance, and to keep the focus on the bottle. The highlight bottle was used twice as a **displacement map** on the background so that the rocks in the background would appear to be distorted through the curved glass.

Rectifying banding

Because the lake image was originally **indexed**, the sky looked banded and ugly. This was rectified by **feather** selecting the sky, and replacing it with a linear **gradient**. A few clouds and a sun reflex was also added.



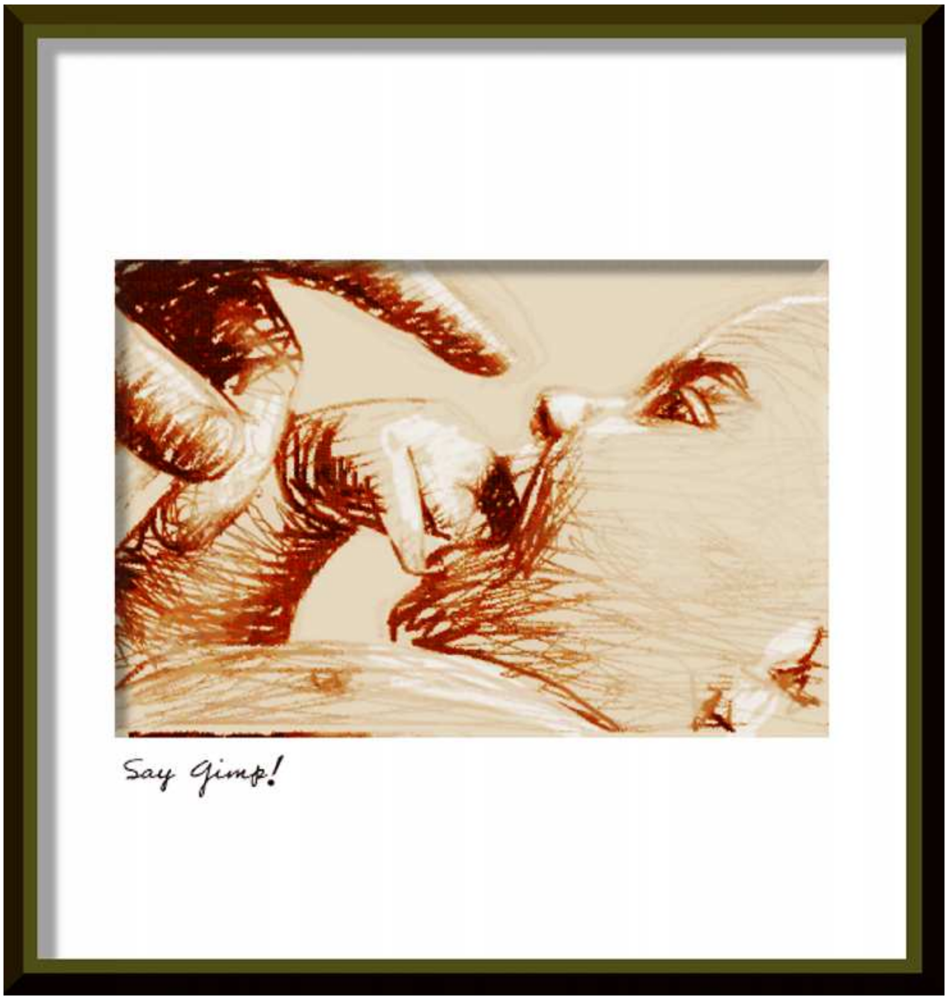
REFLECTIONS

The *water reflection* was made by flipping the image of the merged bottle to a copy of the lower part of the lake, blurring it, adding some ripples with **Distort/Ripple** and lowering the opacity level. The waterline (where bottle meets water) was a little trickier and had to be painted by hand in a couple of **Overlay** layers, and some water glitter was created with the sparkle filter.

A white haze was added to the foreground, and of course, a couple of Larry Ewing's adorable Linux penguins.

TRANSFORMING A PHOTOGRAPH TO A DRAWING

Because black is transparent in Screen mode (also Addition and Lighten Only) black pencil strokes drawn on a white layer will reveal the image underneath, just as if you had sketched all by hand. This is an easy way of creating convincing pencil/ink drawings from a scanned photo.

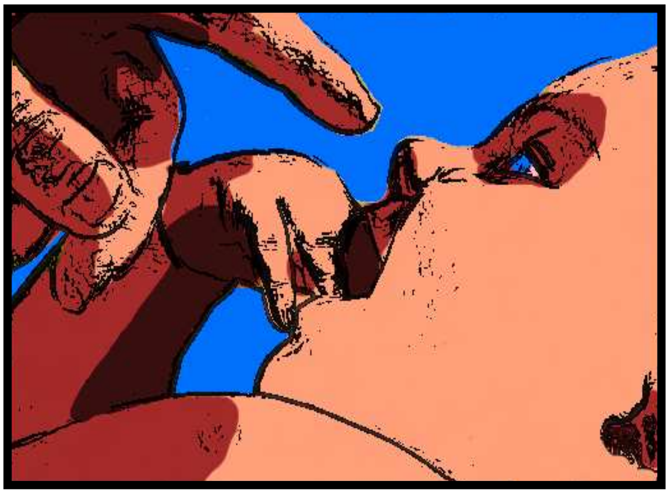


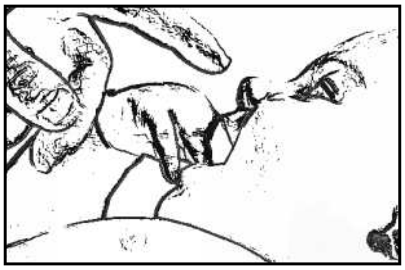
MANAGE WITHOUT ARTISTIC PLUGINS

There are several commercial plugins called things like *charcoal, crayon or ink drawing* that supposedly achieve these things. Certainly, these products can produce nice artistic outputs, but they never come close to the results you get using this method. Naturally, the quality of the final image depends a lot on the drawing in the screen layer, so this isn't an "instant artist" trick. A general advice is to reduce the number of shades in the background, otherwise too much of the underlaying image will show, and this will spoil the illusion. One way of improving coarse computer drawings is to use the **Value Propagate** filter and set it to *more white*. You can also create a crayon or charcoal look to an image by **displacing** or **warping** the pen strokes with a suitable displacement map, or just by using unusual **brushes**.

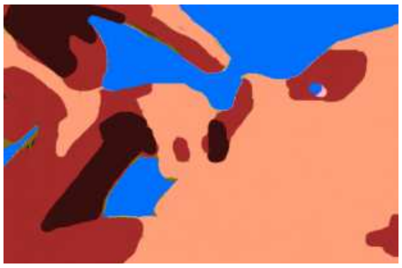
Instant cartoon pictures

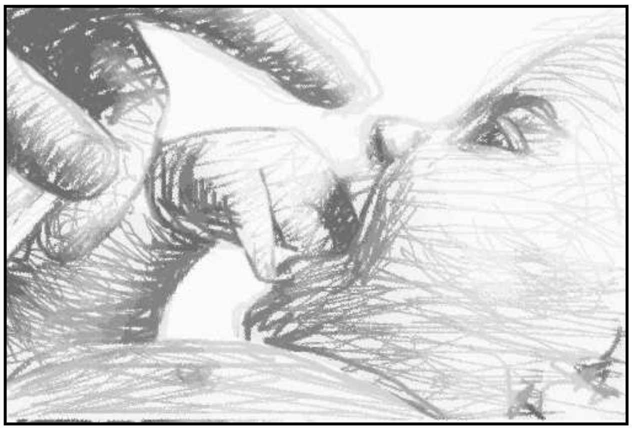
A very simple way of creating drawings from scanned photos is of course to use one of the **Edge-Detect** filters. Running **Sobel** on a duplicate results in a transparent layer with a black outline of the image object. Having done this, it's easy to paint the underlaying layer in large clean areas, and you'll get something very similar to a picture in a comic book.

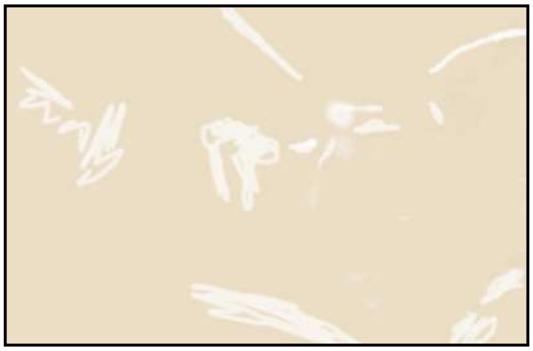


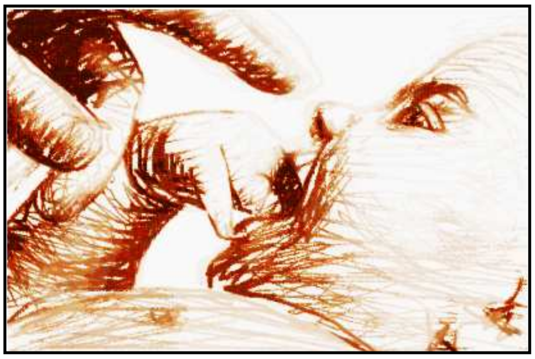


Top right: Original image
Middle right: Transparent Sobel output
Bottom right: Handpainted background







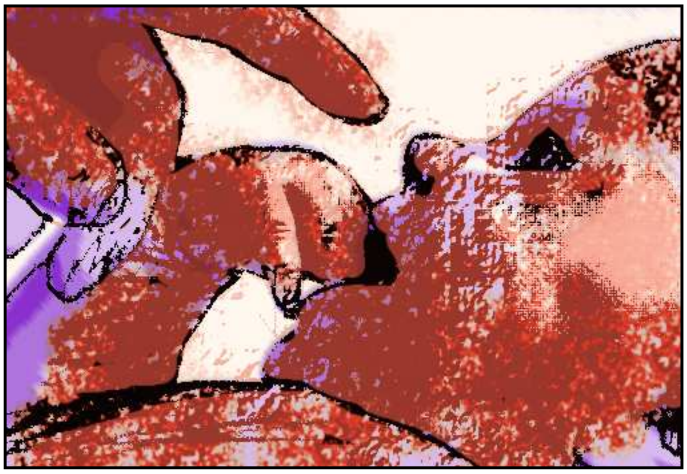


Making a pencil drawing

To make the *pencil drawing*, a white layer was placed on a b/w photo. It was set to **Screen** mode, and the opacity was temporarily reduced, so that the background would be visible. The sketch was drawn with a small, sharp pencil tip, and made to follow the contours and shapes in the photo. The photo's tonal range was limited by using the **Image/Posterize** filter, and the sketch layer was displaced slightly with a canvas structure as map. The image was flattened and adjusted with **Brightness-Contrast** to get the right gray value of a pencil drawing.

From pencil to ink

The *sepia ink drawing* was created from the previous drawing (pencil). Color was adjusted with **Hue-Saturation** and **Brightness-Contrast** to a sepia-like quality, and a beige ”sketch paper” layer was added in **Multiply** mode. The ”white crayon” in the sketch paper layer was painted with the airbrush and several soft edged brushes.

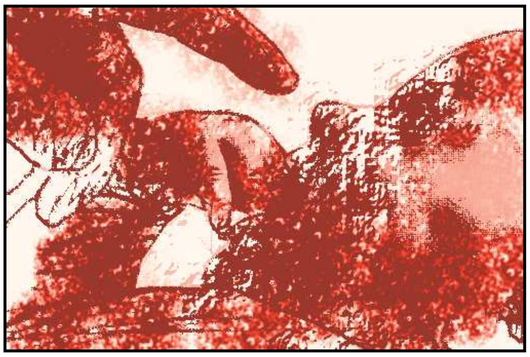


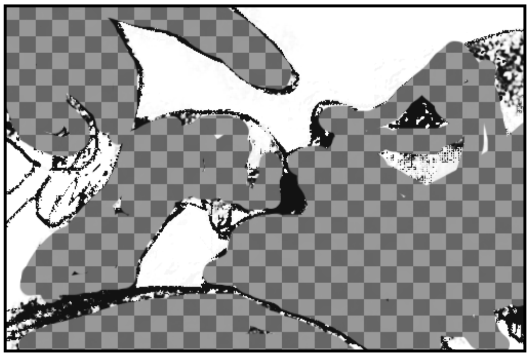
Digital crayons

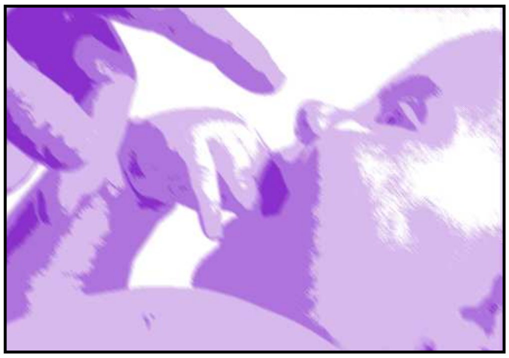
For the *crayon drawing* I used coarser, rugged brushes. I also **displaced** the sketch layer twice to get the right scratchy crayon or charcoal look. The image was flattened and the contrast was increased with **Levels**.

I made a duplicate layer and used the **Dark 1 gradient** in the Gradient editor to map to the image (**Colors/Gradient Map**). I put the old layer on top of it, set it to **Multiply** and erased everything except the contours, and parts which I wanted to keep dark (like the baby's eye and ear).

In the top layer I placed the posterized photo as **Darken Only**, changed the color to violet, and applied some motion blur. This looks somewhat like watercolor, if it's only used in small areas of a composite image.

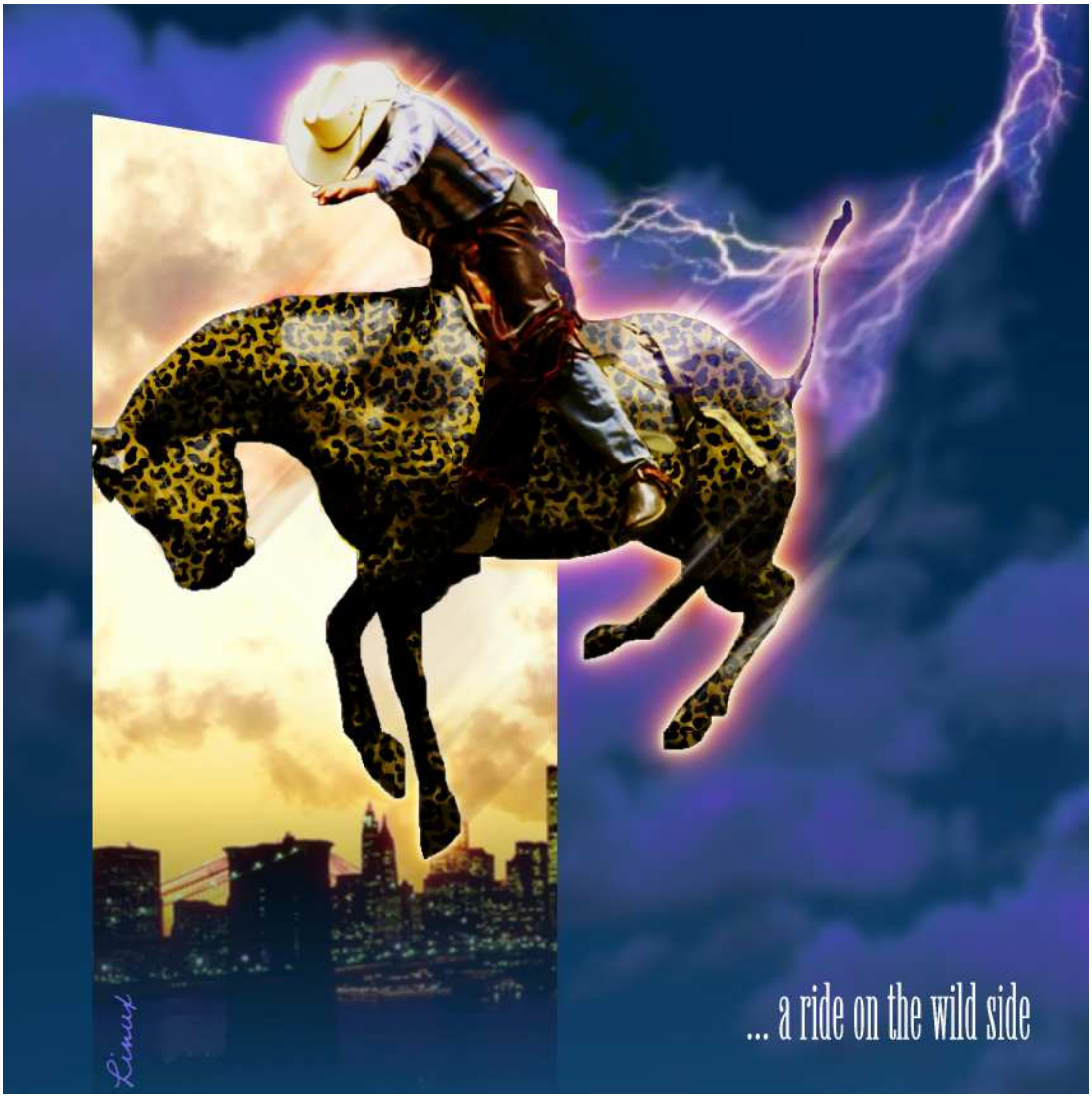


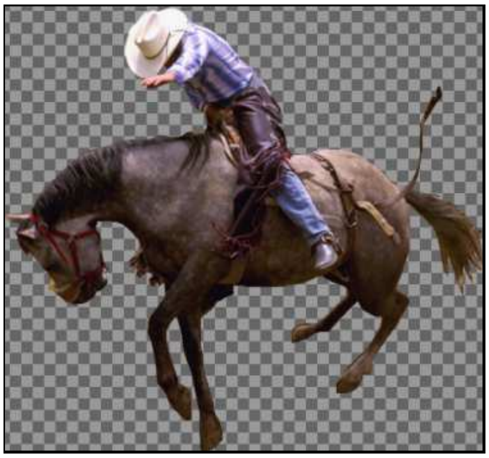


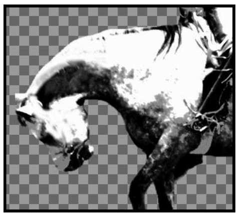


LIGHT, MOTION AND TEXTURE TRANSFORMATION

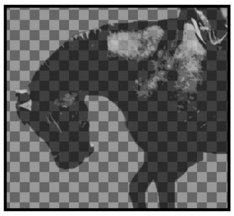
These special effects can provide that little extra which can make a good image great.


... a ride on the wild side
Linux





Shadow layer



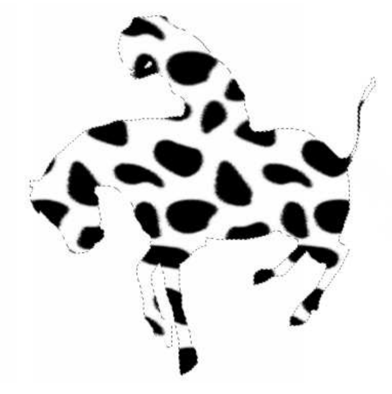
Highlight layer

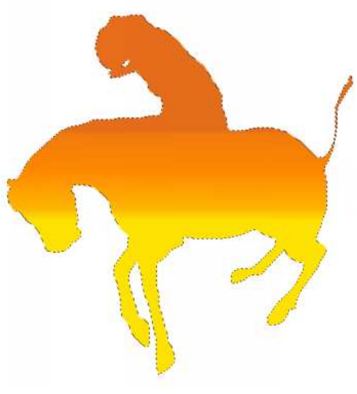
AN ELECTRIC HORSEMAN

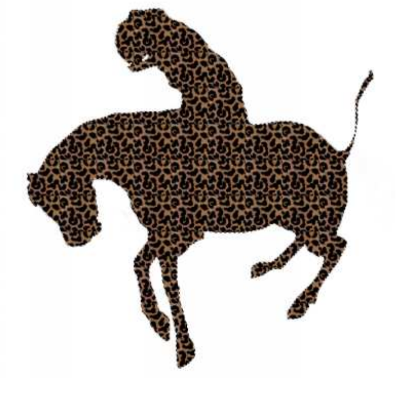
To create *the Electric Horseman*, I started out with a photo of a rodeo rider. The horse and rider were selected with the **bezier** select tool, and pasted to a transparent layer in a new image. The horse's halter, mane and tail was **cloned** away or erased, and the rider was selected separately (using a little feather) and saved as a copy in another layer. The horse and rider layer was duplicated twice, and those copies were desaturated and adjusted with **Levels** to create highlight/ shadow layers.

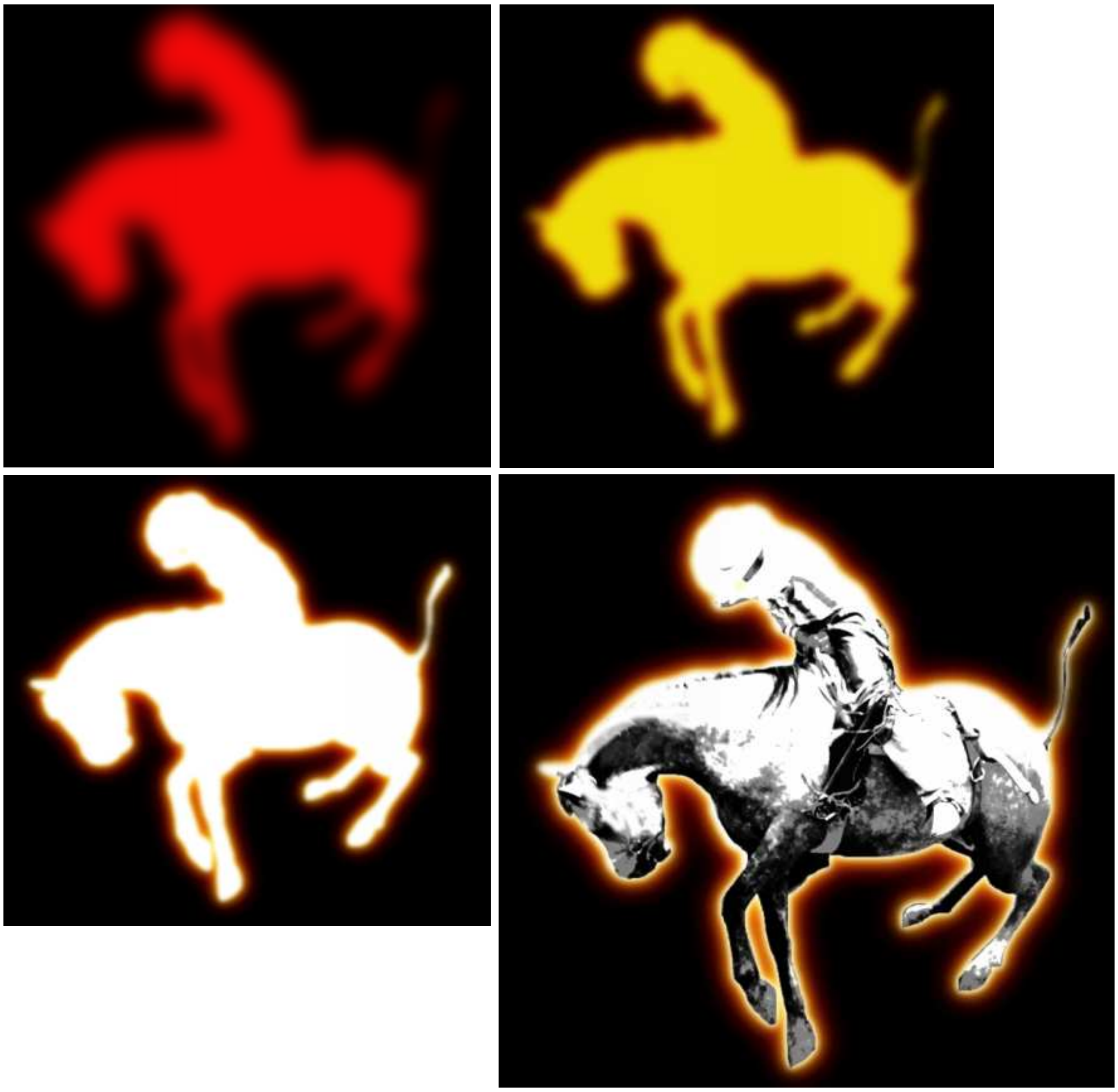
Using a Cow to make a Leopard of a Horse...

To create *the leopard skin*, two more copies were made. In the first copy, the entire horse shape was filled with the Leopard pattern from the **Patterns dialog box**. Since this pattern isn't entirely seamless, it was adjusted by painting yellow and black spots in the visible joints. The leopard layer was set in **Darken Only** mode over the second copy, which was filled with a yellow-orange gradient to add color depth to the leopard. To add some extra glow to the leopard horse, one more texture layer was added. This time I used the black and white cow pattern, and set it to **Overlay**. Note that this does not make the horse look like a cow (!). In Overlay the large dark spots on white background rather gives the illusion of powerful muscles under a shiny coat. Now I turned to the layer with the rider, and changed saturation, brightness and contrast to make it fit the new "horse"



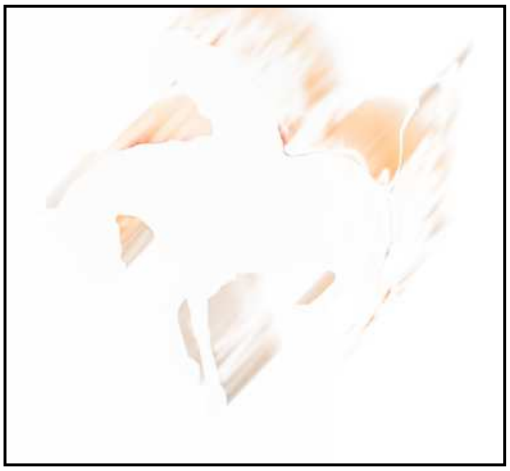




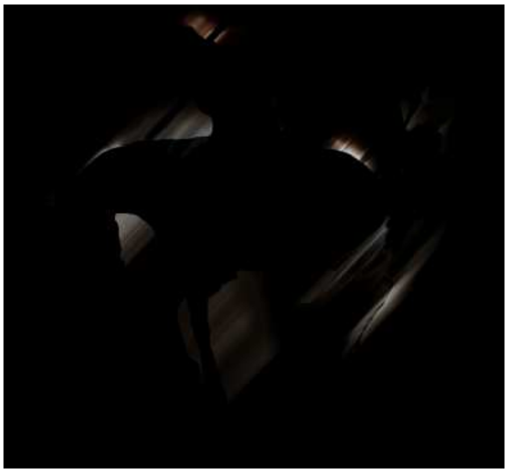


Making things glow

The glow layer was made by filling a feathered horse and rider selection with red, yellow and white, each time with lower feather values, and set the layer to Screen mode.



Dark movement layer



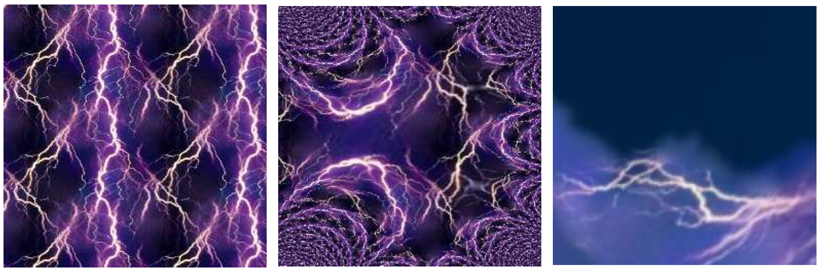
Light movement layer

Adding motion

The illusion of movement was more complicated, because just adding some motion blur wouldn't suffice to create the subtle effect I wanted. I had to create two different motion layers - one dark and one light. For the *Dark movement* layer **Blur/Motion blur** was applied to a copy of the cow glow layer, then I did the same for a copy of the orange gradient layer, but this time I inverted the selection, and only the blurred parts outside of the horse was used. These layers were merged after adjusting the opacity. The dark movement was still a bit to strong in some parts, so a **layer mask** was used to tone down or remove motion glow where it wasn't wanted. This layer was set to **Darken only**. For the *Light movement* layer I blurred the shadow layer and the leopard layer (as with the orange gradient layer before) merged it and adjusted to darker. To protect the rider figure from too much blur, a layer mask was used here as well. This layer was set to **Screen mode**.

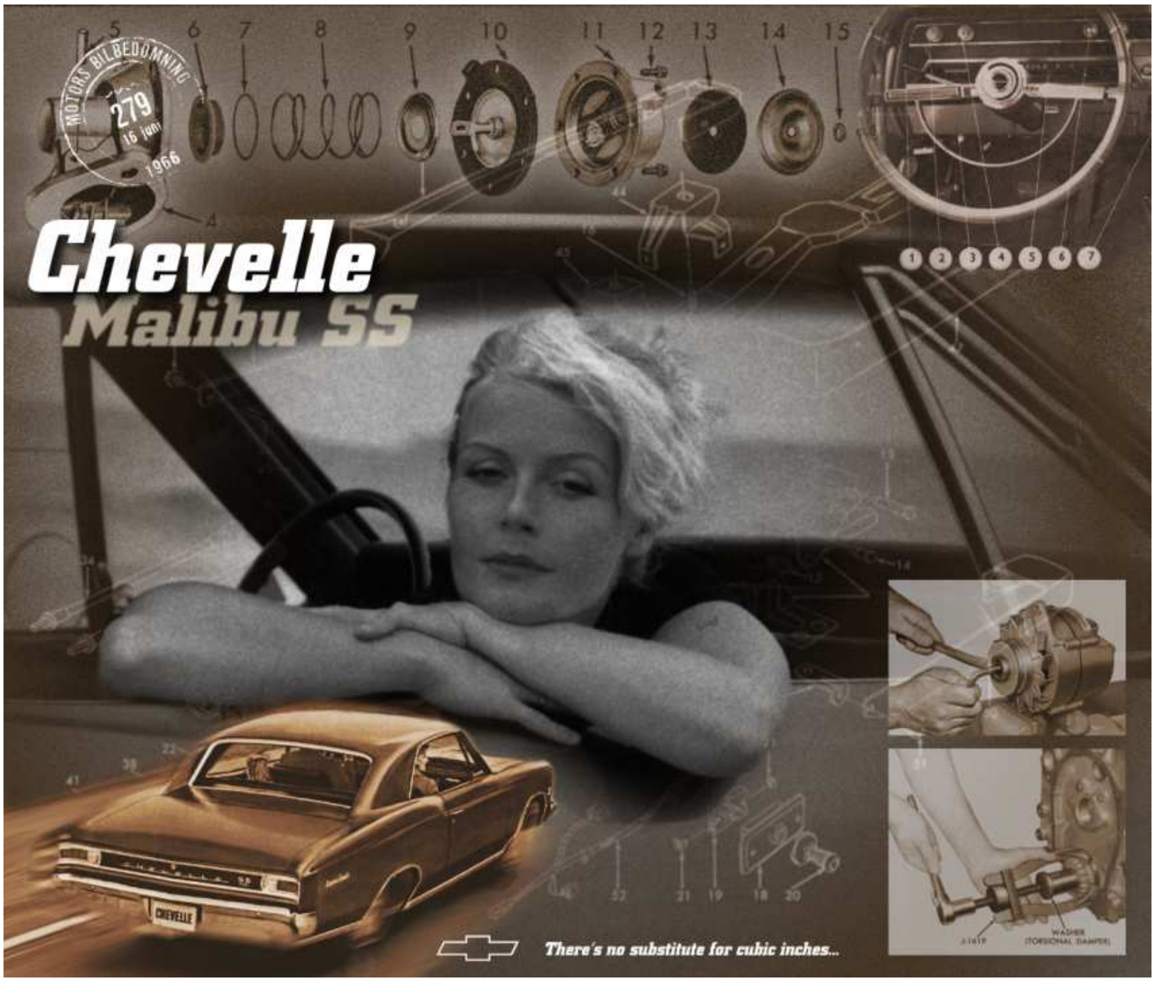
Adding scenery

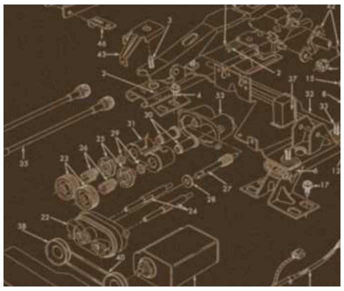
The background was made of a photo of a blue sky, a city panorama by night and a yellow evening sky. The blue sky image was transformed to dark clouds with **Image/Hue-Saturation**, and the yellow sky was merged to the city panorama. The flash of lightning was a bit harder, because the only lightning image I had was the Lightning pattern in the **Pattern dialog**, and that was too repetitive or intertwined with lightning to be used directly. The problem was solved by scaling an image with lightning pattern and then use the **Map/Fractal Trace** plugin. From that image I could feather select a suitable part, and adjust it with **Transform/Perspective** and **Screen mode** to look the way I wanted.

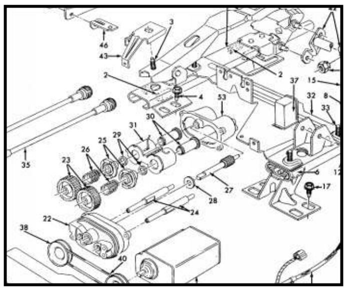


MAKING A MONTAGE

There are many ways of blending images, but for advanced montages, the most versatile method is to use different layer masks.


MOTORS BILBEDÖMNING
279
16 juni
1966
Chevelle
Malibu SS
There's no substitute for cubic inches...
CHEVELLE





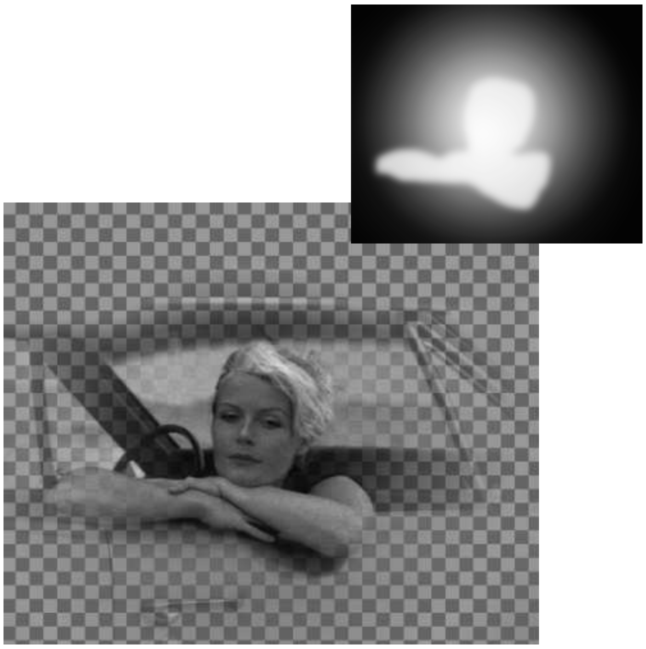
The Background

For the *Chevelle montage*, I used a drawing from the 1966 Chevrolet Chassis Service Manual for background. To make it less dominant, it was **inverted**, **blurred** and **noise** was added. Then color and brightness was changed to a soft ”old looking” sepia tone.

The Vignette

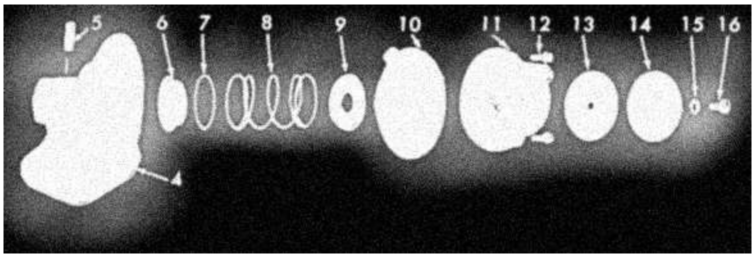
The main collage element was a holiday snapshot of me and our Chevelle. This image was blended to the background with the help of a **layer mask**. I started by making a round vignette shape with a radial gradient. This masked out soft and nice, but it also meant that the gradual transparency was evenly distributed. To put an emphasis on the person rather than the car (I like to keep it that way) some of the mask was painted white to protect face and especially the arms from transparency.

The vignette was still a little bit weak, and too smooth at the edges, so I made a duplicate where the layer mask was brightened up with **Levels**, and noise was added to the mask (not the image). This made the top layer blend well with the spotted background, as well as with the smooth copy underneath.



Adding Noise

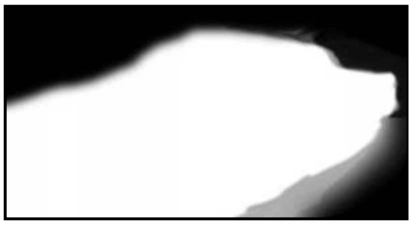
The next element in the montage was the fuel pump assembly. Here, **noise** was added to ***both mask and image***. The Glow Mask was made with a feathered lasso selection which was filled with black and heavily blurred before applying the noise. Finally, the image was tinted with the same ochre or sepia tone as the background.

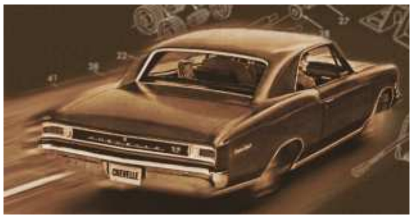


Making an element stand out in a composition

The steering wheel was treated in a similar manner, only this image was taken from an old magazine, so no noise was added (it was already quite noisy).

The picture of the car (from the 1966 Chevrolet dealer album) was desaturated and tinted. To create the illusion of speed, **linear motion blur** was applied to an inverted selection of the car, then a simple **layer mask** was added to blend the car with the background. To make the car stand out more in the composition, **contrast** and **brightness** value was increased.

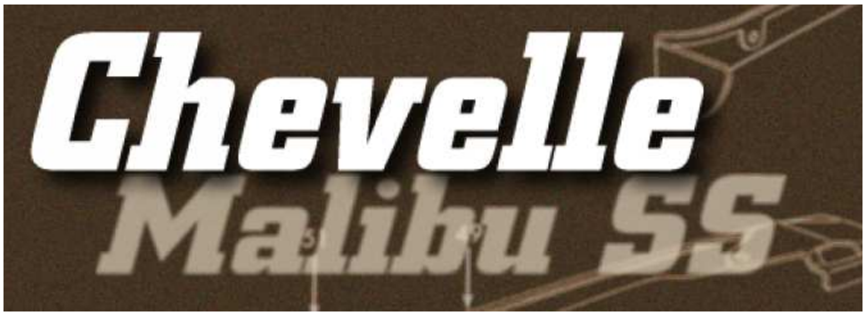


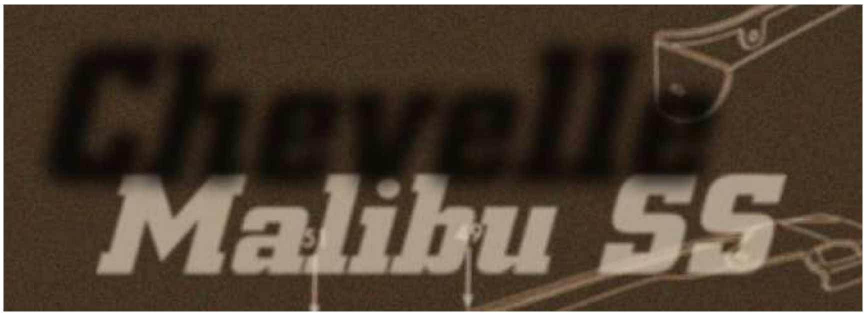


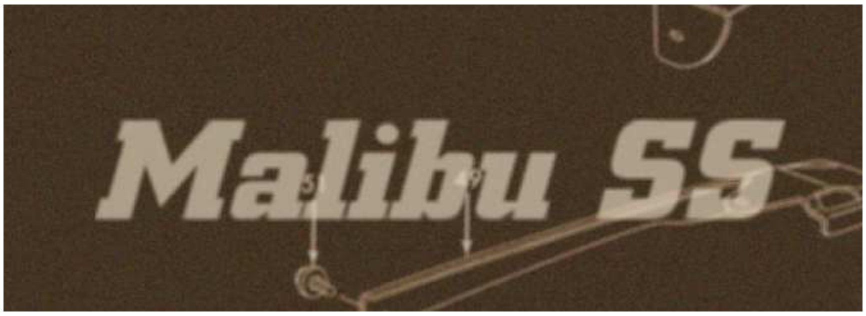
Two photographs from the service manual (removal of drive plate from generator and disassembly of crankshaft gear for the camshaft transmission) were pasted to a layer with low opacity.

Adding depth to text layers

To make the Chevelle Malibu SS text, I used three different layers. The ”Chevelle” layer is supposed to look sharp and close to the observer, while the ”Malibu” layer lies deeper, or further away from focus. This was accomplished by placing the Malibu layer very close to (almost under) the Chevelle layer, tinting and blurring it and placing a very soft shadow on top of it.


Chevelle
Malibu SS


Chevelle
Malibu SS



part II

Gimp installation

- *OBTAINING AND INSTALLING GIMP*

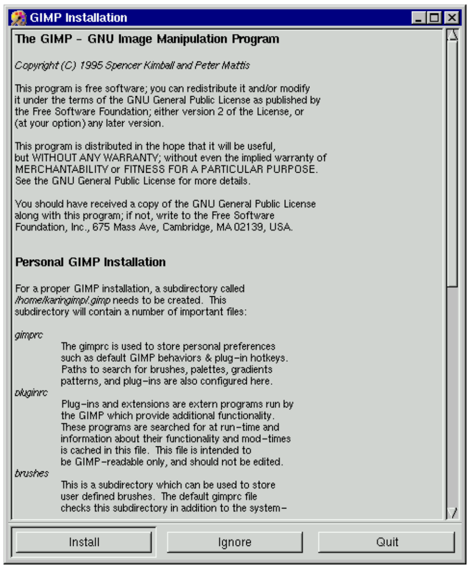
chapter 4

Obtaining and Installing Gimp

In this chapter we will tell you how to obtain a copy of GIMP and how to install it.

HOW TO INSTALL GIMP PERSONAL FILES

If you are working on a system that already has Gimp installed, like a university:



The GIMP - GNU Image Manipulation Program

Copyright (C) 1995 Spencer Kimball and Peter Mattis

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Personal GIMP Installation

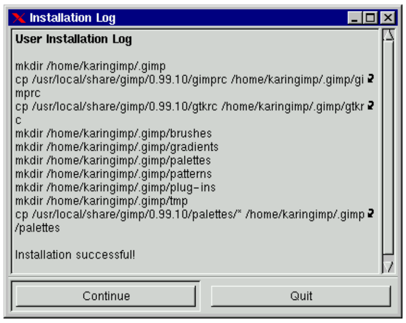
For a proper GIMP installation, a subdirectory called
/home/karingimp/.gimp needs to be created. This
subdirectory will contain a number of important files:

gimprc
The gimprc is used to store personal preferences
such as default GIMP behaviors & plug-in hotkeys.
Paths to search for brushes, palettes, gradients
patterns, and plug-ins are also configured here.

pluginrc
Plug-ins and extensions are extern programs run by
the GIMP which provide additional functionality.
These programs are searched for at run-time and
information about their functionality and mod-times
is cached in this file. This file is intended to
be GIMP-readable only, and should not be edited.

brushes
This is a subdirectory which can be used to store
user defined brushes. The default gimprc file
checks this subdirectory in addition to the system-

At the **Command prompt** type `gimp` and press `enter`. Gimp will bring up a **dialog** box (shown left) telling you that it will install some personal Gimp files in your home directory. If you want to run Gimp, press the **Install** button. The dialog box describes the Gimp **license** and what kind of **files** Gimp will install. It's advisable to read it, but we will discuss it further down in this chapter. Press `Install` now!



User Installation Log

```
mkdir /home/karingimp/.gimp
cp /usr/local/share/gimp/0.99.10/gimprc /home/karingimp/.gimp/gi ↵
mprc
cp /usr/local/share/gimp/0.99.10/gtkrc /home/karingimp/.gimp/gtkr ↵
c
mkdir /home/karingimp/.gimp/brushes
mkdir /home/karingimp/.gimp/gradients
mkdir /home/karingimp/.gimp/palettes
mkdir /home/karingimp/.gimp/patterns
mkdir /home/karingimp/.gimp/plug-ins
mkdir /home/karingimp/.gimp/tmp
cp /usr/local/share/gimp/0.99.10/palettes/* /home/karingimp/.gimp ↵
/palettes

Installation successful!
```

This will bring up a **second dialog** box telling you that all files were successfully installed.

One of the nicest things about Unix is that programs often store personal **initialization files** and **modules** in your home directory. This makes it possible to change and add features in the application you are using without having to deal with the system installation of the application. Gimp is no exception to this model. Unfortunately, most Unix programs do not have a GUI for making adjustments to these files and modules. Gimp is an exception, because it has a GUI for adjusting **user-defined functions**. However, there are some special functions (i.e new plug-in specialities), that you'll have to edit in a normal text editor like **xedit**. You will learn more about this in appendix A.

WHAT ARE ALL THESE FILES GOOD FOR?

Now let's examine the **files** and **directories** that Gimp has installed in your home directory. The first thing Gimp does is to create a directory named`.gimp` The dot means that it's a hidden directory, and you have to use `ls -a` in a terminal window to see it. In this directory Gimp will create three files: `gimprc, gtkrc` and `pluginrc`, along with subdirectories called; `brushes, gradients, plug-ins, scripts, gfig, gflare` and `tmp`. So, what are all these files good for?

- `gimprc` and `gtkrc` are your personal settings files for Gimp and GTK, respectively (**GTK** is Gimp's GUI toolkit). Most of the settings in these files are adjustable via the **Preferences dialog** box in Gimp, but some of them aren't, so you must edit those settings by hand. The Preferences dialog box is discussed in chapter 5, and the manual editing of the settings files is discussed in appendix A.
- `pluginrc` is a file that Gimp needs to store settings about plug-ins, scripts and other external programs. ***You should not edit or change this file***, although you may erase it if Gimp starts complaining about it (see Appendix A).
- The `brushes` subdirectory is where you can store your own personally created brushes. You will learn how to create and install brushes in Chapter 11. When they have been installed, and you have refreshed the **Brushes dialog box**, your personal brushes will show up in Gimp alongside the system-wide brushes.
- The `patterns` subdirectory is where you can store your own personal patterns. You will learn more about how to create and install patterns in Chapter 11. After refreshing the **Patterns dialog box**, your personally created patterns will show up in Gimp alongside the system-wide patterns.
- The `gradients` subdirectory is for storing your own personal gradients. You will learn more about how to create and install gradients in Chapter 11. Your personal gradients will show up in the **Gradient Editor** after refreshing the **Gradients dialog box**.
- The `palettes` subdirectory holds your own personal palettes as well as system palettes that you have edited. So if you want your system default palette back, you have to rename it or erase it in your personal palette directory. You will learn about creating, editing and installing palettes in chapter 11. The new palettes will show up in the **Palettes dialog** *when you quit and restart Gimp*.
- The `tmp` subdirectory stores Gimp's cache of the images you are working on. Gimp does this to support the **undo** capability, and to make it possible to edit large images without consuming too much memory. If Gimp crashes, or something else happens you may be able to find a copy of your image in this subdirectory.
- The `plug-ins` subdirectory holds any plug-ins that you have created or downloaded off the Internet. The plug-ins will show up the *next time you start up Gimp*. You will learn more about plug-ins in chapter 19 to 35, and in chapter 41 you will find a few tips on how to compile them.
- The `scripts` subdirectory holds any personal Script-Fus that you have created or downloaded off the Internet. The scripts will show up when you refresh the **Xtns/Script-Fu** menu. You will learn what Script-Fus are in chapter 37, and in chapter 38 and 39 you will find some tips and how-tos on how to make your own Script-Fus.

- The `gfig` subdirectory holds your personal Gfig drawings (created with the Gfig plug-in). You will learn about Gfig in chapter 35.
- The `gflare` subdirectory holds your personal Gflares (created with the Gflare plug-in). You will learn more about Gflare in chapter 31.

The nice thing about all this is that if you find any new plug-ins, scrips and so on, you can easily install them in your personal Gimp directories and don't have to beg your system administrator to install them on the system.

We encourage you to create your own brushes, palettes, gradients, plug-ins and Script-Fus and to share them with the whole Gimp community. Don't be shy, even small contributions are welcome. You can upload them to `ftp.gimp.org` or to the **plug-in registry** at `http://gimp.foebud.org/registry/`. May the spirit of free software be with you!

OBTAINING GIMP

When Gimp 1.0 is released it will be as a **source distribution**, but for some popular systems there may be **binary distributions**. To get the **source code** you will have to initiate an FTP session to `ftp.gimp.org`. The source code is in the directory `/pub/gimp/1.0` where **1.0.x** stands for the version *(always grab the latest version)*. In the directory `/pub/gimp/fonts` will you find some nice **free fonts** that you can use with Gimp (see Chapter 40 for instructions on how to install them). In the directory `/pub/gimp/libs` you will find some of the libraries that enable some of Gimp's optional features, such as the ability to load and save JPEG images. Also check out the contributions under `/pub/gimp/contrib` where you can find some nice palettes, gradients etc. If you are not familiar with **FTP** you can always use your Web browser to download Gimp, just type the URL `ftp://ftp.gimp.org/pub/gimp/` and take it from there. It must be stressed that the **Gimp FTP site** is often heavily trafficked, and if `ftp.gimp.org` isn't near you, we suggest that you use a mirror site (see Appendix E or `http://www.gimp.org/`). If there is a binary distribution of Gimp you'll find it under `/pub/gimp/binary.`

INSTALLING A SOURCE DISTRIBUTION

If you have downloaded a **source distribution** *(for binary distributions see the next section)* you will now have a file called `gimp-1.0.X.tar.gz.` If you want to, you can always get the extra data distribution as well as the unstable plug-in distribution (they are under than same directory as Gimp). The **data distribution** has some optional palettes, patterns, gradients and brushes that you may find useful. The **unstable plug-in** distribution contains some plug-ins which aren't considered stable enough to be released with Gimp 1.0, so remember that they may be unstable and difficult to compile *(we have used all of them with success here at Frozenriver, and they have also been documented in the manual)*

To unpack the archive, use following command: `zcat gimp-1.0.X.tar.gz | tar xvvf -` This creates a directory in your current directory (i.e `gimp`). Change to the `gimp` directory by typing `cd gimp`.

But first of all we need to have a look at which libraries Gimp needs!

If you don’t have them already, you will have to get hold of the following libraries or programs:

- **GTK** (the most recent version) to *compile* Gimp. You need this library because all of Gimp’s GUI and functions are built on top of it. GTK stands for **Gimp ToolKit**.
- **GNU GhostScript** to enable *PS (PostScript) file* viewing and editing. Type: `gs -v` to output the current version of your Ghostscript distribution. If you receive an error message, either you have not installed GhostScript or it is not in your PATH.
- **Alladin GhostScript** version 5.10 or higher to enable good *PDF (Acrobat)* file viewing and editing. Type: `gs -v` to output the current version of your Ghostscript distribution. If you receive an error message, either you have not installed GhostScript or it is not in your PATH.
- **GNU wget** to enable you to download files off the *Internet* directly into Gimp. To check whether you have wget, type: `wget` into a terminal window. If you receive an error message, either you have not installed wget, or it is not in your PATH.
- **XV** if you want to use *GUASH*. GUASH is a plug-in which lets you browse images and open them in a graphic environment.
- **Gzip** to enable extra file *compression/decompression* of any image format. To check whether you have gzip, type: `gzip -h` If you get an error message, either you have not installed gzip, or it is not in your PATH.
- **Bzip** to enable extra file *compression/decompression* of any image format. To check whether you have bzip, type: `bzip -h` If you receive an error message, either you have not installed bzip or it is not in your PATH.
- **SANE** if you want to *scan* images directly into Gimp.
- **libtiff** to enable reading and writing TIFF images
- **libz** to enable PNG compression
- **libpng** to enable reading and writing PNG images
- **libjpeg** to enable reading and writing JPEG images
- **libmpeg** to enable reading MPEG movies

To determine whether or not you have these libraries, look in `/usr/lib` or `/usr/local/lib` or contact your system administrator. Most of the programs and libraries come as standard with most Linux distributions, except for GTK, Alladin GhostScript, SANE, wget and bzip.

Now, let’s begin to build Gimp.

First of all, fire up a **new xterm** via the command: `xterm -sl 200 -sb &`

In the new xterm window, enter the command `./configure` This command will try to locate the files that Gimp needs in order to compile. Now scroll up and find out if the **configure program** could find all of the files. If not, you will have to tell the configure program where to find the missing files. If you for example couldn't locate the libtiff and libjpeg files, you can do this via the following **command-line** options:

- `--with-libtiff=<where you have your tiff library>`
- `--with-libjpeg=<where you have your jpeg library>`
- `-- disable-debug` *(If you are strictly a user and not a developer, you might want to turn off debugging).*

The command line should look something like this: `./configure --disable-debug --with-libtif=/usr/local/lib/tiff/`

Now, when you're all set (and maybe have run the `configure` program a second time to make sure you have located all the files you need), you will have to fire off the **make** command via the command `make`. ***This will build your Gimp application*** (*If you run into trouble, you will find more information on compiling in chapter 41).*

If there were no errors, Gimp built okay, and it's time to **install** it. To do that, enter the command: `make -install` which by default will install Gimp in `/usr/local/bin`, its plug-ins in `/usr/local/lib/gimp/1.0` and shared data like scripts, brushes configurations etc. `in /usr/local/share/gimp/`

Now it's time to install all of the data files that were in the **gimp-data distribution**. Do this by changing into the gimp-data directory and entering the command `./configure ; make ; make -install` which will install all the data files in `/usr/local/share/gimp`

You can of course install Gimp in different directories than the ones mentioned above. To do so, read the `INSTALL` file and use the command-line options to the configuration program that are specified there (in particular, `--prefix`). Now you can jump back to the first section in this chapter and read about how to install your **personal Gimp files**.

INSTALLING A BINARY DISTRIBUTION.

First of all, grab the latest **binary distribution** for your system and download it from the Gimp FTP site or a mirror site. If you are working on a **RedHat** (rpm) or **Debian** system, download the packages for this type of system (`.rpm` and `.deb` respectively); otherwise download an ordinary `tar.gz` archive and unpack it into the correct directory (usually /) and unpack it using the command `gzip -dc xxxx.tar.gz | tar xvf -` When this is done add Gimp to your PATH and execute it. Then, go back to the beginning of the chapter, where personal Gimp files are discussed.

WilberWorks will make a Gimp CD-rom compiled for all major UNIX systems. If you are new to UNIX, this is probably a nice option because all you have to do is to execute the `install` script. You can buy the CD from either WilberWorks or Frozenriver. For further information see appendix C about commercial support.

INSTALLING EXTRA PACKAGES TO EXTEND GIMP

To install the **extra data distribution** you only have to download it. Unpack it, run the configure script and make a `make install.`

You may also consider downloading and installing the `contrib archive` to install more brushes, gradients etc. You only have to download it, unpack it and install the contents in the right directory. You can for example install new brushes either in your personal brush directory `~/.gimp/brushes` or in the system-wide directory i.e `/usr/local/share/gimp/brushes` Some of the contribution archives come with configure scripts; if that is the case you install them as you installed Gimp. Since it is only data it's quite simple. Just type: `./configure && make install` Most of the time this will be quite sufficient.

It's a good idea to download the `freefont` and `sharefont` archives, so you can get some more fonts for Gimp. Read how to install them in the **Font Install chapter** (chapter 40).

part III

Basic functions

- ***File & Preferences***
- ***Selections***
- ***Paint***
- ***Edit***
- ***Transform***
- ***Text***
- ***Brushes & other dialogs***

chapter 5

Files and Preferences

In this chapter we will discuss how to save, open and create files, set preferences and list the file formats that Gimp supports. We will also look into how you print from Gimp.

THE FILE MENU

In Gimp, you can get to the **File** menu in two different ways: via the *Toolbox*, and via *the right mouse button* in an image window. The menus are slightly different in the two places.

In the Toolbox **File** menu, you will find the following menu items:

- *New*
- *Open*
- *About*
- *Preferences*
- *Tip of the day*
- *Dialogs*
- *Quit*

And in the Image **File** menu you will find:

- *New*
- *Open*
- *Save*
- *Save as*
- *Preferences*
- *Close*
- *Quit*
- *Mail*
- *Print*

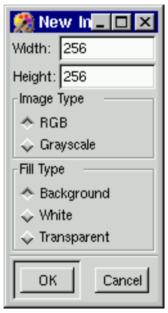
In the **Xtns** menu (in the Toolbox), you will find:

- *DB Browser*
- *Gimptcl Consolio*
- *Guash*
- *Screen Shot*
- *PDB Help*
- *Waterselect*
- *Script-Fu*

- *Web Browser*

We will discuss all of these menu items, except for **Dialogs** and **Script-Fu**, in this chapter.

CREATING IMAGES



Width: 256
Height: 256
Image Type
● RGB
○ Grayscale
Fill Type
● Background
○ White
○ Transparent

Let's start by creating our first image. Choose the **File** menu in the **Toolbox**, and select `New`. This will bring up the dialog box shown to the left:

Here you must decide the **size** of your image in pixels, whether your image should be **grayscale** or **RGB**, and whether it should have a **solid background** or be **transparent**.

As you can see, there are two types of solid background available: **Background** and **White**. *White* will produce a white background, whereas *Background* will produce a colored background based the background color in the Toolbox *(more about that in chapter 7)*. **Transparent** results in a checkerboard-like background that signifies transparency. This is a nice feature if you are creating transparent GIFs.

For now, let's stick to the default values; just press `OK`. Now you have a new image that you can start working with. Let's close the image by choosing `Close` in the **<Image> File** menu. This will close your image, as opposed to `Quit`, which will both close your image and quit Gimp.

You may wonder why you can't create an **indexed** image (that is, an image with a small, fixed number of colors) at once, for the nice GIF for the Web that you're trying to make. The reason is that it's impossible for Gimp to guess which colors you will want to use in your image. ***Also, it's usually a very bad idea to start out with an indexed image.*** You can of course still create GIFs by converting the image from RGB image to indexed color *(see chapter 12)*. The rule of thumb is: Always work with RGB, and don't convert it until you're finished with it. The same goes for every other indexed file format.

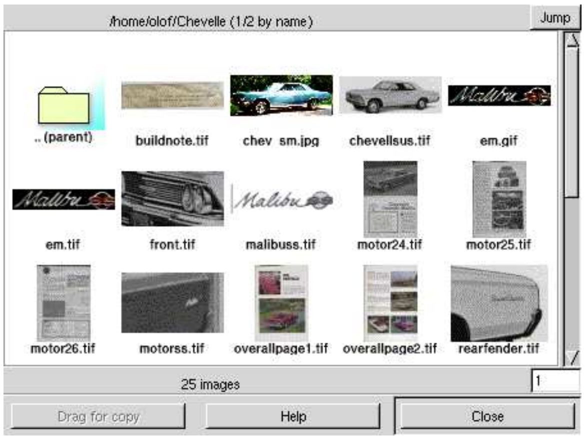
GUASH

One of the best things about Gimp is **Guash**. If you have worked on a Unix system before, you have probably used **XV** and its **Visual Schnauzer**, which enables you to load images via a graphical interface, that shows thumbnail representations of your images. If you are a former Mac and Photoshop user, and are used to the small thumbnails in an image directory, then you're going to love Guash. Guash lets you browse your home directory and see all of your images as small thumbnails.

To start up Guash, select it via `Xtns/Guash`. When you first start up Guash, it will scan your home directory for all of the images that Gimp can read. When it comes across a **PostScript** file, a dialog box will appear in which you may press `Load` to load the image, or select `Cancel` to skip that file, since loading PostScript files can take some time. Because Postscript files are sometimes many pages long, they also have the potential to take up a great deal of space in the `gimpswap file`. ***In other words: load PostScript files only if you need to, and only load the first page in any case.***

The scanning procedure can take anywhere up to five minutes depending on the number of images in the directory being scanned. To load an image into Gimp, simply `double-click` on it (one click will select it, the second will open it), and it will open in a regular **Gimp window**. As you can see, Guash only displays N directories or images at a time. To alter Guash's behavior you must edit the `gimprc` file in your `.gimp` directory:

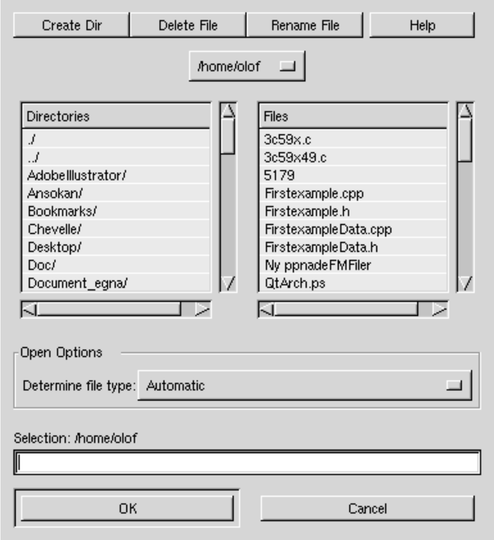
- `(guash-ncol ”5”)` changes the number of columns (valid range is 4 to 10);
- `(guash-nrow ”3”)` changes the number of rows (valid range is 2 to 10); and
- `(guash-keybindings ”emacs”)` enables Emacs key-bindings.


/home/olof/Chevelle (1/2 by name)
Jump
.. (parent)
buildnote.tif
chev sm.jpg
chevellsus.tif
em.gif
em.tif
front.tif
malibuss.tif
motor24.tif
motor25.tif
motor26.tif
motorss.tif
overallpage1.tif
overallpage2.tif
rearfender.tif
25 images
1
Drag for copy
Help
Close

If you click on a **thumbnail**, it will be **selected** and highlighted with a red frame. If you click once more on the highlighted image, it will be **loaded** into Gimp. To **select** or **deselect** further images hold down `SHIFT` and `click` on their thumbnails. If there are no selected images in Guash, pressing the right mouse button will bring up a **root menu**. If any images are selected, a **select menu** will pop up allowing you to perform various operations on the selected images. All of the items on the menus are easy to use and learn. To get the root menu when images are selected, hold down `Shift` and press the right `mouse button`.

In Guash, you can do all sorts of things like **moving**, **copying** and **deleting** images, and **creating directories**. You can even apply Unix commands, and more importantly, **Script-Fu** ”commands” to your images. This is all possible from either menu. The **Jump** menu item brings up a quick way to change directory. In the Jump menu, you have short cuts to directories you've already visited. You can also bring up a file dialog, which makes it simple to move quickly to another directory. We recommend using the Jump menu because moving around within Guash is quite slow, since Guash has to scan every directory for images.

OPENING FILES


Create Dir
Delete File
Rename File
Help
/home/olof
Directories
./
../
AdobeIllustrator/
Ansokan/
Bookmarks/
Chevelle/
Desktop/
Doc/
Document_egna/
Files
3c59x.c
3c59x49.c
5179
Firstexample.cpp
Firstexample.h
FirstexampleData.cpp
FirstexampleData.h
Ny ppnadeFMFiler
QtArch.ps
Open Options
OK
Cancel

Determine file type: Automatic

Selection: /home/olof

Now we’ll open an image using **File/Open**. This will pop up a file selector dialog box that allows you to browse filenames in order to select the image you want. In the **Determine file type** menu you can choose what kind of file you want to open, or let Gimp do it automatically. It’s a good idea to let Gimp figure out what kind of file it is. It’s only when Gimp has a hard time determining what kind of image you are loading that it’s advisable to use the non-automatic options.

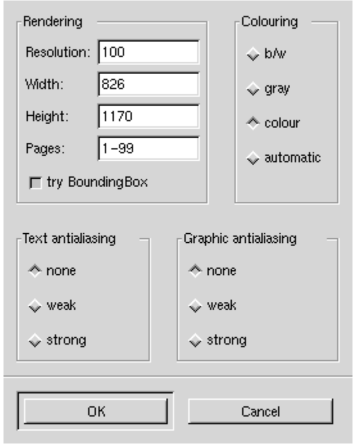
Familiarize yourself with the interface by opening some images in the usual formats like **GIF**, **JPEG** and **TIFF**. Later in this chapter we’ll discuss what kind of formats Gimp can read and write.

Tip: Gimp **expands** your file name so that you only have to type the beginning of the name and then press `Tab` to get the rest, just as you would in `bash` or `tsch`.

The Open dialog box also allows you to **delete** and **rename** files. To do this, simply select the file that you want to delete or rename and press the corresponding button. This will bring up a **confirmation** dialog box. In order to create a directory, press the **Create Dir** button, and a dialog box will prompt you for the name of the directory.

Just remember to double-click on the ”**./**” current directory symbol to update the dialog, otherwise you will not see the new directory *(this goes for files too: If you can’t see the file, double-click ”**./**” and the directory will be updated)*. The possibility to create directories in the Open or Save As dialogs is quite handy, since you often want to save an image in a new directory where you put all new images with a certain theme.

Opening Postscript and PDF Files



Resolution: 100
Width: 826
Height: 1170
Pages: 1-99
☐ try BoundingBox

○ b/w
○ gray
● colour
○ automatic

Text antialiasing: ● none ○ weak ○ strong

Graphic antialiasing: ● none ○ weak ○ strong

The Open dialog box for **Postscript** and **PDF** files is shown to the left. Our advice is to not change any values here if you're going to print it at home or in the office - the default values are fine when displaying the paper formats that are normally used for PostScript files.

If you lower the **resolution** you'll also have to reduce the **width** and **height** of the image canvas (and vice versa for increasing the resolution). If you don't change the width and height when you increase resolution, only part of your PostScript file will be displayed. If the width or height exceeds the size of your PostScript file, the display will automatically be adjusted.

If you uncheck the **Try BoundingBox** checkbox, the pages will be stacked over each other, otherwise, they will be displayed side by side. The **Pages** entry allows you to specify what page, or page interval to display; for example "1-99" specifies page 1 to 99, and "75" specifies page 75. (If your PostScript file has fewer pages than the numbers you specify, they will be adjusted automatically).

Selecting **B/W** will produce a black and white image from a color PostScript file, **Grayscale** or **Color** will produce a grayscale respectively a colored image. **Automatic** will produce whatever type of image the Postscript file was created as. *You can of course never obtain a colored image from a B/W PostScript file, even if you select Color.*

You can also specify how fine the **antialiasing** will be for text and graphics. If you only want a quick look, then select **None** or **Weak**, if you want to edit the file and produce high-quality result then select **Strong**. Naturally, there are exceptions to this rule; if you only want to alter a few minor details in the PostScript file without altering the whole thing, then it's wise to open with None; otherwise it will change the entire file when you save it.

SAVING IMAGES

Gimp naturally lets you save the images you have created. Isn't it wonderful? A ***free*** program with save abilities! It's not like when you get Photoshop for free in demo mode, where *"save and print are disabled, this is a demo copy"*.

You may access the **Save dialog** box via the right-mouse-button menu in the window that contains the image you want to save. The Save dialog is exactly the same as the Open dialog box, except that the **Determine file type menu** is a bit different: the default in this context is **By extension**. This means that the format of the resulting image depends upon the image filename's extension. For example, if you name your image `hello.gif` it will automatically be saved in GIF format. You can of course choose to save `hello.gif` in TIFF format, by selecting TIFF in the Determine File type menu, but this is highly discouraged. If you want to save a file without an extension, then you must choose the file type from the menu. Remember to **flatten** the layers in your image before you save it, unless you are using the **XCF** format (Gimp's native file format) or creating a **GIF** animation.

Gimp supports many different file formats; how many depends on which plug-ins you have installed. Yes, as with many things in Gimp, file format support is implemented via **plug-ins**. Examples of plug-ins are the **mail** and **print** plug-ins, and of course all the **filters**. We will discuss plug-ins in general later on in this manual; but for now, we will concentrate on the file format plug-ins that are included in the base Gimp distribution.

SUPPORTED FILE FORMATS

The list below lists the **file formats** that Gimp supports for **reading** and/or **writing**..

TABLE 1. File Formats

Format	Write	Read	Format	Write	Read
BMP	X	X	PCX	X	X
BZIP	X	X	PIX	X	X
CEL	X	X	PNG	X	X
FAXG3		X	PNM	X	X
FITS	X	X	PSD		X
FLI/FLC	X		PostScript	X	X
GBR	X	X	SGI	X	X
GICON	X	X	SNP		X
GIF	X	X	SunRas	X	X
GZIP	X	X	TARGA	X	X
HEADER	X		TIFF	X	X
HRZ	X	X	XCF	X	X
JPEG	X	X	XWD	X	X
MPEG		X	XPM	X	X
PAT	X	X	URL	X	X

- **XCF** is the native Gimp format. It supports ***layers*** and other Gimp-specific information. If you save your image in a different file format, all Gimp specific information will be lost, and you won't be able to open and edit your layers anymore (GIF supports layers in that each layer becomes a frame in a

GIF-animation). ***Note that only the active layer gets saved when you save an image in formats other than XCF and GIF***. So, for example, if you want to save a layered image in TIFF format, make sure you flatten it before you save it.

The following is a brief description of the different file formats. For more information on file formats, see appendix E about graphics books and links.

- **BMP**: This is an uncompressed bitmap format used by Microsoft Windows for displaying graphics. Color depth is typically 1, 4 or 8 bits, although the format does support more.
- **Bzip**: Lets you open and save bzip'ed images. The name of the file must be `<name>.<well-known suffix like tiff>.bz`.This format is ideal for saving large, multi-layered XCF images. Bzip compression is a little bit better than gzip.
- **CEL**: This is the file format used by KISS programs.
- **FaxG3**: This is the format used by faxes. It's very useful if you have a fax connected to your Unix workstation.
- **FITS**: (Flexible Image Transport System) is mainly used in astronomy. For example, it's used by NASA in their space program.
- **FLI/FLC**: This file format is used by many animation programs. Gimp reads both FLI and FLC file formats. The main difference between these two is that FLI only supports 64 colors at a resolution of 320x320 pixels, whereas FLC supports 256 colors at a resolution of 64Kx64K pixels.
- **GBR**: This is Gimp's native brush format. You will learn how to make brushes in chapter 11.
- **GIcon**: This is Gimp's native icon format, used for the icons in the Toolbox. This format only supports grayscale images.
- **GIF**: (Graphics Interchange Format) trademarked by CompuServe, with LZW compression patented by Unisys. GIF images are in 8-bit indexed color and supports transparency (but not semi-transparency). They can also be loaded in interlaced form by some programs. The GIF format also supports animations and comments. ***Use GIF for transparent Web graphics and GIF animations***.
- **Gzip**: Lets you open and save gzip'ed images. The name of the file must be `<name>.<well-known suffix like tiff>.gz.` This format is ideal for saving large, multi-layered XCF images.
- **Header**: This format is a C programming language header file. This format is for programmers who want to include their image in a C program.
- **HRZ**: This format is always 256x240 pixels and it is, or rather was, used in amateur slow-scan TV broadcasts. The format does not support compression, it's just raw RGB data.
- **JPEG**: (Joint Photographic Experts Group) This format supports compression and works at all color depths. The image compression is adjustable, but beware: too high a compression could severely damage your image, since ***JPEG compression is lossy***. Use JPEG to create TrueColor Web graphics, or if you don't want your image to take up a lot of space

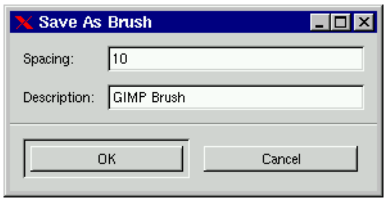
- **MPEG**: (Motion Picture Experts Group) A well known animation format. With this plug-in you can load an MPEG movie into Gimp and play it with the Animation Player plug-in. What's nice about this is that you can open an MPEG movie and save it as a GIF animation (but make sure that you remove all unnecessary frames beforehand). You can't (yet) save an animation created in Gimp as MPEG.
- **PAT**: This is Gimp's native pattern format.
- **PCX**: This is the Zsoft file format, mainly used by the Windows Paintbrush program and other PC paint programs.
- **PIX**: This is a format is used by the Alias/Wavefront program on SGI workstations. It supports only 24-bit color and 8-bit grayscale images.
- **PNG**: (Portable Network Graphics) This is the format that is supposed to replace the GIF format and solve all the related trademark/patent issues. Indexed color, grayscale, and truecolor images are supported, plus an optional alpha channel. PNG also uses compression, but unlike JPEG it doesn't lose image information. We recommend not using this format until all major Web browsers support it, or until your Web page can recognize what kind of browser is reading your page, and based on that information choose the right image to display.
- **PNM**: (Portable aNyMap) PNM supports indexed color, grayscale, and truecolor images. PNM images can be converted to many other formats with the programs that come with the `netpbm` or `pbmplus` distributions. Use this format when you know that you will want to change the image later with a PBM program
- **PSD**: This is the format used by Adobe Photoshop. Great if you are a former Photoshop user and have lots of images in PSD format. (Note that Gimp will preserve your PSD layers now.)
- **PostScript & EPS**: PostScript was created by Adobe. It is a page description language, that is mainly used by printers and other output devices. It's also an excellent way to distribute documents. This plug-in can also read PDF (Acrobat) files. (Read the installation chapter on how to make this work.) This is the format to use when you want to print your image at a professional printing works.
- **SGI**: This is the original file format used by SGI graphic applications.
- **SNP**: This format is used by MicroEyes for their animations. You can load this format and then save the resulting image as a GIF to obtain a GIF animation.
- **SunRas**: (Sun rasterfile) This format is used mostly by different Sun applications. It supports grayscale, indexed color and TrueColor.
- **Targa**: The Targa file format supports compression, as well as 8, 16, 24, 32 bits per pixel.
- **TIFF**: (Tagged Image File Format) This format was designed to be a standard. There are many variations of TIFF, considering that TIFF supports six different encoding routines, and three different image modes: black and white, grayscale and color. Uncompressed TIFF images may be 1, 4, 8, 24 bits per pixel. TIFF images compressed using the LZW algorithm may be 4, 8, 24 bits per pixel. This is a high quality file format that is perfect for images you want to import to other programs like FrameMaker or Corel Draw.

- **XCF**: This is the native Gimp format. Use this format to store all your Gimp images (if you have enough disk space).
- **XWD**: (X Window Dump) This is the format used by the screendump utility shipped with X.
- **XPM**: (X PixMap) This is the file format used by color icons in X. Gimp's XPM plug-in supports 8-bit, 16-bit, and 24-bit images.
- **URL**: (Uniform Resource Locator). With this plug-in you can download a picture off the Internet directly into Gimp. The format of the "filename" (i.e URL) that you must enter into the open dialog is `ftp://<address>/<file>` or `http://<address>/<file>` For more info see the installation chapter.

SAVE DIALOGS

When you save your newly created image, (for instance, `test.jpeg`) the **Save** or **Save as** command will bring up a dialog box asking you what kind of compression and other options you want to set for your image. We will now take look at the different dialog boxes, and discuss what impact these parameters have on the image.

GBR

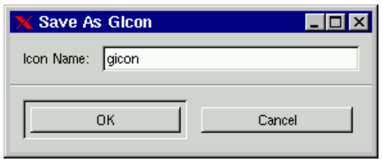


Spacing: 10

Description: GIMP Brush

This is the GBR save dialog. **Spacing** refers to the default spacing your brush will have as shown in the Brush dialog. *(You will learn more about this in chapter 11.)* **Description** refers to the name your brush will get in the brush dialog, e.g. "Olof's test brush"

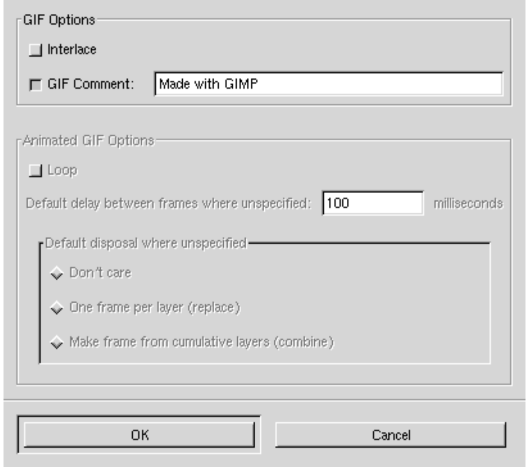
GIcon



Icon Name: gicon

This is the GIcon save dialog. Here all you need to enter is the name of your icon. Note that this is not the file name, it's the internal name of the icon as Gimp sees it.

GIF



☐ Interlace

☑ GIF Comment: Made with GIMP

☐ Loop

Default delay between frames where unspecified: 100 milliseconds

○ Don't care

○ One frame per layer (replace)

○ Make frame from cumulative layers (combine)

This is the GIF save dialog. This dialog has many options. If you flattened your image, you can save it as an **interlaced** GIF image, which enables it to be downloaded incrementally by applications like Netscape. You have probably seen this on the Web, when the image is very rough at first but gradually becomes more recognizable.

You can also set the **Comment** stored in the GIF file to anything you want, like *"Made by Karin with Gimp"*. To alter the default GIF comment, you will have to edit the GIF plug-in source code.

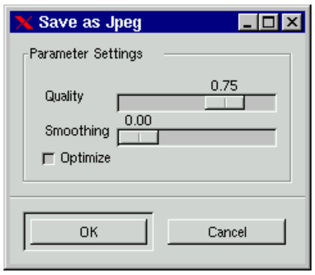
If you have layers in your image, you can create a **GIF animation**. If you only want the animation to display once, uncheck the **Loop** button, otherwise it will loop forever. The **Delay** input is the delay between frames in your animation. The checkboxes in the **Default disposal where unspecified** section do the following:

- **Don't care** and **Make frame from cumulative layers** are the same; they start the animation by showing you the first layer in the GIF image and then add the second layer on top of the existing ones. This mode is useful, for instance, when creating a logo that is going to be "written" one letter at a time.
- **One frame per layer** will use the first layer as the first frame, the second layer as second frame and so on, just like in a movie. This mode is useful when creating a spinning object, like a globe.

If your GIF image is **transparent**, the transparency will be retained when you save it. Transparency works in both flat and layered images. ***Note that GIF does not support semi-transparency***; pixels are either 100% opaque or 100% transparent.

Transparent GIF images will sometimes be displayed incorrectly in some image programs that can't deal with transparency. In that case, a color shows instead of transparency. To forestall this, set the default background color in the toolbox to an appropriate color before saving the GIF.

jPEG



Quality 0.75

Smoothing 0.00

☐ Optimize

This is the JPEG save dialog. This dialog lets you set the **quality** and **smoothing** of the image. There is also an option to optimize the image.

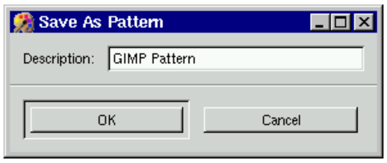
JPEG uses **lossy compression** that takes advantage of the fact that the human brain cannot easily distinguish small differences in the **Hue** part of an image *(see chapter 12)*. In other words, the smaller the number of colors in your image, the lower you can set the quality. A **Quality** setting of 0.75 is generally fine for a full-color image, but it's often as low as you can go without losing too much information in the image. If the resulting image is too inaccurate, go back to the original image and increase the quality. But never go above 0.95 - the file will only get bigger without any noticeable improvement in quality. If you aren't so concerned about the image quality, then you can go down to 0.50. If you just want low quality images, like snapshots of an image archive or some such thing, then you can go down to 0.10. or 0.20 (although personally I think XV or Guash is better suited for this kind of task).

Sometimes sharp, colorful edges can appear a bit jagged due to the nature of JPEG. If this happens, you can increase the smoothing value to more than 0.00. This will blur the edges so that they will become less jagged. But note that the image can get very blurry and that's not good.

The **optimize** check button will enable even further downsizing of the image file.

A bit of advice: don't save a TIFF image of your precious mother-in-law as a low quality JPEG, and then delete the TIFF image, because then your good mother-in-law will be gone for ever. To put it simply, don't save important images in JPEG format.

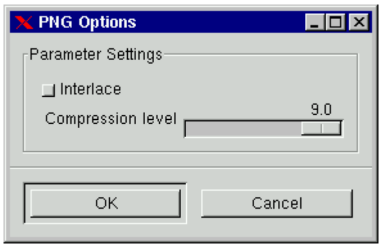
PAT



Description: GIMP Pattern

This is the PAT save dialog. This dialog lets you **name** the pattern. Note that this is name is the one that will appear in the **Pattern** dialog box. Needless to say, you should give it a meaningful name, like ”rock” for a rocky texture.

PNG



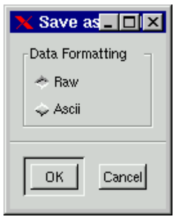
☐ Interlace
Compression level 9.0

This is the PNG save dialog. This dialog lets you set the **compression level** and whether or not the image should be **interlaced**.

The **Interlace** checkbox behaves just like the one in the GIF dialog, which means that the image can be loaded and displayed incrementally.

PNG compression is handled by `zlib`. If you slide the compression to 0, no compression will take place. If you slide it to 9, the image will be compressed by the maximum possible amount. `zlib` compression is lossless.

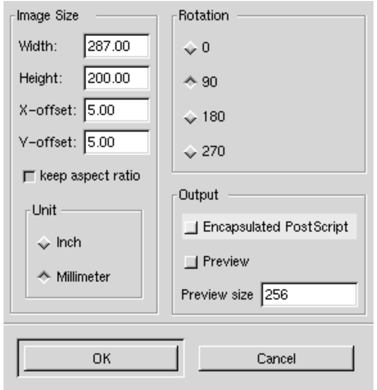
PNM



● Raw
○ Ascii

This is the PNM save dialog;.it's pretty self-explanatory. I have always saved in **Raw** format, because it's smaller (8 times smaller) and faster than the **Ascii** format.

PostScript & EPS



Width: 287.00
Height: 200.00
X-offset: 5.00
Y-offset: 5.00
☐ keep aspect ratio
○ 0
● 90
○ 180
○ 270
○ Inch
● Millimeter
☐ Encapsulated PostScript
☐ Preview
Preview size 256

This is the Postscript save dialog. Note that when you save your image as **PostScript** it's significantly different from saving it as TIFF, GIF, JPEG, etc.

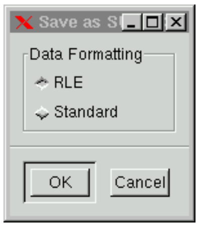
When you save an image as PostScript you must specify the **image size**. You can think of this as specifying the size of the paper that the image is going to be printed on. Then, as you might expect, an image that is 300x400 pixels won't look too good if it's going to be stretched out to 11.30x7.87 inches, which is the default. You have to alter the paper size depending on your image size. To do so, change the **Width** and **Height**. The **offset** is the margin of your imaginary paper. Turning on the **Keep aspect ratio** checkbox means the image size is automatically adjusted to be the same relative shape as your image, so that the image won't be stretched out.

Let's say you are going to save the 300x400-pixel image, and that you decide that the resolution is going to be 100 pixels per inch (suitable for printing to a 300 dpi laser printer). Then the image area will be 3x4 inches, plus 2x0.2 inches at top and bottom and 0.2 inches on each side, which gives you a total image size of 3.4 x4.4 inches with a margin of 0.2 inches. Let's set **Width** to 3, **Height**=4, **X-offset** to 0.20, **Y-offset** to 0.20, **Unit** to inches and **Rotation** to 0. This will produce a pleasing result.

If you want to rotate your image by 90^o then you have to switch *Width* and *Height*, and the image will be saved in **landscape mode**. If you rotate your image by 180^o you'll get an upside-down portrait mode, and if you rotate by 270^o, you'll get an upside-down landscape mode. Read more about this in the prepress chapter.

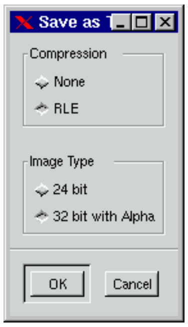
Turning on the **Encapsulated PostScript** checkbox will create an Encapsulated PostScript `(.eps)` file instead of a plain PostScript `(.ps)` file. EPS is useful for **importing** an image into a page layout program or an illustration program. Encapsulated PostScript is also a commonly requested file format when you print at a professional print house. If you're going to import your EPS file into another program, it is wise to check the **Preview** button; if you don't, it's quite possible that the program won't be show what your image looks like on screen, (it will be either be displayed as a gray square, or in very low resolution) and it'll only appear when you print it out.

SunRas



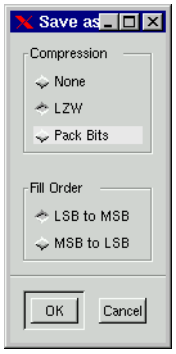
This is the SunRas save dialog. When you save a SunRas file, you'll get a dialog box asking if you want **RLE** compression or **standard** format (no compression). The RLE compression is lossless so it's generally a good idea to select this option, but the RLE format only supports only 4 or 8 bits per pixel.

TGA



This is the Targa save dialog. When you save a Targa image, you get a dialog box asking if you want RLE compression or no compression. As with the SunRas compression, RLE is generally a good idea, as it will cut down on disk space. You may also choose whether you want to save your image with 24 bits per pixel or 32 bits per pixel, which includes an Alpha channel.

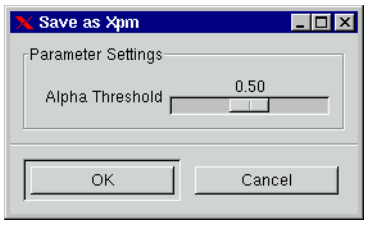
TIFF



This is the TIFF save dialog. This dialog asks what kind of **compression** you want. Since TIFF compression is **lossless** there's no danger in using it. Use **LZW** with 4, 8, and 24 bit images and **Pack Bits** with 1 bit (bitmap) images like FaxG3 images.

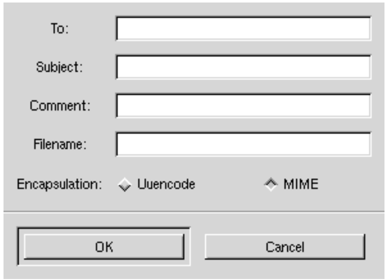
The **fill order** determines how the bits in your image are filled. The first option is Least Significant Bit (**LSB**) to Most Significant Bit (**MSB**). This is the default order on little-endian machines with e.g Intel X86 processors, this is also called **PC filling** in e.g PhotoShop. The other option is the other way around; **MSB** to **LSB**. This is the default order on big-endian machines with e.g Motorola and Sparc processors. This is also known as **Mac filling**. If you want to import the image into Frame Maker or some other page setting program, set Compression to **None** and Fill Order to the right type, depending on the machine the program is running on. Then you are on the safe side.

Xpm



This is the XPM save dialog This dialog is lets you set the **alpha threshold** in order to reduce saving time. This will determine what alpha value to set as transparent and which to set as opaque, e.g threshold=0.5 means that all alpha values under 128 will be transparent and all over 128 will be opaque. According to the source code, 16-bit and 24-bit XPM images are supported on 16-bit displays, and 8-bit images are coded but not tested.

MAILING IMAGES



Gimp can **mail** images. This is a handy feature, particularly if you work in more than one location. Then you can mail your image home or to work. You can also use it to send digital Christmas cards to your friends and family. Note, however that the mail administrator for your friend's network may not be very happy if you mail large images, so use it with care. Now, let's check out how to do it. In the <Image> **File** menu, select **Mail image**. This will pop up the mail dialog box. The **To** field is the e-mail address of your friend. **Subject** is the subject of your email. **Comment** is a comment for the image, like ”Here's a nice picture of my new pet.” **Filename** is the filename of the image that you want to send; it should have the format `<name>.<well-known suffix>`. You also have to choose the **coding type** of the mail; either Unencoded or 64bit-Mime.

PRINTING IMAGES

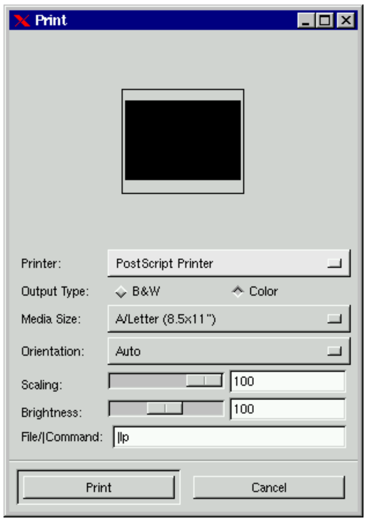
Before you start to print from Gimp, be warned that only the **active layer** of layered images is printed, so you must **flatten** your image before printing. In addition, we're not going to discuss UNIX printing systems, spool systems and so on, because it's beyond the scope of this manual. If you want more information on setting up printers and printing systems in Linux/UNIX, see appendix E for relevant links and books.

What kind of printer does Gimp support internally? To find out, open an image and select the **Print** menu item in the <Image> **File** menu, this will bring up the **Print dialog box**. As you can see in the Printer menu, Gimp supports the following printers:

SUPPORTED PRINTERS

- PostScript printers and PostScript level 2
- HP DeskJet 500, 500C, 520, 540C, 600C, 660C, 68xC, 69xC, 850C, 855C, 855Cse, 855Cxi, 870Cse, 870Cxi, 1200C, and 1600C printers.
- HP LaserJet II, III, IIIp, IIIsi, 4, 4L, 4P, 4V, 4Si, 5, 5FS, 5L, 5P, 5SE, 5Si, 6L, 6P printers, and
- EPSON Stylus Color, Color Pro, Color Pro XL, Color 400, Color 500, Color 600, Color 800, Color 1500, Color 1520 and Color 3000 printers.

SETTINGS



Printer: PostScript Printer
Output Type: ○ B&W ○ Color
Media Size: A/Letter (8.5x11")
Orientation: Auto
Scaling: 100
Brightness: 100
File/|Command: |lp

Gimp supports the most common printers internally, so you don't have to have **GhostScript** installed. If you have GhostScript installed and use it as a printer filter for your printers, then you probably want to print with the **PostScript Printer** option. However, it is recommended that those of you with Ghostscript installed should create a raw printer device, and let Gimp print directly to it, at least for testing purposes, so you can follow the examples in this manual.

The **Output type** selects what type of printer you have. If you have a color printer, use **Color mode**, otherwise use **B&W**.

The **Media Size** is the size of paper you have in your printer. Gimp supports Letter, Legal, Tabloid, A4 and A3 paper sizes, although your printer may only support one of them.

The **Orientation** menu, lets you select whether you want to print in Landscape or Portrait mode, or let Gimp decide in **Auto mode**.

When you open the Print dialog box, S**caling** is set to 100 and the image is stretched so that it will cover the entire page (with a margin of 6mm on each side, and 13mm on top and bottom). *Note that as a result a small image will get so enlarged that it will look very ugly, because the resolution will be extremely low).* There is a lot that could be discussed with regard to printing images, but for now let's stick to what we have to do to get our image to look reasonably good. The key factor is **resolution**. If you have a small image and you enlarge it, it will get a very low resolution. In order to get a nice printout, the image will have to be relatively large; and when you print it, you will have to scale it down to increase the resolution. Read more about this in the chapter about prepress. An image that is 300x400 pixels will look good with a scaling around 30 (on A4 paper and a 300 dpi printer). The procedure is the same as when we saved in PostScript format.

The **Brightness** is by default 100, which is fine for printing to lasers and black and white inkjets. Usually though, you will have to increase this value when printing to a color inkjet. The author of the plug-in says 125 to 150, but for us, that's too bright. We use 110 for our Deskjet 870Cxi. The bottom line is that you will have to experiment and find out for yourself what setting gives the best results for your particular printer.

The **Print menu** is where you set what printer you want to print to *(not the type, but the name of the printer queue)* or if you want to print to file. In the **File/Command** field the printer command or file name will appear depending on your settings in the print menu. If you want **print to file**, just use `<directory>/<filename>` for example `/tmp/hello.ps`. If you print to file, the raw data that otherwise would be sent to the printer is instead sent to a file. ***A warning is in place; if you print to a file that already exists; It will be overwritten without any warnings.***

GIMP PREFERENCES

In the **Preferences** dialog box, you'll find a lot of Gimp-related settings. There are four tabbed folders named **Display**, **Interface**, **Environment** and **Directories**. To save your changes permanently, press the **Save** button. If you only press **OK**, your changes will not show up next time you start up Gimp, and if you press **Cancel** all your changes will be discarded. ***To enable your modifications you must first quit Gimp and then start it up again.***

DISPLAY

In the **Display** folder you can set the default size for new images (in pixels), and the default image type (RGB or Grayscale). You can also set the **preview** size. If you have a small screen and low computer resources you'll generally want to keep the preview size as small as possible. As for **interpolation**; Gimp uses **Linear interpolation** by default. When you scale an image (make it bigger), linear is faster and also leaner on system resources, but it does not produce as high a level of detail in the output image as **Cubic interpolation** does. To get a nice-looking image with a lot of detail when you scale it, check the Cubic interpolation check box. If you are on a machine with low resources use linear interpolation by ensuring that the Cubic interpolation check box is not checked. **Check size** lets you choose the size of the checks on the transparency checkerboard representation, and **Transparency Type** lets you specify the grayscale tone of the checks. We use the default values here.

INTERFACE

In the Interface folder you can set the Levels of **Undo**. A high number of levels of undo requires a lot of disk space, so use it with care if you are low on such resources.

Resize window on zoom is the same as the **Allow Window Resizing** in the **Magnify Options** dialog box. Our advice is to keep this option unchecked since you can easily enable and disable it in the Zoom Options dialog box. Read more about this in the chapter about **Transform tools**.

Auto save is a nice feature; Gimp will automatically save the image in its `tmp dir`, so that if it crashes, you may be able to recover the image by looking in that directory.

Disable cursor update: By default Gimp changes the cursor symbol; for example, if you use the Move tool, your cursor will be the Move symbol. However, this consumes system resources, so if you are extremely low on resources, check the Disable Cursor Update checkbox. Bear in mind that this can be unwise since you won't be able to see what "mode" you are in. You can also set the speed of the marching ants (the blinking dots that appear whenever you make a selection): the value in the input field is the amount of time between updates (in milliseconds), so a lower value makes the ants march faster.

ENVIRONMENT

In the Environment folder you will find the **Conservative memory usage** option. Gimp will usually trade **memory** consumption for **speed** improvement. Checking Conservative Memory Usage will make Gimp more concerned about memory, but it will also slow it down, so only use it when memory is scarce.

Tile cache size is the amount of memory (RAM) that Gimp consumes in order to ensure that it doesn't damage the tile memory when Gimp moves memory to and from disk swap. The memory value is measured in bytes. A higher value will make Gimp go faster and a lower value will make it go slower, but the difference will only be evident when you work with really large images. We stick to the default value and that may work well for you too, but if you have low resources it may be wise to lower the memory value (note that Gimp will work even if you set it to 0). If you're working on an 8-bit display (in other words, you can only display 256 colors at once) you'll definitely want to use the **Install color map** option; otherwise Gimp will be rather useless. Since most of the colors will have already been claimed by other applications like the window manager, Netscape etc., Gimp and the images that it displays will only be allowed to use the leftover colors (unless you install colormap), and the result will of course be horrible.

Colormap cycling affects the "marching ants" or selection border. If you check this option, the "ants" will be replaced with a smooth line. This line is dark as you drag the selection, and turns bright when you release the mouse button.

DIRECTORIES

The most important directories in the **Directories** folder are **Temp** and **Swap**. The Temp directory is where Gimp stores all of its temporary data, like working palettes and images. The Swap directory is where Gimp keeps its swap file for the tile-based memory system.

Suggestions

Here are some suggestions on how to configure these directories for optimal performance. If you are on a system that mounts home directories from a **server** over **NFS** and/or you have only a fixed **quota** of disk space that you can use, you will probably want your **Swap** directories on a local temporary directory like `/tmp`. Otherwise you won't have any disk space left to save your images, since the swap file can get (read almost always gets) nasty big. Another reason to do this is that if you have to send image data to and from a swap file over the net, Gimp will be terribly slow because it has to wait for the net traffic to finish before it can proceed.

Temp is a bit different. Most of the files here will disappear when you exit Gimp, but some will remain (such as working palettes for example) so you will probably want a Temp directory that you don't share with other people, and that isn't cleared if you reboot your workstation. A good idea is to make a personal directory under `/usr/tmp` or `/var/tmp` e.g `mkdir /usr/tmp/your_user_name` and set this directory as your temp dir. The other directories for brushes, gradients, palettes, patterns and plug-ins may be left as they are, or modified to suit your special needs; but if you change them, remember to move your gimp add-ons to the right place, because if you don't, you may not have any plug-ins available the next time you start up Gimp.

MISC. FEATURES & EXTENSIONS

TIP OF THE DAY

Tip of the day is a handy feature for new Gimp users. You can browse the tips by clicking **Prev.Tip** and **Next.Tip**. To turn off Tip of the day uncheck the **Show tip next time** checkbox. The file which contains all of the Tips of the Day is in the Gimp system-wide data directory and is called `gimp_tips.txt`, so if you are a system administrator you can edit this file and add new tips for the Gimp users on your system.

DB BROWSER

In the DB browser you can search for Gimp **PDB calls** and **routines**. This is mainly of use to people who deal with **scripts** and **plug-ins**. Even if you are not writing scripts or plug-ins, this can be a source of information about Gimp's internals. You can search the database for both PDB procedure **names** and **information** (blurb). If you call the DB Browser from the Script-Fu Console then you can also apply the PDB command to the console.

PDB HELP

This is another interface to the Gimp PDB. With this extension you can run a specific PDB call on your image. To do this, simply press **Run** which will bring up a dialog asking for information. As with the DB Browser, this extension is mainly for developers.

GIMPTCL CONSOLIO

Gimp's main scripting language is **Scheme**; one of many scripting languages available for **Unix**. This consolio is for the scripting language **Tcl**. If you are familiar with Tcl you probably don't want to learn Scheme. This consolio is written in Tcl and you can use it when you create Tcl scripting for Gimp. As a user with no technical background, you may well ask yourself why you should bother learning some obscure Unix scripting language? Let us first state that there is no need to learn a scripting language to use Gimp for your designs. You can happily use Gimp forever without a thought of learning any such language. But if you find yourself performing some operation over and over again, wouldn't it be nice to automate it? Take the *drop shadow* Script Fu for example. You can make a drop shadow by hand in Gimp, but isn't it nice that you have a script to do it for you? This is when it becomes interesting for ordinary users to learn a scripting language. At the time of writing, Gimp supports **Scheme**, **Tcl** and **Perl** (although Perl is still in early development stages). There are several books about each of these languages, so you can choose any one of them. If you are totally new to computer languages and scripting, then we suggest that you learn Scheme, because it's Gimp's native scripting language.

SCREEN SHOT

With this utility you can take **screen dumps** directly into Gimp. You have three alternatives; the active window that you select with or without the **window decoration** (the frame that e.g fvwm creates around your window). You can also take a dump of the whole screen. In order to take a dump of the active window, check the appropriate options and press OK. You will now see a cross symbol that you can use to select the window that you want to dump (i.e click in the window that you want to dump). When you want to take a dump of the entire screen, check the right alternatives and press OK. After a little while a new Gimp window with your screen dump will appear.

WATERSELECT

Waterselect is an alternative to the Color Select dialog, when you want to select a special color. This tool resembles the little water cup you use for blending colors when you're making a watercolor painting. The interface is simple and easy to use. At the top you'll find the last ten colors that you mixed. Just press one of these colors, and it will be activated. If you want to mix a color, just move the mouse over the color scale while pressing the left mouse button. If the color gets too dark, release the mouse button and press the right mouse button while moving the mouse. The color will now get lighter, just as if you had added more water. Press OK when you're satisfied, and the chosen color will appear in the active color swatch in the Toolbox.

WEBBROWSER

The **webbrowser** allows you to start up or open a web site from Gimp. You'll find a few short cuts to important locations like Gimp.org, Plugin registry, Gimp news, etc. You can also open the on-line version of the Gimp User Manual this way. To open a site that is not available in the short cut menu, select **open url**. This option pops up a dialog box asking you about the site. You can also choose whether you want to open the site in an existing Netscape session, or whether you want to open a new Netscape window.

Naturally, Netscape must be installed and in your `PATH`. If you want to change or add a short cut, you have to edit the script which controls the behavior of this extension. It's called `web-browser.scm` and is usually located in `/usr/local/share/gimp/scripts/` Just copy it to your personal gimp script directory `cp /usr/local/share/gimp/scripts/web-browser.scm /your/home/.gimp/scripts/` and edit it in a text editor.

chapter 6

Selection tools

In this chapter, we'll learn about making selections, and how to use the different selection tools. We'll also explain why strange things happen when you try to move your selection for the first time.

THE BASIC CONTROLS

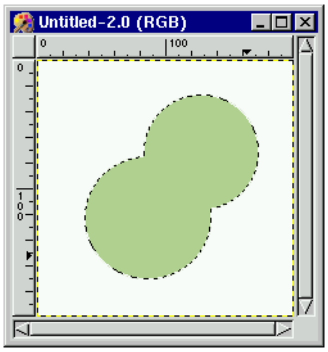
The first six tools in the toolbox are **selection tools**. In order to manipulate a specific part of your image, you first need to **select** that area. The trick is to find the right selection tool, or the right combination of tools to make your selection correspond as exact as possible to the object you want to work with. A sloppy selection is quickly discerned by the eye, and your work will not look very convincing. The selection tools in the toolbox are used for quick, simple selection. For really advanced selection work, read the chapters about **Channels** and the **Select Menu**.

TOGGLE

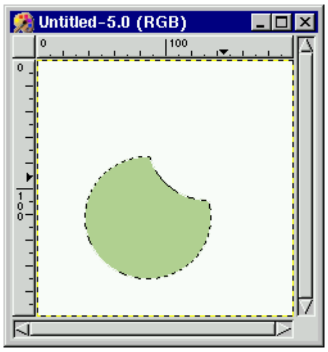
When you have made a **selection**, the **boundary** of the selection appears with a blinking dotted outline, sometimes referred to as *marching ants*. Your selection is now the only active part of your image; the rest is masked and is not affected by your operations. If you find the blinking line distracting, Gimp allows you to switch it on and off with **toggle** in the **<image>/Select** menu. The yellow-dotted **layer** boundaries are also affected by the toggle command.

SELECTION CONTROL

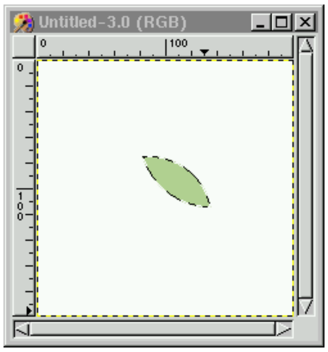
If you're not happy with your selection, just make a new one; the first will instantly disappear and be replaced by your new selection. If you regret making selections at all, just click once in your image, (with the rectangular, ellipse or lasso tool active) and the selection will be gone. However, you might want to make more than one selection, or combine several selections into one - then you'll have to use the `Shift` and `Ctrl` keys.



Pressing the `Shift` key as you drag your selection allows you to **add** a selection. If the selections touch, they will be welded into one single selection (**union**).



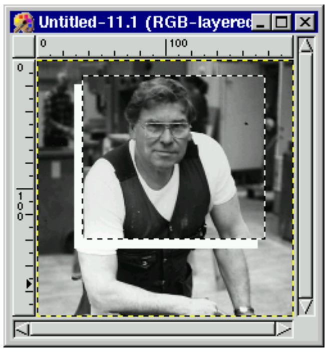
Pressing the `Ctrl` key will do the opposite; dragging a second selection which touches the first will **subtract** that form from the first selection (**difference**). (Just pressing `Ctrl` while drawing selections won't do anything, the selections have to cross each other).



If you press both the `Ctrl` and `Shift` keys simultaneously, you'll get the **intersection** of the two, i.e. only the portion which is part of both selections.

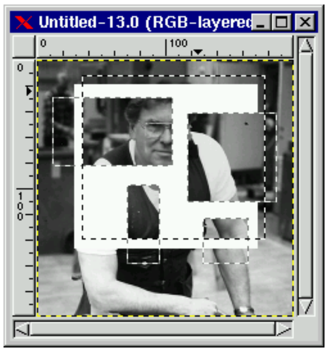
While making these **additions** or **subtractions**, remember *errare humanum est*. It may be wise to keep your middle finger prepared to press the **right mouse button**. If you do this and then *release* the **left mouse button**, the selection you are drawing will disappear. This possibility to **Undo** a selection before it's finished is useful when you add or subtract or intersect. When you decide you don't like the selection you're drawing, you can undo that selection without also erasing the other "good" selections. You can of course do this afterwards with `Ctrl-Z`, it's just a bit quicker to do it with the mouse.

MOVING SELECTIONS



Moving selections in Gimp is not altogether intuitive, and you may well get a bit confused at this. When you have made a selection in Gimp, the cursor will automatically turn into a **move symbol** (crossing arrows), and you can move your selection with its contents (although the selection tool is still active and not the Move tool). The thing is, this makes your selection float! If you don't know what floating selections are - read about it in chapter 18.

ONCE MORE



When you try to do this a second time, you'll just form a new selection inside the old one. This isn't really a selection yet - the dotted outline is gray instead of black, and it doesn't blink. Note that this doesn't happen when you place a text selection, because then no select tool is active.

Mask

This gray **subselection** won't turn into an active selection until you save or delete your float. But you can use it as a **mask** effect. What happens is that your selection turns white (if that's your background color) - it's only inside the boundaries of the gray-lined marquee(s) (you can make lots of them with `Shift`) that you can see the original contents of your selection. This is a nifty little effect, but it can be rather annoying if it happened by mistake (not uncommon) Don't panic - just press `Ctrl-Z` (Undo) and you'll be back where you were. Read about the **Move Tool** for more information about moving selections.

GUIDES

To place your selections exactly where you want them, use the **horizontal** and **vertical guides** which can be drawn straight from the left or upper **ruler**. To change the position of the guides, you must use the **Move tool** (notice how the move symbol changes into a pointing hand when it touches a guide). **Snap to guides** is the default set in the **View menu**. If this option is checked, moving any kind of selection close enough to the guides, automatically causes it to ”stick” or snap to it. You can for example decide exactly from which point of origin you want your square or ellipse selection to start. If you use the `Ctrl` key, and start dragging close enough to the point where the guides cross, that will be the centre of the new selection. Without `Ctrl`, the selection will start from the cross and continue in the direction you drag. You can easily adapt the size, shape and position of a rectangular or ellipse selection to the guides by drawing it within the frame of two vertical and two horizontal guides.

RECTANGULAR AND ELLIPSE SELECTION TOOLS

It’s pretty obvious what these selections look like. If you click the mouse and drag, you’ll get normal **rectangular/ellipse** selections, starting from the corner where you first pressed the mouse button. If you want to create **circles/squares**, or make your selection **spread from the centre**, you must use the `Ctrl` and `Shift` keys. ***Note, to make this work: First press the mouse button, then hold the key and drag.***

SHORT CUTS

- The `Shift-key` constricts the selections to perfect **squares** and **circles**. The selection starts from the **corner**, and continues in the drag direction.
- The `Ctrl-key` draws normal **rectangular** and **ellipse** selections, but with this key, selections will emanate **radially** from the point where you start dragging. This point is now the **centre** of your selection.
- Using both `Shift` and `Ctrl` results in circles or squares (as with `Shift`), but they grow from the centre and out (as with `Ctrl`).

Now, if you want to use `Shift` and `Ctrl` for operations like adding, subtracting or intersecting, and at the same time use `Shift` and `Ctrl` for the operations mentioned above, it gets a bit complicated - but not impossible!

What you have to do is this. First you must decide what the selection will be used for:

• If you want to make **many selections**, or **add** to an existing selection	Use `Shift`
• If you want to **subtract** a selection from another selection	Use `Ctrl`
• If you want to make an **intersection** of two selections	Use `Shift` and `Ctrl`

When you have decided, *hold that key and then press the mouse button. Then release the key but not the mouse button.* With this you have told Gimp what action you want the selection to take. Now, press `Shift`, `Ctrl` or `Shift+Ctrl` and drag. This time the key determines what **shape** or **starting point** you want for your selection.

This procedure makes it easy to add a rectangle to a selection, or make subtractions with squares or circles. It is, however, rather tricky and if you want to do some serious work using these commands I strongly recommend that you plan ahead, and that you always use the guides and rulers to place new selections correctly. You can of course always use **Channels** to perform such operations. By making white circles in a channel and putting black ones on top of them, you'll subtract a circle without having to remember what key to use, except `Shift` for circle. Read more about making selections with Channels in chapter 18.

	K2=Shift circle	**K2=Ctrl centre**	**K2=Both circ+cent**	**No Key 2**
Key1=Shift Adding	Add circle from corner	Add ellipse from centre	Add circle from centre	Add ellipse from corner
Key1=Ctrl Subtracting	Subtract circle from corner	Subtract ellipse from centre	Subtract circle from centre	Subtract ellipse from corner
Key1=Both Intersect	Intersect circle from corner	Intersect ellipse from centre	Intersect circle from centre	Intersect ellipse from corner
No Key 1	Create circle from corner	Create ellipse from centre	Create circle from centre	Create ellipse from corner

Short Repetition:

To create selections:

- A. Press mouse
- B. Press key and drag

To add, subtract or intersect with other selections:

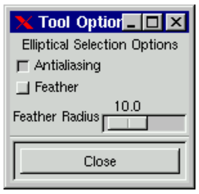
- A. Press key
- B. Press mouse and drag

To create a selection of a certain shape or from a certain direction, **and** use that selection to add/subtract/intersect:

- A. Press key 1 to determine effect

- B. Press mouse
- C. Release key 1, but not mouse
- D. Press new key (2) to determine shape or direction, and drag

SELECTION OPTIONS



Elliptical Selection Options

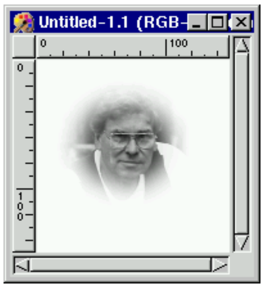
☐ Antialiasing

☐ Feather

Feather Radius 10.0

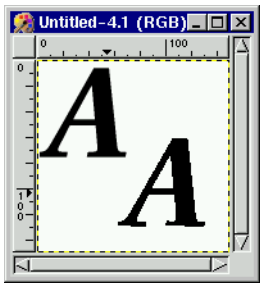
If you `double-click` at the symbols in the toolbox, you'll get a little window showing you the tool options.

FEATHER



For the **rectangular select tool**, the only option is **Feather.** Feather means that you can choose to make the peripheral parts of your selection transparent. It will be **opaque** in the middle, and get more and more **transparent** as you get closer to the edges. This is very useful if you want to make something look soft and blurry, like soft shadows or glowing edges, or if you like to use collage-techniques with the **Paste** or **Paste Into** command. An image you paste into a feathered selection will get soft and transparent at the edges and blend in nicely with your background. It is also usually a good idea to use a small amount of feather if you mean to select something which is to be copied, moved or cut and pasted, because the feathered edges will compensate for an imperfect selection edge and make it blend into the background.

ANTIALIASING



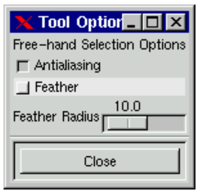
For the other select tools there is also the option **Antialiasing**. Antialiasing is an effect used on curves and boundaries of high contrast areas, which makes the curve or boundary look softer and smoother. The most common use is the antialiasing of letters in bitmapped images, because it takes away the **jaggies** (the visible jagged and pixly edges of the letter's curves in low resolution). What it does is to softly blend the edge pixels with the background, which will make curves look a lot better, but also a bit blurred. So, you gain in smoothness, but lose in sharpness.

THE FREE-HAND SELECTION TOOL

The **free selection tool** works just like Photoshop’s lasso. You create a selection by drawing a free-hand form with it. You close a free selection by ending in the point you started from. If you draw an open shape, the lasso will close it for you.

It’s seldom works to try and select a complex area with just one selection tool. The lasso is an excellent tool to fix up selections with. If you see that you’ve missed some pixels, it is easy to correct this with `Shift` or `Ctrl+` lasso. (As I’m sure you have noticed, the mouse isn’t a very sophisticated drawing instrument. The good news is that the Gimp now fully supports digitized tablets (like the Wacom Art-Pad), and pressure-sensitive pencils. X programs have supported such devices as a substitute for a mouse for a long time. This means that you could use it as a pencil, but it was never pressure-sensitive. Gimp now has patches to make this work. Believe me - working with lassos, pencils and brushes is dramatically different with an accurate tool.)

OPTIONS

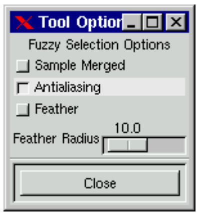


The options for the free selection tool are the same as for the Ellipse selection tool - **Antialiasing** and **Feather**. Read about it in the section above.

FUZZY SELECT

It looks as the Photoshop Magic wand, and it works much the same way. **Fuzzy select** works by selecting adjacent pixels of similar color. It starts selecting when you click somewhere. The wand will select the color on the spot you clicked on, and continue outwards until it thinks the color gets too different.

OPTIONS



There is no control button in the options dialog that determines how sensitive you want the wand to be. Instead, after you position the cursor and press the mouse button, you have to drag the cursor (don’t release the mouse button yet) from the upper left corner, either to the right, or straight down (that doesn’t matter) to go from a small, stingy selection to a very generous one.

Word of warning! Check out your point of departure - if you pointed a little awry you might get the inverted selection of what you wanted - i.e. the wand selects everything except your choice (Black, antialiased object on white background -

you can get a white to gray selection instead of a black to gray one if you are not careful). Needless to say, the wand is the perfect tool to select sharp-edged objects in an image.

It is easy and fun to use, so the beginner often starts out with using the wand a lot. A more experienced user will find that tools like the **Bezier** tool, **Color Select** or **Alpha Channels** are often more efficient for selection, and use the wand more seldom. Still, it's very useful for selecting an area within a contour, or for touching up imperfect selections. The wand is for example very efficient in removing remains of background color from a cut and pasted selection.

In the options dialog box there is a checkbox called **Sample Merged**. This option is available when you use color for an operation. It becomes relevant only when you use several layers. If this option is unchecked, the wand will only react to the color in the **active layer** for your selection. If it is checked it will react as if the image was flattened (layers merged) and use the merged color as it appears on the monitor.

THE BEZIER SELECTION TOOL

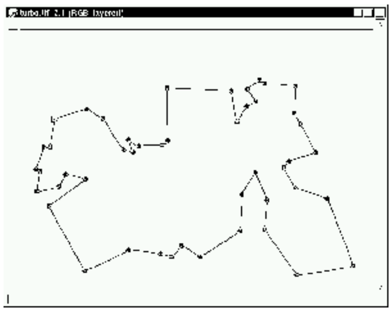
This is to my mind one of the most useful tools. You'll find **Bezier curves** in all self-respecting drawing or imaging software.

USE BEZIERS AS A SIMPLE DRAWING TOOL

turbo.tif-5.1 (RGB)

The Bezier selection is the equivalent of ”Pentool Paths” in Photoshop. Most drawing in programs like Corel, Illustrator, Freehand, or even in 3D programs is done with Bezier curves. Because Gimp is based on **bitmaps** and not **vector** graphics (like Corel or Illustrator), you can't draw with Bezier curves, but you can make advanced selections. I don't count using Stroke, or creating a Bezier curve with a border and then fill the border. If you just want to add a simple drawing, this can be quite sufficient, but if you want to create a more advanced drawing, use the **Gfig** plug-in in the **Filters/Render** menu, or do it in a commercial drawing program, convert it to suitable format and import to Gimp.

CONTROL POINTS

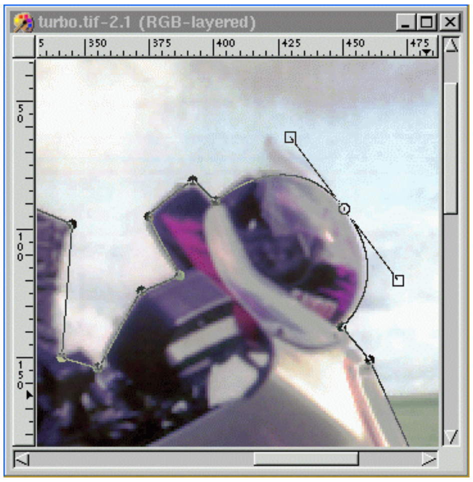


You use the Bezier select tool by clicking out **splines** or **anchor points** in a rough approximation of the shape you want. Don't bother to try and make curves at this stage (unless you're an experienced user), just click out a rough, angular shape and make sure you close it by placing the last point into the first and click.

Plan ahead

I recommend that you plan ahead where to place the anchor points, because in the Gimp, there is no way (yet)to remove or add anchor points with this tool. This means that you have lesser control over your curves. You can't regret anything, so watch out where you put your splines! Also, don't use too many - you only need one for each curve segment.

MODIFYING CONTROL POINTS

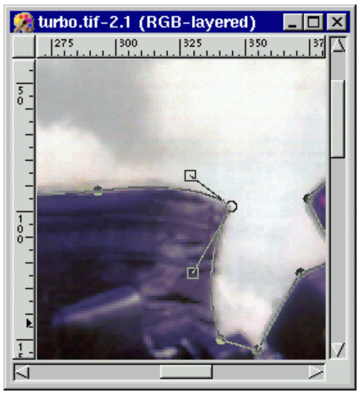


Now, you can start to modify - i.e. to make curves of the straight lines. When you click on an anchor point, two little **handles** appear. If you pull the handles, they will change size and direction and shape a curve. Long handles result in a flattened curve, and short ones in a sharper curve. You can also turn the handles in the angle and direction you want them.

Move

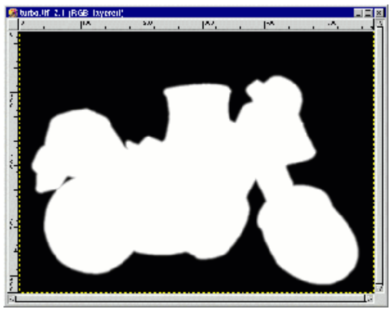
The first thing you'll want to do is to move the splines to their correct position. By pressing `Ctrl`, you can drag and drop an anchor point anyway you like. The other important thing to do is to determine what splines are to be **soft**, and which are to be **sharp** or angled. By default the handles have equal length and create soft, wide curves.

Sharp corners



When you need a **sharp corner**, you have to press `Shift`. Now each handle is managed separately, and you have total control over the shape of your curves.

The final touch



When you're happy with your curve, just click inside the curve, and it will turn into a selection. Remember - always preserve a complex selection like this in an **alpha channel**. You'll probably need it again.

Options

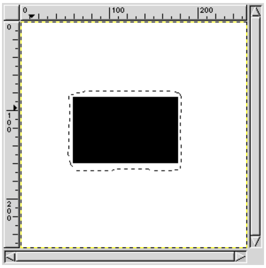
The options for the Bezier tool are Antialiasing and Feather. Read more about it in the section on Rectangular and Elliptical selection.

INTELLIGENT SCISSORS

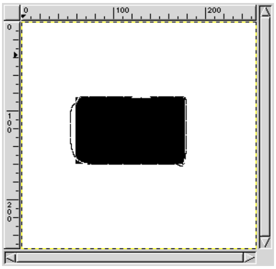
This is a very interesting piece of equipment. What is does is to *guess* the edges that you're trying to select. Does it sound strange? You don't draw an entire selection with the Free-hand tool very often do you? We thought so, it's quite hard to fully control the Free-hand tool, unless you're extremely dexterous. This is where **Intelligent Scissors** comes in. This tool lets you draw the *outline* of an object, just like the Free-hand tool (you can be a bit sloppier), but when you have defined your selection, it will automatically seek out the edges of the object that you are trying to select. If it's not a perfect selection, you have the option to convert the Intelligent Scissors selection to a **Bezier** curve, and make the necessary adjustments. Sounds wonderful, doesn't it? Let us show you how it works.

Make a new Gimp image, select a square and fill it with black. Now make an IS selection around it (like you would with the free hand tool). See, a nearly perfect selection. If you want to adjust it, `double-`

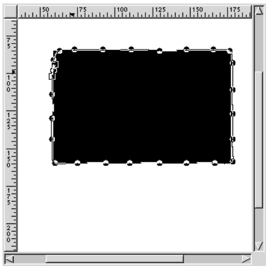
`click` on the IS tool. This will bring up the **option dialog**. Press **Convert to Bezier Curve** and make the final adjustments..



Before it guesses



After IS is done



Final touch with Bezier curve

ACTIVATION AND SHORT CUTS

You have perhaps noticed that there is no *marching ant border* in an IS selection. IS is just like Bezier select, you have to `click` inside the shape to create a selection. This fact will help you to do some fancy tricks, more about that in the Tips paragraph. After you have activated your selection, all the usual selection short cuts work as they do with an ordinary selection.

OPTIONS

There are several options in the IS option dialog. The most important ones are **Edge-Detect Thresh(hold)** and **Elasticity**. Edge-Detect controls how *sensitive* Intelligent Scissors is to the edges in your image. The higher the value, the more sensitive it will be. Elasticity controls how *willing* IS is to *bend away* from the curve you have drawn and snap to the edge. These two options more or less control the whole Intelligent Scissors function. In the images above, you can set both Elasticity and Edge-Detect to a high value since there is only one edge, and you want to select fast and sloppy. The clear contours will enable IS to easily snap away from what you have ”selected” to the edge of the square. On the other hand, if you’re trying to select a certain edge in an image which is full of edges, you have to more careful in your drawing and also lower the Elasticity value so the section won’t snap to another edge instead of the edge you wanted.

Curve resolution controls the roughness of the selection. If you have a lot of curves in your selection, a low Curve resolution will make the curves *rough* or uneven. A higher Curve Resolution will make the curves much *smoother*. **Antialiasing** and **Feather** works like in all the other selection tools.

TIPS

Since an Intelligent Scissors drawing doesn’t turn into a selection until you click in it, you can **undo** what you did *before* the IS operation without destroying your IS selection. This will enable you to run some of the Gimp **Edge -Detect filters** to enhance the edges, so IS will have an easier job. After you have done your IS selection, but before enabling it, press `Ctrl-Z` to undo the Edge-Detect filters (but not your IS selection). When you're finished, click inside the IS selection (or convert to Bezier curves for further adjustment) to turn it into an active selection.

chapter 7

Paint Tools

In this chapter you'll become more closely acquainted with the basic Gimp paint tools

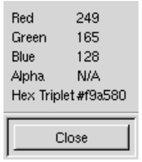
THE COLOR PICKER

The **color picker** is used for choosing colors. If you don't want to use any of the **palettes** available, you can just pick a color in your image (or in any other images that are currently open in Gimp), which gives you a huge spectrum to choose from. Just click on a color in an image, and this color will appear as the **foreground** or **background** color at the bottom of the toolbox.

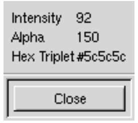
COLOR INFO

A small information window shows you the **RGB**- and **Alpha** values of your chosen color. *Alpha N/A* means that your image contains no Alpha channel. (For more information on Alpha Channels read chapter 18).

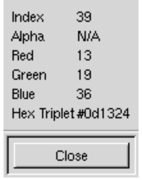
The **Hex Triplet** is the color coded in **hex**. You can use this information to set the background color in your **Web** pages, since hex is used for coding color in HTML. For a **Grayscale** image the information box displays an **Intensity** value ranging from 0 to 255, where 0 is black and 255 is white. An indexed image also has an index number for the specific color. Note that the Alpha value of an indexed image is either 0 or 255, there's nothing in between.



Red	249
Green	165
Blue	128
Alpha	N/A
Hex Triplet	#f9a580

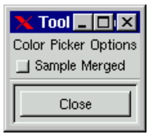


Intensity	92
Alpha	150
Hex Triplet	#5c5c5c



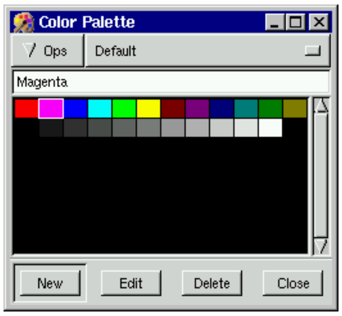
Index	39
Alpha	N/A
Red	13
Green	19
Blue	36
Hex Triplet	#0d1324

OPTIONS



The color picker also has a **Sample Merged** option. Use this option if there are *several layers* in your image and you want to pick a color which represents the color you see on the monitor, not just the color in the active layer. *Only colors in visible layers are used.*

PALETTES



When you want to pick a color to work with, you can always use the **color picker**, but most of the time you'll be working with a **color palette**. You can choose one of the many ready-made palettes, but you can also create your own palette. Another way is to create a palette from an indexed image, i.e `<image>/Image/Save Palette`, or use a temporary palette by bringing up the Indexed Color Palette in the `<image> Dialogs` menu.

To create a personal palette, open the **Color Palette** in the `File/ Dialogs` menu (or press `Ctrl-P`). You'll find a **Default** palette in the palette window, which contains some standard CMYK/RGB colors and a range of grayscales. You can add colors to this or another custom palette, or you can create and name a new palette by selecting **New** in the **Ops menu**. To add colors from an image, drag the color picker over your image until you find a color you wish to add to the palette (look in the **foreground color square** - you'll see the colors change as you drag). To add a color with a specific RGB or HSV value, `double-click` in the FG color icon to access the **Color Selection dialog**.

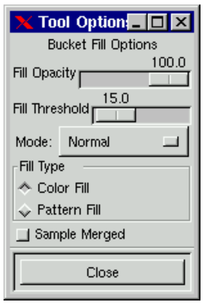
ADDING COLORS

When you find the color you want, press the **New** button, and a new color square will be created in the palette. You can **name** and **save** your new palette, and you can **edit** or **delete** colors. To use colors in a palette, you don't need to select them to the foreground color square with the picker, just click on a new color with whatever paint tool you're using at the time. More information about palettes can be found in chapter 11

THE BUCKET FILL

This tool fills a selection with the current **foreground color**. If you `Shift-click`, it will use the **background color** instead.

OPTIONS



Bucket Fill Options

Fill Opacity 100.0

Fill Threshold 15.0

Mode: Normal

Fill Type
- ● Color Fill
- ○ Pattern Fill

☐ Sample Merged

It will also fill a layer which already contains color and alpha information, in which case the amount of fill depends on what **Fill Threshold** you have specified. The fill threshold determines the **seed fill**, just like the magic wand. It starts at the point where you click and spreads outwards until it thinks the color or alpha becomes ”too different”.

With the maximum threshold value it will fill the entire layer (but you have to choose **Select all** to fill where Alpha is 0).

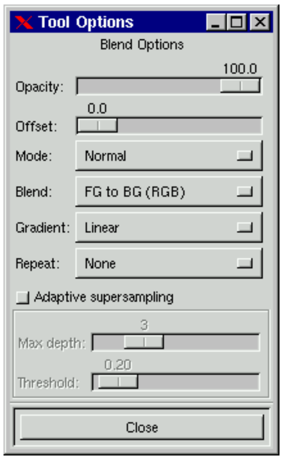
In a **feathered** selection, the bucket will make a soft concentric fill, which can be made more intense by clicking several times. If you use different colors each time, they’ll blend softly into each other in an opalescent effect. There is also a **Fill Opacity** option, where you can specify the transparency of the fill, as well as a **Mode** option (read more about modes in chapter 16), and a **Sample Merged** option. Sample Merged means that the Bucket fill reacts to the color borders in the *entire composite image*, and not just the ones in the active layer (but it will only fill in the active layer). The **Pattern Fill** radio button enables you to fill your selection with a pattern instead of a color. Open the **Pattern dialog** in the `<image>Dialogs` or <toolbox>`File/Dialogs` menu to choose a fill pattern.

THE BLEND TOOL OR GRADIENT FILL

This tool fills the selected area with a gradient blend of the foreground and the background colors by default, but there are many other options in the Gimp blend tool. To make a blend, you simply drag the cursor in the direction you want the gradient to go, and release the mouse button when you feel you have the right position and size of your blend. The softness of the blend depends on how far you drag the cursor. The shorter the drag distance, the sharper it will be.

OPTIONS

The **Offset** option controls the fuzziness of the blend. A high offset value gives a harder, sharper blending edge, and the foreground color becomes more dominant. Offset moves the blend’s point of departure (that is, the point where the FG color starts to blend with the BG color). Offset works with all gradient types, *except for Linear and Shapeburst*. The offset blend is a good choice if you want to control the appearance of the fluorescent, shiny or metallic objects you can create with the blend tool.



Tool Options

Blend Options

Opacity: 100.0

Offset: 0.0

Mode: Normal

Blend: FG to BG (RGB)

Gradient: Linear

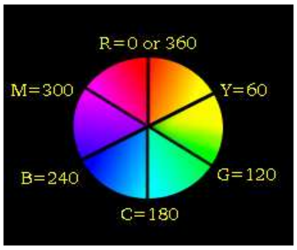
Repeat: None

☐ Adaptive supersampling

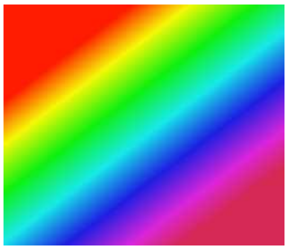
Max depth: 3

Threshold: 0.20

Aside from the standard **RGB** option in the **Blend menu**, there is also the **FG to BG (HSV)** option, where HSV stands for **Hue**, **Saturation** and **Value**. HSV is a color model based on a 360 degree spectrum circle. This means that a gradient using HSV blend, will not simply make a transition from red FG to blue BG via shades of violet. HSV starts with red, and follows the color circle clockwise with yellow, green and cyan until it reaches blue. The gradient will go clockwise if FG is in the right part of the semi circle, counter-clockwise if it's in the left half.



HSV Circle

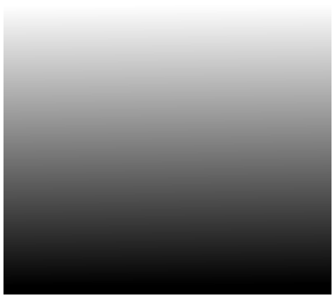


HSV Gradient

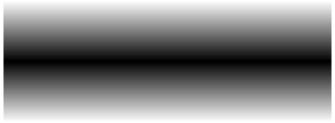
FG to Transparent only uses one color. It gradually changes the alpha value from 255 to 0, causing the color to get more and more transparent. This option is very useful when working with softly blended collages or fog effects. You can also choose use one of the many custom gradients, or make your own gradient with the powerful **Gradient Editor**, in the **Custom from Editor** option.

GRADIENT TYPES

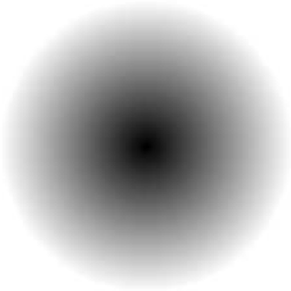
There are no fewer than nine different gradient types, as described below.



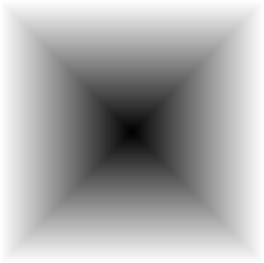
- The default gradient type is **Linear**, which produces a smooth transition from one color to another.



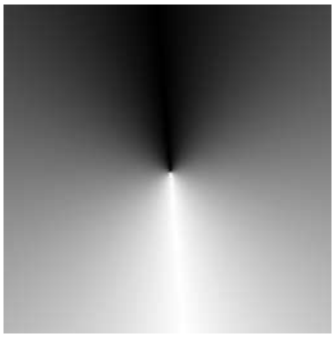
- **Bilinear** mirrors linear. By this we mean that Bilinear places the FG color in the middle, and then makes the transition to BG color in both ways. The result often resembles metal pipes, especially if you drag only a short distance.



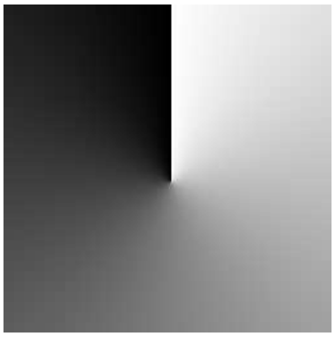
- **Radial**, as the name implies, makes a radial transition from FG (in the middle) to BG (peripheral). The radial gradient type is necessary for certain gradients in the Gradient Editor, such as ”Eyeball” etc.



- The **Square** gradient produces a square blend, imitating the visual perspective in a corridor.

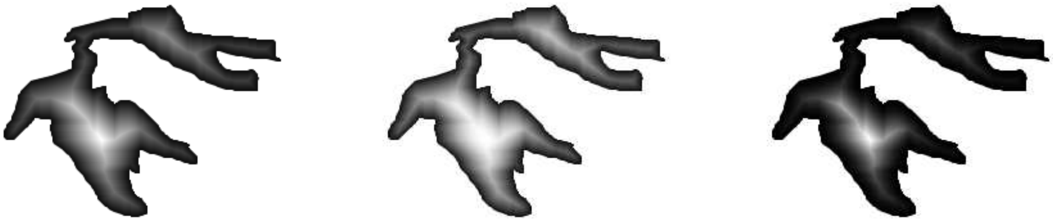


- The effect of the **symmetric** and **asymmetric conical** gradients is like looking at a 3-D cone from above.



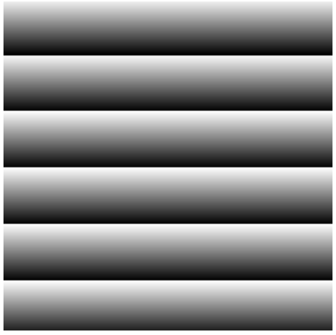
- The **asymmetric** conical gradient imitates a ”drop shaped” cone, i.e. a cone with one sharp edge. The dragging distance is irrelevant for Conical gradients, what matters here is the point of departure (determines where the top of the cone is placed) and the drag direction (determines from which way the FG color, which represents light or shadow is coming from).

- The **Shapeburst** gradients are not affected by drag direction, distance or position. The purpose of shapeburst gradients is to give a defined shape (selection) a 3D-looking fill. You can see the different shapeburst gradients below. **Spherical** gives a flatter, rounder looking surface than **Angular** and **Dimpled**.

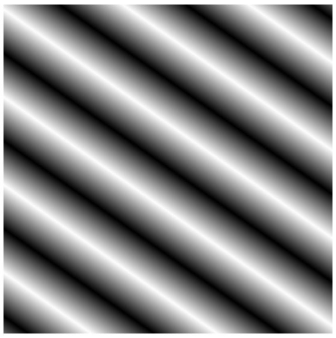


Angular Spherical Dimpled

The repeat options **Sawtooth** and **Triangular** allow you to repeat a gradient fill. The shorter the drag distance, the more often the fill repeats. The examples below show the sawtooth and triangular repeating options with a linear gradient. *Repeat cannot be used with conical or shapeburst gradients*.



Sawtooth



Triangular

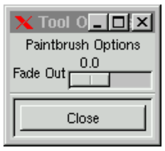
The Blend tool also has an **Opacity** option. Using a semi-transparent opacity value, and then dragging several times in different directions gives an interesting shimmering, bubbly surface to a background. Tip: Also check out the **Difference** option in the **Mode** menu, where doing the same thing (even with full opacity) will result in fantastic swirling patterns, changing and adding every time you drag the cursor. To understand Modes, read chapter 16.

Finally, the **Adaptive Supersampling** is an option which slightly improves the smoothness of the blend by adding a few new intermediate colors. The effect is often quite subtle, unless this option is used for very small selections (especially when using shapeburst), with a gradient from the editor. You can read more about the editor in chapter 11.

THE PENCIL AND PAINTBRUSH

The **Pencil** and **Paintbrush** are very similar tools. The main difference is that although both tools use the same **Brush Selection**, the pencil tool is unable to produce fuzzy edges, even with a very fuzzy brush. If you want to draw straight lines, use `Shift`. Pressing `Shift` and then clicking the mouse button results in a straight line from the last pencil position to the current one. The `Shift-click` option also works for the **eraser**, **airbrush**, **clone tool** and **convolver**.

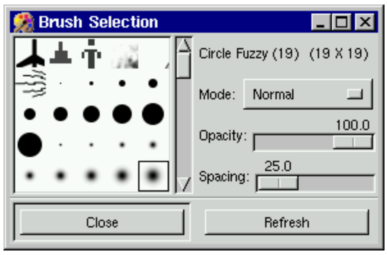
OPTIONS



Fade Out 0.0

The Paintbrush also has a Fade Out and an Incremental option. **Incremental** only affects semi-transparent paint, so in order to see it you must change the opacity value in the **Brush selection** dialog. Incremental allows you to increase opacity in the area where two brush strokes cross, a little like painting with water color.

Fade out means that a continuous sweep of the brush, without releasing the mouse button, will cause the paint to get more and more transparent until it is completely invisible - just as if you made a real brush stroke and ran out of paint. The amount of fade-out determines where the fading starts as well as the length of the ”comet tail”. If you choose the lowest Fade Out value; 34.5, the brush stroke will start to fade after 35 pixels, it will fade to near transparency after 70 pixels, and it will completely fade out when the stroke is about 105 pixels long. You will find more information on brushes in chapter 11.



Circle Fuzzy (19) (19 X 19)

Mode: Normal

Opacity: 100.0

Spacing: 25.0

THE ERASER TOOL

Just as you can paint with pattern-shaped brushes in the Brush Selection, you can also **erase** with the same patterns. This is a great way to create convincing **texture** effects, especially if you use it in many layers. A common technique for isolating a complex selection is to use the **Eraser tool** and just erase the background around the object’s outline, then use the wand or lasso and easily select either the object or the surroundings and delete or cut it.

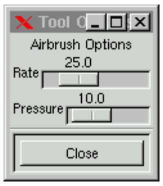
Hints

Floating selections (like text strings) are affected by the eraser tool so that erasing reduces the selection. On the other hand, if you choose **Select/Invert** before floating, you’ll increase the selection instead (provided you invert it back again of course). This happens because the eraser tool affects alpha values in floating selections.

THE AIRBRUSH TOOL

This tool creates soft, semi-transparent spray strokes. There are two options in the Options dialog: **Rate** and **Pressure**.

OPTIONS



Rate 25.0

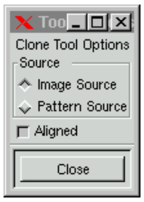
Pressure 10.0

The **Rate** value determines how spray distribution follows movement. A low value produces a smooth, even brush stroke (pausing does not affect it). With a high value this tool behaves more like a real airbrush - if you pause in mid-stroke, the airbrush will keep on spraying and leave a darker stain at that position. **Pressure** regulates how much paint there is in the ”spray”. Note that low pressure isn’t the same as low opacity on the brush. Low pressure is more uneven, or ”airbrushy” looking compared to the smoother appearance that results from high pressure and low brush opacity.

THE CLONE TOOL

The **Clone tool** gives you the ability to paint with **patterns** or **part of an image** instead of just a solid color. To paint with a pattern from the **Pattern Selection** dialog (see chapter 11), simply click at the pattern you want and start painting.

OPTIONS



Use the **Aligned** option if you want a continuous pattern. If the Aligned option is not checked, you will paint a new part of the pattern every time you let go of the mouse button and start again. If you use the **Image Source**, it’s important to know that in Gimp you use `Ctrl` instead of `Alt` (Photoshop) to choose the point of departure from your image source (just `Ctrl-click` at the part of the image you want to start your clone-painting from, return to the image you wish to retouch, and start painting). You can choose to clone from another part of your image, you can even clone from one layer to another, or you can clone from a completely different image.

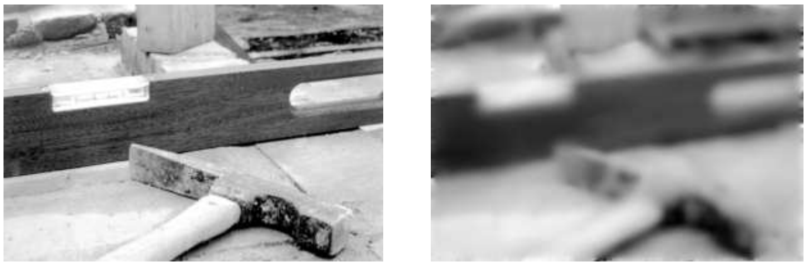
RETOUCHING HINTS

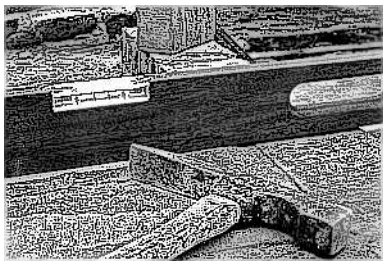
Needless to say, the Clone tool is the ultimate **retouching** instrument. If you for example would like to retouch a birthmark from a face, it would be hard, almost impossible to imitate the complex color varieties of skin texture with regular paint or blends. The cloning of skin in another part of the face is the only way to make it look good. We’re not saying that cloning is easy though; to make it look convincing you may well have to clone from different directions, with different brushes, and maybe even in different layers with appropriate opacities and modes.

THE CONVOLVER

The Convolver's **blur option** is somewhat similar to *Smudge* in Photoshop, though not quite so smeary. It softens sharp edges and blurs whole areas as you paint.

It always blurs to a darker, never to a lighter color.Blurring transparent areas with the convolver will only transform them to opaque color. The convolver will use the nearest color and smear it to full opacity.





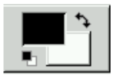
The **Sharpen** option doesn't really sharpen in the traditional sense of the word, it is too ”dotty” or ”pixely” for that. If you want to sharpen up something out of focus, it's better to use **Sharpen** in the `<image> Filters/ Enhance` menu. The convolver sharpen will, however, produce some very interesting graphic results (especially in a grayscale image) resembling engravings, prints or ink drawings.

Another oddity about this tool is that it will sometimes blur out of hue. A ”swimming pool blue” color (45,230,247) on white background results in an almost violet blur color. This probably occurs because the convolve tool uses RGB and not HSV values. Going from a color with high values for green and blue, but low for red, to a color with maximum value for all colors (i.e. white), means that the intermediate colors will get lighter, but they will also get visibly redder.

BRUSH SELECTION

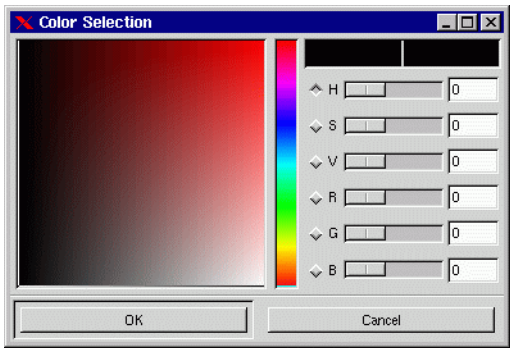
The **Brush Selection** isn't only available for the Pencil and Paintbrush tool, but also for the Eraser, Airbrush, Clone tool and Convolver. You'll find the **Brushes** menu item in the `File/Dialogs` menu in top of the **Toolbox**, or in the **Dialogs** menu within the menu you get from right-clicking in you image. You'll find a variety of brushes. Note that brushes may have any shape! Besides the **Mode** and **Opacity** options there is a **Spacing** option by which the brush stroke can be decomposed into a line of individual brush shaped objects. If you like, you can make your own brushes and add them to the Brush Selection dialog. There is more information on this in chapter 11.

THE FOREGROUND/BACKGROUND ICON



The last tool in the toolbox is the **Foreground/Background** color selector. The foreground color is the color you use to paint or (bucket) fill with. The background color is the color you'll get if you **Erase**, **Cut** or **Edit/Fill** a piece of your image. Pressing the **double arrow** symbol swaps the foreground and background colors. You can also swap the colors by pressing `X` or you can choose to restore the default values (Black/White) with `D`.

THE COLOR DIALOG



H 0
S 0
V 0
R 0
G 0
B 0

Double-clicking on the **foreground** or **background** icons brings up the **Color Selection dialog**, where you can easily change to another color.You can do this manually, by clicking on a Hue in the spectrum box, and then dragging the cross in the large color box to the exact color you want. If you want a specific RGB or HSV color, you can type the exact value in the **HSV** or **RGB parameter fields**. There are also other ways to find the color you want. You can try altering *S* for Saturation, (if you're after pastel or neon colors for instance) or *V* for Value (washed out or gloomy colors). You can also search using the *R* (Red), *G* (Green) or *B* (Blue) sliders for the shade you want.

chapter 8

Edit & View

This chapter covers the important basic functions found in the Edit and View menu. Some of these functions are well known to users of similar software, but some of the features are quite unique for Gimp

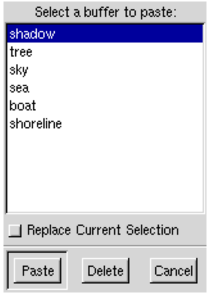
CUT, COPY, AND PASTE

These three commands are so common that they hardly need a presentation. As with ordinary word processing, **cutting** and **copying** a part of an image will place that selection in a local **buffer**. When you choose **Paste**, Gimp takes that selection from the buffer and turns it into a **Floating Selection**, which you can either **anchor** to a layer, or put in a layer of its own. Notice that the default shortcut keys for these commands are the same as in most other programs; `Ctrl+X` for Cut, `Ctrl+C` for Copy and `Ctrl+V` for Paste. I hope that you have noticed the fantastic possibility with **dynamic key-bindings** in Gimp. If you don't like the default settings, you can choose any shortcut key you like for a certain command by simply typing the new shortcut in the menu! Read more about it in chapter 2.

PASTE INTO

Paste Into is a very useful command when you want to insert an image into a defined shape in another image. To use it, you first need to copy or cut the image you want to paste. Then you make a selection shape in the target image (if you use a little feather, you'll get wonderfully soft edges to your selection). Choose `Paste Into`, and the image you copied will appear inside the selection. The initial position isn't fix, you can easily move the image with the **Move tool**, to place it exactly where you want it before you anchor it.

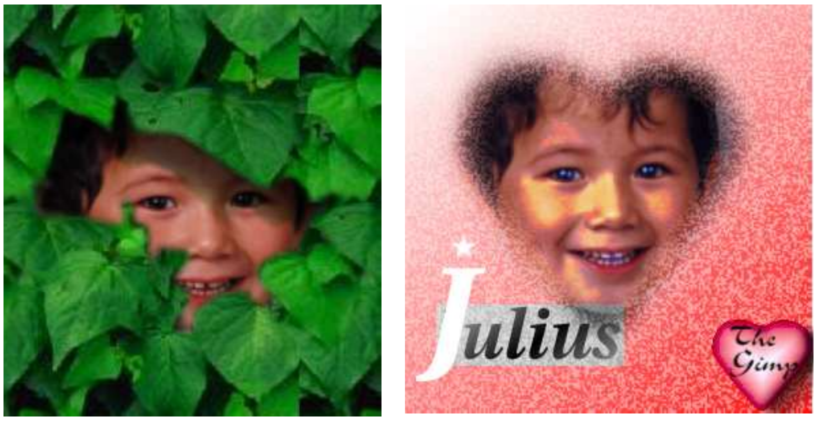
CUT, COPY AND PASTE NAMED



You're probably used to that as soon as you copy or cut something new, the old copy will be replaced by the new one. You don't have to worry about this anymore, because **Cut Named**, **Copy Named** and **Paste Named** allows you to use a **buffer** which can hold a great number of cut images. If you're cutting and pasting a lot of images this is great!

How to

This is how it works: When you have made a selection of an image that you want to cut or copy, choose `Cut/Copy Named`. A small **dialog box** pops up and asks you to enter a name for your selection. This will place your selection in a buffer. When you want to paste one of the selections you have named and put in the **Named buffer**, choose **Paste Named**. This command pops up a dialog where you can see a list of all the selections you have saved here. To paste a selection, just choose its name and press **Paste**, and your selection will appear as a **Floating Selection** in the active image. The option **Replace Current Selection** allows you to replace a (non-floating) selection in the image with a selection from your list. Replace Current Selection unchecked gives the same result as **Paste Into**, if you already have a selection in the image. If you try to do the same with a floating selection you'll just anchor the old float to the image, and make the new selection into a new float in the same place as the old one. You can also delete a selection from the list if you no longer need it. The Cut/Paste Named command is very useful, just remember that *the Named buffer only exists as long as your Gimp session*. If you want to save a selection more permanently, you'd better save it in a separate layer in your image.



These images were created by using the Cut, Copy and Paste Named option.

CLEAR, FILL AND STROKE

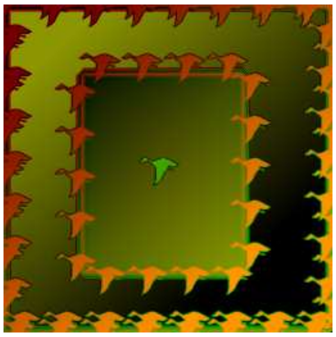
CLEAR

Clear works much like cutting, without saving what you cut away. It's a powerful command. Used in a layer, it will delete everything and leave an empty, transparent layer. Used on a selection it will delete or cut everything inside of it, leaving only the "marching ants". Used on a (non-alpha) background it will delete all background information, and leave only the background color in the BG color swatch in the Tool box.

FILL

Fill is the equivalent of the **Bucket Fill** with a maximum fill threshold. It's not as sophisticated as the Bucket Fill, but it's quick, easy and efficient. Note that this command always uses the **background** color in the Tool box color swatch, while the Bucket Fill uses the **foreground** color.

STROKE



Stroke is an interesting command. It creates a **color frame** around the selection edge. This frame is based entirely of what brush you're using, and what brush options you have set in the **Brush Selection Dialog**. Try using Stroke with different brushes and different settings for Spacing, Mode and Opacity. You'll find that Stroke can be quite versatile.

UNDO AND REDO

These options should be self-explanatory. The only thing I'll say about them is that you use them so often that it's a good idea to learn their key shortcuts by heart. The default is `Ctrl+Z` and `Ctrl+R`.

COPY VISIBLE

This function will copy all visible layers to a flat copy which you can paste into a new image. It's like executing both **Copy** and **Merge visible layers** at the same time. This function is quite handy when you don't want to alter the original layers, but only work with a quick copy. The copy can then be pasted into a new image.

ZOOM

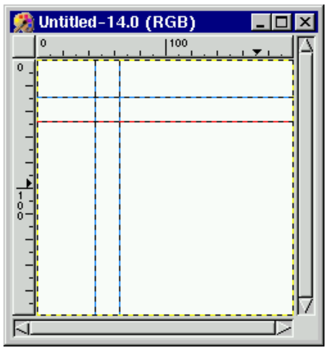
Zooming with the **Magnify tool**: If the Magnify tool is active, you just `click` in your image to zoom in, and `Shift-click` too zoom out. To zoom a specific part of the image, drag diagonally across that part (draw a rectangle), and release the mouse button. You can also **pan** a zoomed image with the **middle mouse button**.

Zooming with the **View menu**: There are several shortcuts to zooming in the View menu. Whether or not the Magnifying tool is active, you can always zoom with the **Zoom in** and **Zoom out** shortcuts. The default hotkeys are (−) for zoom out and (=) for zoom in, but since Gimp has **dynamic key-bindings** you can for example choose (+) instead of (=), if you like that better. The Zoom option gives you access to nine different scales; from 16:1, 8:1, 4:1, 2:1, 1:1,... - 1:16. To get scale 1:1 press (`1`). Pressing (`minus`) zooms in smaller steps than **Automatic zoom**, so to display your image in scale 1:3, you press (`minus`) three times. 16:1 is the lower limit and 1:16 the upper limit for zooming in Gimp.

GUIDES AND RULERS

At the top and left side of the image frame, you'll find the Gimp **Rulers**. In Gimp, all images are measured in **pixels**, which is quite OK if you're designing for the Web, so the default ruler unit is in pixels. You can change the ruler unit to millimeters or inches in `.gimp/gimprc` (see appendix A). However, there is not much point in doing so, because in Gimp you can't set **image resolution** and **canvas size** the way you can in Photoshop. **Size** (and thereby **Resolution**) for a printed image is determined by the **Print** dialog or **Save as (Postscript)** dialog.

TOGGLE AND SNAP



The horizontal and vertical **Guides** can be dragged straight from the left or upper **Ruler**. To change the position of the Guides, you must use the **Move tool** (notice how the move symbol changes into a pointing hand when it touches a Guide). **Snap to Guides** is the default set in the **View menu**. If this option is checked, moving any kind of selection close enough to the Guides, automatically causes it to "stick" or snap to it. You can for example decide exactly from which point of origin you want a selection to start. If you use the `Ctrl` key, and start dragging close enough to the point where the Guides cross, that will be the centre of the new selection. Without `Ctrl,` the selection will start from the cross and continue in the direction you drag. You can easily adapt the size, shape and position of a selection to the guides by drawing it within the frame of two vertical and two horizontal Guides. If you find them disturbing, or just don't want to use them at the time, both Guides and Rulers can be toggled on/off with the `Toggle Ruler/Guides` command in the **View menu**.

WINDOW INFO, NEW VIEW AND SHRINK WRAP

NEW VIEW

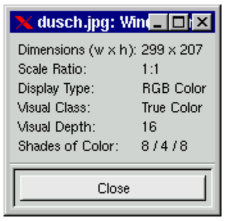
There is another very useful item in the View menu called **New View**. With New View, you open up a new window displaying the same image. This enables you to watch your creation from several windows, each with different focus and zoom. It is still the same picture, and the changes you make appear in all of the New View windows. If you want to try out different versions of your image, press `Ctrl+D`, to **Duplicate**. Contrary to New View, duplicate windows are separate, changeable copies of your original image.

SHRINK WRAP

Shrink Wrap means that the canvas size will always adapt to the amount of zoom you use, so you'll always be able to see the entire image regardless of zoom level. If you want this to happen automatically every time you change the zoom level, you can use The **Magnify Tool** option **Allow Window Resizing**

WINDOW INFO

There is also a **Window Info** dialog in the View menu, which will give you up-to-date information on the active window, such as **Dimensions**, **Scale Ratio** and **Display type**.



Dimensions (w x h): 299 x 207
Scale Ratio: 1:1
Display Type: RGB Color
Visual Class: True Color
Visual Depth: 16
Shades of Color: 8 / 4 / 8

chapter 9

Transform Tools

In this chapter, we'll take a look at the different transform tools, including the Move tool, the Magnifying glass and the Crop tool

THE MOVE TOOL

The fact that the **Move tool** moves selections should be obvious, but this tool works in different ways. You can move **floating selections**, you can move the entire **image** or **layer**, and you can move an empty **selection shape**. We have already discussed the fact that Gimp creates a ”fake” Move symbol, which transforms your selection to a float when you use it. This means that if you don’t want a float, you have to remember to change to the Move tool as soon as this happens. You can also ignore the ”fake move symbol” and turn straight to the **Float** command in the `<image>Select` menu. This procedure assures you of the float’s position, *and IF you decide to move it, you can use the Move tool in the normal way.*

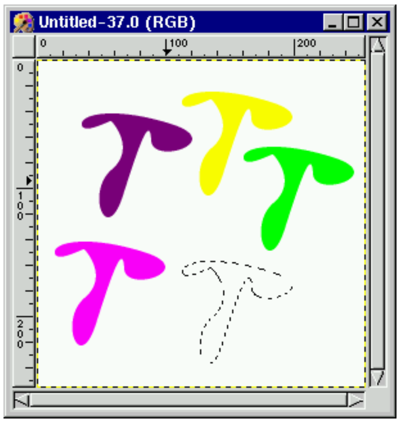
MOVING FLOATING SELECTIONS

When you use the Move tool on a **Floating selection**, you’ll notice that the **double-arrow** symbol changes to a **single arrow** when the pointer is outside of the selection. `Mouse-clicking` when the single arrow is visible causes the float to **anchor to** (merge with) the layer which was last active.

MOVING THE WHOLE IMAGE OR A SINGLE LAYER

If you don’t have a selection, the Move tool causes the **entire image** to move, or the **active layer**, if you have layers. *If you try to move a* ***transparent*** *layer, the Move tool won’t do it.* Instead, it will move the top layer in the layer stack, and this can be very irritating! This happens because to move a layer, the Move tool needs something solid to grab on to. If you paint a solid dot of color in the empty layer and you turn off all the other layers (make them invisible) there will be no problem to move it, but this isn’t very convenient, is it? There is a solution to this problem, and that is to press `Shift` as you drag. You’ll see the yellow **layer boundary** turn blue, and you can be sure that you’re moving the right layer. Also notice that if you have **grouped** one or more layers (with the little anchor symbol), they’ll all move regardless of which layer is currently active. This means that if you for example have grouped a text string with its drop shadow, you’ll have to ungroup them before moving another layer, otherwise you’ll move all three layers.

MOVING EMPTY SELECTIONS



If you don't want to move a floating selection or a layer - when you just want to ***adjust the position*** of the selection, use the `Alt`-key. Observe the difference! You move only the empty form of the selection with the `Alt`-key, not the selection with its contents! This option is only available for normal (empty)selections, you can never move a floating selection with the `Alt`-key. This option allows you to use the selection as a template - you can move the selection around and fill it every time you move it. Note, the **Move tool** or a **selection tool** must be active for this to work.

Hints

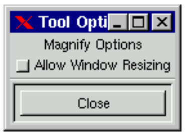
The Move-tool has an extra feature which I really like. You can **nudge** the selection with the keyboard's cursor keys (just like in Corel Draw), *nudge=moving very precise in small steps.* Pressing `Shift` will increase the length of these steps.

One more thing about moving - when you make **multiple selects**, Gimp treats them as a **unit**. That means that you can't move one selection closer to another with the Move tool, because the whole lot will move together as one.

MAGNIFYING GLASS OR ZOOM TOOL

To **zoom in**, you simply `click` in your image, - to **zoom out**, press the (`-`)key, or `Shift click`.To zoom a specific part of the image, drag diagonally across that part, and release the mouse button. There are also several shortcuts to zooming in the **View/Zoom** menu. You can get nine different scales, from 16:1 - 1:16. To get scale 1:1 press (`1`).

OPTIONS



☐ Allow Window Resizing

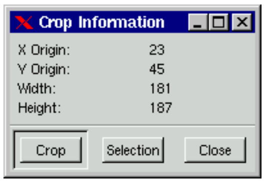
I recommend the excellent option **Allow Window Resizing**. This means that the canvas size will always adapt to the amount of zoom you use, so you'll always be able to see the entire image regardless of zoom level. If you don't want to use this option, you can press `Ctrl+E` for **Shrinkwrap** - this procedure will also adjust the canvas to image size.

There is another extremely useful item in this menu; the **New View** command. With New View, you open a new window showing your image. This enables you to watch your creation from several windows, each with different focus and zoom. It is still the same picture, and the changes you make appear in all of the windows. If you want to try out different versions of your image, press `Ctrl+D`, for **Duplicate**. The duplicate windows are separate, changeable copies of your original image.

THE CROP TOOL

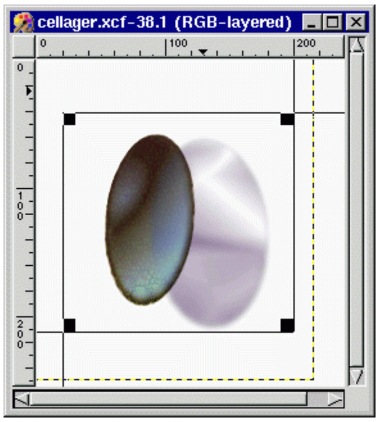
This tool corresponds to the ”scissors” in Photoshop. You can **crop** a scanned image, if you just want a part of the picture. To crop, you drag the marquee diagonally, and release the mouse button when you are satisfied.

THE DIALOG



X Origin: 23
Y Origin: 45
Width: 181
Height: 187

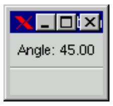
Press the **crop button** in the dialog box, or click inside the markings, and whatever’s inside of the cropmark will be your new image. The **Crop Information** dialog box informs you of the crop placement in **X-Y origin** and **width/height** in pixels. The cropmark can be placed very accurately. You can **move it** (lower left or upper right corner) or **resize** it (opposite corners). It appears very clearly against the rulers, and you can use **nudge** (see Moving empty Selections) and **Snap to Guides** to help you further. Another option is to click the **Selection** button in the dialog box. This will automatically place the cropmark as close as possible to your selection.



THE TRANSFORM TOOL

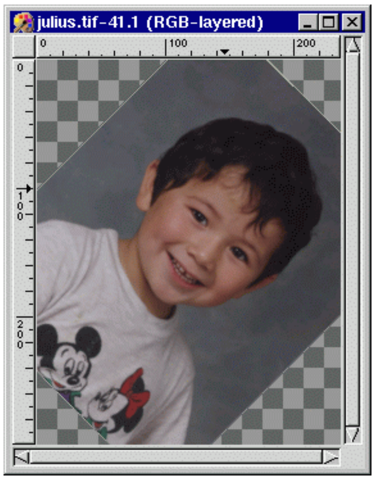
It’s much like ”Effects” in Photoshop. You can **rotate**, **scale**, **shear** or **distort** a selection with this tool. With the **Rotate** tool, you rotate your selection manually. Note that parts of the image will end up outside the drawable area, so you may have to resize the image afterwards.

ROTATE

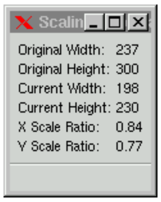


Angle: 45.00

The **rotation angle** is displayed in a small dialog. You can’t write any numbers there, so this is no high precision tool. You can, however, use the `Ctrl` key, which makes the selection turn 15 degrees at a time, which in most cases is quite sufficient.

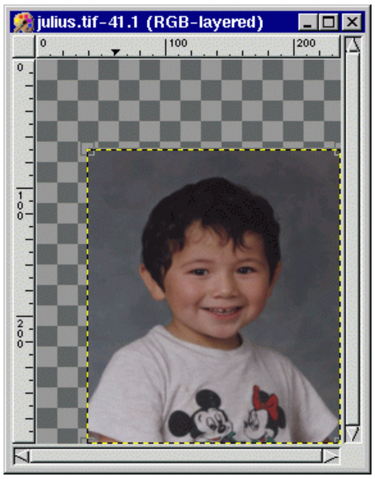

julius.tif-41.1 (RGB-layered)
0
100
200
0
100
200

SCALE

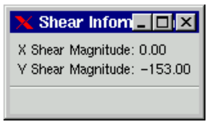


Original Width: 237
Original Height: 300
Current Width: 198
Current Height: 230
X Scale Ratio: 0.84
Y Scale Ratio: 0.77

The **Scale** tool is quite advanced though. The little information box tells you both the original **height** and **width** of the selection a well as the new measurement. It also informs you of deformation rate in the **X-and Y scale ratio**. The `Ctrl` key locks the scale in the **X-term**, `Shift` locks in **Y-term**. `Ctrl+Shift` returns to the original shape, and **scales proportionally**; xchange=ychange.

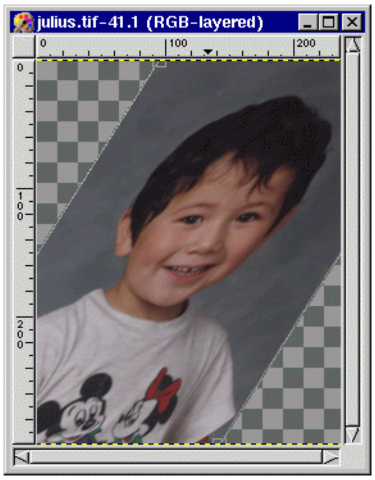

julius.tif-41.1 (RGB-layered)
0
100
200
0
100
200

SHEARING



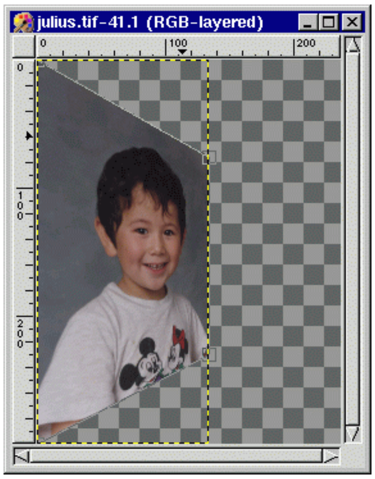
X Shear Magnitude: 0.00
Y Shear Magnitude: -153.00

Shearing can be done in both X- and Y-term, but only one at a time! If the X magnitude is other than zero, then Y must be zero. If you want it sheared both ways, you have to change to another tool icon, and then return to **Transform**. Dragging up and down shears to Y, and to the sides to X.

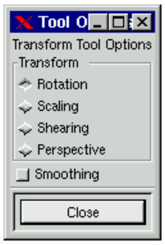


PERSPECTIVE

The last option is called **Perspective**, which isn't quite accurate, because there is no actual perspective transformation (small in the far end and large in the other). I would rather call it distort, as you can change all four displacement points independently. You can certainly make something look like a perspective, but you have to do it manually. Observe that you can turn the "perspective" inside out if you like. If you let the lines of the perspective square cross or top each other you'll mirror and wrench the selection so much it'll soon be unrecognizable.



OPTIONS


Tool O
Transform Tool Options
Transform
Rotation
Scaling
Shearing
Perspective
Smoothing
Close

Smoothing is as you might have guessed an **antialiasing** option, which makes the distorted edges look better. If you don't use smoothing, you selection might turn out very rough and pixly in some edges.

THE FLIP TOOL

It does exactly that - it **flips**, or more accurately, **mirrors** the selection **horizontally** or **vertically**. If you are working with an image where you want the illusion of reflection, whether in water, glass or metal, this is the tool to use.

chapter

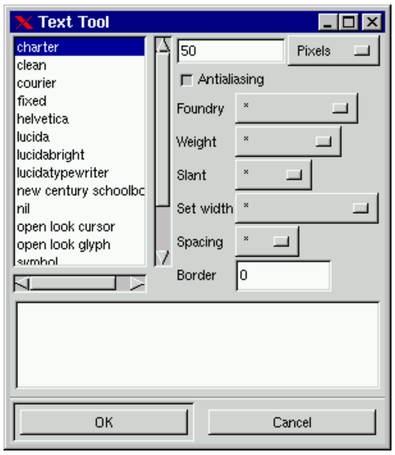
10

Text and fonts

Gimp text is based on the way X manages fonts. In this chapter, we'll try to shed some light on the matter.

THE TEXT TOOL

To use **text** in Gimp, you click in the image with the **Text tool** active. This opens the **text dialog** box where you can see a scroll of the installed **fonts**, and at the top, the current **text size** in either pixels or points. There is also an **antialiasing** option and six parameters (Foundry, Weight, Slant, Set width, Spacing and Border).



50 Pixels

☐ Antialiasing

Foundry *

Weight *

Slant *

Set width *

Spacing *

Border 0

- **Foundry** refers to the origin of your font (usually the manufacturer). This parameter is used to separate fonts with the same name, but from different companies. This is not unimportant, there are lots of different versions of well known fonts and you need to specify which one you mean if you have several versions installed.
- **Weight** shows what typographic ”blackness” options you have for the font you have chosen (**black**, **bold**, **demibold**, **medium** and **regular**). Options which are not available for this font are grayed out.
- **Slant** is the ”posture” option. The letter ”**r**” signifies (**Roman**) *upright* posture. The letter ”**i**” (**Italic**) or ”**o**” (**Oblique**) are two versions of *slanted* posture, where ”**i**” gives a good representation of the letters, and ”**o**” gives a simpler, more jagged sort of italic. What version is available depends on the font.
- **Set width** is an option if your font has the built-in possibility of **horizontal width** like ”**semi-condensed**” etc.
- **Spacing** informs you of what sort of in-built spacing you font has - ”**c**”, ”**m**” or ”**p**” (for **character cell**, **monospaced** and **proportional**) The ”**c**” and ”**m**”-fonts are mainly for programmers, and are used in terminal display windows. The proportional fonts are what you might call the ”real” or ”normal” typographical fonts, because they are not constrained to a fixed width.
- **Border** - This option is not to be confused with the Select Border option in the Select menu, and it does not offer a beveled effect to your letters if that’s what you thought. Instead it ***increases the size of the text layer*** (that yellow-dotted rectangular box around your text). A border of zero fits your text

very snugly, while larger borders increases the size of the layer size. The reason for using **Border** is that you'll need some space if you want to hand kern your letters (adjust the spacing between the letters), or if you just want to move them around separately. If you find that your border size wasn't large enough after all, you can always change it by using **Resize Layer** in the **Layers menu**. More information on floating selections, and how to move letters in chapter 18.

BORDER MATTERS

Note! You might be fooled to think that you got ”a real border”. When you fill letters in a text layer with a different color than before, it might actually look like a sort of border around your letters. This has nothing to do with the **Border** option in the **Text tool** dialog. Instead, it is due to a **low fill-threshold** in the **Bucket fill** options dialog. With a low threshold, the Bucket tool won't fill semi-transparent pixels, and they will stand out against the fill because they have kept their original color. With a higher threshold everything will be filled with the same color.

Text and fonts

chapter 11

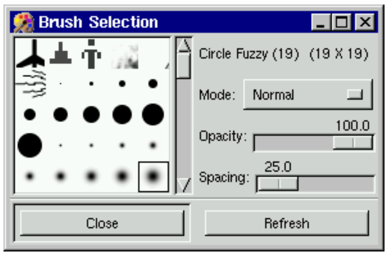
Brushes, Gradients, Palettes and Patterns

In this chapter will we discuss how to create and use these basic tools.

BRUSHES

In the **brushes dialog**, accessible from the **Toolbox** file menu or **<image>**menu dialogs, you will see all the brushes you can use in your Gimp sessions. You may notice that there are some strange brushes, like text, little figures and so on. These brushes are not very difficult to make yourself, if you are a bit artistic. This is what we will discuss in this section, but first a bit about the Brush dialog.

THE DIALOG



Circle Fuzzy (19) (19 X 19)

Mode: Normal

Opacity: 100.0

Spacing: 25.0

If you select a brush you will se it’s name and size in pixels. In the **Modes** drop down menu, you can select what kind of mode your brush should use, *more about this in chapter 16*. If you can’t see the whole brush, press your left mouse button in the brush square, and the entire brush will appear. The **Opacity** slider sets the amount of transparency. With the max value of 100, your brush paints with solid color, whereas a value of 0 gives you total transparency (or invisible paint). **Spacing** is the distance between your brush marks. If you set the spacing to zero, there will be no distance between the marks of your brush - i.e. a fluent line.

Let’s test the **finger brush** and set the spacing to 150. You will get lots of fingers with some space between them, but you can still see that they are fingers, but if you had set the spacing to zero you wouldn’t have. You will normally use low values of spacing when you paint with ordinary brushes, but when you paint with special brushes a bit of spacing can be useful.

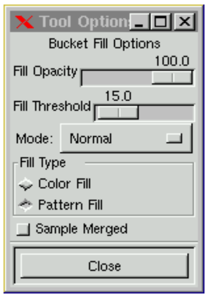
MAKING A BRUSH

How to make a brush: Create a new grayscale image, the default values will be fine, and make the background white. Now, pick up a pencil and draw an X over the image frame. Invert the image by applying `Image/ Map/Invert`. Rescale it to say, 20x20 in the **Image/Resize** menu. Save the image as xxx.gbr. In the GBR save dialog, set the default spacing to 15,and call your brush Olof. Move the xxx.gbr to your `.gimp/brushes` directory. Open the brush dialog and press `refresh`. The new brush will now appear in your Brush Selection dialog under Olof (20x20).

A general tip on how to make good-looking brushes is to make it real big and then resize it to the size that you want your brush to be.

Note that solid black (before inverting) makes your brush look hard, if you use a gray color it’ll look softer. The same goes for when you create a brush with lots of blur, the new brush will get soft. So keep experimenting with different kinds of brushes, create them with or without blur, soft edges and so forth.

PATTERNS



Fill Opacity 100.0
Fill Threshold 15.0
Mode: Normal

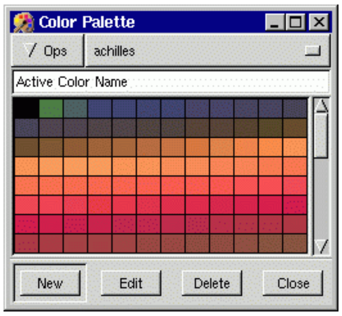
It's nice to be able to fill selections or backgrounds with a pattern instead of a solid color. Gimp comes with many patterns, and you can also find nice patterns at many Gimp-people's home pages. To use a pattern you have to select one in the **Pattern Select** dialog. If you can't see the whole pattern, press your left mouse button to get a full preview. The name of the selected pattern is typed in the upper left corner and you can browse the different patterns by clicking on them (selecting). To use your selected pattern, you have to bring up the **Fill tool** option dialog. Do so by `double-clicking` on the `Fill tool icon`. In this dialog select `Pattern Fill`. Now you can fill with your selected pattern. You can also paint with patterns, using the **Clone tool**, but you can't use patterns with any other tool.

MAKE A PATTERN

To make your own pattern, just open a new RGB image and create your pattern (or take a nice pattern and change it, system patterns are in `/usr/local/share/gimp/patterns`). To create seamless tiles, use **Make Seamless** in the **Filters/Map** menu or the **Offset** option in the **Image/Channel Ops** menu. When you're ready, resize the image to fit the tile size of the pattern you want to create. Save the image as `pat file` and name your pattern in the **Save option** dialog. Move the pat file to `.gimp/patterns (mv test.pat .gimp/patterns)` and in the **Pattern select** dialog, press `Refresh` and your newly created pattern will be available. Go on, experiment and make different patterns.

A great thing about patterns is that you can also use them as a texture when you paint in Gimp. To do so you will have to learn about Modes/Value in chapter 16.

PALETTES



achilles

Active Color Name

In the palette dialog, you can create a new palette in the **Ops** drop-down menu. In the same menu can you also merge and delete palettes. When you make a new palette you always start out with a black palette with no color in it. You also have to name it.

When you press **New** to get a new color to your palette, the color is always visible as foreground color in the **color icon** in the toolboox. You can edit a color in your palette by selecting it and pressing **Edit**. *This will bring up the Color Selection dialog, see chapter 12*. You can delete a color by selecting it and pressing **Delete**.

As soon as you edit a **system wide palette** (other than your personal palettes), it will end up in your personal palette directory. If you delete a system wide palette however, it will not be deleted, only acknowledged that you don't want to use it in this session. But if you delete a **personal palette**, it will be gone forever. If you have loaded a lot of palettes, it will take some time before they will be displayed in the dropdown menu, so be patient when you have pressed it.

CREATE A PALETTE FROM AN IMAGE

You can create a palette from an indexed image. This is quite handy if you only want to use the specific colors in the image. Here's how you do it:

- Convert the image to Indexed mode with `<image>/Image/Indexed`.
- In the Index dialog you select how many colors your palette should contain, click `OK`.
- Invoke `<image>/Image/Save palette`
- Name the palette and press OK

You now have a new palette, but to use it you first have to restart Gimp.

Hints

A tip: Use palettes when you want to create images with a fix number of colors, for example when you are creating icons.

Say that your screen depth is only 8bit (256) colors, and you don't want your icons to use them all up. If your icon manager reduces the number of colors it may look awful. So, what you want to do is to create a palette with say, 50 basic colors for your icons.

The **palette format** is like the format of the `rgb.txt` file delivered in your X window system. First comes the number values of the color; say, 255 134 56 and then the name `<name>`. This makes it possible to edit palettes in an ordinary text editor.

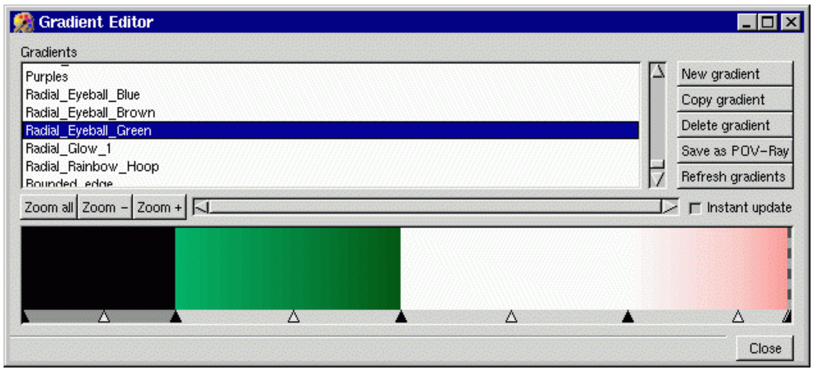
GRADIENTS

When you talk about gradients, you may think about the **blend tool**, which lets you ”flood” a selection or image, starting with one color and smoothly changing into another. Gimp lets you do even more complex blends with a tool called the **Gradient editor**.

THE GRADIENT EDITOR

In this editor can you specify what your gradient will look like, and what colors should be used in it. To use it, select **Custom from editor** in the **Blend** drop down menu in the Gradient tool option dialog (just `double-click` in the gradient tool icon).

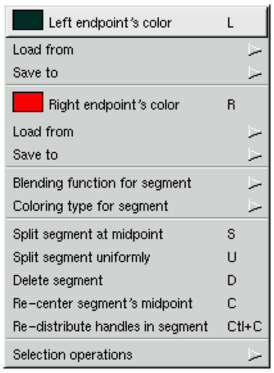
The gradient editor is a highly flexible tool, and you can create very advanced gradients with it. Let's take a look at the user interface. The first thing you'll see when you bring up the dialog is the main view. The current gradient can be seen in a **preview** window, and there is a **selection browser** for all gradients. To the right, you'll find some buttons that lets you save and copy etc. Pressing the right mouse button in the gradient window brings up a menu with editing tools for your gradient.


Gradient Editor
Gradients
Purples
Radial_Eyeball_Blue
Radial_Eyeball_Brown
Radial_Eyeball_Green
Radial_Glow_1
Radial_Rainbow_Hoop
Rounded_edge
New gradient
Copy gradient
Delete gradient
Save as POV-Ray
Refresh gradients
Zoom all
Zoom −
Zoom +
Instant update
Close

How to use the gradient editor

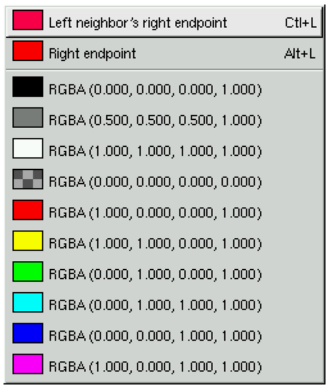
So what can you do with it? Let's start by copying a gradient and play around with it. Select a gradient, choose `copy`, and remember that undo is disabled in the gradient editor. (*if you edit a system wide gradient, it will end up in your personal gradient directory, just like the brushes and palettes*). This brings up the **naming dialog** (New will bring up the same dialog), name it and off we go. The triangles at the bottom are **color section** markings. There are two kinds of markings; **endpoints** are black and **midpoints** are white. The area between two black points is called a **segment**. As you see, I haves selected a segment by clicking on it, which turns it gray. Now, if you move the midpoint it will move the breaching point towards one of the end colors. Dragging at the endpoints *(not the ones to the extreme left or right since you can't move them)* will make the selection wider or smaller. You can move the whole selection by clicking in the gray field and drag. If you `click` at an endpoint and then press `Shift` and drag you will **compress** the selection. You can also **extend** a selection by `Shift - clicking` on another segment. You can use the same manipulations on an extended selection as you can on a normal segment.

The popup menu


Left endpoint's color L
Load from
Save to
Right endpoint's color R
Load from
Save to
Blending function for segment
Coloring type for segment
Split segment at midpoint S
Split segment uniformly U
Delete segment D
Re-center segment's midpoint C
Re-distribute handles in segment Ctl+C
Selection operations

As mentioned before; if you press the right mouse button in the gradient, a menu. will appear where you can edit the colors in a selected segment. You can edit both endpoints by selecting **Left/Right endpoint's color**. This will bring up a **color edit** dialog (*see chapter 12*) where you can select a new endpoint color.

Endpoints


Left neighbor's right endpoint Ctl+L
Right endpoint Alt+L
RGBA (0.000, 0.000, 0.000, 1.000)
RGBA (0.500, 0.500, 0.500, 1.000)
RGBA (1.000, 1.000, 1.000, 1.000)
RGBA (0.000, 0.000, 0.000, 0.000)
RGBA (1.000, 0.000, 0.000, 1.000)
RGBA (1.000, 1.000, 0.000, 1.000)
RGBA (0.000, 1.000, 0.000, 1.000)
RGBA (0.000, 1.000, 1.000, 1.000)
RGBA (0.000, 0.000, 1.000, 1.000)
RGBA (1.000, 0.000, 1.000, 1.000)

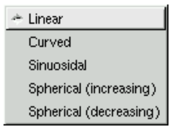
You can also load and save an endpoint color to/from an RGBA channel in the **Load from**, or **Save to** menu. You will find that you can load some nice basic colors and transparent backgrounds. The saving menu is quite useful if you want to use a certain end color in another part of the gradient.

Segments

The **Split segment midpoint/uniform** command will split a selected segment. **Midpoint** creates a duplicate and places it beside what's left of the original. Both of the parts will be half the size of the original. **Uniform** lets you decide how many splits you want to make in the (gray) selection. If you have selected more than one segment, **Split** will not split them as a unit, it will treat each selection separately.

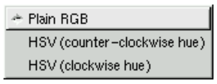
Delete segment deletes the entire selection, not just the segment you press the right mouse button in. **Re-center segment's midpoint** will re-center your midpoint in the selected segments. **Re-distribute handles in segment** is good if you have played around with the points/handles in a segment or selection. It will restore them to their original positions ***(undo!)***.

Blend



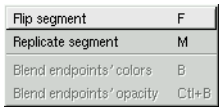
In the **Blending...** submenu, you will find some **Modes** that control how the gradient in your selection behaves. **Linear** is straightforward (and default). It will make the color change in a straight direction from one endpoint to another. **Curved** causes the end section's colors to change in a different "speed" over the middle section. You can think of it as a semi circle. The color change will appear to happen fast in the ends of the half circle and slower in the middle. **Sinusoidal** acts the other way around. Slow in the beginning and fast in the middle. **Spherical inc**. will let the transition happen fast to the left and slow to the right. **Spherical dec.** is the opposite of inc., slow left side and fast right side.

Coloring



The coloring submenu lets you choose a color model for your selection or segment. There's Plain RGB and two kinds of HSV. *You will have to look in chapter 12 to learn about different color models.*

Flips



Flip lets you flip a segment or selection. This can be quite handy if you want to change the order of two segments. First select two segments and flip both as one. Deselect one of them and flip the other one once more. Select the first one and flip that too. Now you have changed the order of your segments. This can be done to more than two segments, and gives you the freedom to change the structure of the gradients.

Replicate

Replicate pops up a little dialog where you can make a copy of your segment or selection to X copies. **Blend endpoint's colors** will only work if you have selected more than one segment. It will blend the end points of your selections and it will also "merge" your selections so they will blend gradually from one end point to the other. Blend end points opacity does the same, but for Alpha values.

Save, Save as, and POV gradient format

You can also save a gradient in POV-ray format. **Save as POV-Ray** is nice for POV-raying people. If you have downloaded a gradient from the Internet and placed it in `.gimp/gradient` you will have to press **Refresh** to see it and use it. Every edited system wide gradient will end up in your personal gradient directory. So if you want your system wide gradient back, just delete or move it from your personal directory. If you delete a system wide gradient, it just means that you don't want to use it in this session. But if you delete a personal gradient it will be gone forever.

Gradients can be very versatile. You can use them in advanced fountain fills or patterns, but you can also create object-like things with them, like an eyeball, a hole or a pipe etc. You will just have to experiment and find out for yourself.

part

IV

Color knowlage

- COLOR MODELS
- PREPRESS

chapter 12

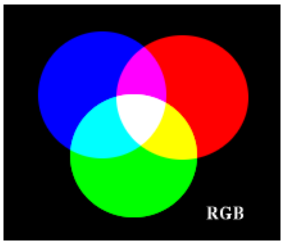
Color Models

If you want to work seriously with digital imaging and/or PrePress, you need to understand color - how it works in real life and how it works in your computer

COLOR MODELS

Color models are used to classify and standardize color. In this chapter we're going to discuss a few of them:

RGB

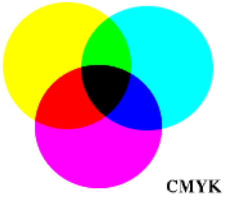


RGB stands for the three colors used in this system - **Red**, **Green** and **Blue**. Perhaps an "**L**" for **Light** should be included as well, because this color model is very much based on light. Imagine three different spotlights, red, blue and green, directed at the same spot on a white screen. Because each spotlight adds more light, the resulting color of two spots will be brighter than just one.

Where all three spots meet, the color is at maximum brightness - white. As you may have guessed, the RGB model is used in a television or computer **monitor**. The colored spots of a TV screen emit three colors, and the sum of these colors determines the impression to the eye. If the color spots shine with equal strength, the visual impression is white or gray.

This is called an **Additive** color model, because light is added to light, which results in brighter colors. Each color in the RGB system has a value for Red, Green and Blue. This value goes from 0 to 255, where 0 for all three colors equals black, and 255 for all equals white. Thinking of the spotlights this is quite logical - no light, or weak light means black or dark color, and full light from all three must result in a strong white light. This means that you can get more than 16 million colors (**TrueColor**) because 256*256*256 = 16.777.216, but you can only get 256 shades of gray. If you only have an 8-bit output to your color monitor, this is of course academic, because you'll never be able to see more than 256 colors anyway.

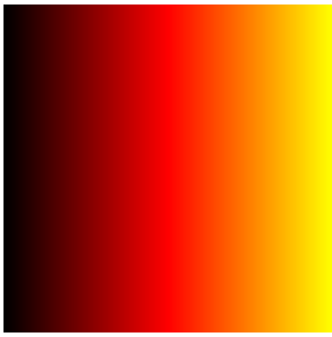
CMYK



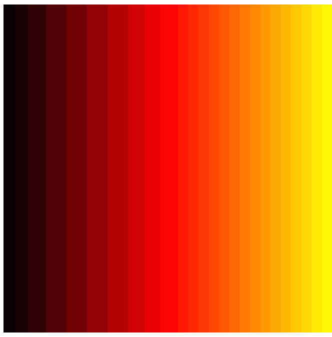
The other important color model you need to understand is called **CMYK**. CMYK stands for **Cyan**, **Magenta**, **Yellow** and **Black** (**K** for **Key** color). These colors are sometimes called **process** colors, because you use them in four-color process printing. If you have a color printer, you know that the toner in the machine consists of these primary colors.

Primary colors mean that all other colors can be created by mixing these colors together. Cyan, magenta and yellow are theoretically all you need, but to make a print look sharp and crisp you also use a black plate in the printing press. This is called a **Subtractive** color model, because the process ink pigments ”subtract” light when mixed, or absorb certain colors and reflect others (for your eye to see). Naturally, red, green and blue are the primary colors in the RGB color space.

INDEXED



RGB Image

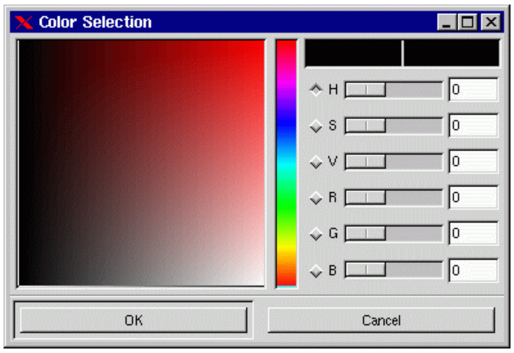


Indexed Image

Indexed, or mapped color doesn’t have RGB channels. It works with a fixed color value for each pixel. Each color is put in a specific **color table** or **color map** (containing 256 or less colors if the image is in GIF format) which comes with the indexed file, and this color table is then used to map color to pixels in the host application. Indexed files contain less data and therefore use less disk space. If you are designing for the web, you’ll have to consider that many computers can’t display 24-bit color. There are a lot of people out there who are limited to 256 colors on their monitor. This means that your full-color image will be indexed on their screen to a measly 256 colors to represent your wonderful ”millions of colors” image. It may well come out looking bad, and it still takes as long to download.

There are two kinds of image file formats you can use on the web - **JPEG** and **GIF**. There are advantages and disadvantages to both formats, and I’m not going to get into the big discussion about GIF vs. JPEG here. The important thing to know is that GIF is indexed and JPEG isn’t. There are many reasons to use GIF:s, the most important being the small file size of indexed images. Also remember that if you assign a **WWW-indexed palette** to a GIF you know that ”what you see is what you’ll get”. Your website will always look the same in **Netscape**, whether you have an 8, 16 or 24-bit output. Word of warning: ***Never ever index your image before you’re finished with it.*** Always keep an RGB copy of your original before converting to Indexed color. The reason should be obvious - most of the image information is lost forever as soon as you convert it, and you can’t get it back by changing to RGB again. You also can’t work with (most) filters on an indexed image.

HSV

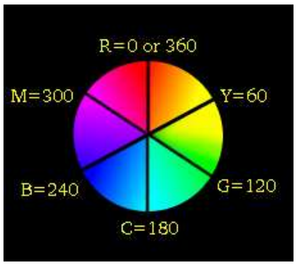


H 0
S 0
V 0
R 0
G 0
B 0

HSV color stands for **Hue**, **Saturation** and **Value**. This color model is best understood if you open the **Color Selection** dialog and look at the three top slide bars marked H, S and V.

The HSV color model is used in the **Image** Dialog for **Colors** and **Channel Ops**. It is also used in certain **filters** and **Modes**.

HUE



Hue describes where the color lies along the **spectrum** - i.e. what ”rainbow color” it is; red, orange, indigo or green. As in the rainbow, the starting and ending color is red. Hue can be described as a color circle with red at 0 degrees, yellow at 60 degrees clockwise, continuing with green, cyan, blue, magenta and red again at 360 degrees.

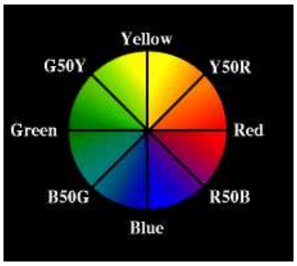
SATURATION

Saturation is about how **pure** or ”loud” the color is. The saturation value goes from 0 (grayscale) to 100 (max loudness), where a low value indicates a neutral, dull color, and a high value means a strong, pure color.

VALUE

Value is all about **brightness**. Zero for Value means totally black, and 100 is the brightest Value a color can have. Max value doesn’t mean white though, (unless saturation is zero)- it is simply the brightest value a color can have with a certain saturation.

NCS



NCS, or the *Natural Color System*, is the system used by architects and interior decorators. This is also the system they use at the chemist's when you buy paint for your house etc. (if you live in the U.S you'll probably use the Munsell-system instead). The NCS color model is based on the six **elementary** colors; **Yellow**, **Red**, **Blue**, **Green**, **Black** and **White**.

All colors can be described by their likeness to the elementary colors. The elementary colors and the color scale between them form a **3-dimensional body**. From this color body you can derive a **color circle** and **color triangles**. In the color circle (horizontal section of the color body), the colors are arranged according to their position in the spectrum, evenly distributed between the four basic colors Yellow, Red, Blue and Green.

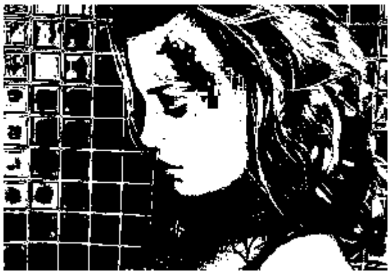
For every color in the circle there is a vertical section, showing the different shades of a color, meaning **Whiteness**, **Blackness** and **Colorfulness** (saturation). In this way you can name a color: `Y70R s70c20`, meaning a dark brown orange color with 70% red in it, s=70 means it has 70% blackness and c=20 that it has 20% colorfulness.

Even if you don't deal with this system on your computer, it is important that you understand it, because it is the single most used color system for professional designers and architects (At least in many European countries). If you get an assignment where the client want you to suggest different colors to a pattern, an object or a building (by using a program like Gimp or Photoshop) you must normally be prepared to relate to NCS values to the client, not RGB or HSV.

SPOT COLOR

Spot color or **Custom color** means a color in a commercial custom color system used for printing in color when you don't want 4-color process printing. You may want a non-rasterized smooth silver gray for text on a poster; and then you have to specify a predefined spot color. PANTONE and TRUEMATCH are the most common custom color systems used by professional printers. Read more about spot color in chapter 18.

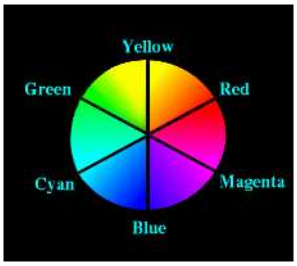
GRAYSCALE AND LINE ART



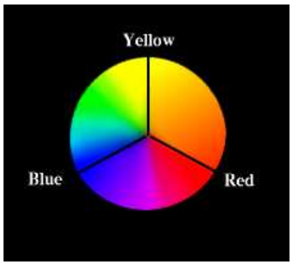
It is pretty obvious what a **grayscale** image is. It is an image consisting solely of shades of gray (max 256). This means that grayscales have a small file size compared to **RGB** files, and that they can convey more detail than **indexed** color images. **Line Art** is the most simple, and smallest bitmap file there is. A Line Art image consists of white or black pixels alone. They are extremely simple, but can be quite expressive. You can either scan in Line Art or use the **Threshold** or **Posterize** command in the **Image/Colors** dialog.

COMPLEMENTARY OR INVERTED COLORS

The definition of **complementary** colors are two colors which produce a neutral gray when mixed together. Complimentary colors in real life are not the same as on your monitor. Most people are taught as children that mixing yellow and blue results in green. This is perfectly true if you're talking about inks, oil paint and pigments, but it doesn't apply to the RGB color model. Try painting yellow in a layer on a blue background and watch it in 50% opacity - no green appears, just a neutral gray. Do the same with watercolor or oil paint and you'll definitely get a greenish color! The best way to understand this is to look at the **RGB color circle**. In the circle model the complimentary colors are found opposite each other. In the RGB color system the opposite of red is cyan, the opposite of yellow is blue and so forth. Look at it this way; the **inverted** or complimentary value equals 255 minus the original value in the RGB model.



RGB Model



Natural Model

This doesn't work with pigments in real paint. Here you must use another color circle. In the **Natural color circle** you can see that the complimentary color of yellow isn't blue but violet, while the complimentary color of blue is orange. Water color artists often work by adding a complimentary color to neutralize an overstrong background without dulling it, and why do you think some old ladies have "blue" hair? They are just trying to neutralize unwanted red/yellow shades in their gray hair by using a dye with the complimentary color - which is blue or violet.

chapter 13

Pre-press and color in Gimp

This chapter will help you prepare your Gimp images for the printing press. You will also learn some simple color calibration.

WHAT IS PREPRESS?

When people think about **prepress**, it's mostly a question of bringing their image or book to the local printer so it can go to press. The digital prepress business has for a long period been dominated by a company with an apple in their logo. Programs for prepress has also traditionally been dominated by companies like Adobe, Corel, Macromedia etc., and there are often tight bonds between **graphics software** and **printing hardware**. This makes it quite easy for the average Mac user to create a digital design and send it straight off to prepress.

Since the market has been dominated by Mac for so long, most of the scanners, printers and monitors come with drivers and calibration tools adjusted and written for Mac. The only thing the graphic designer has to do is to get a **printing profile** from the ”print man”, put it in the program, and everything is set as it should be. People can argue about whether this is the right way to go, or whether the user should have more control over settings or parameters, but using preset Mac profiles is certainly very handy.

When you meet with prepress people for the first time, you'll come across a lot of buzzwords like **lpi**, **dpi**, **ppi**, **Pantone** colors etc. You'll also have to solve the problem of how to make your Gimp images available for Mac users and Mac print shops, and how large image files should be transferred.

We will try to answer these questions, and sort out some of the buzzwords. And relax, you really don't need a Mac or a Windows box to prepare for prepress, when there is such a thing as Gimp and UNIX...

PRINTING FROM GIMP

Compared to Photoshop, Gimp is not as intuitive when it comes to prepress issues. Gimp is more focused on Web design, and you always see the image size as the websurfer would. Gimp has no dialogs where you can put given **printing values** before your start creating your design. You simply have to use your good sense, and a few calculations. Don't worry, it's not very difficult if you just plan ahead a little.

If you want to **adjust/calibrate** your scanner, monitor etc. this has to be done by hand. Does it sound complicated? Maybe, but on the other hand you'll probably understand a lot more about prepress when you've read this chapter, and you shouldn't have a hard time setting up your own prepress strategy with Gimp.

FILE FORMATS FOR PRINTING

There are a lot of image file formats, but as far as most professional printers are concerned there are only two: **EPS** and **TIFF** (even if many printshops can manage JPEG and compressed EPS).

The other option is of course to import the image file to a **lay-out program**. Most print shops that accept PC-files, support FrameMaker, Illustrator and Corel Draw.

If your local print shop can manage **Encapsulated Postscript files**, then go for it. EPS is device/platform independent, and you can be sure that the outcome will look just like the image you created, assuming of course that you have calibrated your monitor and that you're printing to a **PostScript** printer. If you go for EPS, first make sure that your print shop can manage **color adjustment** in an EPS file, and that they are capable to **preview** it. If they don't, you have to be sure that all settings and color in your file is perfect from the start, because then the printer won't be able to fix it for you afterwards.

If it is a large print job, you also need to know that they can **rearrange** the pages in the postscript file. Otherwise, you may have to pay for two printing plates instead of just one. Gimp has a powerful **postscript file saver** where you can specify paper size, resolution as well as type of postscript. To create an EPS file, just check that option in the `File/Save As PS` dialog. Most print shops can take plain postscript files, so this may also be a option for you, but you still have to check the same things as with EPS. Don't use the **Print** command in the File menu to **print to file**, because the print tool is just for printing a proof at home, so you or your client will get an idea of what you will get at the print house.

If your printer doesn't support EPS/PS then you have to save your image in an ordinary image format like TIFF or JPEG, but you must be aware that some image formats differ in PC and Mac environments, TIFF does for example. The advantage of using TIFF is that you can make a high quality proof even on a non-postscript printer. Also, TIFF is a good choice if your image has a white background (EPS sometimes has a tendency to produce a weak tint in large white areas).

If you ship your images like ordinary image files you must remember to tell your printer what **size** you want the images to be printed in. It's important that you inform them of this, because they normally expect **plain image** files to be created in Photoshop. In Photoshop the image size/resolution is present in the image file, but that's not the case with Gimp, which only comes (as yet) in one resolution (72 ppi or monitor res).

RESOLUTION AND IMAGE SIZE

Gimp images have only one **resolution**, and that's 72ppi (pixels/inch) which is the **screen** resolution. Obviously, this is ideal for web-publishing, but a 72 ppi resolution is far to low for any kind of serious printing. Suppose your printer tells you that to make your 50x60 inch poster look good, you have to use a minimum resolution of 200 ppi. Well, then you have to make a larger image in Gimp, and ”scale” it down when you save it as a .ps/.eps file, or the printer will do it when they import it to one of their programs. In the `File/New` dialog box, you always specify the image size in **pixels**, so it's really quite simple. All you have to do is to multiply the image width and height by 200 and set those values in the dialog box.

50x200 is 10.000pixels and 60x200 is 12.000pixels. The equation is simple:

Gimp image size = wanted ppi x wanted height or width in inches

The bottom line is to *adjust the image size before you start making the image*. If you need a high resolution for your print job, your images will get an enormous size on screen. The good news is that Gimp is smart enough to adjust the canvas size so it will never be bigger than your display. So please, before you start making your big poster, check with your printer, and calculate the image size you need to produce a nice printed output.

So what if I haven't done the necessary calculations before I created the image, or if the printer tells me that my image resolution is too low? There is a way out of this dilemma. You can scale the image to a certain amount, because Gimp can manage interpolation. **Interpolation** means that when you enlarge an image by scaling it, Gimp compares neighbor pixels, makes an assessment of their color and calculates an intermediate color for the new pixels. This works just fine if you only have to resize the image a little bit, but don't try it for scaling the image to the double size. That will only make the image look blurred and fuzzy at the edges. To get better (but slower) interpolation check cubic **interpolation** in the Gimp **preference** dialog.

PREPARING FOR THE PRESS

DPI, LPI, PPI AND SCANNING RESOLUTION

If you've never done a high-end print job before, the first thing you should do is to make some phone calls to your local print shops and ask a few questions. Let us give you an example: Say that you want to print a 22x34"poster, then you must first make sure that the print shop can manage to print in that size. Furthermore, you have to check that they can print in the resolution that you want.

Often people think that a high resolution is the better choice, but that's not always true. For a poster, you'll have to take **viewing distance** into consideration. For a large image, like a commercial poster, the beholder will observe it from a certain distance and will never see the coarse halftone dots unless he or she gets very close. For that reason, you don't have to print a poster in a very high resolution. On the other hand if you're making a cover for a monthly fashion magazine, then you'll need a very high resolution indeed.

lpi& dpi

You have probably often heard the expression **dpi** (**dots per inch**) when it comes to home or office printer resolution - a 600 dpi laser printer, or a 300 dpi inkjet printer for example. However, the term ”dpi” is not as commonly used in professional printing.

The main term here, and one you'll most certainly come across as soon as you enter the print shop is **lpi**. Lpi stands for **lines per inch**, and is another term for **screening**, or the fine **halftone** pattern you see when you look closely at a picture in a newspaper. The higher the lpi value, the more (and smaller) halftone dots are squeezed in per inch.

Each halftone dot is made up of very small ”dpi dots”. Dpi can simply be described as the maximum number of these microscopic dots the printer can manage to print per inch. Inexperienced users tend to think that 600 dpi in a printer is the same as 600 ppi in an image file, which can mislead them horribly. It takes a lot of dpi to represent a ppi.

For a relation dpi/lpi see the table at the end of the chapter. This table can serve as a guide for what lpi you need for producing different types of printed material, or more exactly, which output device can manage an lpi suitable for your needs.

ppi & Scan resolution

When you have chosen an appropriate lpi for your print job, you can start calculate what **ppi** (**pixels per inch**) resolution you'll need in your digital image.

- The rule of thumb is that ***image ppi = 1,6 x lpi.*** This equation is of course not exact, it all depends on what kind of quality you want (see table at the end of this chapter). We've chosen a 150 ppi resolution for our poster, assuming of course that the image file and the printed output have a 1:1 ratio. If you want to make a large print from a small scanned image, you'll need a much higher resolution. If you scan an image which has a natural size of say, 2 x 2 inches, and you want the printed size to be 5 x 5", you need a scan resolution of 375 dpi to get a 150 ppi resolution in the enlarged image (5/2 x 150).
- Note that **scan dpi** is not the same as printer dpi, so don't think that you have to scan at 600 dpi to print to a 600 dpi laser printer! Scan resolution is measured in "scan dpi", but this "scan dpi" is really the same thing as ppi!
- The calculation is: ***scan resolution = wanted ppi * (wanted size/real size)***

Now when we have all the basic facts, we can make our phone calls to the local print shops asking if they can produce our poster with an lpi around 100.

Read more

Want to learn more about this? Check out the in-depth information about lpi, dpi, ppi and screening later in this chapter.

AT THE PRINT SHOP

COLOR

or why the colors that looked so good on your monitor don't match the color of the output.

Here's the hard part of printing; *never give your image file to the printer and simply ask them to print it.* If you try this, you can more or less take for granted that the colors will be screwed up. **Color calibration** of your computer environment is one of the hardest parts when it comes to image production.

Make it simple

The most simple solution to this problem is to make a small **proof** at home with a color inkjet. If the proof's color is correct as far as you're concerned, bring this example to the printer and ask them to adjust the color so the printed outcome will look as your proof. We will describe how to calibrate scanners, monitors etc. later in this chapter, but remember that it's always a good idea to bring a proof or dummy as a reference. Then there will be no problems to demand a reprint if the colors of the printed product should be incorrect.

Spot color

If you only can afford or want to use one color plate in the printed product, there is another way of getting past this problem, and that's by using **spot colors** from a commercial color system like **PANTONE**. Gimp doesn't support spot colors natively, so if you want to use custom colors with Gimp you'll have to buy a PANTONE color map. Just decide which spot color you want to use, and give a **grayscale file** and a **spot color specification** to your printer. If you want to use more than one spot color and you don't have a spot color separation program, you must give the printer an appropriate grayscale file for each color plate.

HOW TO TRANSFER IMAGES TO THE PRINT SHOP

Removable drives

The most common form of **transporting** images to a printer is to use a **SyQuest** removable drive. This device is a de facto standard in the Mac world. However, in the last three years new removable drives been introduced such as **jaz** and **zip** drives from **Iomega** and **Ezdrive** from SyQuest.

The most common removable drive in the PC world is now the zip drive which is capable of storing 100MB of data. To decide what drive you should buy, it's wise to phone around to your local printers asking what kind of drives they support or consider supporting, and make a list of it. Based on this information you can make a conclusion of what kind of drive you should invest in.

Based on our own experience, we would suggest a **scsi Zip drive**. It's fast, the old SyQuest drives are expensive and on their way out. The new drives from SyQuest have not been so successful as the Iomega ones, and most printers seem to support or are willing to support Zip drives.

Internet and BBS

Over the **Internet**: It's not uncommon that print shops have some sort of Internet connection, or a **BBS**. If they have an Internet connection, it's also likely that they can support file upload by the **FTP** protocol. The only thing you have to do then is to get an ftp program and their address so you can upload files to their server.

If they have a **BBS**, then you need a **modem**, which is probably the case if you have a dialup Internet connection. You also have to have a **terminal program** that supports their BBS. In order to figure out what kind of program you need, you'll just have to ask them. There are several modem/terminal programs available for UNIX/Linux such as **Kermit**, **Seyon** etc.

A warning should be in place here, since it can be rather expensive to use this kind of file transfer, at least in Europe where even local calls can be expensive. We don't suggest uploading files over 20MB if you only have an ordinary modem. Some printers also charge you for each MB you upload, making it even more expensive.

Email

Email the file? We don't recommend it, because when you code your attachment it makes the file bigger and this makes the "upload" time last longer. It's also not uncommon for files to get damaged when you send them as email.

Filesystem format

What kind of **filesystem format**? Nowadays nearly all print shops support the **Fat** (msdos filesystem/disks) filesystem even if they are running their software in a Mac environment. If they don't support fat, the only chance is Internet or BBS.

However, we think you should seriously consider turning to another printer who supports Fat. If they do support Fat, well then everything is ok, and you can just create a Fat filesystem on your removable drive (Zip and Jaz come preformatted if you buy a PC disk).

If you are running a **Linux** system, there are some nice tools which makes it real easy to manage Zip and Jaz drives through a GUI. There are also some nice howto:s on both Iomega and SyQuest drives if you are running Linux.

SCANNING UNDER UNIX/LINUX

SANE

Since we are working with Gimp, the natural choice is to use **SANE** (scanner access now easy). SANE can be run as a stand alone program, but it can also act as an **extension** to Gimp. This enables you to open SANE from Gimp, and scan straight into Gimp. It's like having a **Twain** interface under Photoshop or similar program in a Mac or Win environment.

SANE supports a wide range of scanners on nearly all major Unix/Linux platforms. It also has a nice interface where you can adjust nearly everything, like **gamma**, c**olor curves**, **resolution** etc. It may not be perfect at this time, but remember that SANE is only in v0.72 and will certainly get even more intuitive and powerful when it reaches v1.0. If you want to know more about SANE, please read SUM (sane user manual or the Readme and Install files that comes with Sane) about installing and using SANE.

OTHER SCANNER PROGRAMS

Naturally, there are other freeware scanner programs. Most of them are not as GUI'fied as Sane, but if you have a **mustek scanner** then you can try a program called **Tkscan**. If you have a scanner which isn't supported by Sane, then it's always a good idea to take a look at `ftp://sunsite.unc.edu` and see if you can find another program that supports your scanner. It's better to have a plain **CLI** program than no scanning program at all.

COMMERCIAL SCANNER PROGRAMS

There are of course a lot of commercial scanner programs. **XVScan** is a very nice program that supports **HPscanners**, and it has a very moderate prize tag. It's a good scanner program that is integrated with **XV** and has all the functions of a modern scanner program. Most of the older pictures in this manual have been scanned with XVscan, and we are quite satisfied with it. Nowadays we use Sane and a **Umax** scanner and we're equally satisfied with that solution.

In the professional class you can choose from scanner programs from **Caldera** (in France `www.caldera.fr`) and **Mentalix** (`www.mentalix.com`). Both companies offer everything from a scanner interface to a total graphic studio with programs that are Gimp-like but with a focus on prepress.

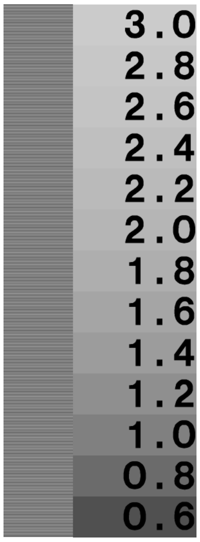
If you're considering serious prepress in a Unix/Linux environment, then you may want to buy one of those programs. There are Light versions (not all functions and with a lower pricetag) of the programs from Mentalix (if you are working with Linux).

We will try to get a copy of the programs so we can give you a full review of the capabilities here, but for now the only info is their websites. There are most likely other providers of scanner programs in the Unix community, but we don't know of anyone else that also provide it to Linux.

CALIBRATION

Before you calibrate anything that has to do with your monitor, let it be turned on and warmed up for at least 30 minutes. Also make sure that you have a normal light in your room (not too dark or too light).

GAMMA CALIBRATION



Every monitor has a **gamma** value. We will not go into the background of gamma, but when you scan an image it has a gamma value of 1 (if you haven't altered it) and this may not be suitable for your monitor. In order to find out your monitors gamma value check out two files at `ftp.gimp.org/pub/gimp/manual/` gamma.tif and gamma.gif. When you have downloaded the file (gif or preferably tif), you must find out which of the squares to the right that has the same brightness as the correspondent strip to the left. Now you have to open the image in a program that lets you alter the gamma by value, e.g with **Display** from **Imagemagic**, you do the following:

- `display gamma.tif <enter>`
- right mouse button and down to gamma
- enter your gamma value and OK

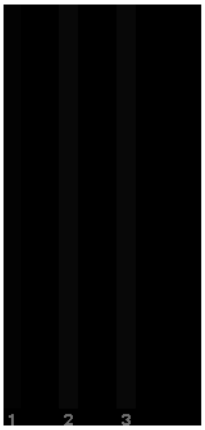
If your gamma is okay, the 1.0 square should now have the same brightness as the correspondent strip to the left. When you have figured out your monitor's gamma value you can apply this value when you scan images, so they will display correctly when it comes to gamma. It won't affect color mismatch, but it's much better than no correction at all.

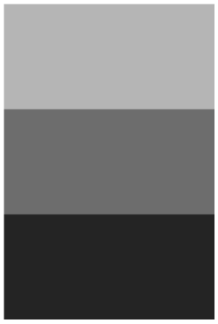
BLACK LEVEL AND WHITE LEVEL ADJUSTMENT

Besides the gamma value, the **black level** is one of the most important things to adjust. Here's how to do it. First download the image `black_lev.gif` and `white_lev.gif` from `ftp.gimp.org/pub/manual/` and load it into Gimp. Adjust your screen until the image covers most of it (do not adjust image size, adjust the screen resolution).

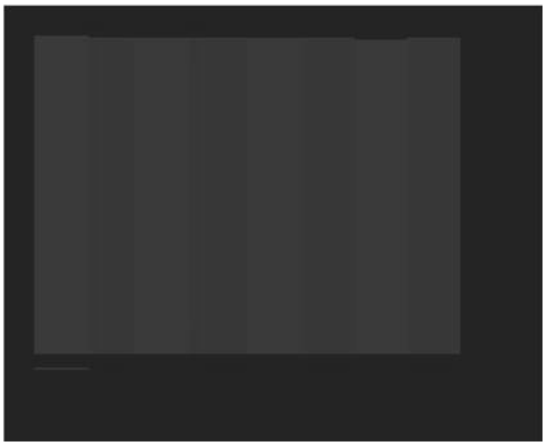
- Load the `black_lev.gif`
- Bring up the brightness control
- Adjust (turn up) the brightness, until the gray stripes marked 1, 2 and 3 are visible
- Now gently turn down the brightness until strip 1 fades away.
- Carefully bring the brightness up to the precise moment when strip 1 is visible again

-





- Load `white_lev.gif`
- Adjust your contrast so the gray bars in the middle square appear of even density
- Now re-check the black level because black and white level are interdependent
- For the same reason re-check the white level again
- Now you’re hopefully finished after a few adjustments and readjustments



COLOR CALIBRATION

There is no easy way of color calibration, so you have to buy some sort of **color calibration kit**. When you buy your kit, make sure that it is platform independent (or that it's made for your UNIX version).

Plain

Simple color calibration kits are usually made up of *a printed color image*, a *color image slide* and a *color image file*. You load the color image from the file, compare it with the printed image, and adjust your **monitor** until the colors match (For more specific information, read the leaflet which comes with the kit).

To calibrate your **scanner** you scan the slide or the printed image and compare it with the image file. To calibrate your **printer**, you compare the output from your printer with the printed image from the kit, and adjust it accordingly.

This is a very time consuming task, but before you have done it you can't really trust your system at all when it comes to color fidelity. Note that even if you have made all of these corrections, it's still not bulletproof, since it's a human correction and humans are always subjective when it comes to color representation.

CMS

If absolute color correction is a high priority to you, then you have to get a **CMS** (Color Management System) There are CMS systems available for Unix, and Caldera (in France `www.caldera.fr`) sells CMS- like systems. Mentalix (www.mentalix.com) also provides CMS. These products work for most Unix systems, including Linux. There are other providers in the Unix community, but we don't know of anyone else that also provides CMS to Linux.

Color correction kit's are expensive, and if we are talking about CMS systems, you should be prepared to pay big time money. However, for a commercial company dealing with advanced image manipulation and prepress, the best thing would certainly be a real CMS system.

Frozenriver's plain color calibration

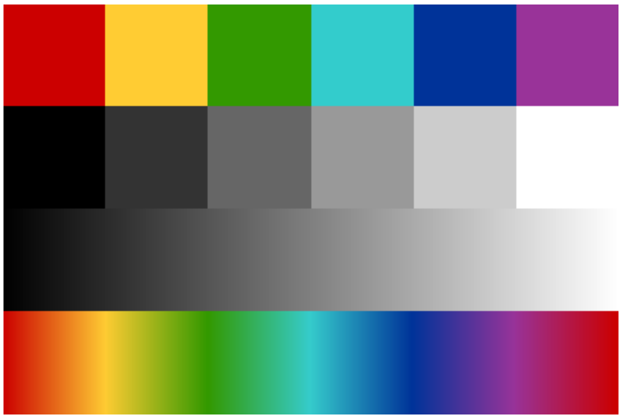
We have put together a plain and easy color calibration kit. You can't use it if you are working with colors professionally, but it's good enough for private and semi-professional use, and it's definitely better than no color calibration at all.

If you want to use this kit, you must have *a printed copy of the manual*, and the printed copy must be *approved by us*. As we've just explained, you need a (high quality) printed or photographic color image for calibration, so you can't use the kit if you have a web, .ps or .pdf edition of the manual.

If you have a printed and approved copy of the manual, you can start by downloading the color calibration file **colorcal.tif** at `ftp://ftp.gimp.org/pub/manual`. and make sure that your X-display (i.e X-server) is running in at least 16-bit color mode.

- Load the file to Gimp
- Take the color image that comes with the manual, and place it beside the monitor in a room with normal working light. The light must be natural, don't use cold, white fluorescent tubes as light source
- Set the monitor's color temperature to 5000 degrees Kelvin as a start (if your monitor doesn't have this feature, skip it and move to the next step)
- Start comparing the colors and make the necessary adjustments in the different RGB channels
- When you have finished calibrating the colors (when they are equal to the colors in the test image, or as close as you can manage) you have to re-check the white and black levels in your monitor
- If you needed to adjust the white or black levels, re-check the colors and adjust again if necessary
- When you have checked, and re-checked and everything is okay, your monitor is now calibrated and you can trust it
- To calibrate your scanner: Scan the calibration photo, and compare the scanned image with the colorcal.tif image that you downloaded.
- Depending on your scanner program, you have to adjust the color curves to correct the colors in the scan.
- If you want to, you can try to adjust the printer driver, but don't expect to get a print that will look exactly like the calibration photo. A homeoffice printer can never get close to photographic output, or the output of a high quality photosetter. The important thing is that you can trust your scanner and monitor to produce colors that look the same when you print them at the print shop.
- You are now hopefully finished with your calibration. Please remember to re-calibrate the computer system regularly.

A word about the **calibration photo**. This photo is produced to represent the output of a high quality image setter. As you can see, to make sure the colors are **CMYK-legal**, the colors used here are a bit dull looking compared to some of the clear, fluorescent colors you can get on your RBG screen. The reason for this is that many of the bright colors in an **additive** color system (like in your monitor) can't be truthfully represented with CMYK inks. More about this later in this chapter.



Poor man's color calibration

If you can't afford or get hold of a real calibration kit, here's a cheap and easy (but not very exact) method. Simple color calibration is better than none, just don't think that your system is properly calibrated after this. It helps, but it can never compare to a professional color calibration.

The key is that **big brand names** always use a specific color in the product name or **logotype**. Volvo uses a very a special "Volvo blue" color, Marlboro cigarettes has its Marlboro red, Sun always has their own particular color in the Sun logotype etc. These colors can often be downloaded from the web, and the printed color can be picked up by a phone call or a walk to their sales office.

Now you have the color on file as well as on print, and you can start calibrate just like you would with a plain color calibration kit.

Once more, don't expect miracles from using this method. It is only as a rough approximation of real color calibration.

Why don't the colors look like they should, even if I have calibrated the system?

When you work at your computer monitor you work in **RGB**-mode, and when you print your file at a print shop or at home you work in **CMYK**-mode. All RGB colors can't be transferred to a CMYK color (for more info read the in-depth paragraph later in this chapter).

Since Gimp is more focused on Web creation than prepress, you don't have a **CMYK preview** where you can find out whether your color image can be printed without some loss in the RGB information.

Our advice is to be careful when using highly saturated colors in your image, because they probably aren't CMYK safe, and always make a proof before giving the file to the print shop.

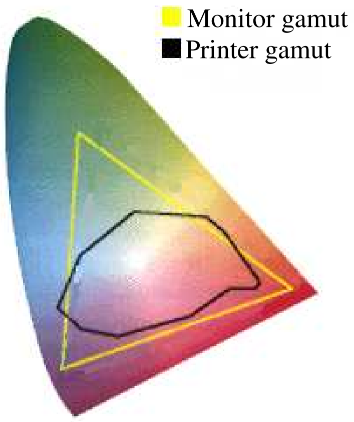
If you want to use colors that you suspect aren't CMYK-legal, choose a high quality printer to print a proof, because a home office printer doesn't have the capacity to make a good translation from RGB to CMYK.

IN-DEPTH INFORMATION

COLOR

Before reading any further, have a look in the chapter 12 to understand different color models.

COLOR MANAGEMENT



Why don't the colors on my monitor match the output on my printer?

RGB and CMYK

This is a common question when you are talking to color device end users. As we've discussed before, the answer lies in the fact that a monitor is based on the RGB color system and a printer is based on the CMYK color system. Because monitors and printers use different **color models** it is impossible to print the monitor data directly to the printer. When you print your RGB image it will automatically be converted to CMYK.

Gamut

Gamut is the total number of colors a device can produce. The human eye has a higher gamut than a 24-bit color monitor; i.e the human eye can perceive more colors than the monitor can produce. The gamut of a monitor is higher than the gamut of your color printer.

You now understand that all colors in an RGB image can't be represented in a CMYK image. An RGB color that can't be represented in CMYK will therefore be converted to the nearest CMYK color.

How well this is done depends on the software doing it (i.e the printer driver which in most Linux installation is **Ghostscript**). Gimp also provides you with a **built-in printer driver**. If you decide to use it, the conversion quality will depend on the quality of Gimp's native printer driver.

CMS

At this time **CMS** systems make their entrance. A CMS system makes the appropriate **gamut mapping** between all the different devices in the system. **Profiles** of different devices have been made by color scientists using special equipment. This profile is based on factory defaults. When a device starts aging, the profile is no longer correct, and a recalibration is a must (even if the device is brand new, variation in manufacturing quality often makes a recalibration necessary)

Profiles

All of these **preset profiles** are used by the CMS system as the image travels from the scanner to the monitor and finally to the printer. The CMS system often uses an independent **color space** when the images are transmitted from one device to another. This results in consistent colors, even if the devices have different color systems. Sometimes its impossible to keep the colors consistent, e.g you can't reproduce a highly saturated monitor image with CMYK ink, wax or toner printer.

The printer gamut area is simply to small to cover highly saturated colors (see picture). This type of RGB color is not **CMYK safe**. When a non-CMYK safe color has to be reproduced, the system has to do a gamut mapping which selects the nearest reproducible color.

High quality CMS systems often provide you with the option to select a certain type of rendering for the gamut mapping, because there is a vast difference between reproducing images and reproducing business graphics.

RESOLUTION

Resolution can be described as image quality in images which are made up of pixels. A digital image appears to be of good quality when you can't see the individual **pixels** that it's built of. If the resolution drops (i.e the image is magnified) individual pixels will be visible to the eye, and the image will be considered ”jaggy” or of low quality. In other words, image quality is based on the **number of pixels** it's made up of, and the **size** of the image. These two factors determine the resolution.

Printed material, like a newspaper looks good when you view/read it from a reading distance. If you pick up a magnifying glass and look at it, you will see the halftone dots that the images are made of. Therefore the resolution of printed material is determined by the **distance** from which you look at it, and by the need for smooth image **quality**.

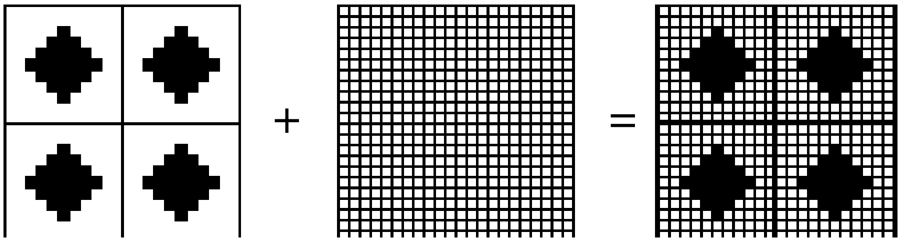
It's important that the resolution of the print and the resolution of the digital image are adapted to each other. If the image resolution is higher than the output machine can handle, the **ripper** (the device that creates the halftone pattern) has to throw away a lot of information that it doesn't need.

The quality of the selection depends on the quality of the ripper, and it's always better to create the image with the proper resolution. This way, the ripper doesn't have to choose what information it should throw in the trashcan. On the other hand, if the image resolution is too low, the image pixels will be visible in the print. Even if you can't see the halftone dots, this will make the print look jaggy and cheap.

Lpi, dpi and screen frequencies

Screening is a common buzzword when it comes to printing. Imagesetters make a print based on **halftone screens**. These halftone screens are measured in **Lpi**, or lines per inch, and the ”resolution” or quality of an imagesetter or laser printer is often measured in **Dpi**, or dots per inch.

The halftone screen can be represented by a **grid**. In each grid square there is a **halftone cell**, capable of holding one **halftone dot**. Over the this grid, there is a superimposed grid called the **resolution grid**, with a high amount of grid squares for each cell. The amount of squares in the resolution grid determines the imagesetter's resolution in dpi.



Halftone grid

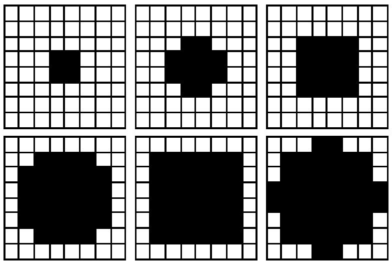
Resolution grid

Final grid system

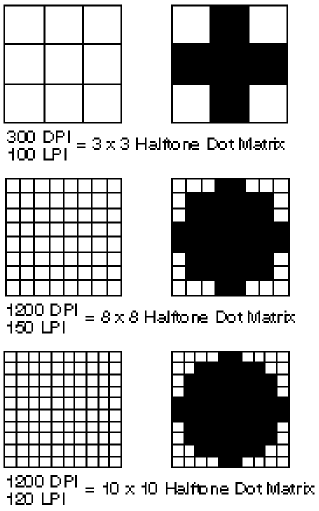
Halftone dots

If you look real close at a newspaper you will see that it's built up on different **halftone** dots (see picture X). A halftone dot can vary in size from very tiny to the full size of the halftone cell. The tiny ones represent a light gray"color" and the big ones represent a dark gray"color".

Each halftone dot consists of a number of smaller dots, and each square in the halftone screen is built up of a subgrid (**halftone cell matrix**) with a fix number of available squares to put the small "dpi" dots in, (see picture X). The natural conclusion is that image "roughness" depends on how large the halftone cells are, and the number of grayscales depend on how many small dots you can fit into the cell matrix.



Different sized halftone dots


300 DPI
100 LPI = 3 x 3 Halftone Dot Matrix
1200 DPI
150 LPI = 8 x 8 Halftone Dot Matrix
1200 DPI
120 LPI = 10 x 10 Halftone Dot Matrix

Different sized matrixes

A halftone matrix is measured in 1x1, 2x2, 3x3, 4x4, 5x5 squares etc. A 3x3 matrix is capable of holding 10 shades of gray, but a 5x5 is capable of holding 26 shades and a 8x8 matrix can of hold 65 and so on.

The number of squares in a halftone matrix can be calculated like this:

DPI/LPI = N x N Halftone matrix.

Therefore a 300 dpi printer combined with an lpi of 100 only produces a matrix that is 3x3 and contains 9 grayscales (not very impressive). An image that is built up of 10 shades of gray looks quite bad.

It would be better to combine the printer with a lpi of 60, which will give you 16 shades of gray (not good but better, it’s like a print in a cheap newspaper). Now, if we take a 600 dpi printer and set it to 100 lpi, we’ll get a 6x6 matrix capable of producing 35 levels of gray. So you see, Dpi isn’t all. To achieve the really high quality of 256 shades of gray with an lpi of 133 (magazine standard) you’ll need a 2400 dpi printer.

Why doesn’t my inkjet look like halftone screens?

FM Screening

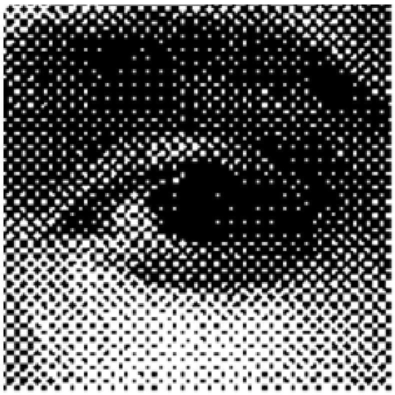
When we discussed screening in the paragraph above, we were talking about **halftone** screening, also referred to as **AM screening**. Even if it not totally correct, you can say that inkjet printers use **stochastic** screening, or in other words **FM screening**. While AM screening uses different sized dots in a fix grid, FM screening dots have exactly the same size, but the distance between dots varies.

Because stochastic screening is based on the dot **frequency**, it’s called Frequency Modulation screening. Halftone screening which is based on dot **size** or amplitude is called Amplitude Modulation screening. FM dots are usually 1 to 2% the size of halftone dots.

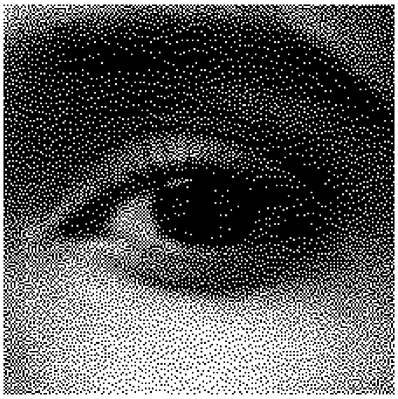
The advantage of using FM screening is that you can achieve a higher level of detail, even with a low res printer (600 dpi). There are also other advantages, like the fact that you don’t have to worry about **rosette** and **moire** patterns like with AM.

Also, you don’t need the same level of resolution in the images that you provide for printing. Usually half the resolution that you would have used for AM printing suffice for producing excellent results with FM printing.

Then why doesn't everybody use FM? Well, the biggest problem has been to get the"printing presses" to produce small enough dots. These problems are now minor, and nowadays it's quite common to print in FM.



AM screening



FM screening

TABLES

LPI TABLE

Document type	paper type	necessary lpi
High end advertisement and high end brochures, fine art books and fine art reproduction. High end magazines	Sheet-fed/coated	150 to 300 lpi median 200 lpi
Catalogs, monthly magazines, commercial grade advertisement, ordinary books	Heat set web/coated	100 to 150 lpi median 133 lpi
Newletters, forms and flyers	Sheet-fed/uncoated	100 to 133 lpi median of 100 lpi
Small magazines, catalogs, direct mail	Heat set web/uncoated	90 to 133 lpi median of 100 lpi
Newspaper supplements of high quality	Newspaper/coated	65 to 100 lpi median of 90 lpi
Newspaper supplements of ordinary quality	Newspaper/uncoated	65 to 100 lpi median of 65 lpi
Newspaper, low quality spare parts catalogs	Newspaper/newsprint	65 to 100 lpi median of 65 lpi

PRINTER TABLE

Printer resolution	Recommended lpi	Best choice/number of gray shades
2400 dpi	133 to 150 lpi	150/257
1200 dpi	90 to 110 lpi	100/145
600 dpi	60 to 80 lpi	75/65
300 dpi	40 to 55 lpi	53/33

IMAGE TABLE

Lpi	Recommended image ppi	Median
150	240 to 300	240
133	210 to 266	210
100	160 to 200	160

75	120 to 150	120
53	85 to 100	85

SHADES OF GRAY

X=” Screen frequency”, Y=”Printer resolution”, Z=”number of gray shades”

$$Z = \left(\frac{Y}{X}\right)^2 + 1$$

SCREENING MATRIX GEOMETRY

X=”X*X Halftone dot matrix”, Y=”Printer resolution”, Z=”Line screen”

$$X = \frac{Y}{Z}$$

part

V

Extend Gimp

- IMAGE MENU
- SELECTION MENU
- MODES
- LAYERS
- CHANNELS

chapter 14

Image Menu

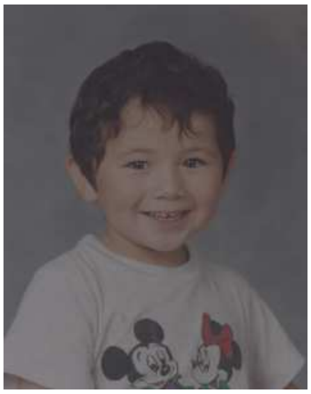
The Image menu offers some of the most useful image manipulating functions in Gimp. In this chapter we're going to discuss what you can do with Colors and Channel Ops.

COLORS

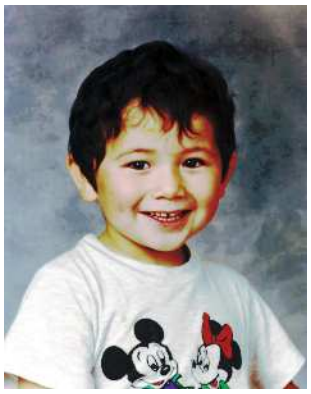
The first part of the Colors menu deals with how image pixels are mapped to different RGB values. The second part deals with color correction.

EQUALIZE

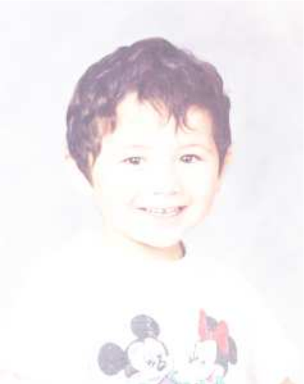
Equalize is often used to correct over- or under-exposed photos. This command finds the darkest and lightest pixels in the image, and sets the darkest value to black, and the lightest to white. The intermediate colors are then translated to the correspondent **histogram** values on the new scale. By doing this, it **equalizes** the image pixels on a wider scale or spectrum than before. The result is usually harder and much clearer, with more saturation and contrast but often less fine detail. *Equalize can be used to find ”hidden” colors in old, fading photos, because colors are also equalized.* If there is a weak shade of green somewhere in a seemingly solid blue area, equalize will find it and strengthen it, while the blue color will lose in intensity. Sometimes this behavior means that colors will look a bit unnatural after equalization, so you might have to correct it further by using the color correction tools. For adjusting old photos, it is sometimes better to use **Autostrech HSV**, **Contrast Autostrech** and **Normalize**. You’ll have to try all and see which is most suitable for your picture.



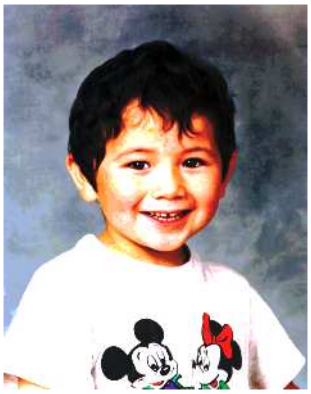
Dark input



Output after equalize

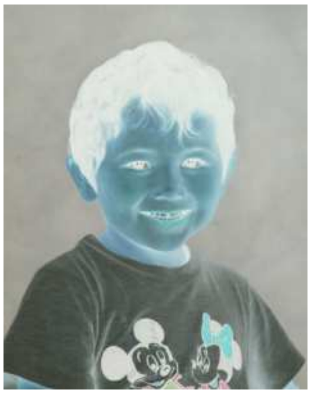


Light input



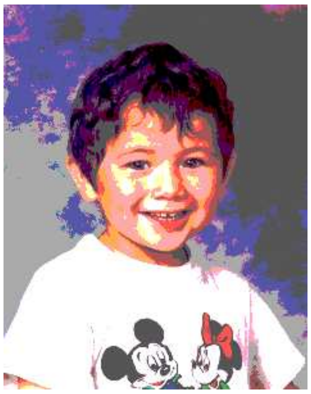
Output after equalize

INVERT



Invert creates a color negative of your image (or a positive of your negative). This is very useful if you have a **slide scanner**, because then you never have to worry about expensive developing. You need only develop the negative film, and then you can do your own developing in the computer. The calculations are quite simple; the inverted RGB value of a pixel is 255 minus the former channel value. *Read more about inverted or complimentary colors in chapter 12*

POSTERIZE

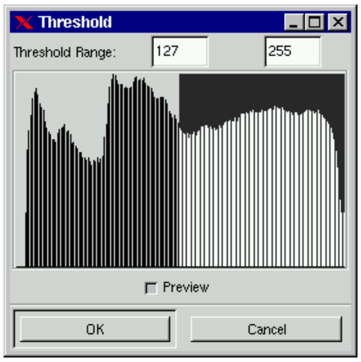


Posterize is a way of creating an ”indexed” image, where the colors are not derived from the image, but from the number of possible combinations the three RGB channels can produce. The available colors depend on what **Level** you set in the **Posterize dialog**.

For a grayscale image it’s easy. Level 0, 1 and 2 produce the minimum of colors possible - black and white only. For each level you get one more grayscale. Level 4 equals four shades of gray, level 16 equals 16 shades of gray etc.

In an RGB image, Level 4 equals four different shades in each color channel. This means that level 4 equals 4*4*4, which is 64 colors. Because those 64 colors have nothing to do with the original colors in the image, only a few of them will be used in the image. A posterized image will be a lot less representative, but (perhaps) more ”artistic” than an indexed image, where the chosen colors are taken from the image.

THRESHOLD



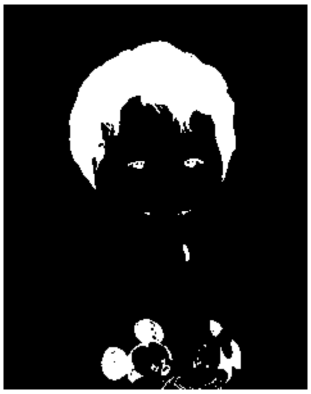
Threshold Range: 127 255

☐ Preview

The **Threshold** dialog displays a brightness histogram of the image. Each spike in the histogram represents an **intensity value** ranging from 1 to 255. The longer the spike, the more pixels with that particular value. In the preview you can see where those pixels are situated in the image.

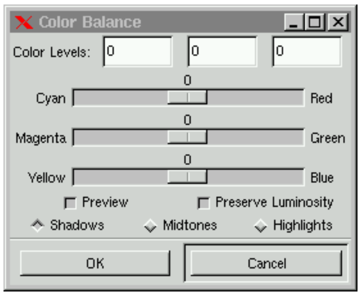
Clicking on a single spike will only display pixels with that particular value, while dragging from one spike to another, will display all pixels with a threshold range from the first value to the second.

Lineart



If you want to make a **Lineart** (Black and White) image, this is the command to use. It is also an excellent tool for **selecting by Value**! If you copy your image to an **Alpha Channel**, and use threshold on that, you can for example select all pixels dark enough to represent a certain shape, and use that to make a selection.

COLOR BALANCE



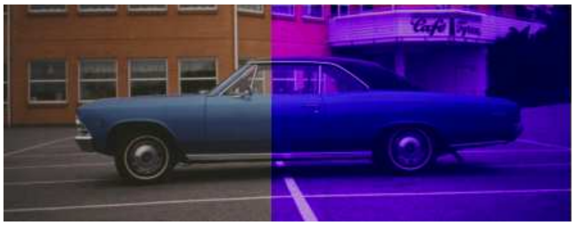
Color Levels: 0 0 0

Color Balance allows you to adjust the **color levels** in your image. Color Balance changes colors in the image, but not so drastically as **Hue-Saturation**. Use Color Balance when you want to make subtle changes in color.

In the dialog box you'll find three slide bars ranging from the three **RGB** colors to their complimentary colors (**CMY**). As you know if you have read the chapter about color, the sum of two complimentary colors is neutral gray. You can consider your current pixels ”neutral” and any change you apply will draw the pixel color either toward Red or Cyan, Green or Magenta, Blue or Yellow.

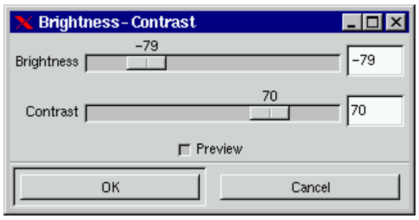
Note, that drawing all three slide bars against the CMY (additive) colors will darken your image, while drawing them toward the RGB (subtractive) colors will lighten the image.

There are also three buttons for **Highlights**, **Midtones** or **Shadows**. By pressing one of these buttons, you choose whether you want to change the darkest pixels, the medium pixels or the brightest pixels. The **Preserve Luminosity** button makes sure that the brightness value of your image doesn't change. By default this option is unchecked, the colors can get very unnatural if you insist on keeping the original values while changing the color balance.



Output from color balance

BRIGHTNESS-CONTRAST

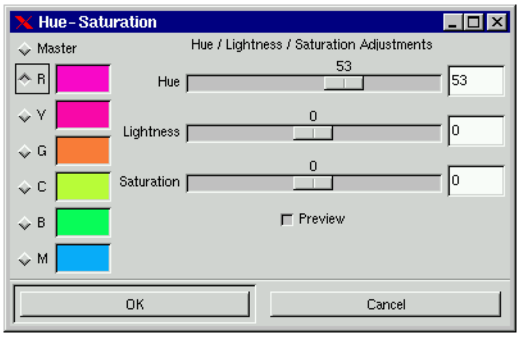

Brightness-Contrast
○ Preview
OK
Cancel

Brightness -79

Contrast 70

Brightness-Contrast is easy to understand. The zero values represent the current values of your image. From that point of departure, you can raise or lower the amount of **contrast** and **brightness**.

HUE-SATURATION


Hue-Saturation
Hue / Lightness / Saturation Adjustments
Master
R
Y
G
C
B
M
☐ Preview
OK
Cancel

Hue 53

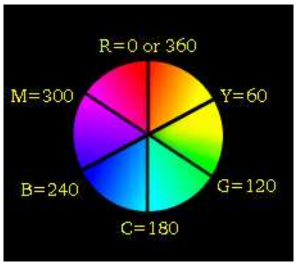
Lightness 0

Saturation 0

The **Hue-Saturation** dialog lets you adjust **Hue**, **Saturation** and **Lightness** (Value). It is important to understand that this option is entirely based on the **HSV** color model. *Read more about HSV in chapter 12.*

When you change **Hue** with the **Master** button checked, all pixels in the image or selection will change color according to how many degrees you have turned the HSV color circle. If you just want to change part of the spectrum, you can choose one of the color swatches.

Hints



You must remember once again, this is HSV - not RGB. This explains why checking the Yellow button doesn’t necessary change the yellow parts in your image.

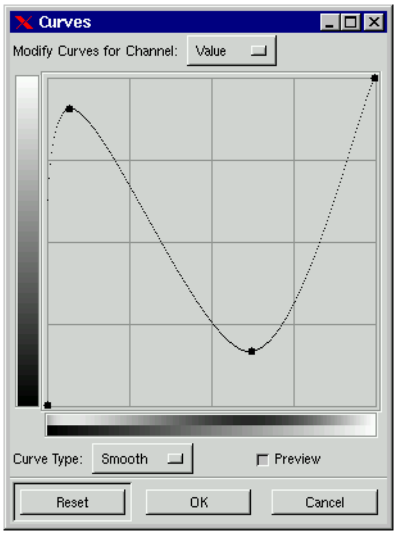
Because this is HSV, the **yellow swatch** starts at 100% pure yellow on the HSV color circle (no orange shade whatsoever allowed) and continues toward **Green**. Everything in the yellow-green cake slice is affected by changes to hue, saturation or value. Yellow pixels get the color you see in the swatch, and greenish pixels get the color next (clockwise) to the swatch color, more and more so the greener they get. To affect the yellow pixels which were slightly to the red side, choose the red swatch instead. The red swatch changes all color between pure red and yellow. This fact can make Hue-Saturation unsuitable for certain images. For most images, it’s very useful indeed, but if you have trouble with it, try **Colormap Rotation** in the **Filters/Colors** menu instead (there are very few things you can’t achieve with that filter).

CURVES

The **Curves** tool is a fantastic instrument for changing **color** and **brightness** of an image or a certain color/brightness range in an image. As a matter of fact, it's so versatile that it is quite hard to describe.

Workflow

Simply put, the image's **RGB values** (and possible **Alpha values**) are represented by **curves**, and you can change a curve anyway you like by dragging at different parts of it.



Modify Curves for Channel: Value

Curve Type: Smooth

When you first open Curves, you'll see a straight linear curve. This curve is called **Value** and represents the values of all three RGB channels of the current image. Value in this option does not mean value as in HSV. Curves is based entirely on **RGB** values, and when you change the Value curve, you'll affect all of the color channels.

Remember the **Threshold** command? When you looked at your image with Threshold, you could choose an area from spike A to spike B, and that area represented pixels of intensity A, ranging to intensity B. It is the same way with Curves; The left part of the curve represent the darkest pixels in the image, and the right part represents the lightest pixels. There is a grayscale gradient to the left of the curve. This gradient tells us what **brightness value** the curve follows. To understand Curves better, you have to open an image and try it.

Try dragging at the middle of the curve, up to the left, and down to the right. While you are doing this, take a look at the gradient below the curve. First of all, you'll find that dragging down makes your image darker, and dragging up makes it lighter. Remember that you dragged in the middle of the curve. The middle represents pixels with **midvalues**. By making a soft curve in this fashion, you have changed the midvalues a lot, but you have only changed **highlights** and **shadows** a little, so you have less **contrast** now. If you check out the lower gradient, you'll see that the balance of dark and light has been changed, you can also see where on the scale this happened and with how much. If you are in a color channel, this procedure will turn pixels a lot more/less (green/red/blue). In a Color channel *dark* means pixels with *little* of that color, and *light* means pixels with *a lot* of that the color in them. If you are changing Alpha, it's the same - **dark** stands for **low** Alpha value (or very transparent).

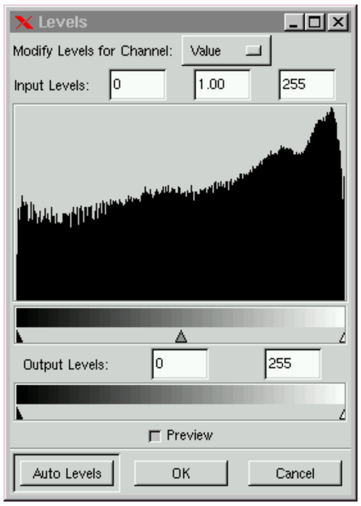
You may have noticed that there are two **dots** at the ends of the curve. Drag at these dots. You'll find that the line to the left/right of the dots is perfectly straight. In practice, this means that dragging the left dot constrains a range of dark pixels to the same value. How **large** this range or interval will be depends on the **horizontal** length of the line. How **dark** it will be depends on where you put the dot on the **vertical** scale.

Click at different places on the curve. Each click produces another dot, and you can move each dot independently of the others. You can move them up, down or along the curve. These dots are like **Bezier anchor points**, but without the handles. This means that by moving a dot, you'll produce a little curve between the two nearest dots. This way you can easily pick a suitable **pixel range**, and only change the values within that range. To get rid of dots, drag them toward one of the end dots. You can choose to get rid of all dots except one, but having only one dot means that your curve is flat. A flat curve causes all values to be the same; and this means your image will turn gray and disappear (or, if you're in the red channel, all pixels will get the same red value).

Because this is a Value based on **RGB**, you can sometimes get an unwanted color change where you only wanted to change the pixel brightness. This happens when you try to make extreme changes, like turning a very dark object light without disturbing the other colors in the image. Doing this will only **invert** the color, and you'll just end up with something red if the object was green. There are many solutions to this problem. You can for example isolate the object in a selection and use **Value Invert** in the **Filters/Colors** menu. This filter works in **HSV** space, and inverts Value without changing Hue or Saturation.

The option **Curve Type: Free**, lets you create or modify your curve with a pencil, which offers unlimited possibilities of fine-tuning colors or values. Pressing **Smooth** will get you back to a smoother version of the same curve. To learn more, you have to keep experimenting by yourself.

LEVELS



Modify Levels for Channel: Value

Input Levels: 0 1.00 255

Output Levels: 0 255

☐ Preview

Levels is another way of manipulating **RGB** properties. Levels is an excellent tool for making **highlights** or **shadow** areas for 3D enhancement. It is also good for putting emphasis on a certain range. While Curves only affect the chosen range, Levels also changes the pixels outside of that range. When you use Levels, you're usually only interested in a defined range, like the brightest or darkest parts of the image.

When you open the Levels dialog, you'll see a **histogram** of your image's Value (all RGB Channels) or of the Red, Green, Blue or Alpha Channel. Take a look at the histogram and the little **arrows** beneath it. The black arrow represents **minimum** value in the RGB channels, the white arrow represents **maximum** value and the gray arrow represents the **mid value** (127 on the RGB scale). The area between the black and white arrows define a certain pixel value range.

Levels on Grayscale

If you're working with a grayscale image, every pixel to the left of that range turns black, and everything to the right of it turns white. Within that range you can decide how the brightness value is going to be distributed through the transition from light to dark. If you're working with an RGB color image, the areas outside of the range will consist of the **primary colors** and their **derivatives** (**CMY**, **RGB**, **BW**).

If you drag the black arrow to the start of the histogram, the white arrow to the end of it and put the gray arrow in the middle, you have pretty much done the same thing as **Equalize** does, i.e. broadened the **spectrum** of the image. If you keep moving the arrows toward the centre, you'll get more contrast and less detail, because the range is narrowing down. Try and move the gray arrow. Moving it to the left adjusts the brightness scale so that more and more of the pixels get brighter, and vice versa. Also try the **Output levels** gradient. The output levels control the overall brightness value, the entire image or drawable gets darker or lighter - always with less contrast than before.

Levels on RGB or Alpha

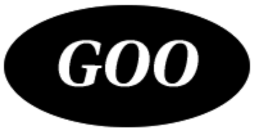
When you switch to a **color channel**, you must remember that we're no longer talking about dark and light, we're now discussing the **quantities** of color in the image. Dark in the grayscale gradient means low values of the color in questions, and light is for pixels with a lot of that color in them. The outcome of moving the arrows in a color channel isn't as obvious as with Value, it depends on how much there is of the color, and in what range most of the color is found. The **Output gradient** controls the value range pixels are allowed to use for a certain color. By this I mean if you set the Output to 100 - 120, all pixels with a little green in them, (even if it's a very low value) get a new green value, which is at least 100, and no pixel gets a higher value for green than 120. Dragging the white arrow to the left causes the image to get less green (i.e. more red), and vice versa for the black arrow.

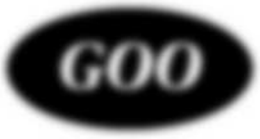
What is true for the color channels is also true for **Alpha**. Here, dark means low Alpha (transparent) and white is high Alpha (fully opaque or solid). When you use Levels on Alpha, you should use it for **advanced selections** from an Alpha Channel or a Layer Mask, i.e a selection with great variation in Alpha values. Otherwise it will be a bit like trying to use Levels on a solid white square on black background - there isn't really much you can do with it.

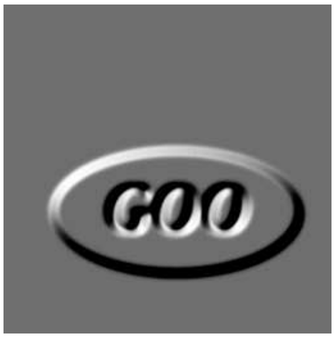
Example: Making Carved Text with Levels

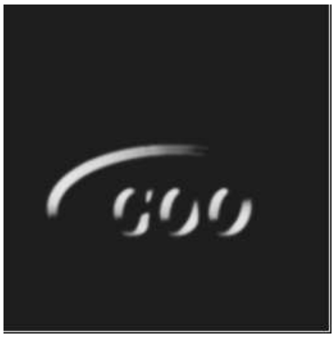
To make you understand what **Levels** are good for, check out this example.

- 1. Create a new image with a white background
- 2. Create a white layer in this image.
- 3. Choose the **Text Tool** and write a few (black) letters on the background. Use some **Gaussian blur** (the more you use, the more bevel and softness you'll get).
- 4. Activate the white layer and choose **Bump Map** in the Filters/Map menu.
- 5. Choose the background to bumpmap.

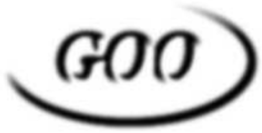


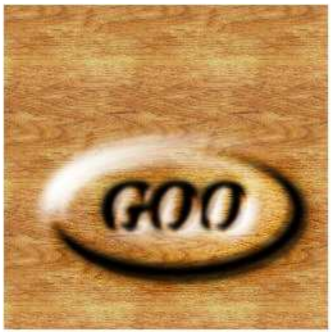






- 6. You should now have a nice 3D image of the letters in your layer. Use some more Gaussian blur if you like, and duplicate the layer.
- 7. Now it's time to use **Levels**! Activate the top layer and move the black and gray arrows to the right, until you see the bright highlights of the letters on top of a black background.

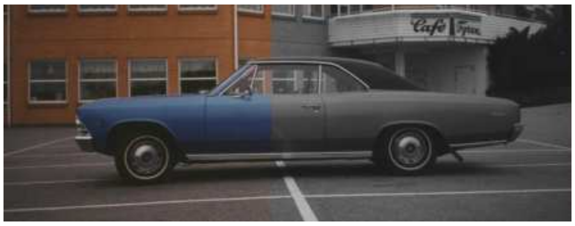




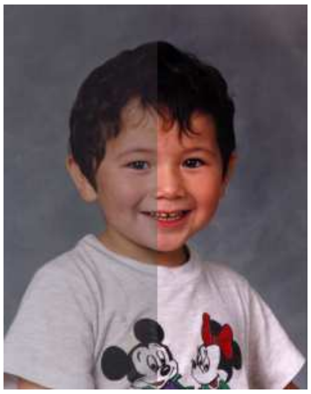
- 8. Activate the other layer and move the white and gray arrows to the left, until you see the shadows on top of a white background.
- 9. Change the Mode of the highlight layer to **Screen**
- 10. Change the Mode of the shadow layer to **Multiply**
- 11. Fill the background with a suitable color or pattern.

DESATURATE

With **Desaturate** you can choose to remove color from a layer or selection without disturbing the rest of the composite image. Desaturate does not turn your image into a grayscale, it's still RGB, even if there is no visible color.

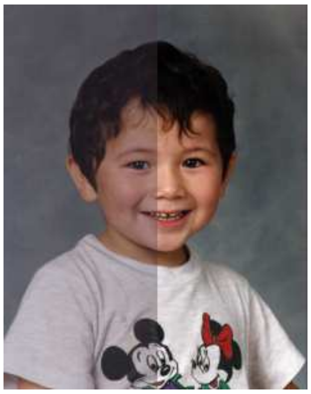


AUTO-STRETCH HSV



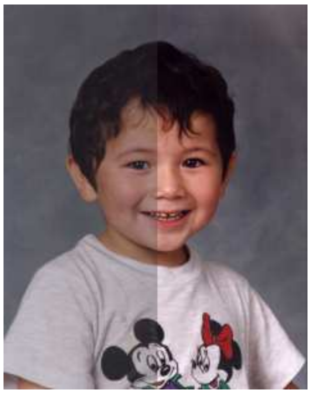
This filter makes an **automatic contrast stretch** of your image. It does this by finding the lowest and highest values of each **HSV** channel, and stretch this to the full contrast range. This is similar to Contrast Auto-Stretch but Contrast Auto-Stretch works in **RGB** space. Auto-Stretch HSV is a great filter for enhancing or correcting old pictures. If you test it on some old, faded images, and it doesn't work, you can always try **Equalize** to see if that does the trick. The filter can also be used to do HSV stretching for other purpose than image enhancing. If you really want to know how it works, just bring up a histogram before and after using it.

CONTRAST AUTO-STRETCH



Works just like Auto-Stretch HSV, but in **RGB** space. This plug-in does a great job removing **undesired color tints**. As you may have noticed, this plug-in is also a wonderful image enhancer for poor quality photos, or faded images. It's also good for badly scanned images.

NORMALIZE



This filter is rather similar to **Contrast Auto-Stretch**, but it won't allow the **RGB** channels to stretch **independently**. Instead they are treated as a **union**. The technical difference is that with Normalize, all channels will not stretch over the whole range from 0 to 255. This is a filter you will use a lot if you are dealing with scanned images and other image enhancements.

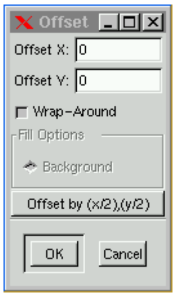
CHANNEL OPS

The **Channel Operations** deal with channels, but in very different ways. **Duplicate** includes all layers and channels in a composite image, while **Offset** only affects the active layer/channel. **Decompose** uses channel information from the active layer and creates new grayscales, and **Compose** transforms grayscales to channels, thereby creating a new RGB image.

DUPLICATE

Duplicate creates a copy of your image. This is a very useful command, so make sure you use it every time you want to experiment with different solutions and variations in an image.

OFFSET



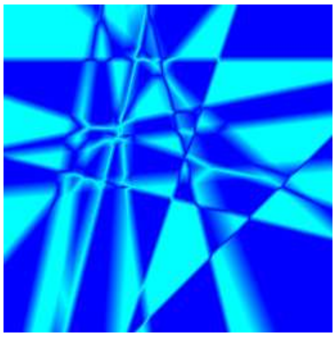
Offset X: 0
Offset Y: 0
☐ Wrap-Around
Fill Options
● Background

Offset can be used for placing layers or floating selections. Offset is useful if you want to move layers in a very exact way, or if you'd like to move them without changing the layer border. **Wrap-Around** means that the parts of the image which ended up outside the layer border, will turn up on the other side of the image. If you don't want this, you can choose to fill the empty area with background color or transparency.

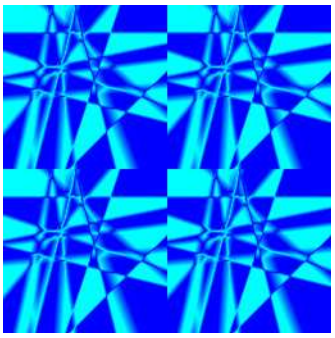
Tips

Another useful aspect of Offset is the possibility of making **seamless tiles**. If you want to make a wallpaper pattern of your image, it's a good idea to check how well the edges fit, before you tile it. If you choose **Offset by...**, and check the **Wrap-**

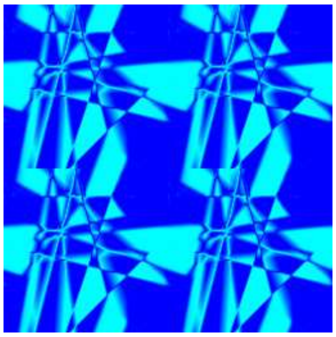
Around button, you'll split your image in four equal tiles. To achieve a seamless tile, you'll have to try and erase the sharp borders between the tiles. There are many ways of doing this, depending on how complicated the image is. You can use the **clone** tool and the **transform** tool, you can **paint**, **copy** and **paste** suitable selections with different opacities, or use special filters. When your happy with the result, choose the **Offset by...** option again, and your image will return to normal, but now with seamless edges which will produce a perfect result when tiling the image. There is a filter called **Make Seamless** in the **Filters/Map** menu which also produces seamless tiles. **Tiling** looks very good with this filter. The drawback is that you can never get really sharp patterns with it - everything looks a bit blurred and double-exposed



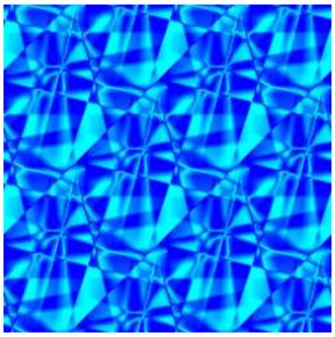
Original



Tiled original

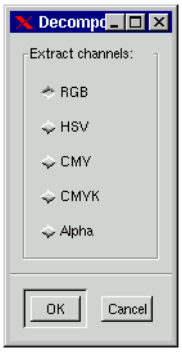


Corrected with Offset

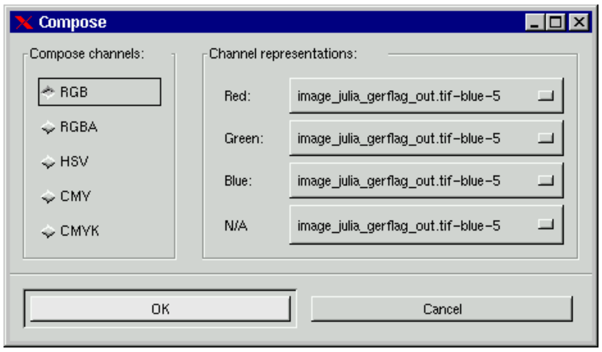


Corrected with Make Seamless filter

COMPOSE AND DECOMPOSE


Decompo
Extract channels:
RGB
HSV
CMY
CMYK
Alpha
OK
Cancel

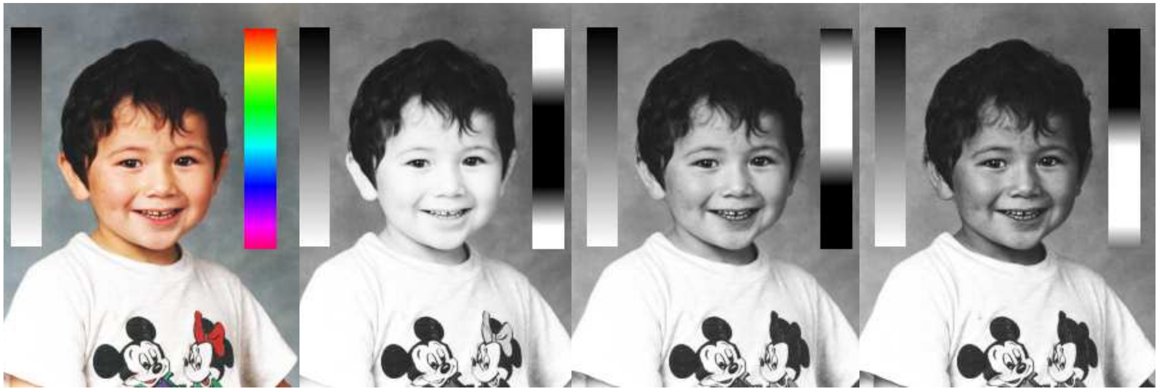
Decompose is a way of **extracting channels** from an image. **Compose** merges (at least three) grayscale images to one color image. You can break up your image into its constituents, by choosing to extract RGB, HSV, CMY(K) or Alpha channels.


Compose
Compose channels:
RGB
RGBA
HSV
CMY
CMYK
Channel representations:
OK
Cancel

Red: image_julia_gerflag_out.tif-blue-5
Green: image_julia_gerflag_out.tif-blue-5
Blue: image_julia_gerflag_out.tif-blue-5
N/A image_julia_gerflag_out.tif-blue-5

RGB decomposing

Decomposing to **RGB** is really the same as activating a color channel in the **Layers & Channels** dialog, but now you'll get them as three different grayscale images. You can do a lot of things with decomposed images. The first thing that comes to mind is to edit them in different ways, and then use **Compose** to put them together again. Another way of using Decompose is to save an extracted grayscale as a selection or **mask** in a **channel**. Sometimes there is very little of a certain color in the background compared to an object in the image. This gives you the perfect opportunity to create very exact selections of intricate objects. You can also play around with compose/decompose, and create interesting patterns.



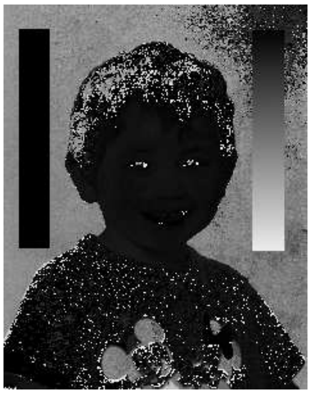
Original *R* *G* *B*

HSV decomposing

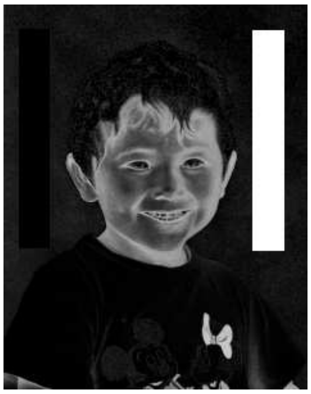
The **HSV** decompose option creates three different grayscales, one for **Hue**, one for **Saturation** and one for **Value**.

It may seem strange that a grayscale image can describe color, as in Hue, but consider the HSV color circle. Imagine a circular gradient where white and black meet in the start and ending point on the circle. Pure black and white represent red, which is on top of the HSV color circle. Dark gray equals orange or yellow, and medium gray tones represent green or cyan. Accordingly, light grays represent blue and magenta.

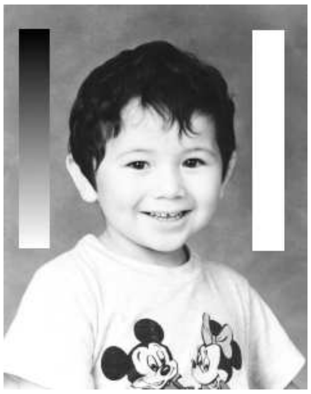
The Saturation and Value grayscales are easier to understand. White represents pure, strong color (saturation) and maximum brightness (value). Black means completely desaturated gray(saturation) and, well, black (value).



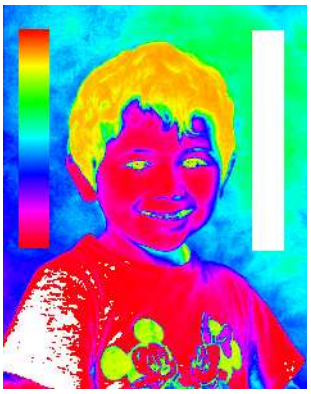
H



S



V

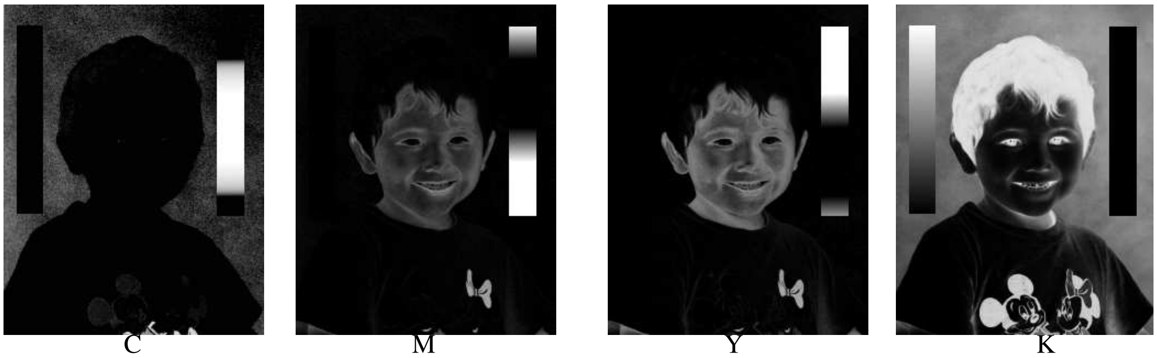


Value composed as Hue

From this we can derive, that if you want something to be white in a HSV based image, Saturation in that area must be set to black, and Value to white. The big difference between RGB compose and HSV compose is that the grayscale information means very different things for Hue, Saturation and Value, while gray in RGB channels just represents the concentration of a certain color. This also explains why it is possible to manage with just one grayscale, when you are composing to HSV. If you were to use the same grayscale for composing RGB, the result would be exactly the same as the original grayscale. If you did the same for CMYK, you'd just end up with an overbright negative of the image (if the image was inverted you'd get a very dark grayscale).

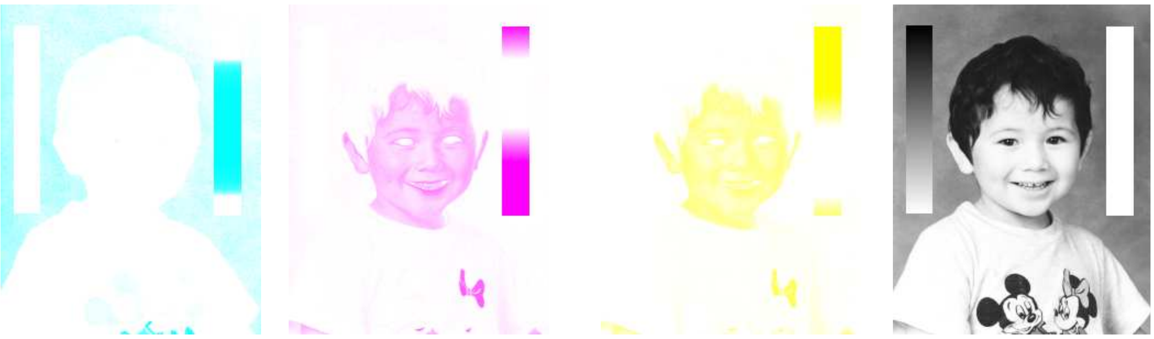
CMYK & CMY decomposing

The **CMY** or **CMYK** decompose option is interesting if you are thinking of putting your work to print. Using CMYK decompose is actually the same as making your own color separation for four-color printing plates. In principle, you can use CMYK decompose, print the outcome on a laser printer, and give it to a professional print shop. However, the professional printer would most likely do a much better separating job than you can, so I don't recommend this solution. You can certainly achieve very interesting results with CMY/CMYK decomposed images, even if you don't use them for printing purposes.



C M Y K

This is the outcome of CMYK decompose and what the printing plates will look like



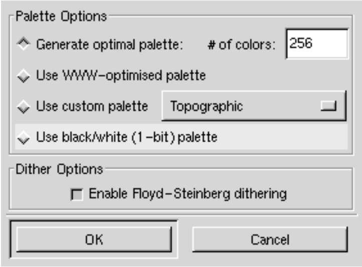
C M Y K

The printed result of each CMYK plate

Alpha decomposing

Alpha decompose is a fast and easy way of converting semi-transparent layers to the equivalent grayscale. You can certainly do the same thing if you copy and paste the layer to an Alpha channel or a Layer mask, but with Alpha compose/decompose you don't have to worry about copy/paste to Channel, or transforming to selection and filling when you want it back in the layer again. It's a real convenient way to edit **Alpha layers**, and a worthwhile alternative to editing in the **Channels** menu. Another great advantage is that decomposing an Alpha layer twice (Alpha decompose as well as RGB decompose) allows you to retain original structure, color and Alpha after editing, because you can compose to **RGBA**, which includes all channel information.

RGB, GRAYSCALE, INDEXED



Palette Options

● Generate optimal palette: # of colors: 256

○ Use WWW-optimised palette

○ Use custom palette: Topographic

○ Use black/white (1-bit) palette

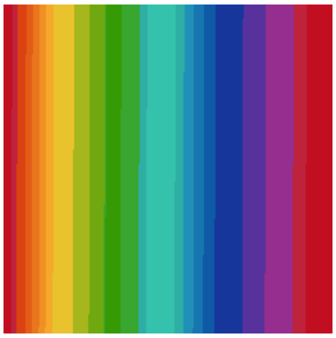
Dither Options

☐ Enable Floyd-Steinberg dithering

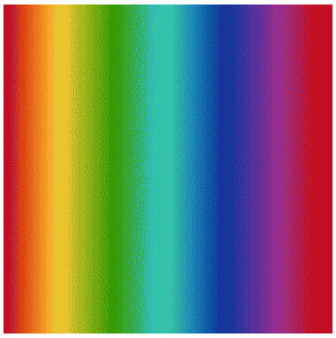
These are the three conversion options you have for your image. The natural choice, when you start working with an image is of course **RGB**. If you know that you want a **grayscale** or an **indexed** image, you can easily convert your image later. Some commands, filters and Script-Fu:s require grayscales, while others require RGB images. Just remember that all color information is irreversibly lost as soon as you convert to Grayscale, and a lot of color information is lost when converting to Indexed. Remember that most commands, filters and Script-Fu:s give horrible or no result at all when used on indexed images, so if you are using an indexed image - ***convert it to RGB immediately, and don't convert it back to Indexed until you're finished with it!*** *Read more about Color Models in chapter 12.*

INDEXED OPTIONS

When you think of indexed images you probably think about **GIF** images with a maximum of 256 colors. However, this is no limit for indexed images in general, you are free to specify any number of colors. If you are dealing with web design then it's a good idea to use the **WWW-optimized** palette to map the colors. You can also use a **personal palette** for color mapping. A nice option is the **Floyd-Steinberg** dithering that helps make indexed images look ”good” even if you only have a few colors in it.



Without dithering



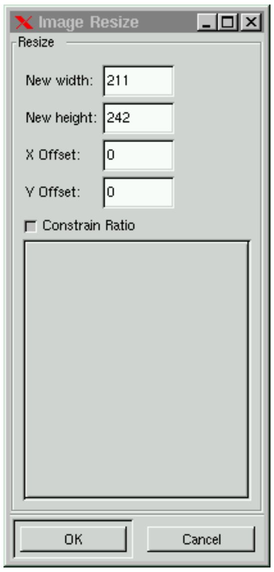
With dithering

RESIZE AND SCALE

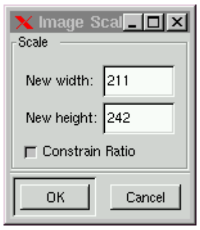
RESIZE

The major difference between **Image Resize** and **Image Scale** is that Resize only changes the **canvas size**, while Scale changes the actual size and **scale** of your image. Note that when you enlarge your image with Scale, Gimp *interpolates* the new image. **Interpolation** means that an additional pixel between black and white original pixels will get a gray color. This means that the enlarged image will look better than the original image zoomed to the same size, but it will also lose in sharpness and clarity.

SCALE



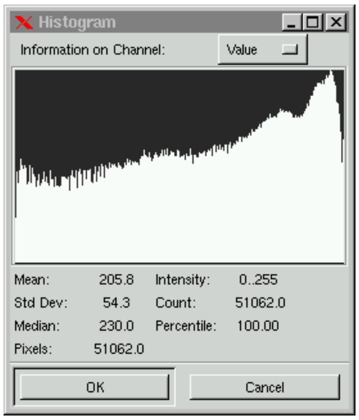
New width: 211
New height: 242
X Offset: 0
Y Offset: 0
☐ Constrain Ratio



New width: 211
New height: 242
☐ Constrain Ratio

Scale makes everything in your image larger or smaller, including layers, but **Resize** doesn't change the size of the layers. To do that - use the **Resize Layer** option in the **Layers menu**. You have to change the layer size if you mean to use the new canvas in a meaningful way (unless you're happy with the layers you have, and only wish to use the new size for new layers). You can determine where on the canvas you want to place the image by using **X- and Y Offset**. The dialog window gives a preview of what the proposed changes in size or offset will look like, and you can choose to keep the proportions of the old image by checking **Constrain Ratio**. If you uncheck Constrain Ratio and then change the proportions, Scale will stretch or compress the image to fit the new size, thus distorting the image. Resize will never distort the image, it will either crop or add a transparent area to the image to fit the new size.

HISTOGRAM



Information on Channel: Value

Mean: 205.8
Std Dev: 54.3
Median: 230.0
Pixels: 51062.0
Intensity: 0..255
Count: 51062.0
Percentile: 100.00

A **Histogram** gives you valuable statistic information on the channel values, or of a certain pixel range in a channel. The histogram always refers to the active layer.

- **Intensity** refers to what value from 0 to 255 a certain pixel spike has, or the value of the first and last spike in a chosen interval.
- **Mean** informs you of the mean value of that interval.
- **Count** refers to the number of pixels in a spike or interval
- **Standard Deviation** tells you how homogenous the pixel spike height is in a certain range.
- **Percentile** tells you how many percent of the total number of pixels there are in a certain range
- **Median** tells you what the median of the interval is.
- **Pixels** tells you the total number of pixels in the active layer.

ALPHA

HOLES:

This plug-in requires a **layered image**. As you may have noticed, indexed images like GIF don't support **semi-transparency**. Holes creates a rather convincing illusion of semi-transparent pixels (at least if you compare it to a dithered Netscape image in 8-bit mode). You can of course use **Dissolve** to achieve a similar effect, but this is a slicker way. In order to make this look good, you must use a small **Hole Size** (like 1) and a medium to l**ow Density** (like -1.50) with the **Keep Opaque** button checked. If you don't use Keep Opaque, you'll risk getting spillout or ”dot pollution” on the transparent areas of the image. This is undesirable behavior if you're trying to create a semi-transparent look in a GIF, but if you're not, this option can result in quite nifty effects. Depending on what effect you want, you may change the shape and size of the holes, you can choose from square, round and diamond, though round and diamond shaped holes require a minimum size of (2).

THRESHOLD ALPHA

This filter lets you remove pixels by their **alpha value** (transparency). The outcome of this filter is always totally transparent or solid (never semi-transparent), so it's a great tool for controlling the appearance of an image that you want to save as a transparent GIF. The slide bar value determines what values you want to remove. If you set a high value, only pixels with an even higher alpha (more solid) will remain after the filter is applied.

SAVE PALETTE

When you are dealing with image manipulation you often want to transfer the colors in your image to a **palette** so you can easily pick the color you want. This tool will extract a palette from an indexed image. The number of colors in the index dialog also sets the number of colors in your palette. Here is the workflow: **Index** your image and choose how many colors you want for your palette. Invoke `Save Palette` to bring up an ordinary **Save** dialog. Type the name of the palette, and press `Save` (the dialog is already in your personal palette directory so there is no need to change the directory). Now, press `Ctrl-Z` to bring back your image to RGB. Now you're almost ready to use your new palette. You just need to restart Gimp before you can use it, because Gimp needs to refresh the palettes.

TRANSFORMS

AUTOCROP

This filter **crops** images **automatically**. If you paint or paste something in a large solid-color or transparent background, you may find that you have put too much space around the image object. If you then apply **Autocrop** it will crop your image so that everything but the important parts are cut away. Note, if you work in a layer, Autocrop will still crop the entire image with no concern of the other layers.

IMAGE

Lets you **rotate** the **image** 270 or 90 deg.

LAYER

Allows you to **rotate** a **laye**r 270 or 90 deg. Remember that when you rotate a layer, some parts of it will always disappear outside of the image boundary.

ROTATE

A simple way to rotate your image or selection (must be single layered) 0, 90, 180 and 270 deg. Quite nice when you are scanning images.

ZEALOUS CROP

This filter works much like the **Autocrop** filter, but there are some significant differences. While Autocrop cuts away peripheral parts which have no color or alpha variation, **Zealous crop** uses **compression** before cutting. When this plug-in finds objects of different color on a solid-color or transparent background (in the same layer), it will **squeeze** those objects together as tight as possible without overlapping, but they'll keep the same formation (relative position) as before. This way you get to keep all your stuff inside of your image, which is now as small as it can possibly be without scaling. Zealous Crop cuts through all layers, but it only uses compression on the active layer. Z-crop always crops inactive layers at the upper left part, while Autocrop crops all layers in the position determined by the active layer.

chapter 15

Selection menu

Here we will explore Gimp's selection possibilities further

TOGGLE

The **toggle** option allows you to turn off/on the ”marching ants” border around layers and selections. This border is sometimes distracting, and prevents you from seeing what you are doing, so this is a nice option.

INVERT

The **Invert Selection** command is used when you want to select everything but the previous selection. This usually means the background or surroundings of the selected objects. Remember that feathered selections are equally feathered when inverted, but the feathering will now do the ”fuzzying” inwards, toward the object border.

SELECT ALL OR NONE

Select all is used together with **Keep transparent** when you have several objects in a transparent layer, and want to fill all of them at the same time (most commonly a text layer, where you want to select all of the letters). You may also want to select an entire layer (without Keep Transparent) if you want to fill all of it with paint, and discard the objects in it. After having used the Select All option you can use **Select None** to get rid of it. You can of course just click at the layer with the rectangular or ellipse tool for the same result, but if you happen to drag by mistake while doing this, you will transform the selection to a float.

FLOAT

This is the command to use when you want your selection to **float**, but you want to make certain that it doesn’t **move**. We have discussed the fact that you can transform a selection to a float by moving it when the select tool is still active. This will inevitable change its position by ever so little, and it will show against the background. This is really important if you took the float from the background, and there is now a ”hole” in the background image where the float used to be.

FEATHER, SHARPEN AND BORDER

FEATHER

A **feathered** selection is a selection which gets more transparent the closer you get to the edges. Feather allows you to blend a color or image softly with the background. You can set a **feather radius**, where a low radius value produces a selection with a bit more softness to the edge than usual. A maximum feather radius means that the fuzziest parts of the selection will reach as far as more than twice the radius of the equivalent ”sharp” selection, and the core of the selection will only have about half that radius. In other words, this is a v-e-r-y fuzzy selection. It is so soft that you can fill it with patterns, then with colors and then paste images into it, and all these things will blend together. If the **Keep Transparent** option is checked, filling or painting within the selection will only affect areas which have an alpha value>0. Also remember that feathering ”blunts” the shape of the selection. **Edge detail** disappears in a fuzzy fog, and a hand-shaped selection with fingers will look like a softball mitt at best. You can use this to your advantage if your selection isn’t as smooth as you’d like. Use feather to smooth, then grow a little to compensate for loss in size, and last - use sharpen.

SHARPEN

Sharpen selection removes the **fuzzyness** in a feathered selection, and leaves only the **core**, which is now sharp edged. Sharpen does not restore edge detail in a feathered selection though. Sharpen is used when you change your mind about the feathering, or maybe more commonly, when you have made a fuzzy fill and want to enhance it by making a sharp fill or paste inside of the fuzzy one (a sharp selection is always smaller than a fuzzy selection).

BORDER

Border transforms the selection to a border area. The new selection surrounds the former selection and reaches a number of pixels outside and inside of the selection edge boundary. Border is useful when working with special effects for text, or when you want to make drawings or frames from selections. When you use **Stroke** the effect may seem similar, but the big difference is that with Stroke, you never change the shape of the selection. Border creates a new selection based on the outline of the old one, and using invert on a border selects both what’s inside and outside of the old selection.

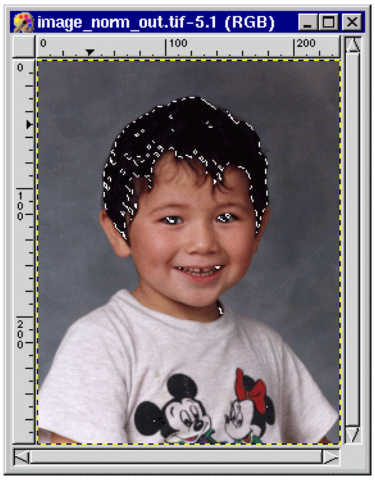
GROW AND SHRINK

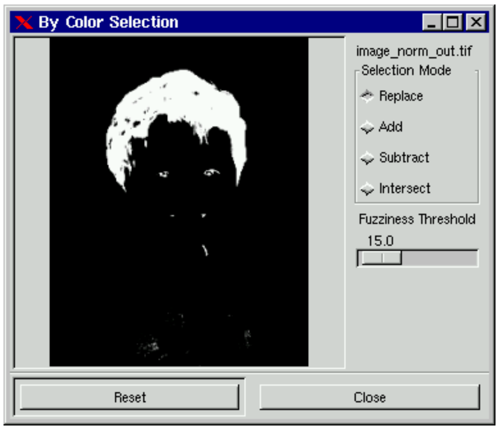
Grow and **Shrink** allows you to change the size of the selection. If the selection edge has fine details, they will disappear if you shrink/grow by many pixels. If you just want your selection to be bigger or smaller, but otherwise look the same, it’s better to save the selection to a layer and apply the **Scale layer** command.

SAVE TO CHANNEL

If you are working with a complicated image, **Save to channel** should be used almost every time you make a selection! In this way you save the selection’s shape and transparency and store it in an **Alpha Channel**. You can later activate this Channel and use the selection again if you need it. You can also do advanced selection editing in Alpha Channels. *Read more about it in chapter 18.*

SELECT BY COLOR





Select by color is a very advanced selection instrument. You can use it to modify an existing selection, or you can create entirely new selections with it. This tool is based on color, but it works fine for grayscale or indexed images too.

To select an area or areas of a particular color in an image, you just click at it, and all pixels with that color, or similar color will be selected. The **Fuzziness Threshold** slide bar lets you decide how generous the ”similarity” classification is going to be. A low fuzziness value selects only pixels of the exact, or nearly exact color, while a high value accepts a lot of colors as being similar.

REPRESENTATION

While you are doing this, you can see a grayscale representation of your current selection in the **By Color Selection** dialog. This preview works just like an **Alpha Channel**. White parts represent sharp selections (opaque), gray parts are for more or less feathered selections (semi-transparent), and black areas represent unselected areas (wholly transparent).

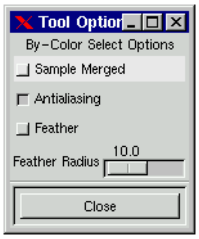
MODES

If we take a look at the **Selection Modes**, there are four choices; **Replace**, **Add**, **Subtract** or **Intersect**.

Replace is the standard mode, and also the most important one. With Replace you choose the **basic color range** which your selection will be composed of. If you are not happy with the basic selection, you can click at another place in your image, or change the fuzziness value to create a more accurate selection.

Add and **Subtract** are used to modify the basic selection. By pressing Add and clicking at an unselected part of the image, that color or color range will be added to the selection. Likewise, Subtract will **deselect** areas of unwanted color from your selection. Intersect is a little trickier to understand, but you can say that **Intersect** chooses, depending on the fuzziness threshold, the small stripes of intermediate color between the selection color and the adjacent color you clicked at (if you click at a color which doesn't touch the selection, you end up with no selection at all). Using Intersect usually results in selecting the thin, antialiased edges between different shapes.

OPTIONS



☐ Sample Merged
☑ Antialiasing
☐ Feather
Feather Radius 10.0

Select by Color also has an **Options dialog**. The **Sample Merged** option refers to the possibility to base the color selection on the entire composite image, instead of just the colors in the active layer. The **antialiasing** option is checked as default, but if you uncheck it the selection will get sharp, no feathering whatsoever is allowed. The **Feather** option is quite interesting though. Watch the **Alpha Channel preview** closely when you select with this option. You'll create soft cloud- or nebulae-like selections with Replace, and lightning or metal highlight effects with Intersect. Feel free to experiment with the Feather option checked, the results are always interesting. If you really like the grayscale interpretation of the selection, and want to work with it as an image, you can save it by the **Save to Channel** command, and then copy that channel to a layer.

chapter 16

Modes

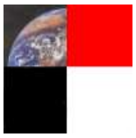
Modes is a powerful instrument for applying Layers and Paint. There are fourteen different ways of combining layers, and they're all very useful. If you learn how to use Modes, you've learned a lot about computer graphics.

WHAT IS MODES

Blending modes or **Transfer** modes can be described as different ways of controlling how the pixels in a foreground layer blend into the background layer(s). Modes is mainly used with layers, but they can also be used more directly as paint modes with the **Paint bucket**, the **Blend tool** or any tool which uses **Brush Selection**. There are 15 different modes in Gimp;

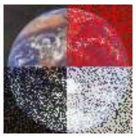
- **Normal**
- **Dissolve**
- **Multiply**
- **Screen**
- **Overlay**
- **Difference**
- **Addition**
- **Subtract**
- **Darken Only**
- **Lighten Only**
- **Hue**
- **Saturation**
- **Color**
- **Value**
- **Behind** *(not available for Layers Mode)*

NORMAL MODE



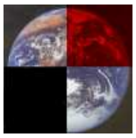
Normal Mode doesn’t do anything special, as you might expect. The normal mode layer covers all other layers, unless there are **transparent** or **semi-transparent** areas in it.

DISSOLVE MODE



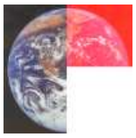
Dissolve Mode is actually very similar to Normal Mode. You must move the **Opacity** slide bar and make the layer or paint transparent to be able to see the effect. When in Normal Mode at 50% opacity, you'll see a smooth semi-transparent surface, Dissolve Mode will produce a *dotty, grainy looking surface*. Instead of using semi-transparent pixels, Dissolve Mode produces **entirely opaque** or **entirely transparent** pixels, 50% of each (if that's the opacity value). This mode creates noise effects similar to grainy film or granite rock. If you want a transparent GIF to look semi -transparent, using Dissolve mode (or Holes in the Image/Alpha menu) is the only way you can achieve something of the kind. As you may know, GIF:s only go two ways - transparent or non-transparent pixels. However, this isn't a great solution. Most likely, your semi-transparent surface (glowing text for example), will look much better with a solid background, if you index it to match the background on your page. ***Conclusion: This mode is used with semi-transparency, where it divides pixels into wholly transparent or fully solid, according to the level of transparency.***

Multiply Mode



Multiply Mode amplifies **shadows** or dark areas of the image. Multiply is the mode for creating shadows. You may compare it to putting two **slides** on top of each other on a fluorescent clipping board. As in a slide, white areas are transparent in this mode, and as putting together two slides would - the result is always darker. As to colors, they also react as layers of colored glass (or if you prefer, CMYK color, yellow+magenta=red). The difference between Multiply and Darken Only will be discussed later in this chapter. ***Conclusion: This mode works with shadow, much like putting two slides on top of each other.***

Screen Mode



Screen is the mode for creating **highlights** in an image, and is in many ways the opposite of **Multiply**. You may compare it to **projecting** two slides on a film screen. As you may have understood, this mode depends on light, so black is transparent (just like a black spot on one slide will allow the image on the other slide to show) and *the result is always lighter*. Accordingly, the darker a layer is, the less it will affect the composite image and vice versa. In Screen mode, white covers all, gray shades get more transparent the darker they get, and color is only visible if the other layer's color permits it. (Now we are talking RGB color, yellow + magenta = white, just as on your computer monitor or television set). For more information on Screen mode, read about the difference between Screen, Addition and Lighten Only further on in this chapter. ***Conclusion: This mode works with light, much like projecting two slides on the same screen.***

OVERLAY MODE

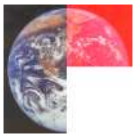
Overlay Mode is something of a combination of **Screen** and **Multiply**. Medium gray is transparent in this mode. The background is the most important layer; the overlay layer is only used for modifying the background. Light and dark areas in the foreground affect the background by intensifying **highlights**, **color** or **shadows**, so white, black or RGB/CMY parts of the background are not affected *(you can't intensify the shadow of black, highlight white areas or intensify a color with maximum color value)*. In this mode all FG colors appear in a paler, more washed-out shade. Overlay color and saturation depend mostly on the background. You can say that the foreground intensifies or modifies the background color; a green FG on a green BG makes the image a bit greener, while a red FG makes the green color slightly less green. Overlay Mode is good for adding shadows and highlights to an image, or changing the color temperature. What it really does is to add information to the existing (brightness) Value in your image, so you can"paint with light". Hue and Saturation are also changed, but add no more than a cold shadow or reddish sunlight would. ***Conclusion: The Foreground affects the dominating Background by intensifying color, highlights or shadows.***

DIFFERENCE MODE

In **Difference Mode**, foreground and background pixels have a dramatic effect on each other. Black pixels in either FG or BG are transparent, but as color gets brighter, the effect increases, white being the most powerful difference color (a white pixel always **inverts** the correspondent pixel). What Difference mode does, is to evaluate corresponding pixels in both FG and BG and calculate the difference between them. Now, if we're talking of grayscale pixels in one of the layers, it's easy to predict the outcome. If for example the foreground pixel is brighter than the one in the BG layer, the background pixel gets inverted, and if it's darker, it keeps its color. A greyish cast is however added to the image - the more the grayscale pixel goes toward white or black, the weaker this shade of gray gets. If the pixels are colored or grayscale in both layers, the result is harder to predict, but it works the same way; |FG-BG|=Outcome i.e the absolute value of the difference between FG and BG, *(you count with negative values and dismiss the minus character).*

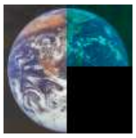
Difference mode is usually very **colorful**, and can produce psychedelic results, but it is also a powerful tool for telling the **difference** between two layers. This becomes interesting when you for example want to compare the size, shape and relative position of grayscale masks in different layers. Where the result is black, the layers are identical, and any small difference will appear very clearly in this mode. ***Conclusion: This mode displays the difference between the RGB-values in the two layers.***

ADDITION MODE



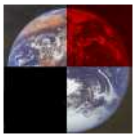
Addition Mode is rather similar to **Screen Mode**, and is built on the simple principle of adding the RGB values of foreground and background pixels. The result is always lighter and often results in white areas and unsharp edges. ***Conclusion: This mode adds the foreground RGB-value to the background RGB-value.***

SUBTRACTION MODE



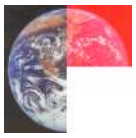
Subtraction Mode is the opposite of Addition, and sometimes produce results similar to **Difference**. Here you **subtract** the foreground color from the background color - If your background is White (255,255,255), and you subtract Red (255,0,0), the result is (0,255,255) which is Cyan. A white shape in the upper layer against a white background will for the same reason produce a black shape. So far the result is the same as for Difference, but as soon as a foreground value exceeds the background value, the results start to diverge, since what is zero for Subtract becomes a”negative” value for Difference. ***Conclusion: This mode subtracts the foreground RGB-value from the background RGB-value.***

DARKEN ONLY



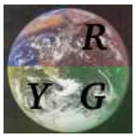
Darken only is somewhat similar to **Multiply**. It is a mode where color can only get darker, but Darken doesn’t work like two slides on top of each other. Here foreground and background pixels are compared, and Darken chooses the **darkest** RGB values in each channel. The result of choosing between a bright red (243,83,47) and a turquoise blue (47,239,201) would be: (47,83,47) - a dark moss green. Had you used Multiply instead, you would have gotten a similar but somewhat darker color (44,77,37) ***Conclusion: This mode compares the pixels of foreground and background and chooses the lowest RGB-value.***

LIGHTEN ONLY



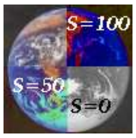
Lighten only is of course the opposite of Darken only. If we choose the same example here, bright red and turqoise, the result would be: (243,239,201) - an eggshell bright beige. Lighten works by choosing the **highest value** of each pixel pair, and the result is always lighter (similar to Screen). If you’d used Screen instead, you’d have gotten a similar, but somewhat brighter color (246,245,211), and if you had used Addition, you’d have got a similar but much brighter color (255,255,248). ***Conclusion: This mode compares the pixels of foreground and background and chooses the highest RGB-value.***

HUE



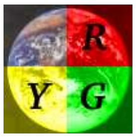
Hue mode makes it possible to change the **color** of an object without changing brightness or saturation. This mode acts on color alone. Your foreground color is the color you'll get, but you'll keep the general feeling in the image; a loud green on soft dark blue will be translated into a loud yellow on soft dark yellow. White, black or grayscale information in the background is not affected and cannot be tinted. Consider this if you wish to tint a background with complementary colors next to each other. Fuzzy blue dots on yellow means that somewhere around the edges there will be gray (on your screen, transparent blue on top of yellow doesn't turn green, it gets gray!), and your dots will look much like frog's eggs or germs in a microscope. Note that you cannot shift a color to grayscale in this mode either. Grayscale information in the foreground comes out as a scratchy red. ***Conclusion: The composite image uses the Background for Value and Saturation information, while the Foreground is only used for determining Hue.***

SATURATION



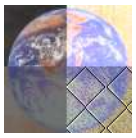
You can use any color you like in the foreground, background color doesn't change, only **saturation**, which assumes the same saturation as the foreground color. The hue can't be changed, except for pure grayscale background which becomes shades of red if the foreground contains color. Because you can change saturation, this mode is often used with gray paint because this desaturates the background. Note, using black, white or gray makes no difference, what matters is that those colors have no saturation. ***Conclusion: The composite image uses the Background for Hue and Value information, while the Foreground is only used for determining Saturation.***

COLOR



Color Mode will change both **hue** and **saturation** in an image. Black or white background pixels are not affected, but all other colors, grayscale or not will assume the hue and saturation of the foreground color. Color mode does not affect the **Value** (brightness) of the background. The dark and light information is left intact, but not hue or saturation. This is a nice mode for tinting grayscale photos if you want strong, clear colors, for a softer tinting (old photo look) consider using Overlay instead. ***Conclusion: The composite image uses the Foreground for Hue/Saturation information, while the Background is only used for determining Value.***

VALUE

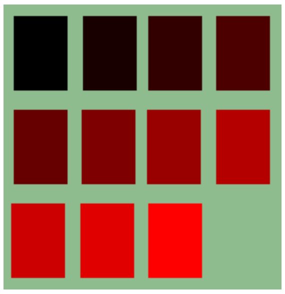


Value is the same as”Luminosity” in Photoshop. It doesn’t change the hue or saturation of the image, but it changes **brightness** and **shadows**; i.e the structure or 3D dimension of the image. As opposed to Overlay, this mode can’t distinguish a dark red from a bright red, all shadows or highlights disappear from the background, and Value mode can’t see grayscale BG at all - because all of these things are associated with Value, and BG value is ignored in this mode. Value is good for correcting overbright/dark colors, or it can be used to transfer patterns or structures into an image, without changing the color of the image. ***Conclusion: The composite image uses the Background for Hue/Saturation information, while the Foreground is only used for determining Value.***

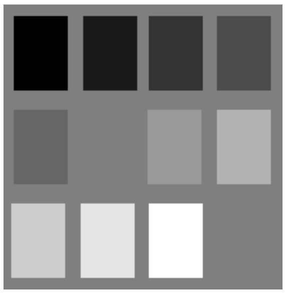
BEHIND

This Mode is a pure **paint-mode**, and has as such no function in the Layers dialog. Behind is used with Blend, Paint bucket or in the Brushes dialog. There is no use trying this mode on solid layers, because it only affects **transparent** or **semi-transparent** areas. When you paint with Behind, it’s like painting on the other side of the layer. If you compare the layer with a wall with windows in it, the Behind paint would be applied to the outside of the house, and only be visible through the windows. Accordingly, if the windows are dirty (semi-transparency), the dirt will show against the Behind color. ***Conclusion: This is a paint-mode where only transparent areas are affected by your painting (does the opposite of Keep transparent in the Layers dialog)***

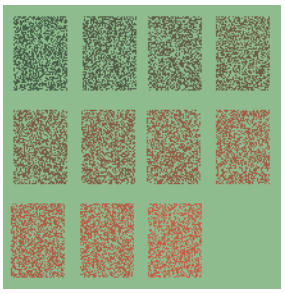
COMPARING PICTURES IN DIFFERENT MODES



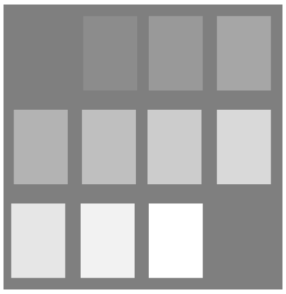
Normal Mode (color)



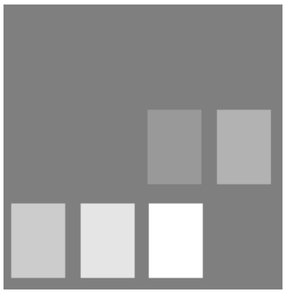
Normal Mode (b/w)



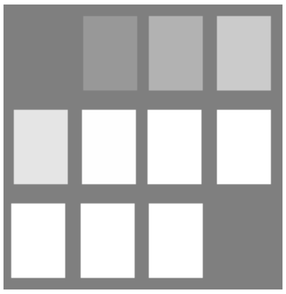
Dissolve Mode (color)



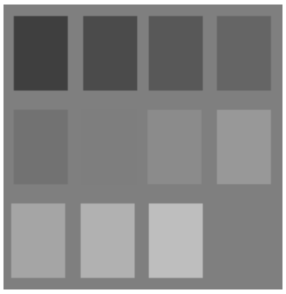
Screen



Lighten Only

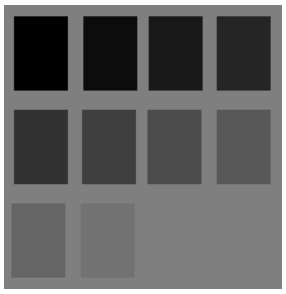


Addition

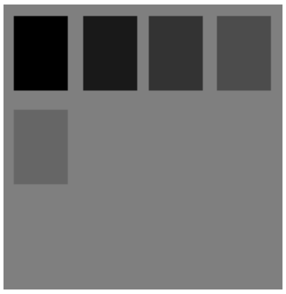


Overlay

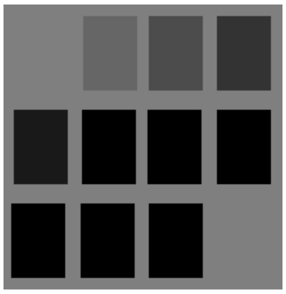
As you can see, Screen has kept the value variation of the grayscale squares, but all shades are brighter than the original. Lighten hasn't changed the brightness of the visible squares, but all dark squares are invisible. Addition brightens up so much that most of the squares are white, and black is invisible for all three modes. Overlay is something between Screen and Multiply. Medium gray is invisible in this mode.



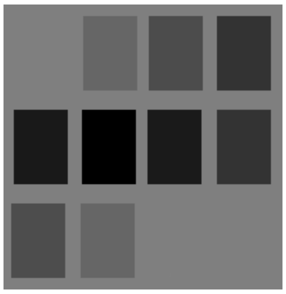
Multiply



Darken Only

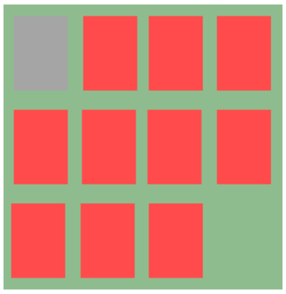


Subtract

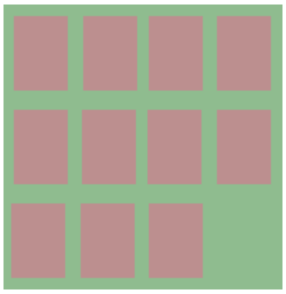


Difference

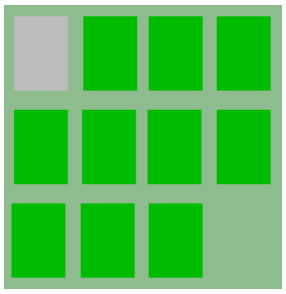
Multiply, Darken and Subtract (grayscale) are the dark equivalents or opposites of Screen, Lighten and Addition Color and Hue both use the hue of the foreground, but not the value. As you see, Color also uses the saturation of the FG (as Saturation does). Value is the opposite of Color, and Difference displays the difference between FG and BG.



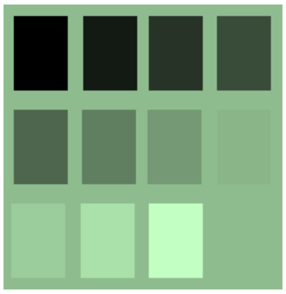
Color



Hue



Saturation



Value

WHY ARE THERE SO FEW COLORS IN SCREEN, ADDITION, AND LIGHTEN ONLY?

With pure process colors (**CMY** or printing ink colors) as back/foreground (**Cyan**, **Magenta**,**Yellow**) shades of that color is the only color you're going to see in **Screen**, **Addition** or **Lighten** mode. The reason for this: Just as White has a maximal RGB-value for all three channels, the RGB value of a process color is maximum for two RGB-channels. As a matter of fact, all RGB colors with max values for two channels are more or less bright versions of a certain process color, no matter what the third value is. So, if you fill a layer with Magenta (255,0,255), all colors in the composite image will end up Magenta. The more green they have in them, the paler the shade will be (until white) Check it out with the color selection dialog! Using pure Red, Green or Blue produces similar results. Since The RGB-colors have max value in one channel, output colors are restricted to those with max values for that channel, e.g. Using Red in a layer means only colors in the red spectrum (from magenta to yellow) are visible in the composite etc. Conclusion: White=max in all three channels - only white is visible, Process colors=max in two channels - only shades of that color are visible, RGB color=max in one channel - only shades of that spectrum are visible.

WHAT IS THE DIFFERENCE BETWEEN SCREEN, ADDITION AND LIGHTEN ONLY?

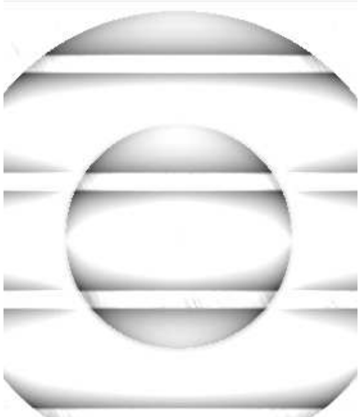
Screen, **Addition** and **Lighten** may appear very similar, sometimes almost as the same mode, but there are certain important differences. **Lighten Only** compares the FG and BG RGB-values, and chooses the **higher value** for each channel. Addition just **adds up** all values, and thus comes up with a brighter image.

Screen works by **mapping** the foreground against a background scale from 0-255 in each channel, where the FG pixel maps its value to 0 (black) on that scale. Compare it to putting two scales of equal length on top of each other. The background scale goes from 0 to 255, while the foreground scale goes from x to 255 (x=FG value). The composite value can be found somewhere on the FG scale, and that placement is determined by the other scale. If the background is a grayscale with an Intensity of 100, and the foreground's Intensity is 178, the composite **Value** can be calculated like this:

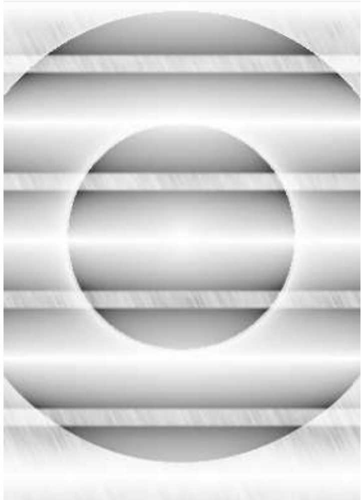
- The Background scale consists of 255 equal parts
- The Foreground scale consists of 77 equal parts, because 255-178=77
- If you choose 100 on the BG scale, the FG scale can be calculated: 100/255=FGscale/77
- FGs=(77x100)/255, so in this case the FGscale=30
- Composite Value=178+what the FG scale shows: 178+30=208

The formula for how Screen works: Composite Value = FG + ((255 - FG) x BG) / 255

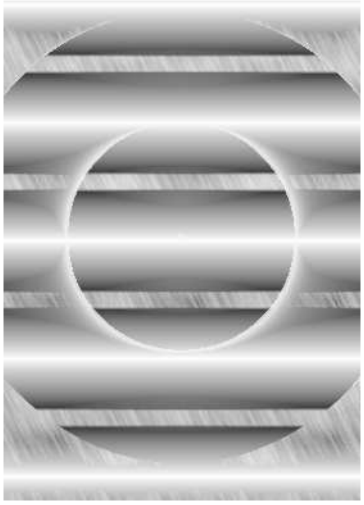
The most obvious difference between the three modes is that **Screen** color is *darker* than **Addition**, but *lighter* than **Lighten Only**. The main (in practice) difference between Lighten Only and Screen is that Screen is brighter, and it allows more shades than Lighten does. This is because Lighten is seldom



Addition



Screen



Lighten Only

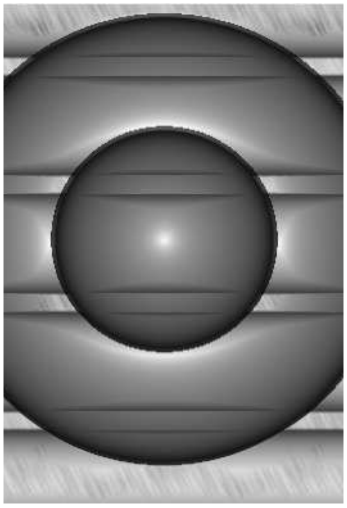
affected by dark colors, it'll just choose the color that is lighter in all RGB values if there is one. The advantage of Screen vs. Add is that Screen colors don't white out as often as Add (because 255 is often the result when adding up).

WHAT IS THE DIFFERENCE BETWEEN MULTIPLY AND DARKEN ONLY?

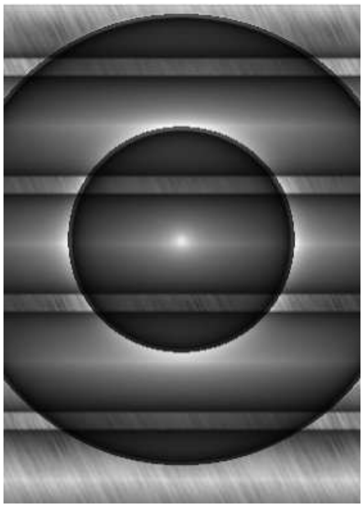
These two modes often result in similar outcome, where **Multiply** is always a little *darker*. As you know, **Darken** chooses the darkest RGB values in each channel to produce its result. You can compare the difference and similarity of Multiply and Darken Only with those of Screen and Lighten only (Subtract would also correspond to Addition, if you inverted the top layer). Multiply is mathematically the opposite of Screen. Where in Screen the foreground value was mapped to zero and up, it is mapped to 255 and down in Multiply. This means that the Foreground value is the lightest possible value, just how dark it's going to be is determined by the Background Value. ***The algoritm is simple: The Composite Value = FG x (BG/255).*** For grayscale images, the difference is much more evident. If you take a look at the pictures on the next page, you'll find that Darken Only looks semi-transparent where Multiply looks as transparent as slides. The Multiply Mode is darker and shows a lot more of the background, and in color images, you'll get more color variation (for lighter colors) with Multiply than with Darken Only.

WHAT IS THE DIFFERENCE BETWEEN COLOR AND HUE?

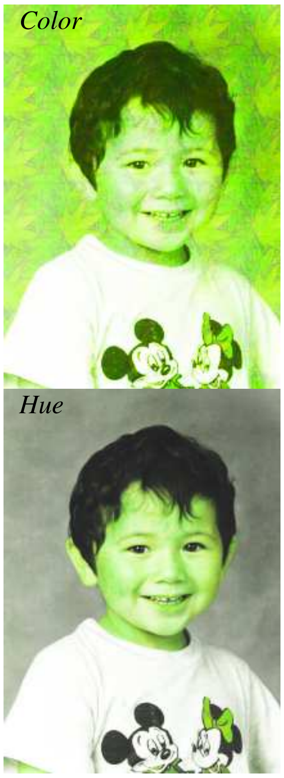
Hue and **Color** sometimes produce similar results, because in both cases the **Foreground** controls the composite **Hue**, and the **Background** determines **Value**. The difference consists in what layer controls **Saturation**. In Hue Mode it's the background, and in Color Mode it's the foreground. Generally, Hue and Value are the most important parameters in an image, and if Saturation is roughly the same in both layers, it'll be hard to tell the difference between Hue and Color Mode. You will however certainly be able to tell the difference if one of the layers is grayscale, or if there is a lot of variation in the saturation of a layer. You can also think of it this way; *Color Mode is often used for adding color to a grayscale*, in which case it's important that the top layer controls Hue and Saturation, while it must not interfere with the dark/bright values of the background. *Hue, on the other hand is often used for changing the existing color of a certain object.* In this case you don't want to change the Saturation in the background, because then the object would look very unnatural.



Darken Only



Multiply



In Color Mode (top, right) you can see the leaf pattern of the foreground clearly, although no brightness Value from the FG has been used in the composite image.

In Hue Mode, the only visible effect of the foreground is the green color of the leafs. Because Saturation is taken from the background, the only part of the image which is really green is the face (which had the highest saturation).

chapter 17

Layers

In this chapter we'll discuss the very foundation of effective image manipulation - Layers. What they are and how to use them.

INTRODUCTION

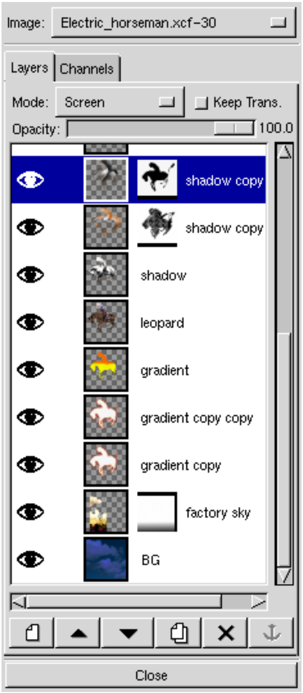


Image: Electric_horseman.xcf-30

Mode: Screen

Opacity: 100.0

I have heard that some Gimp-users find the concept of layers hard to grasp. It shouldn't be that way though, because using layers is really as simple and straightforward as wearing layers of clothes on your body. Once you have started to use layers, you won't understand how you could ever do without them!

ADDING LAYERS TO YOUR IMAGE

Open the `Layers & Channels` dialog box in the `Image-Layers menu`. The dialog box has an **Image menu**, where you can see the name of your image. If you are working with several images this is very useful, because all open images are listed here, and you can choose the one you want to work with by clicking its name.

THE DIALOG

There are two **tabbed folders** - one for *Layers*, where you are now, and one for *Channels* (Channels will be discussed in chapter 18). There is also a drop down menu for **Modes**, an **Opacity** slide bar, and a **Keep Transparent** button.

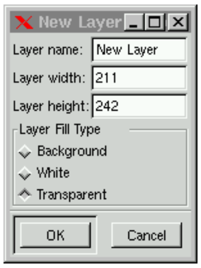
There are also five *short cut* buttons at the bottom of the dialog. They are (counted from the left) **New** layer, **Raise** active layer, **Lower** active layer, **Copy** active layer, **Delete** active layer and **Anchor** floating selection, all of these options will be discussed later in this chapter.

LAYERS

In the window, you will see a **thumbnail icon** of your image. There is an **eye symbol** to the left of it, and you can see that your image is called *Background*. Try clicking at the eye, and you'll find that your image disappears! Click again and it comes back. The eye symbol allows you to **toggle on/off layers**, so you can work with a particular layer without having to see all other layers. If you `SHIFT-click` on an eye icon, all layers except that one will be hidden from view. To toggle on the other layers, `SHIFT-click` again. Now, press the `right mouse button` with the cursor placed on the thumbnail image or the layer name. This brings up a **Layers menu** with a few more options than the regular Image Layers menu (the one you get when you right-click in your image). Choose the first option on the list called **New Layer**.

LAYER OPERATIONS

New layer



Layer name: New Layer

Layer width: 211

Layer height: 242

Layer Fill Type

○ Background

○ White

● Transparent

When you choose **New Layer**, you'll get a little dialog box asking you some questions about the new layer. Here you **name** your layer. You can for instance call it "shadow" if you mean to create a shadow of an image object in this layer. It's a good idea to name your layers if you work with many of them, especially when you use similar layers with different functions. You can also change the **size** of the layer, but you don't really have a reason to do this, unless you know exactly what to put in it. The last thing to decide is what **layer fill** you want. The options are; **Background**, meaning your current background color in the toolbox, **White** and **Transparent**, where transparent is the default fill. Press `OK`, and you have created a new layer on top of your background.

The active layer

Now, when you have two ore more layers, it's really important to know which one is the **active layer.** The active layer is marked with blue color in the dialog box, and you can make another layer active by clicking its name beside the thumbnail. As soon as you have several layers in an image, Gimp ***reacts only to the active layer***, so you can only work in one layer at a time

If a layer is smaller than the background, you can also make it active by clicking in it with the **Move tool**. If you want to move a transparent layer, or just make sure that you move the right layer, press the `Shift key` as you drag. This will ensure that only the active layer is moved. When the selected layer's boundary changes color from yellow to blue, you know that it's safe to move it. Also see chapter 9 about the Move tool.

Symbols and explanations

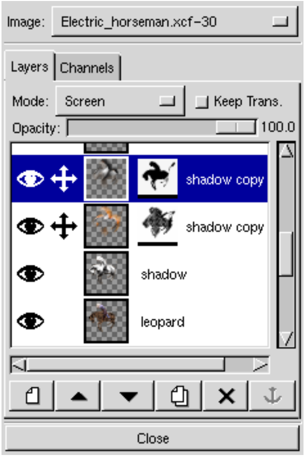


Image: Electric_horseman.xcf-30

Mode: Screen

☐ Keep Trans.

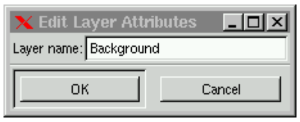
Opacity: 100.0

If you click beside the eye symbol, you'll get a new icon - a little **anchor** symbol. This enables you to *link* some of your layers. **Linking** layers means that all layers with the little anchor symbol are locked or grouped in relation to each other, so you can't move one layer without moving the others at the same time.

Only remember to unlink the layers before moving another layer, otherwise you'll move all of the layers even if the linked layers aren't active.

To understand the **Keep transparent** button, you first have to draw something in a transparent layer. Make the layer active, and paint something with the **paintbrush** (with the button unchecked, which is default). Now, press the button, change the color of your paint, and start painting again. You'll find that you can only paint the **opaque** areas - i.e. the ones you already painted. The purpose of Keep Transparent is to protect the boundaries of opaque shapes in a transparent layer. It also makes it very easy to change their color, you never have to worry about painting outside of edges etc. If you want to use a filter on a layer, you usually don't want this option checked, because then you'll only get the filter effect inside of the shape.

Naming



Layer name: Background

If you want to **rename** a layer you just `double-click` on the layer's old name and you'll get a dialog where you can type the new name.

Remember that your original image was automatically named *Background*? The **background layer** doesn't react quite as the other layers, because it isn't really regarded as a layer - it's a fix and solid area for layers to work against. The background works the same way as it did when you didn't have layers. When you cut parts of it, or use the eraser, the background color shows - not a transparent surface. You also can't use **layer masks** on it or raise it. Normally, you have no need to do these things on a background, but if you want to anyway (for transparent GIF:s for example); `double-click` its name, rename it and it will behave like any other layer. You can also use **Add Alpha Channel**, or duplicate the background and erase the original.

RAISE LAYER AND LOWER LAYER

These commands move the active layer to a higher or lower position in the layer hierarchy. You can only raise or lower a layer one step at a time, so if you want to move a layer near the background to the top, you have to keep raising it step by step, until you are satisfied with the position. The background layer can't be raised, unless you **rename** it or use **Add Alpha Channel**.

DUPLICATE LAYER

Makes a **copy** of the active layer, and places it immediately above the original.

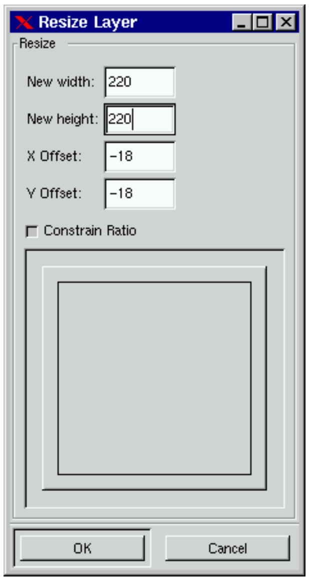
DELETE LAYER

Deletes the active layer.

SCALE LAYER

Shrinks or enlarges the layer *and its contents*

RESIZE LAYER



Resize Layer

Resize

New width: 220

New height: 220

X Offset: -18

Y Offset: -18

☐ Constrain Ratio

Changes the size of the layer, *but not its contents.* If the contents of the layer is larger than the new layer size, the image or selection will be clipped to fit the new size. There is also an **offset** option, where you can decide how the layer content is to be placed inside the new layer.

ADD LAYER MASK

DESCRIPTION

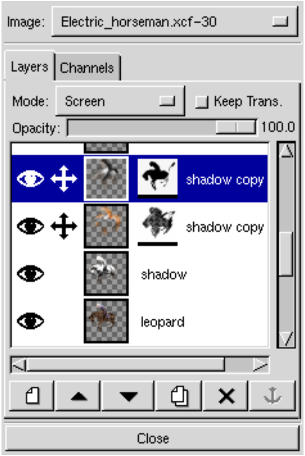


Image: Electric_horseman.xcf-30

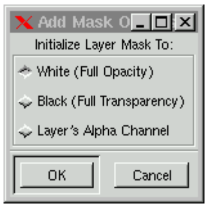
Mode: Screen

☐ Keep Trans.

Opacity: 100.0

This is an advanced option. You use a **Layer Mask** when you want to change the **alpha values** in a layer. If you don't know what alpha values are, read the chapter on **Channels**.

HOW TO



When you add a **layer mask** to a layer, you always work with a **grayscale** image (you can use colors, but they will appear as shades of gray in the mask). In the layer mask, black stands for transparent and white for opaque. When you choose `Add Layer Mask`, you'll first be asked what initial **fill** you want; white, black or layer's alpha. If you choose **black**, you start out with a *transparent* mask, **white** gives an *opaque* mask, and **layer's alpha** means that the mask will use whatever alpha values your layer has. When you have chosen the initial fill, a small layer mask icon appears next to the layer thumbnail. To switch between layer mask and layer you click at the appropriate thumbnail. To create the mask, you can do a number of things.

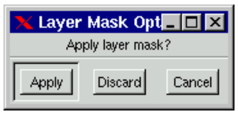
With gradients

A very useful mask can be made from a **gradient fill**. A gradient results in the top layer gradually blending with the background. You can paint degrees of transparency where you want it, or use filter or a pattern to create an interesting mask. You can use selections, or paste an image into the mask. In short, this is an extremely versatile and powerful instrument.

Green and Red Mask display

To work with the mask, the little thumbnail is quite insufficient! Use `ALT-click` on the **mask icon**, and you'll see the grayscale mask instead of your image. *Note that the mask icon is now lined with **green***. When you want to check the result of the mask, `ALT-click` again, and you'll see what your image looks like with the mask effect. To see your image without the mask `CTRL-click` on the icon; *The thumbnail is now lined with **red***. `CTRL-click` again to return to the mask. You can also alternate between **green** and **red** display by `ALT-clicking`, but to enter and exit this mode you have to switch to red first.

APPLY LAYER MASK

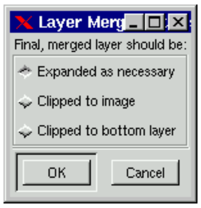


Apply Layer Mask is used when you're happy with your mask, or if you're dissatisfied, and don't want to use it.The choice is clear - **apply** or **discard**.

ANCHOR LAYER

Anchor layer is used for merging **floating selections** with a layer (the layer which was active before you placed the float.). If you don't know what a floating selection is, check out chapter 18. Most of the time it's quite sufficient to mouse-click in the float to make it stick to a layer. Sometimes the **Move tool** has to be active for this to work (this is the case for anchoring floating **Text** strings), but the Anchor layer command always works, and it's fast to use, if you remember the short cut key for it (`Ctrl-H`, if you don't assign another key to this function).

MERGE VISIBLE LAYERS



Final, merged layer should be:

- ● Expanded as necessary
- ○ Clipped to image
- ○ Clipped to bottom layer

This function *merges* all layers which have the little eye icon turned on. Invisible layers are not affected. The options **Expanded as necessary** and **Clipped to image** result in the same: the final layer is just big enough to fit all layers, but no larger. **Clipped to bottom layer** results in a final layer with exactly the same dimensions as the bottom one.

FLATTEN IMAGE

Merges *all* layers without exception. Remember that you must use **Flatten Image** before saving your image, if you mean to save it in any other image format than `XCF` and `GIF`. (Also, remember to erase your **Alpha Channels**, if you have made any in the **Channels** folder - many image formats and printers don't understand Alpha Channels).

ALPHA TO SELECTION

Alpha to Selection is a very powerful command. Alpha to Selection is used when you want to transform opaque or semi-transparent shapes in a transparent layer to a selection.

This command is so useful, that you use it nearly every time you work with a layered image. Think of it, the form, or shape of things is what you mostly work with in the different layers. If you work with an image of, say, a saucer, you'll want to place a copy of the saucer shape under the actual saucer to create a saucer-shaped shadow. Then, you want another saucer copy to accentuate highlights in the porcelain, and another to show dark areas, and another with a special pattern on it, and another...etc. Get the hint? Every time, you just want the empty shape or the selection of the saucer, not a copy of an image with a saucer in it. This is why Alpha to Selection is so important. The pure shape of an object in the form of a selection goes through all layers, you just activate the layer you want it in, and start working!

MASK TO SELECTION

Mask to Selection is the equivalent of **Alpha to Selection**, but for *masks*. You usually don't use this option as often as Alpha to Selection, but if you've made a good mask it is great to be able to use it again in another layer.

ADD ALPHA CHANNEL

If you wish to add alpha information (be able to work with transparency) to the background layer, you can use this option. There are other ways to accomplish this, but **Add Alpha Channel** is fast and easy.

ALIGN VISIBLE LAYERS

This command is useful when you have lots of small layers, and you want to position them in a very exact way. It is, of course, quite ideal for correcting frames in `GIF`-animations. The **Align** command only works with visible layers (with the eye icon turned on), so if you only wish to align certain layers, that's no problem.

HORIZONTAL STYLE

Horizontal style controls how layers are positioned (horizontally) in relation to each other.

- **None:** No change in horizontal position
- **Collect:** Top layer controls where the lower layers end up. **Collect** causes all lower layers to snap their chosen edge to the horizontal coordinate of the top layer's chosen edge.
- **Fill (left to right):** Places the layers from left to right, starting with the top layer. The distance from the chosen edge of a layer to the next is always the same. If the top layer's chosen edge is already to the left of the others, it will not change horizontal position.
- **Fill (right to left):** Corresponds to the above
- **Snap to Grid:** The chosen edge will snap to the nearest horizontal grid line. A chosen corner will snap to the nearest node in the grid system you made.

HORIZONTAL BASE

Horizontal base controls what edge the layers snap to or fill from

- Left Edge
- Centre
- Right Edge

VERTICAL STYLE

Vertical style controls how layers are positioned (vertically) in relation to each other

- **None:** No change in vertical position
- **Collect:** Causes all lower layers to snap their chosen edge to the vertical coordinate of the top layer's chosen edge.
- **Fill (top to bottom):** Places the layers from top to bottom where the distance from the chosen edge of a layer to the next is always the same. If the top layer's chosen edge is already to the top of the others, it will not change vertical position.
- **Fill (bottom to top):** Corresponds to the above
- **Snap to Grid:** The chosen edge will snap to the nearest vertical grid line. A chosen corner will snap to the nearest node.

VERTICAL BASE

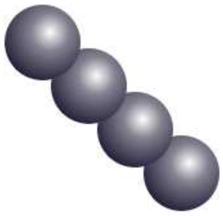
Vertical base controls what edge the layers snap to or fill from

- Top Edge
- Centre
- Bottom Edge

COLLECT

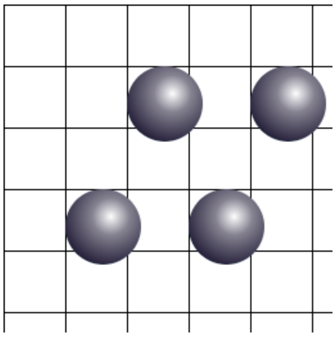
Collect is good for stacking objects in horizontal or vertical rows. If you want to place your layers exactly on top of each other, this is the option to use (choose **Collect/Centre** for both **Vertical** and **Horizontal style/base**)

FILL



Fill is good for animations where you wish to move an object with equal steps for each frame. If the layers are placed on top of each other, or very close together when you use Fill, you'll first have to move them apart to get an acceptable result. *Moving small layers* is very easy. Clicking on a small layer with the **Move tool** automatically makes that layer active, so you can drag and drop layers easily without having to choose them in the Layers dialog. There is no sure way of controlling the distance between layers, but as a rule, the maximum distance is achieved by spreading the layers as far as possible before aligning. Aligning with Fill several times, reduces the distance between layers. Each time you use it, the layers get closer, until the layers' chosen edges touch. A tip is to lock the layers after using Fill. If you *link* them with the **anchor** symbol, you'll be able to move all of the layers in the sequence to a suitable place in the image.

SNAP TO GRID



Snap to Grid is an indispensable option when you're planning an animation. The problem is that the grid you create with Grid size is invisible. To be able to use this option in a productive way, you have to make the grid visible. The solution to this problem is to create a new full-sized layer, use **Grid** in the **Filters/Render** menu, and set the size in this grid to the same as the invisible Align-grid. Remember that different layers may well snap to different nodes in the grid - each layer will snap to the node/line which is nearest to its chosen edge/corner. The best way to work with Snap to Grid is to place the layers close to the place you want them, and then align them perfectly with this option. If the grid system isn't enough for your needs, you can always correct the position afterwards by dragging **Guides** to the position you want, and use **Snap to Guides** (default in the **View menu**).

MISC.

Ignore the bottom layer even if visible is set as default. This is wise in most cases. When you have created layers with a fix background, you don't want the background to start moving.

Use the (invisible) bottom layer as the base treats the bottom layer as any other layer, it aligns the background or bottom layer together with the other layers. Use this option if you want to make a `GIF`-animation with transparent background.

Grid size: Here you can set the size of the invisible grid. The grid squares measure up to 200 pixels.

IMPORT LAYERS

With this tool can you **import** one or more layers from a another image. The **source** image can be bigger or smaller than the **target** image. The interface is simple and easy to use. There is a menu where you select an image to import layers from, and several options for selecting what layer(s) you want to import. You can make importing easier by bringing up the **Layers & Channels** dialog. As you can see in the interface, you can import the layers in the way they appear in the source image, or in reverse order. You can also choose what layers you want to import; **linked** (the little anchor), **visible**, **selected** or **all layers**. To import a layer, press **OK** or **Apply**. Pressing Apply will keep the dialog open after importing, so you can go on importing layers from other images.

ADJUST LAYERS

When you add a small layer to an image, you may want to adjust that layer in relation to another layer or the image border. This tool will help you do this in several ways. You must first choose what layer(s) or image border you want **Adjust** to be relative to. The second choice is to specify the layer(s) you want to adjust. You specify these adjustments with four check buttons. As you see in the table below, there are sixteen different combinations:

Left	Right	Top	Bottom	Result
X				Adjust to the left. No vertical change
	X			Adjust to the right. No vertical change
X	X			Centre layer horizontally. No vertical change.
		X		Adjust to the top. No horizontal change
X		X		Adjust to the top left corner
	X	X		Adjust to the top right corner
X	X	X		Centre layer horizontally, and adjust to the top
			X	Adjust to bottom. No horizontal change
X			X	Adjust to the bottom left corner
	X		X	Adjust to the bottom right corner
X	X		X	Centre layer horizontally, and adjust to bottom
		X	X	Center layer vertically. No horizontal change
X		X	X	Centre layer vertically, and adjust to the left
	X	X	X	Centre layer vertically, and adjust to the right
X	X	X	X	Centre layer in all directions

You can also select a layer in the drop down menu. It’s wise to have the **Layers & Channels** dialog up, so you can set the layers that should be **linked**, **visible** or **active**. All these functions makes it easy to adjust several layers against the image border or another layer. You can for example use a combination where you adjust *all linked layers* against *all visible layers*.

MOVE LAYER

Here’s a possibility to move one or several layers in a more exact way than you can achieve by hand. In the drop down menu you select a layer to move. The slides set distance and direction of the movement. If you want to move more than one layer you have to check **Affect visible layers** or **Affect linked layers** or both. If you don’t check any of these buttons, only the layer selected in the drop down menu will be affected. You must also choose how to move the layer(s); **relative** or **absolute**. With Relative, you will move the layer(s) with the ***selected layer’s current position*** as reference. With Absolute mode, you will move the layer(s) with the ***image border*** as reference.

chapter 18

Channels, Floating Selections and Duotones

Channels are a resource that beginners often leave unused in programs with layer capability. Learn how to use them, and you can't be without them!

CHANNELS

If you click on the tabbed folder marked Channels next to Layers, you'll get closer acquainted with the wonderful world of RGB- and Alpha Channels.

RGB CHANNELS

When you open the **Channels dialog**, you'll see the three **RGB-channels**. Each channel represents the Red, Green or Blue color value of the pixels in your image. Think of the RGB channels as grayscale representations, where"white" is replaced with 100% color. Pure red in the red channel is the equivalent of a red value of 255 for that pixel, black means that the pixel has no red in it at all. As opposed to Layers, *the RGB channels are all active at the same time*. The RGB channels are also equipped with **eye icons**, so you can choose to watch your image in a single color channel. If you want to, you can choose just to work in one or two specific color channels, by clicking at them (click at an active channel and it becomes inactive, and vice versa).

Normally, there isn't much use in working directly in the color channels. If you are very skilled, and like to experiment with patterns and advanced coloring, this can be of interest though. You can for instance erase or add parts to the red channel, or try a filter on the blue channel etc. Just remember that the channel operation only affects the active layer. Also note that you can't use the standard edit functions like **cut/copy/paste** in a single RGB-channel. Moving selections in a color channel is also not possible without moving the content of all three RGB channels. If you want to do any of these things, use the **Compose/Decompose** option in the **Image/Channel Ops** menu.

ALPHA CHANNELS

No, the real reason to use the Channels dialog is that you can **store** and **edit** selections in Alpha Channels! Whenever you make a selection which is more complex than a square - save it! You are most probably going to need it later.

WHAT ARE ALPHA CHANNELS?

The **Alpha** value describes the amount of *transparency* in a pixel. Just as each pixel has a value for the Red, Green and Blue channels in the RGB system, it also has an Alpha value. Check this out with the Color Picker! As with the RGB-channels, pixel values in an Alpha channel range from 0 to 255. The max value (255) refers to 100% opaque, and 0 means totally transparent. When you create a new Channel, you create a mask which can be translated to an Alpha selection and be applied to a layer. These channels are always grayscale images (you can use colors, but they will appear as shades of gray in the channel). Black represents transparent and white opaque, the grayscale in between represents different levels of transparency.

STORING SELECTIONS

In the Select menu there is an option called **Save to Channel**. When you have made a selection, choose the Save to Channel option, and open the **Channels dialog**. You'll find that you now have a new channel called **Selection Mask copy**. The channel is a grayscale version of the selection you just made. Don't forget to name the channel - this is important, because selections stored in channels aren't always easily recognized. To name the channel, double-click its old name, and a dialog box appears. Name the new channel *saucer shadow* or whatever your selection represents (or will represent when you have edited it). You can set a **fill opacity** value in the popup dialog box, but that doesn't affect the alpha values in your channel. The reason you set the fill opacity to a transparent value is that this enables you to see your image through the black parts of the channel. If this was not the case, it would be impossible to edit the channel properly.

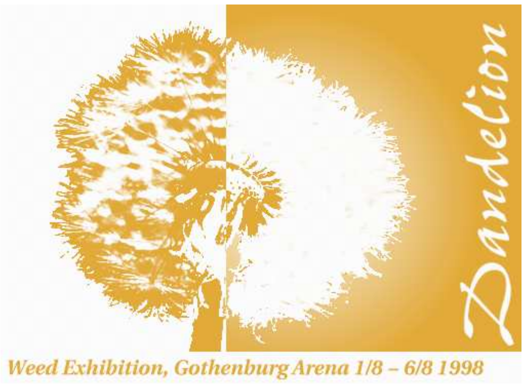
When you want to use the selection you just stored, click the right mouse button on the channel's name, and the **Channels menu** appears. Choose the last option on the drop down list called **Channel to Selection**. This option creates a selection based on your channel - just switch to the **Layers folder** and your selection is ready for use!

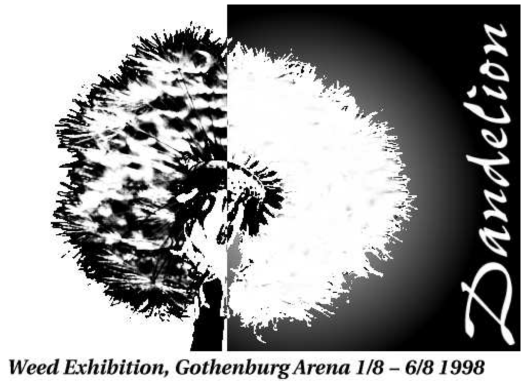
EDITING ALPHA CHANNELS

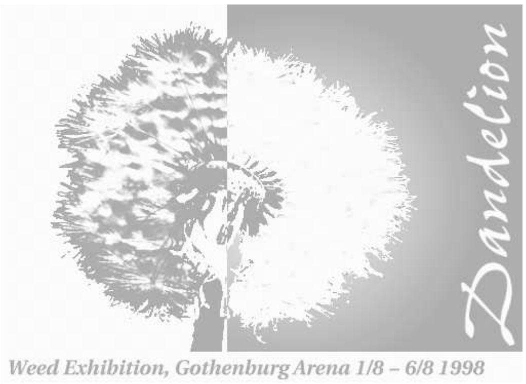
The Channels menu should be easy to understand. You can create a **new** channel, **raise** or **lower** your channels, **duplicate** or **delete** them and change them into **selections**. You can't merge channels, unless you first copy them and paste them into layers. You can merge those layers with the **merge visible layers** command, then you copy or cut the layer and paste it into a channel again. As you see, the contents of alpha channels and layers can be edited just as an in ordinary images. You can paste an image or a layer into an alpha channel, and vice versa. You can paint, adjust image values, make selections or use filters on Alpha channels. This makes Channels the most powerful tool for creating advanced selections there is!

USING CHANNELS FOR CREATING SPOT COLOR SEPARATION

ADDING COLOR TO A CHANNEL







We have already discussed the fact that you can change the name and fill opacity value in the Channel pop up dialog box, but you can also change the **mask color** of an alpha channel. Note that setting another color than black will not turn your alpha channel into a regular color channel like the RGB channels. The RGB channels use an *additional* color model, where maximum color in all three channels equals white. Alpha channels however, can be compared to grayscale layers in *Multiply* mode. White areas are transparent, gray areas are more or less transparent and black areas are totally opaque.

A colored channel can never get darker than 100% of the mask color, no matter how you change *Contrast*, *Curves* or *Level* values. Where 100% cyan and 100% yellow produce a green color in a layer with Multiply mode, a yellow channel on top of a cyan channel will only display yellow in the parts which are 100% black in the yellow channel. So as with printing plates, it is very important to know in what order the channels (or plates) come.

The pictures to the left show the difference between colored **layers** and colored **channels**. Starting from the top: a yellow channel or a yellow layer looks just about the same (like the top image). Now, if you were to copy the layer/channel and paste it into a grayscale image, the channel representation would look as picture number two; crisp, clear and black where there was 100% yellow in the colored channel. The layer representation in picture number three, however, looks washed-out and gray, because the yellow brightness values have been translated to light shades of gray. This grayscale image cannot be used for making a printing plate.

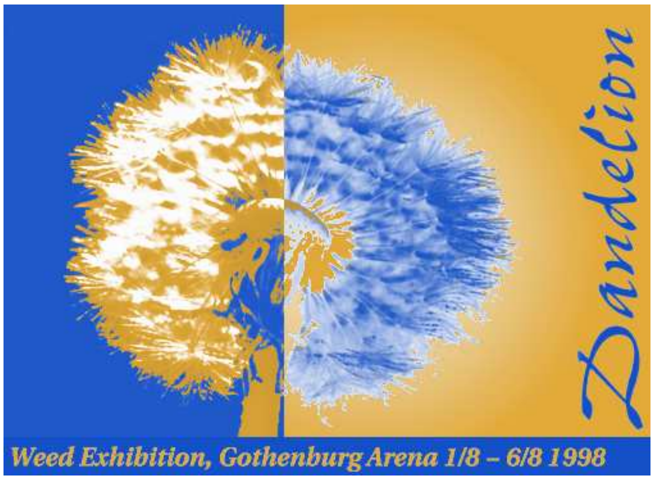
SPOT COLORS

The reason I mentioned printing plates is that the prime use for colored alpha channels is to create **spot color separations**. Spot or custom colors like PANTONE or TRUMATCH are used when you don't want, or can't afford four-color CMYK printing. Custom colors come in all possible shades, including special inks and varnishes (like gold, silver or fluorescent inks). Designs for spot colors can be very attractive, and has the great advantage of letting you print solid color without a halftone pattern. Using fewer plates than four is also a cheaper alternative when printing smaller series, and it will often look much better than a plain black and white print.

DUOTONES

What is a **Duotone**? Simply put, a duotone is an image where you use the same grayscale image to produce two (almost) identical printing plates. The difference is that the screen angle is set differently, so that the little ink dots will not end up in the exact same spot. Remember that when you're using several plates to compose one image, you have to use less ink (especially black ink) to prevent the printed result from getting too dark.

The result is often a very eye-pleasing, softly tinted image, with more depth to it than a plain black&white image. Typically, duotones are created with one black plate and one colored plate, where the chosen color is often blue or sepia brown (the classical inks used in traditional ink drawing and lavure painting). You don't need Channels to create a duo- or multi-tone, but you'll need them to edit and preview your work, and you'll definitely need them if you want something more than a uniform color distribution.

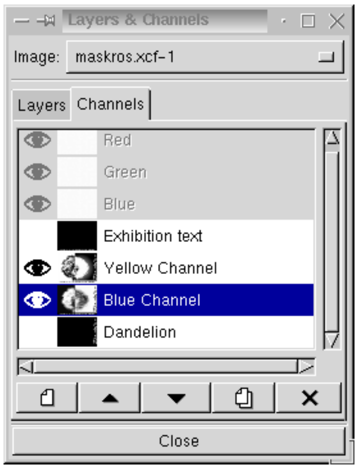


CHECKLIST

To create a spot color multi-tone, there are a few things you must first take into consideration.

1. What spot colors should I use? You can choose from a great variety of custom colors, so you can go about in two different ways. You can go to your local print shop, take a look at their color charts and ask them to make a (color calibrated) **proof** of your chosen colors to take home, or you can buy a PANTONE **color chart** (quite expensive), and specify (to your printer) the custom colors that look most like the RGB colors you have chosen for your alpha channels. Remember that color in alpha channels is for preview only. All channels are grayscale.

2. In what order should I print the plates/put the channels? Always ask your printer's advice about this. Ask in what order they prefer to print the different inks, and the approximate density of the last ink to be applied. You'll probably also better discuss what screen angle and dot gain your printer thinks would be appropriate for this print job and whether you should use AM or FM screening. As always, make a color proof that you're satisfied with, and tell the printer to adjust the final settings to look like your proof.



For more information on the printing procedure and pre-press, see chapter 13.

HOW TO CREATE A DUOTONE

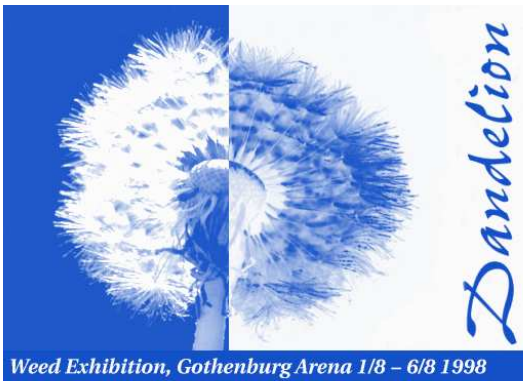
Start by **opening** the image you want to turn into a duotone. If the image contains color information, you should first **desaturate** it and increase the contrast.

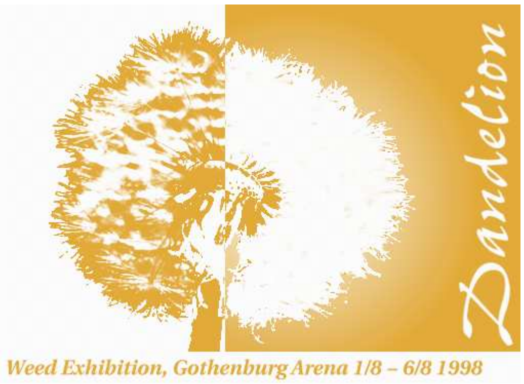
Copy the image and then **clear** the background to white. Make sure that the little eye icon is turned on in the white background layer.

Turn to the Channels tabfolder and create a new **Channel**. Set this channel to 100% **Fill Opacity**. Click in the color swatch to access the color selector dialog box, and choose a nice color (the first plate to be printed). Name the channel ”navy blue” or similar, depending on which color you have chosen.

When this is done, you can **paste** the image into this channel. Repeat these steps for the other channel (the second plate to be printed), but set the Fill Opacity to 85% this time (or the value that your printer thinks is closest to the density of this ink).

Now you can start manipulating the content of the channels to a result of your liking.





If you want the pure color of a certain custom ink or dye to show in a certain part of the image, remember to save the correspondent selection in a new (black) channel. Load the selection on the target color channel and fill it with black, then change to the other color channel and fill the selection with white. This way you'll make sure that the selection is placed in the exact same position in both channels and you'll avoid trapping problems due to differences between the two grayscale images.

When you're satisfied with the result, **duplicate** the image file, **copy** and **paste** the contents of the two channels to the white background (channel no 1 to the original and channel no 2 to the duplicate). **Delete** the channels and convert to **grayscale**. **Save** grayscale no 1 as i.e ”blue.tif” and grayscale no 2 as ”yellow.tif”

Give these files to your printer or service bureau, specify the custom colors and tell them in what order the plates should be printed. Ask your printer to set an appropriate screen angle and dot gain. And remember, they are the experts, you are well advised to follow their instructions.

FLOATING SELECTIONS

WHAT IS A FLOATING SELECTION?

When you first place a text string, move a selection or choose **Float**, your free, empty selection is transformed to a floating selection with a pixel content. This means that the selection contains information which isn't attached to any layer - it floats independently, and changes to a floating selection can't affect the rest of the image (all layers are grayed out). You're restricted by the ”marching ant” border in floating mode, but you can still do a few things:

A. *Change the color, Image options or use filters on your selection*

B. *Use the Eraser to reduce the selection, or edit it with cut/copy/paste etc.*

C. *Move the selection.*

The one thing you can't do with a floating selection, is using any of the **Select** commands.

Anchoring a floating selection

If you choose **Anchor layer** in the **Layers menu**, the float will merge with the layer that was active before you placed the float.The simplest way to anchor a floating selection is of course to mouse-click anywhere in the image (outside the floating selection if the Move tool is active). If you save a float as a **new layer** *by double-clicking its icon in the Layers dialog*, the new layer's **size** will be the same as the float. In Photoshop, all layers are the same size as the image, but Gimp saves disk space by adapting the layer size to the size of the floating selection. The dotted yellow layer border constitutes the boundaries of the drawable surface in a layer. This means that you can't paint or make a selection outside of this border, and if you use the Move tool, you'll move the entire layer around.

Moving objects in a floating selection layer

To move an image object in a layer made from a floating selection (like a character in a text string), you must first select it (the lasso or the wand is a good choice). As soon as you try to move this letter, it will instantly turn into a new floating selection. When you have moved it and released the mouse button, you have to change to the **Move tool**, or you'll just get irritating subselections (as discussed in the Moving Selections chapter). To get around this, don't move the selection with that first automatic Move-cursor. Instead, select **Float** in the **Select menu**, then choose the **Move tool**, and feel free to move your letter. When you're happy with the position of the letter, anchor it to the layer with **Anchor Layer**. Remember that a text layer will be very small and snug, unless you have specified a **Border** in the **Text dialog**, This means that to be able to anchor single text characters to the text layer, you may have to increase the layer size with the **Resize layer** command.

Tips on working with floating selections

As you may have noticed, floating selections work a bit differently in Gimp than in Photoshop. If you want to apply any of the selection options in the **Select** menu to an image object on a transparent layer (like a text string for instance), it's a good idea to work like this:

- Use a different layer for each effect.
- Put the float in the top layer. Just ***double-click*** at the floating selection in the Layers dialog, and you'll automatically place it in a new layer called **Floating Selection** or **Text layer** if you're placing a text string. Note that, in this layer, you can't use the Select menu options yet, because nothing is selected.
- To achieve effects like **Border** or **Feather**, you must first select the image object with the **Alpha to Selection** command. This results in an ordinary non-floating selection, because this command selects everything with an alpha value > 0, i.e. everything opaque or semi-transparent in the layer (If Alpha = 0, that pixel is transparent)
- If you stay in the floating selection layer after doing this, you'll have to uncheck the **Keep Transparent** button, to work with a changed selection size. Remember that in this layer, you must use the Keep transparent option if you want to fill the text with color (you also must remember to **Select All** to paint all letters at the same time if you use the **Bucket fill**). If the Keep transparent option is unchecked, you will fill the entire layer with color and you won't be able to see your text anymore (not too smart).

- The floating selection layer has a reduced **size**, which means that filter effects and large selections are limited, they get ”cut off”, because there isn’t enough space. Also, if you make too many changes in the original layer, the pure form of the shape (especially for letters) will soon disappear.
- It is better to change to a lower layer to make special effects (the selection goes through all layers, and you can choose any layer you want to work in). In these other layers, there is no tiny layer size to restrict you, and you don’t need to think of Keep Transparent.

part VI

Filters

- Animation
- Artistic
- Blur
- Color
- Combine
- Cryptographic
- Distort
- Edge-detect
- Enhance
- Generic
- Glass effects
- Light effects
- Map
- Misc.
- Noise
- Render

chapter

19

An introduction to filters

A short description of how filters generally work in Gimp

PLUG-INS

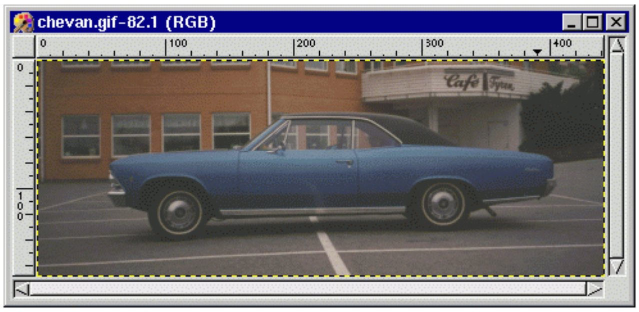
When an ordinary Photoshop user thinks of plug-ins, things like **Eye Candy** and **Kai's power tools** comes to his or her mind. Gimp plug-ins does the same; they permit the user to add *extra features* to Gimp. With features, we mean **filters**, **printer drivers**, **mail interfaces**, **save/write modules** etc. This is also true for Photoshop and similar Windows programs. As far as we know, the Gimp is highly modularized, so nearly every function beside basic Gimp stuff has to be done by plug-ins.

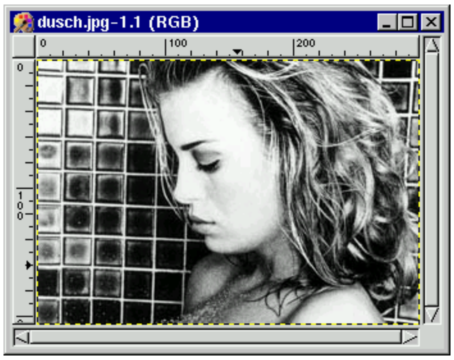
Many Gimp users/developers have made Gimp plug-ins, which are available to the Gimp community. We encourage you to do the same. If you have made your own plug-in, submit it to the Gimp community under GPL licence! It will make Gimp even greater.

In this chapters, we will discuss the **filters menu**, and we will call these filters plug-ins, because that's what's most people think of when the hear the word plug-in. The Script-Fu image menu is similar to the filters menu because these Script-Fu:s can be applied as ordinary filters to your image. You'll find that you can make your own filters quite easily, without expert knowledge of C programming and GTK libraries.

Since plug-ins and Scripts develop rapidly in the Gimp community we at Frozenriver can't keep this chapter as updated as we would like to do, so we encourage developers to send us a mail about their new or changed plug-in so we have a easier job updating this chapter. In chapter 41 you'll find some tips on how to compile plug-ins. When we use the term ***image*** we mean (most of the time) both image and drawable. It can also be wise to visit the filter developers home pages to get uptodate information about the filter.

In this chapter and following filter chapters we'll often use the same pictures when we want to show the result of the filter. Two colored images and one grayscale. The screen dump of the plug-in dialogs and their value isn't necessarily the same as we used to generate the outcome. The reason for this is that we sometimes like to exaggerate a bit so you can really see what the filter does.





THE MAIN CATEGORIES

In the filter menu you'll find the following sub-menus, which also groups the plug-ins by function.

- **Animation**: Here you'll find an animation player, which lets you play your Gimp animation. You will also find an animation filter that can optimize your animation, so that it will use much less disk space.
- **Artistic**: Here you'll find filters that create instant artistic effects. You easily create cubist paintings, mosaic patterns etc. You mainly use this kind of filter for adding special effects to an image, but you can also create nice patterns and many other things. Just to use your mind and imagination, and there will be no end to what you can achieve.

- **Blur**: Includes many different types of blur filters. Blur is useful when you want to soften a part of picture. Real shadows are seldom hard and solid, so to create realistic shadows you'll want to soften them up with an appropriate blur filter. A portrait may look too honest and show all the skin imperfections and wrinkles of the model - then blur comes in handy. (This trick is used all the time in modeling magazines, that's why they always look so perfect. (But if your model is ugly, blur won't help).
- **Color**: Here you'll find a whole bunch of tools that can manipulate color and HSV values, just as if you were standing in a darkroom.
- **Combine**: Here you'll find many different ways of combining several images to create a new image.
- **Crypt**: These filters let you sign, encrypt/decrypt your image or send hidden files.
- **Distorts**: Experimenting with distort filters is like entering a gallery in the hall of funny mirrors. Some of these filters are great for adding special effects to an image, like making ripples in a water surface for example, and if you want to create textures, you'll find many useful filters here.
- **Edge detect**: These filters help you find the edges or color boundaries in an image. It can be quite useful when you work with layered images, and you want to strengthen (or smooth) the contours of an object. You can also use it for making easy selections with the magic wand, or easy fills with the bucket fill tool.
- **Generic**: Here you'll find mathematical filters that use a matrix for image manipulation. You can perform all kinds of manipulation with these filters, but you may need some math up your sleeve.
- **Glass effects**: These filters create different kinds of lens- or curved mirror effects.
- **Light effects**: Light effects lets you add a little glamour to your design, like extra shine or lustre, glitter or star reflections.
- **Map**: If you want to bump map, displace or alter you image in a relation to a map, this is where to look.
- **Misc.**: Here are all filters that don't fit anywhere else; currently stereogram filters and video screen simulation.
- **Noise**: If you want to add some noise to your image, this submenu offers three different noise filters.
- **Render**: These filters will render all kinds of shapes or objects, and are extremely useful for creating textures or patterns.

Notice the handy short cuts; `Shift+Alt+F` to bring up the last plug-in you used (if you haven't reassigned the short cut), and `Alt+F` to apply the last filter once more.

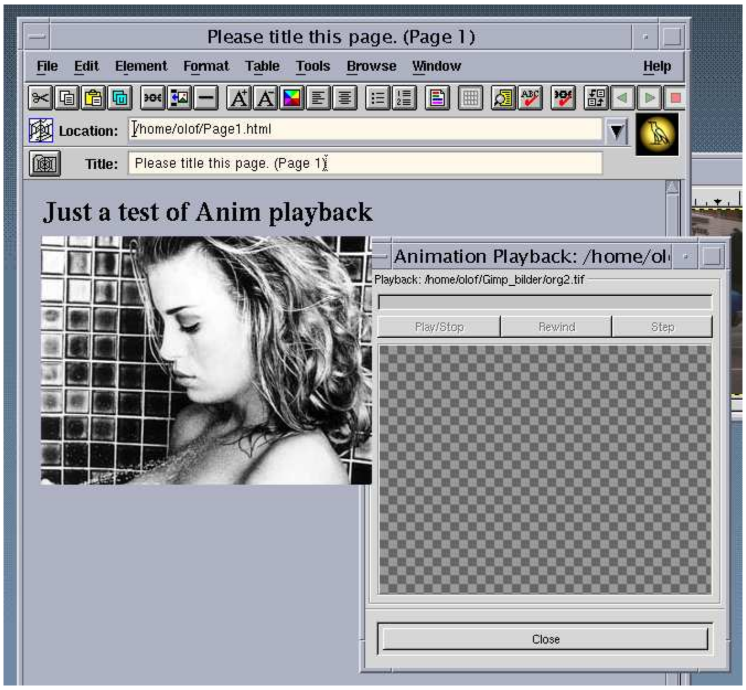
chapter

20

Animation filters

Different animation filters and functions as well as a description of Gif animation in Gimp.

ANIMATION PLAYBACK



Location: /home/olof/Page1.html

Title: Please title this page. (Page 1)

Just a test of Anim playback

Playback: /home/olof/Gimp_bilder/org2.tif

This plug-in will play back your **layers** or **GIF animation** like a film. If the image isn't a GIF, it will run each layer in **Combine mode**. You can also step each frame.

This is the basic function of Animation playback, but there is another extremely useful function in this plug-in:

You can *grab* the image in the playback frame, *drag* it out of the frame and *drop* the image on a web page to see what it will look like. Note that the image doesn't need to be a layered (i.e playable image) in order to use this function. You can use it on any ordinary jpeg or tiff image.

ANIMATION OPTIMIZE

As a **GIF animation** is built of many **layers**, some of the layers will probably repeat much of the information in the previous layer (the previous frame in your animation). Wouldn't it be great if you could skip all that unnecessary information? This is quite hard and time-consuming to do by hand, so here's a filter that can do it for you! Apply this filter to your animation, and take a look at your layers afterwards. You'll find that the layers are much smaller, only additional or diverging information is displayed. Now your homepage GIF animation will be remarkably smaller and faster to download. You can also use this filter on large, multi-layered XCF files; they will occupy a lot less space on your hard disk.

ANIMATION UNOPTIMIZE

When you have an optimized an image (by the optimize filter or by hand), it's usually quite hard to scale and your manipulations will often result in a bad output.The **unoptimize** filter solves this problem, so use this filter before you make any further alterations to an optimized image.

FILTER ALL LAYERS

This filter is used for creating animations with the **Gap plug-in**, (see Chapter 36 **Anim Frames)**, but you can of course apply it to any kind of multi-layered image. When you select this filter, a **browser** will appear. It is much like the DB Browser, except that only plug-ins are listed. Select the filter that you want to apply, and specify how by selecting *constant* or *variable*. When you have made your choice, the normal filter dialog will pop up asking for values. If you chose **constant**, the dialog will pop up only once, and the values you specify will apply to all layers. If you chose **variable**, the dialog will pop up for every layer so you can apply different values to each layer.

HOW TO CREATE A GIF ANIMATION

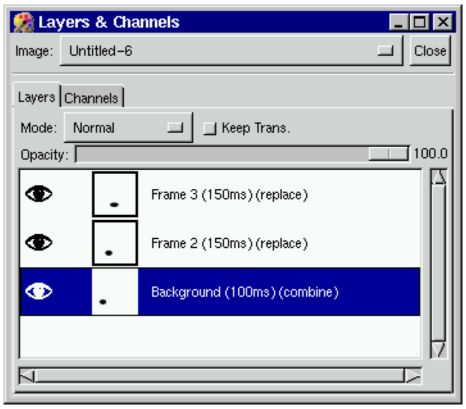


Image: Untitled-6

Mode: Normal

☐ Keep Trans.

Opacity: 100.0

Frame 3 (150ms) (replace)

Frame 2 (150ms) (replace)

Background (100ms) (combine)

If you want to make a **GIF animation**, Gimp is the ultimate tool. This is how it works:

Gimp treats each **layer** as a **frame**. The Background layer is Frame 1. and each new layer will be a new frame. When you add a new layer to the background layer, name it Frame 2, Frame 3 and so on.

SPECIFYING THE DELAY OF EACH FRAME

Edit the layer name by double-clicking in the Layers dialog, and rename it **Frame X (xxxxms)** where *X* is the frame number and *xxxx* is the delay in milliseconds. Naming a frame **Frame 5 (100ms)** will give that frame a delay of 100 ms.

COMBINING FRAMES

To make each layer combine *(Combine means that Frame 2 will be added to Frame 1 (the background), Frame 3 will be added to Frame 2 and Frame 1, and so on)*, just name the layer **Frame X (xxxxms) (combine)**.

REPLACING FRAMES

To make it work like in a real movie, i.e. each new frame will replace the former, add **(replace)** instead of **(combine)**. When saving your GIF, don't check **Don't care** because that will make the layers combine without showing up in the Gimp layer dialog as (combine). You can (combine) and (replace) in any order.

An example:

- **Background** (100ms) (combine)
- **Frame 2 (**100ms) (replace)
- **Frame 3** (100ms) (replace)
- **Frame 4** (100ms) (combine)
- **Frame 5** (100ms) (replace)

will be played back like this:

- Background
- Background + Frame 2
- Frame 3
- Frame 4
- Frame 4 + Frame 5

all with 100ms delay.

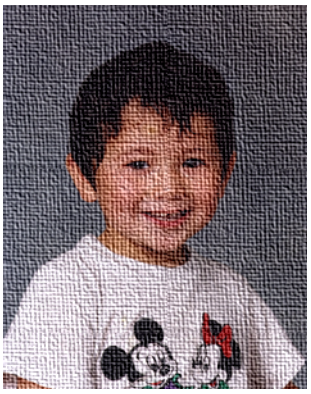
chapter

21

Artistic filters

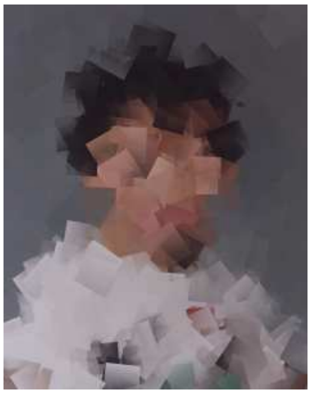
Filters that create instant artistic effects such as oil paintings or mosaic floors. In general they are ready-to-run filter that demand little attention from the user.

APPLY CANVAS



This filter applies a **canvas** structure to your image or selection. This will obviously making it look more like a painting. You can specify four different canvas directions, as well as how ”rough” the fabric should be. The **Depth** slide bar controls the canvas structure. A high value will get you a very rough and prominent canvas texture while a low value results in a softer, smoother canvas.

CUBISM

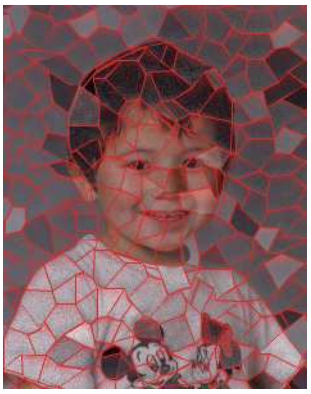


Transforms your image to cubist art. If you check **Use background color**, the background color will appear between your tiles, otherwise this area will be black. **Tile size** determines how ”cubist” you want your image to be; higher values result in a more abstract image. The **saturation** value decides how colorful the image will be.

Tip

To achieve interesting effects, try applying cubism to several layers with different modes over the original image.

MOSAIC



With this plug-in you can imitate everything from a stained glass window to a ceramic mosaic floor. This plug-in has many parameters that lets you control the final result; **Size** and **height** of the tiles, the **spacing** between them, and the **neatness** which controls the appearance of the stones.

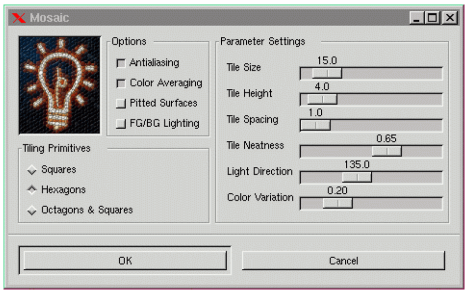
If you use a hexagon shape, the stones will be hexagonal with a high neatness value. If you lower the neatness value, the hexagonal structure will fade, and the stones will look more like the natural stones you find in the open.

Light direction controls how the daylight will appear to shine on the mosaic edges. **Color variation** adjusts how much the color is allowed to fluctuate. With a low value, the original color from the image will be preserved.

Tiling primitives is what kind of mosaic tiles (stone structure) you want as a base for your mosaic. If you set a low neatness value and a high color variation value, you will get a very abstract mosaic. **Antaliasing** produces smooth edges.

Color Averaging creates a true mosaic. If this option is unchecked, the original picture will only get a mosaic texture. **Pitted Surfaces** will get you a surface that looks old and used. **FG/BG** controls edge color. If unchecked, it will use the foreground from the toolbox, BG will use the background color.

TIP



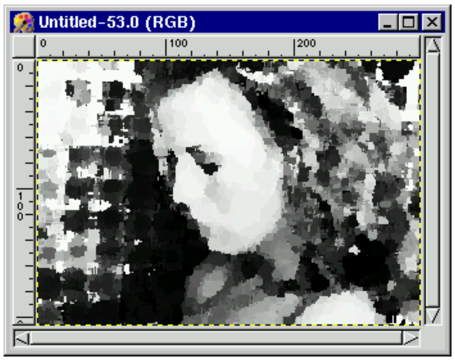
Options
- ☑ Antialiasing
- ☑ Color Averaging
- ☐ Pitted Surfaces
- ☐ FG/BG Lighting

Tiling Primitives
- ○ Squares
- ● Hexagons
- ○ Octagons & Squares

Parameter Settings
- Tile Size 15.0
- Tile Height 4.0
- Tile Spacing 1.0
- Tile Neatness 0.65
- Light Direction 135.0
- Color Variation 0.20

You can create very interesting surfaces with Mosaic. You can for example take a stone pattern and combine it with Mosaic to get something that looks like an old stone floor, or with another combination, perhaps a cracked glass/windshield etc...

OILIFY



As you might guess, it makes your image look like an oil painting. **Mask size** sets the outcome. A high value gives the image less detail (as if you had used a larger brush).

Tip

This tool is nice to use with several layers. You can also use it to help you select intricate objects by running this filter on a duplicate layer; a simplified shape is much easier to select.

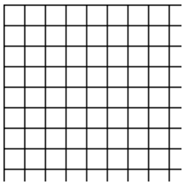
VAN GOGH (LIC)

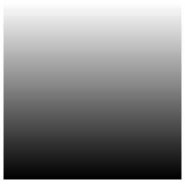
Van Gogh can be used as a **blur** tool or as a **texture** maker. In short, this plug-in has more in common with the texture or displace filters than the artistic ones, even though you can achieve quite artistic results with it.

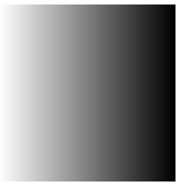
- To create a blur, check **Convolve with source image** before applying the filter.
- To create a texture, check **Convolve with white noise**.

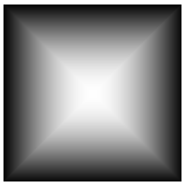
MAP IMAGE

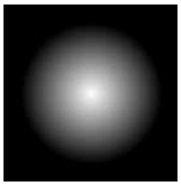
Whether you want to make a pattern or a blur effect, you must first create a **map image**. **Effect Channel** determines which **HSV** channel should be used (brightness is generally best). **Effect Operator** controls the direction of the pattern or blur. **Derivative** sets the direction to the opposite of **Gradient**. If your map image has a certain direction, the Effect Operators work much the same way as **Flip** in the toolbox.


Target image


Vert. Map


Hor. Map


Square Map


Radial Map

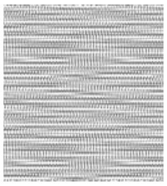
BLUR

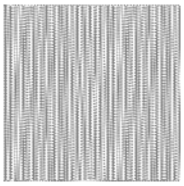
Blur: The advantage of this filter is that the map image determines the **direction** of the blur. This means that you can adapt a gradient to create variation, movement and a sense of direction in blurring. A radial gradient creates a circular blur movement, a horizontal linear gradient (meaning when you drag left to right) puts an emphasis on vertical lines in the image because it blurs horizontal lines, and vice versa.

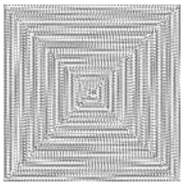
A solid color object in the map image doesn’t create a blur effect (except for the antialiased edges) - you must use some sort of gradient in the areas you wish to blur. I recommend that you use the default settings with the exception of **Filter Length** which controls the strength or depth of the blur, and in some measure also **Integration Steps**. Use a **black/white** gradient map and nothing else. (You can of course use all sorts of image maps, but in most cases that doesn’t accomplish more than a general blur over the entire image).

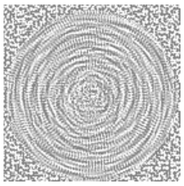
TEXTURE

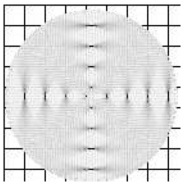
Texture: You can create many interesting patterns and textures with Van Gogh. This filter is especially good for making patterns that look like **woven** or **knitted textiles** or fabrics. Use a target image with a solid color or a color pattern you think would fit your texture. Use a **grayscale gradient**, or **blurred mask** as map image, set **Integration Steps** a bit higher than default, and experiment with the other settings to get the sort of texture you want.

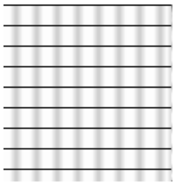

Vert. Pattern

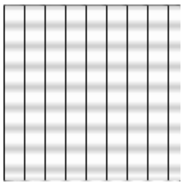

Hor. Pattern

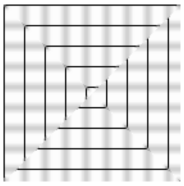

Shapeburst P.

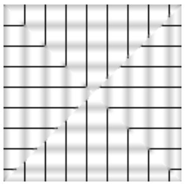

Radial Pattern

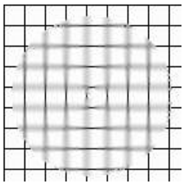

Filter length, Blur


Vert. Blur


Hor. Blur


Shapeburst B.

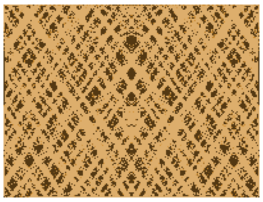

Derivative ShB


Radial Blur

When you use this filter as a texture maker, the settings are more important than when you use it for blurring.

- **Max/Min Value** controls **contrast**. If the Max/Min range is large, contrast is low. The contrast increases when you shrink the interval. *Does not affect Blur*

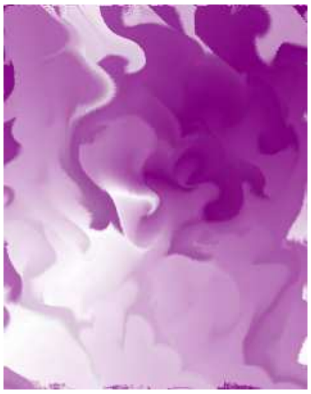
- **Integration Steps** control how much the map gradient is allowed to influence the shape of the pattern (large integration = large influence). *Don't set this parameter to low for Blur, Default is fine*
- **Noise Magnitude** controls the **amount** and **size** of **random noise** (that breaks up the regularity of the pattern). Low values produce finely grained surfaces (sand), and high values produce coarser materials (large bumps and holes). *Does not affect Blur*
- **Filter Length** controls depth just like in the **emboss filter**. Low filter lenght=smooth surface, High filter length=rough surface. *Controls the strength of the Blur (compare the two images to the right)*



Creating a nice burlap cloth texture:

Run the filter twice on a white image. Check **White noise** (for pattern), select a quite small interval for **Max/Min**, and put the other slide bars somewhere in the middle (higher for creating a coarser fabric). Use a *diagonal linear gradient* as map, and set **Gradient** the first time and **Derivative** the second time, this inverts the direction of the pattern. Paste the second result image to a new layer in the first one, and set Lighten only mode (or other suitable mode).

WARP



Warp (or syrup in sour cream) is a nice little filter. I'm not kidding about the sour cream, it's actually a lot like the patterns you made as a child with a spoon in a bowl of thick cream and juicy berries or syrup. To make this work properly, you'll need a **Displacement map**. To create syrup curls, you should use the **Solid Noise** filter in the **Filters/Render** menu for a displacement map (don't use Noise filters from the Filters/Noise menu, that's another kind of noise).

The more detail and small turbulences you have in the map image, the more, frizzier and smaller curls you'll get. Make sure that the map image is the same **size** as the image you want to work the filter on. If you only want to warp a small part of the image, you can erase or cut part of the map image, and only leave an area which is about the same size and shape as the area you wish to warp in the target image. When you wish to specify a more exact position, size and shape of the warpable area, use a **Magnitude Map** instead.

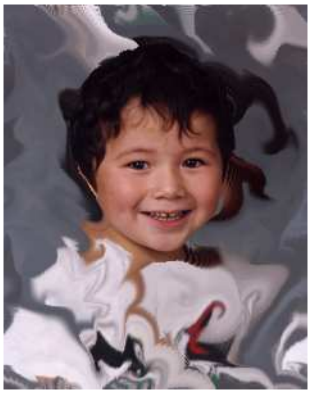
MAIN OPTIONS

- **Step Size** controls the amount, or strength of the filter. The value you set here affects the image for every iteration, or warp step you make.
- **Iterations** sets the number of times the filter should repeat the effect.

- **On Edges** refer to the way the plug-in deals with background color at the distorted edges (see Displace Filter chapter 32)

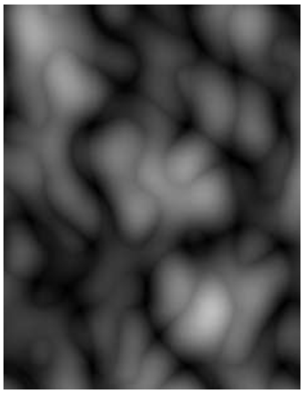
SECONDARY OPTIONS

- **Dither Size** causes pixels to displace randomly, thus decomposing the image. A moderate dither size just makes the warp curls look more grainy. Larger values scale from fuzzy particle clouds to total disintegration of the image.
- **Rotation Angle** determines what the ”curls” will look like. 90 degree rotation, which is the default setting, makes them look like small whirlpools. A 0 degree rotation looks more like the kind of solid distortion you’d see through a bathroom window. Every other angle are combinations of these two, more or less so depending on how close they are.
- **Substeps** increases calculation time for each flow step. There is a slight improvement in warping, but it is much slower.
- **Magnitude Map** is a more subtle way of controlling what parts of the image should be warped and how much. The magnitude map should be a grayscale image, where areas you do not wish to affect are black. Warp magnitude is determined by a brightness scale. White represents 100% (or normal warp), black stands for no warp at all, and the different shades of gray weaken the warp effect.



Warped image

Magnitude Map

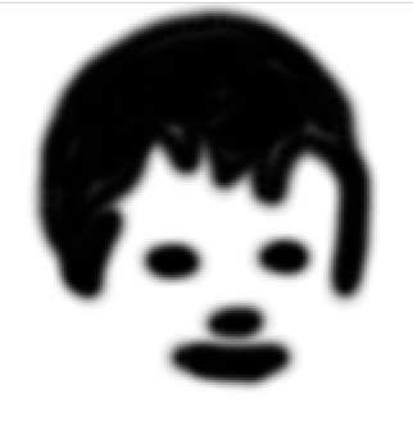


Displacement Map

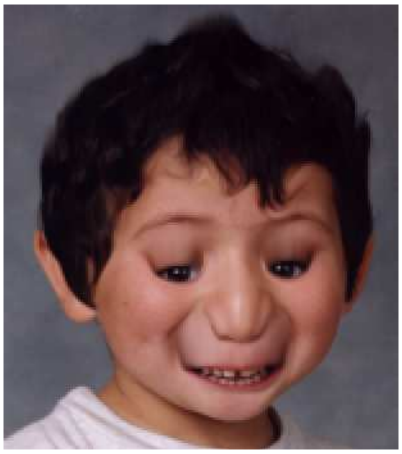
OTHER OPTIONS

- Besides the Warp displacement map, you can add a **Gradient displacement map**. This map will not displace in the curly warps you have seen before. Here displacement depends on the direction of the gradual transition which makes the distortion straight and angular. The **Gradient Scale** sets the amount of influence the gradient map should have.
- You can also displace along a fixed direction by using a **Vector map**. This option lets you displace a chosen map one step per iteration in a certain direction. **Vector Magnitude** determines by how many pixels the image should move for each iteration, and the direction is specified in the **Angle** swatch. A Vector map is something in between a Displace and a Magnitude map. It protects black areas and the rest of the image moves along the vector direction. The smoothness of the stretch is determined by the number of pixels specified, a low value here gives a smooth disfigurement of the image

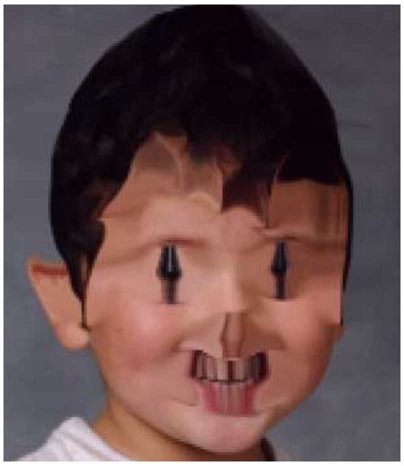
.



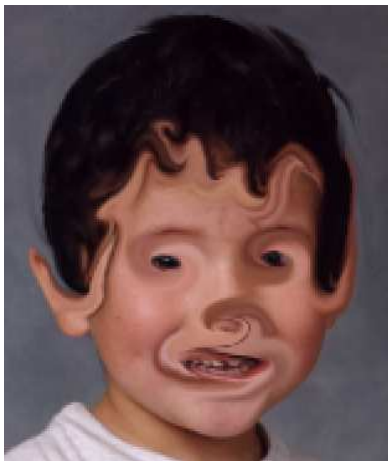
Displacement Map



As Vector Map



As Gradient Map



As normal Warp Map

Artistic filters

chapter 22

Blur filters

There are many ways of creating blur effects. Here you can learn what filter to use, and how to get the kind of blur you want.

BLUR

Blur puts your image out of focus, making it look softer. There are no parameters in this plug-in, so you'll often have to blur more than once to get the desired effect.

Blur filters are often used on certain parts of an image to shift the focus to the sharper parts, to soften hard edges, or to create an illusion of depth or distance.

GAUSSIAN BLUR (IIR)

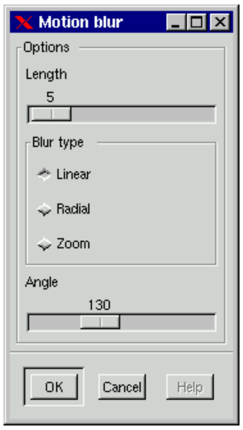
Untitled-53.0 (RGB)
0
100
200
0
100

This is a variable blurring method, based on the **radius** of the blur. Higher radius produce a higher amount of blur (values less than 1.0 are invalid). You can also choose to blur in **vertical/horizontal** direction or both. This makes it possible to create a motion blur with this filter. The **IIR** blur type is best for **scanned images** and other natural (photographic) images.

GAUSSIAN BLUR (RLE):

This is the same as above, but gives the best result on **computer-made images**.

MOTION BLUR



Options

Length 5

Blur type

● Linear

○ Radial

○ Zoom

Angle 130

LINEAR

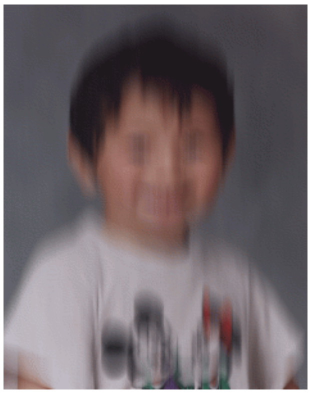
This plug-in simulates a snapshot of a moving object. If the blur type is **Linear**, it will look as if the object was moving beside you. **Angle** is the direction of the motion, and **Length** is the speed of the motion, so more length means higher speed.

RADIAL

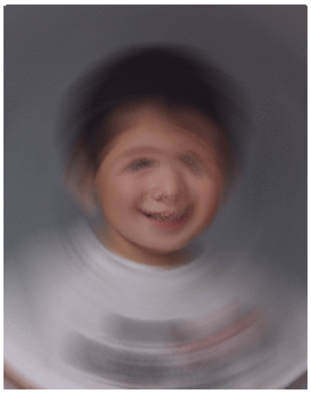
If the blur type is **Radial**, it will look like the object was *rocking* in front of you. In this case, **Angle** determines the amount of ”rocking”. A high value for Angle will make the image spin. **Length** determines how fast the object is rocking/spinning.

ZOOM

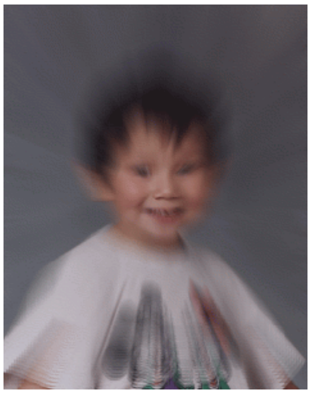
If the blur type is **Zoom**, the object will look as if it was moving away from you. Here, **Length** is the speed with which the object is moving away from you. **Angle** seems to have no impact on this blur mode..



Linear

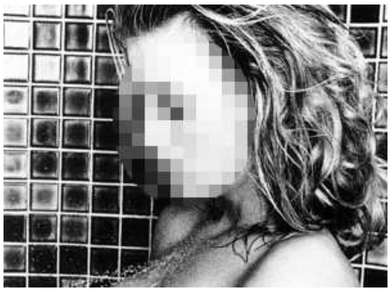


Radial



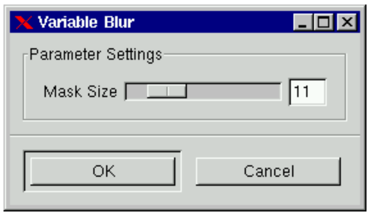
Zoom

PIXELIZE



This filter makes your image look as if it was made of really large pixels (like on TV when they don’t want to show a certain person’s face etc.). You have to set the new pixel size, e.g. **3** makes 3x3 pixels in the original image look like a single large pixel.

VARIABLE BLUR



Variable Blur
Parameter Settings
OK
Cancel

Mask Size 11

Much like ordinary blur, but with the possibility to regulate the amount of blur.

chapter 23

Color filters

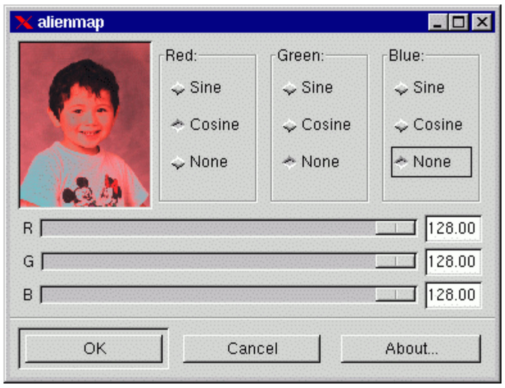
Welcome to the color darkroom! With these filters you can do what you would normally achieve in a darkroom; changing colors, masking colors etc.

ALIEN MAP

This filter applies *trigonometrical* functions to **RGB channels**.

Let’s take the Red channel and see what we can do with it. First set Green and Blue to **None (linear)**. If you now check **Cosine** in the Red channel, the image will get a lot redder. Why? As you can see in the picture, cos will boost the ”low” red values. That’s what makes the difference, because even the tiniest little redness in a pixel will get a lot redder. The parts that already are quite red will also be boosted, but the parts with middle value for red are subdued. The change in mid-values will not be as apparent as the boost in low values, because you already saw that they were red.

If you check **Sine** you will notice that the image will get less red. This is because Sine subdues the low red values and boosts the already high red values. As with Cos it’s a lot easier to detect when a low value is altered. If you haven’t checked **None**, you can also set the intensity of each channel.



alienmap

Red: ○ Sine ● Cosine ○ None

Green: ○ Sine ○ Cosine ● None

Blue: ○ Sine ○ Cosine ● None

R 128.00

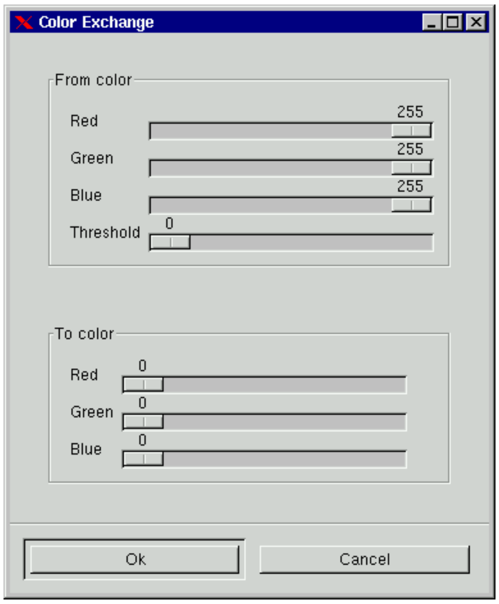
G 128.00

B 128.00

COLOR EXCHANGE

This filter is a lot like **Color Rotate**. Well, it is much simpler, but it is also faster and slimmer. You define the color to exchange in the upper field by setting the slide for each **RGB channel**, and a threshold fuzziness like in **Select by Color**. In the lower field you select the color you want to exchange to.

A tip is to have a color dialog up so you can see what color you are choosing.



From color

Red 255

Green 255

Blue 255

Threshold 0

To color

Red 0

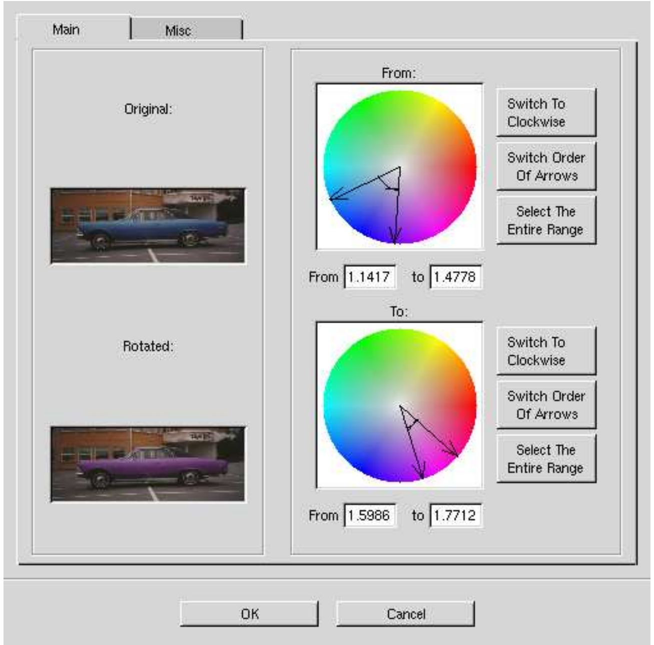
Green 0

Blue 0

COLORIFY

The effect of this filter is much like looking at your image through **colored glass**. There are some predefined colors, but there's also the possibility to set a color of your choice in the **Custom Color** swatch.

COLOR MAP ROTATION


Main
Misc
Original:
Rotated:
From:
To:
Switch To Clockwise
Switch Order Of Arrows
Select The Entire Range
Switch To Clockwise
Switch Order Of Arrows
Select The Entire Range
OK
Cancel

From: From 1.1417 to 1.4778

To: From 1.5986 to 1.7712

Oh no! Our Chevelle is now purple.

This filter lets you change one color interval to another.

MAIN WINDOW

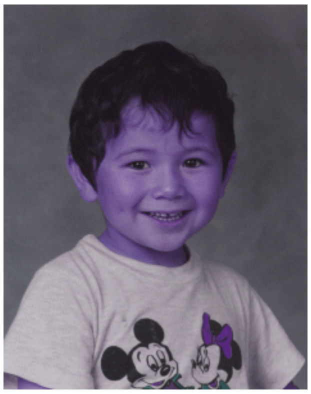
In the **From:** and **To:** color circles you define the color range that you want to rotate (From) and the color range you want to rotate (To).

RANGE

By default the range is going **counter clockwise**, so if you define a range that you want to rotate, the rotation will map your pointers in the **From:** and **To:** color circles. If one of the fields is switched to **Clockwise**, the first pointer in a field will map the second pointer in the other field.

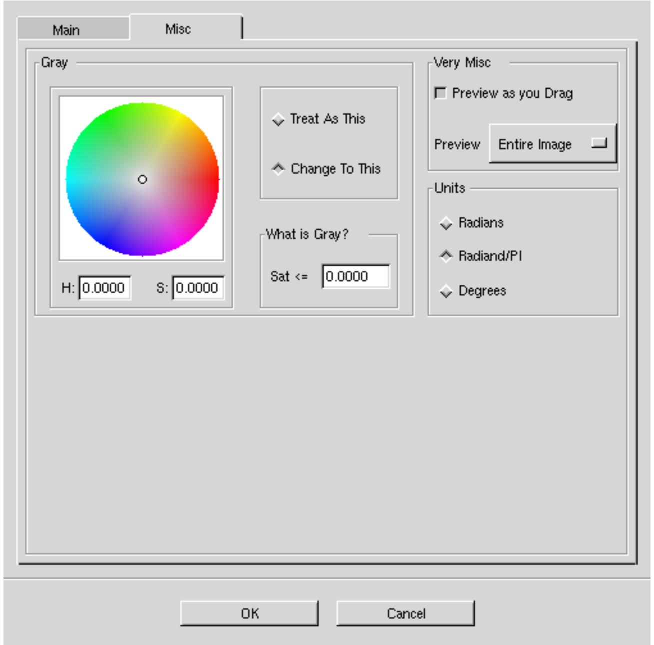
EXAMPLE

If you in the **From:** circle select a range from cyan to magenta, and the type is **counter clockwise**, and in the **To:** field select the same range (but **clockwise**), then all cyan colors will be rotated to magenta. If you press the **Switch Order...** button, the range will be inverted.



Let’s say you have a selection range that goes from cyan to magenta over blue, and you press **Switch Order...** Now the range will go from magenta to cyan over red and green. The button **Select The Entire Range** will of course select the whole range. You can also select what kind of **Unit** you want to use and what kind of **preview** you want.

THE MISC. WINDOW


Main
Misc
Gray
Treat As This
Change To This
What is Gray?
Very Misc
Preview as you Drag
Units
Radians
Radiand/PI
Degrees
OK
Cancel

H: 0.0000 S: 0.0000

Sat <= 0.0000

Preview Entire Image

In the **Misc.** tab folder you determine what colors should be regarded as gray.

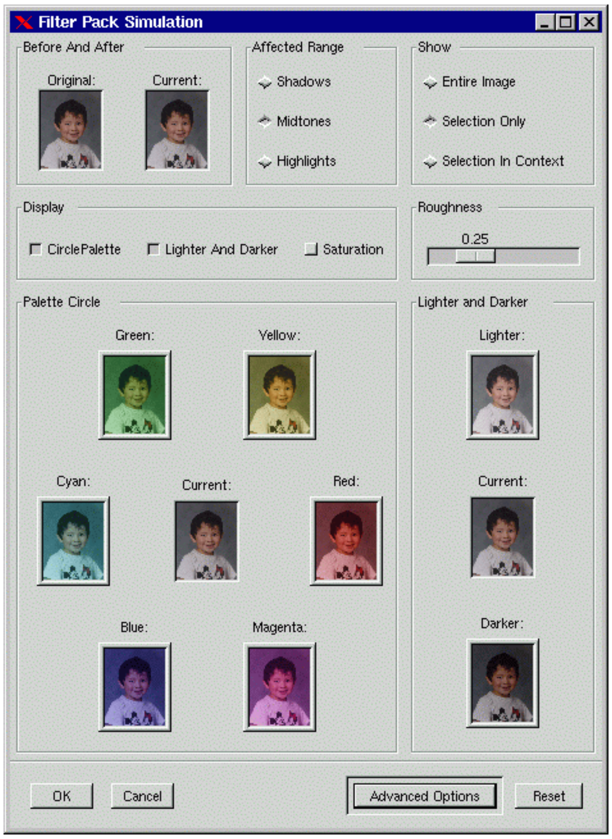
You define what gray is, and how you are going to deal with it in the Gray field. The field **What is Gray** is where you tell the filter what you think are, or should be gray. You define this in a **Saturation scale** from 0 to 1, where a *pure gray is 0*. This is often too narrow, so you will probably have to set gray to less or equal to 0.1 or 0.2.

As you set this value, the circle in the Gray color circle will expand and cover more saturation values. The check boxes **Treat As This** and **Change To This** is for what you want to do with the gray areas you defined in **What is Gray**.

With **Treat As This** you tell the plug-in what color you want gray *to be like*. You do this by paning the little circle cursor in the **Gray color circle**. You can for example drag the define circle to the red part of the color circle, and all gray shades will be treated as red color when you perform a color rotation.

If you check **Change to This** all gray shades will be changed to the color you have selected within the gray define circle.

FILTER PACK


Filter Pack Simulation
Before And After
Original:
Current:
Affected Range
Shadows
Midtones
Highlights
Show
Entire Image
Selection Only
Selection In Context
Display
CirclePalette
Lighter And Darker
Saturation
Roughness
0.25
Palette Circle
Green:
Yellow:
Cyan:
Current:
Red:
Blue:
Magenta:
Lighter and Darker
Lighter:
Current:
Darker:
OK
Cancel
Advanced Options
Reset

This is a real darkroom tool. Here you can do most adjustments that you could do in a real lab. Filter Pack is a real masterpiece. It is true that you can do all these things by using other tools in Gimp and maybe even with better precision, but this tool has a natural interface. It's just perfect for fixing up old and faded pictures, or correcting other kinds of color problems.

- First decide what **brightness** value you want to work with, **Shadows**, **Midtones**, or **Highlights**.
- **Show Art** is for preview. If you use a selection, you can press **Selection Only** to only preview the selection.
- **Display** determines what kind of values you are interested of changing, and therefore want to have a view of.
- The **Palette Circle** lets you change the color of the image. If you want your image to look a bit bluer, just press the blue preview.
- **Lighter and Darker** lets you change the amount of light in the image.
- **Saturation** (not in the screen dump because of the size) lets you alter the color dullness/loudness.
- **Roughness** refers to how much you want to alter your image. A higher value will allow you to change a lot, but on the other hand you will lose in precision.

ADVANCED OPTIONS

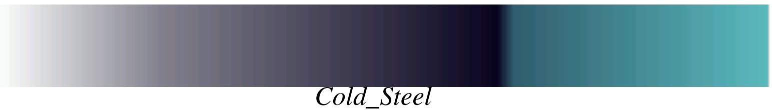
In **Advanced Option** can you set the **preview** size (good if you have a large screen). You can also check **Pixel Selection Menu** which lets you select the **HSV** value you want to change. The pixel selection lets you change one of the HSV values. The *pointers in the curve* lets you alter the range of **shadows**, **midtones** and **highlights** in a brightness scale from 0 to 255. *The slide bar* alters the **smoothness** of the transition from one shade to another. If you set it to zero, only the selected range (shadows, midtones, highlights) will be affected by your interaction. If you set it to 1, every range will be affected. A value in the middle is the most appropriate to create a smooth transition.

GRADIENT MAP

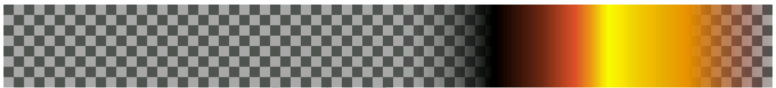
This is a really cool filter. It lets you **map** the image against the **active gradient** in the **Gradient Editor**. The lightest pixel in the image will get the color which is to the right in the Editor, and the darkest pixel will get the color to the left (except for transparent areas).

You can see it as mapping the luminosity of your image against the gradient in the gradient editor. Say, that in your image, the luminosity stretches from 50 (dark) to 235 (light). Then pixels with a value of 50, will get the left end color, the pixel with a luminosity of 51 will get the next color to the left in the gradient editor, and so on until you reach 235, which will get the right end color.

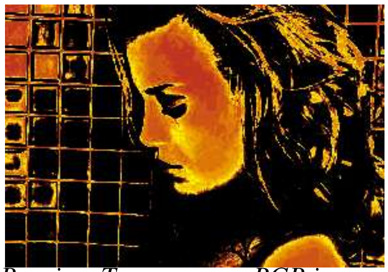
Here are some examples to make you understand it better: 1. A grayscale image in RGB mode and Cold_Steel as gradient. 2. The same picture with the gradient Burning_Transparency. 3. And now the same image in grayscale mode mapped against Cold_Steel and Burning_Transparency.



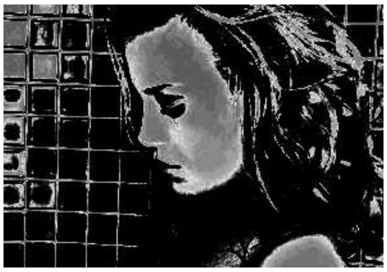
Cold_Steel



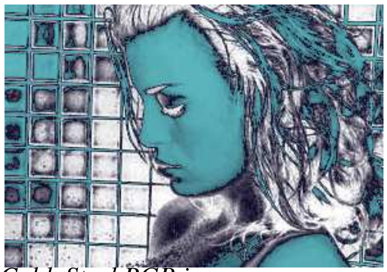
Burning_Transparency



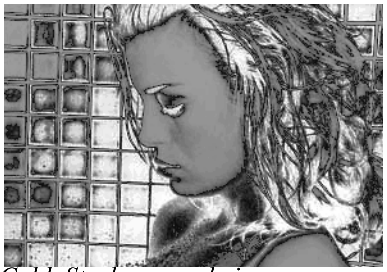
Burning_Transparency RGB image



Burning_Transparency grayscale image

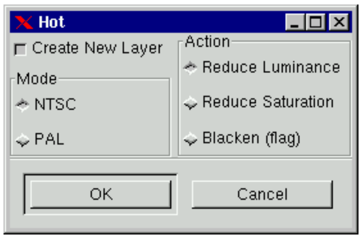


Cold_Steel RGB image



Cold_Steel grayscale image

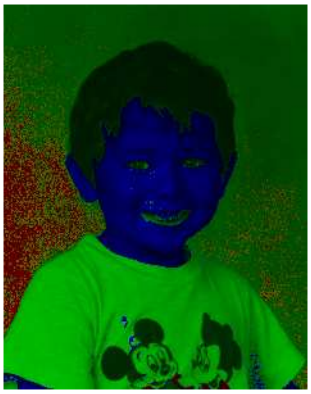
HOT

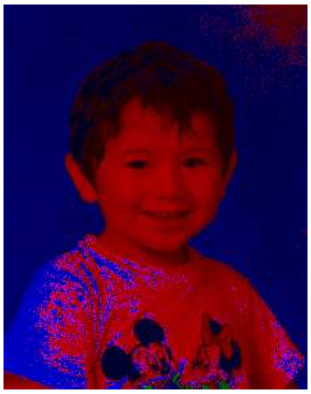


This filter will identify pixels which can be hard or troublesome to show on a **NTSC** or **PAL** monitor. You can do this on the original image, or as a new layer on top of your image if you check the **Create New Layer** check box. You can reduce the pixels' **Luminance** or **Saturation** or simply make them black. It's generally good to use this filter when you publish images on the web, but the obvious drawback is that different surfers use different monitors.

MAX RGB

This plug-in detects the **RGB channel** with the highest or lowest value and maps every pixel to either **Red**, **Green** or **Blue** - with maximum value! If you have checked **Hold the maximal channels** you select by maximal value. If there was **Cyan**, **Magenta** or **Yellow** in your image, those colors will remain even after the filter has been applied, because those colors already have maximal values for two RGB colors (the filter doesn't know which of those to choose). If you check **Hold the minimal channels** you select by minimum value. With this button checked, Cyan will not show, because the filter will choose the lowest value, which is 0 for Red. Black and White will remain in both cases, because they have no RGB value which is lower or higher than another.

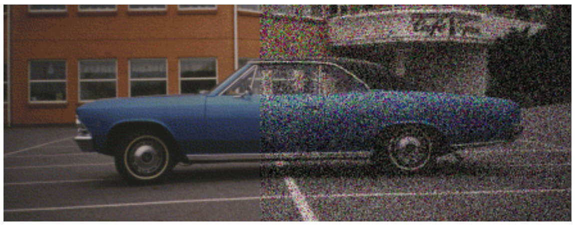

Min


Max

QUANTIZE

The effect of this filter is similar to making an **Indexed** image from an **RGB** image, but now you can stay in RGB space. You choose between two different color spaces; **CIE** or **RGB** (read more about it in the Color chapter). There is also a check button for **principal quantization** if you want the filter to quantize to slightly different colors. Which method to use depends on the colors in the original image; try both!

SCATTER HSV



Scatter HSV is a powerful and sophisticated tool for creating **noise** in an image. There's a *slide bar* for scattering each of the HSV components. **Hue** changes the color of the pixels in a random pattern. The color spread starts with the colors nearest to the original color in the HSV color circle, and continues to spread until all colors are available. The **Saturation** slide increases saturation, and the **Value** slide scatters hue-changed pixels *(If Hue is 0, Saturation/Value will only affect a few random pixels)*. The **Holdness** slide bar controls how far the scattering is allowed to go, or how different the new value is going to be, compared to the original value. Low Holdness allows maximal scattering, high Holdness results in a subdued noise effect.

SEMI-FLATTEN

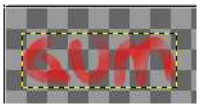
This is an extremely versatile filter for web graphics. A majority of the web-published images are **indexed** (i.e **GIF** images) because GIF supports **transparency**, and that is very handy when you for example place a logo over the background color in a web page.

Most of the time this works fine, but just try to create a **semi-transparent** glow around your logo. When you convert the image to **Indexed**, you'll find that the image looks horrible, and you will probably get surprised and disappointed. What happened? Isn't GIF supposed to handle transparency? This is how it works: Yes, GIF/Indexed images do handle transparency, but only total transparency (alpha value=0). Every other alpha value will automatically be converted to either fully opaque or completely transparent. This is why you can't make a smooth transition from solid color to transparency in an Indexed image.

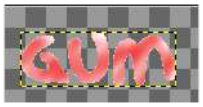
SEMI TRANSPARENCY IN WEB IMAGES

This filter will save your day, all you have to do is to apply the filter and then convert the image to Indexed.

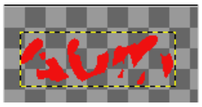
This is how it works: Every pixel that has an **alpha value** between 1 and 254 is **flattened/merged** to the current background color in the toolbox. If your semi-transparent logo will be on a white web page, then you have to set the background color to white before you apply the filter. To appreciate the difference this filter makes, see the images below.



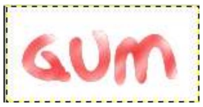
Original logo

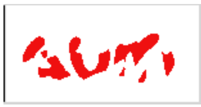


Indexed with filter



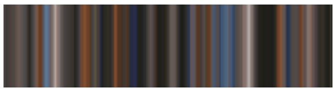
Indexed without filter





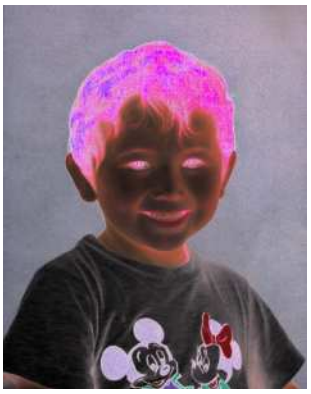
SMOOTH PALETTE

Smooth palette makes a *striped palette* of the colors in your image. This is a nice tool when you want to see what kind of colors you have used in an image, but the main purpose of this filter is to create colormaps for use with the **Flame filter** (see Flame filter in the Filters/Render menu).



Output of the Chevelle image

VALUE INVERT



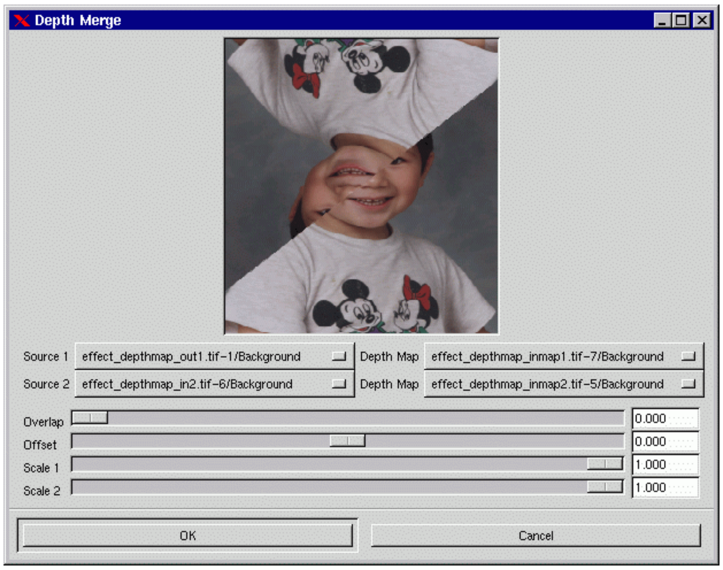
This filter inverts the image in **HSV space**. It will not alter **Hue** and **Saturation**. It inverts the brightness value without changing the basic color in your image. This can be quite interesting if you are using **Curves** to change a certain value dramatically, and you get stuck with the wrong (inverted) color. Try it yourself and see what you can use it for.

chapter 24

Combine filters

The smash-together factory. If you want to combine several images to one, these are the filters to use. You can fade images into each other and even create a piece of celluloid film...

DEPTH MERGE



Source 1: effect_depthmap_out1.tif-1/Background
Depth Map: effect_depthmap_inmap1.tif-7/Background
Source 2: effect_depthmap_in2.tif-6/Background
Depth Map: effect_depthmap_inmap2.tif-5/Background
Overlap: 0.000
Offset: 0.000
Scale 1: 1.000
Scale 2: 1.000

This is a very nice tool for **combining** two images.This is done with the help of two **map images**. The maps should be **grayscale** images (so you can control the outcome better).

AN EXAMPLE

This example will hopefully make matters more clear:.

- Take two images of equal size. Then create two new empty images with the same size as the original images.
- In one of the new images, take the **blend tool** and blend from black in the top left corner, to white in the opposite corner. For the other grayscale image, do the opposite. Black is bottom right and white is top left.
- Now, bring up the **Depth merge dialog**. Set **Overlap** to 0, set **Offset** to 0 and **Scale1** and **2** to 1.000. Use one of the first opened images as **Source 1**, and the other one as **Source 2**. Use the first grayscale image as **Depth Map** to source 1, and the other one as map to source 2. Now the two images will get **combined** and a sharp border will appear diagonally from the to right to the bottom left.

Why? Well, the plug-in looks at both map images, and compares every pixel. The map with the darkest pixel will win, and the source to that map will get to show its image pixel. When we created our grayscale images, there was a clean line going diagonally Top Right to Bottom Left where the outcome flipped from one map to the other. Now when you understand the basics, you can create all kinds of maps.

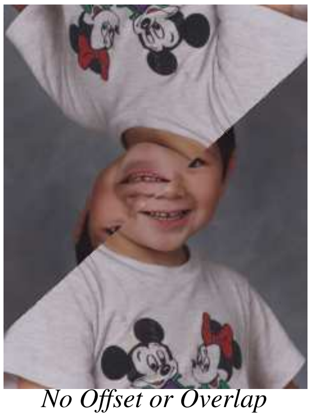
Let’s see what the **slides** can do for us. As in you saw in our example, there was a sharp border between the images. This was because the **Overlap** was set to zero. If we slide it up a bit, you will see that the border will be get more fuzzy and transparent. This is the variable you use to get *soft transitions*. Offset changes the darkness value of your maps. If you slide it to a negative value, the map to source 1 will get *darker* and therefore get more of the outcoming image. If you slide it to positive, the other map will get *darker* and get more of the outcoming image. The slides **Scale 1** and **Scale 2** make **map 1** respectively **map 2** darker or lighter. They have the same effect as **Offset**, but they are more sensitive to the touch

Source 1

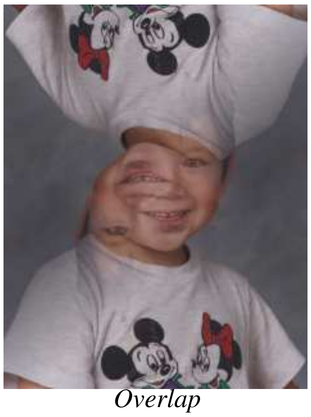
Map 1

Source 2

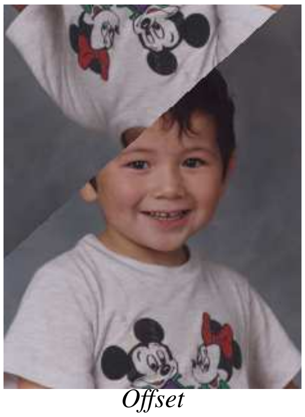
Map 2



No Offset or Overlap

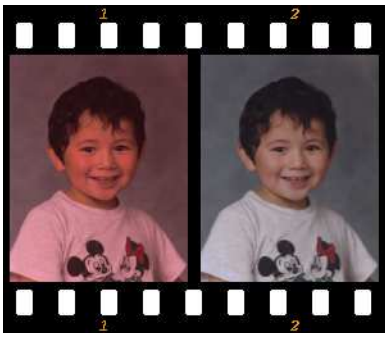


Overlap



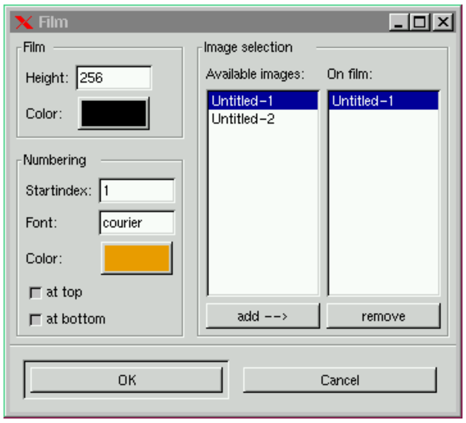
Offset

FILM



This plug-in creates a *film celluloid* of two or more images. This is a nice special effect, and I bet that you have seen something like this before in magazines or papers.

HOW TO



Height: 256

Startindex: 1

Font: courier

Available images: Untitled-1

On film: Untitled-1

The user interface is quite simple. In the image selection dialog you have a view of all available images. You just click **Add**, to use it in your film.

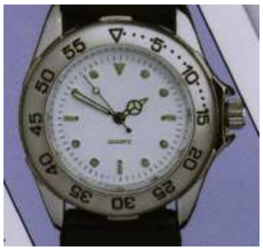
On film shows the pictures on your film. To change the order of the pictures, you have to add and remove pictures so they will come in the order you want.

Height is the height of the outcoming film. **Color** is the film color (black). The plug-in will automatically adjust your images so that they will fit.

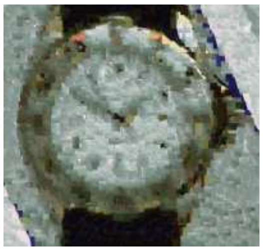
Numbering lets you specify a start number on the roll, and the **Font** (warning! - you have to have the font installed), color is the number color in your film (orange). Both the color on the film itself (usually black though) and the color of the numbers can be changed by clicking on them. Check the **at top** and **at bottom** if you want numbers there, otherwise uncheck.

FUSE

This plug-in creates a new image from **tiles** of one or several input images. You can use this filter in two ways:

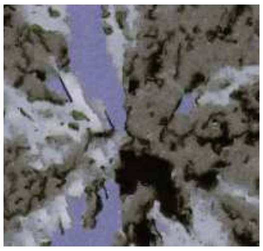


Original

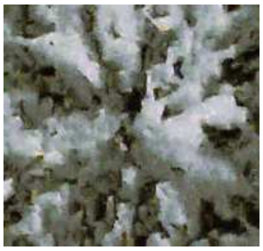


Fused as template

- You can use the **target image** as a **template**. This will result in an image that looks like a mosaic or cubist version of the original image, and where the mosaic stones or cubes consist of pieces of the input images (a little like modernist collage art where you create a composite image by cutting and pasting pieces of a newspaper).



Fused original only



Fused orig+other image

- You can also create a new shape by **fusing** one or more images. This filter uses *associative image reconstruction*, meaning that it will search out pieces that match where they overlap, thus creating a pattern with more or less ”solid objects” in it. The new pattern is a random mix of matching tiles from the source images, with a direction tendency towards the centre of the image.

If we take a look at the **Fuse dialog**, you’ll find a window at the top of the dialog with a list of available images (Indexed images are not accepted). You can select one or all of the images in this list by mouse-clicking, click again to deselect.

*Note, that the image you opened Fuse from is not used in the fusing process unless you select it in this menu or check the **Use target as template** button.*

PARAMETERS

The parameters list starts with **Tile Size**, which controls the size in pixels of the square tiles. Small tiles produce smoother images, but makes the operation very slow.

Overlap determines by how many pixels the selected tiles should overlap in the fused image. If Overlap is too low, you’ll get small black gaps in the composite image, and it it’s too high it’ll slow down the process a great deal. The recommended amount is somewhere between a quarter and a third of the tile size.

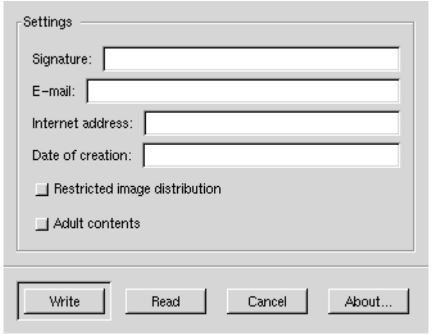
Search time refers to how long the filter should search for the tile that best fits the overlap. Search time also slows down the process, but results in a smoother output. When you check **Use target as template**, Fuse has to determine what’s most important - to find a tile that matches the template, or a tile that matches the overlap. If you set the **Template Weight** slider high, you’ll get a good representation of the target image, but the image will look much more ”cubist” than a fused image without template (but with the same parameter value). If you set the Template weight lower, you’ll get a smoother image, but it will not bear as much resemblance to the template image.

chapter 25

Cryptographic filters

So, you want to be a double 0 seven? Then you may want to use these filters to encrypt and hide your information

DIGITAL SIGNATURE

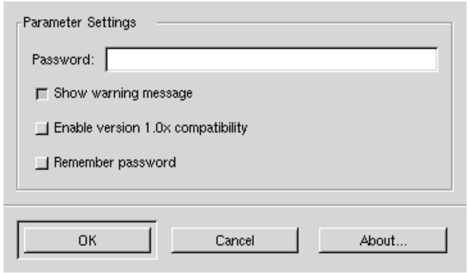

Settings
Signature:
E-mail:
Internet address:
Date of creation:
☐ Restricted image distribution
☐ Adult contents
Write
Read
Cancel
About...

This plug-in lets you make or read an invisible **signature** in the image. In order to read a signature just press **read**. If you want to create a signature, you have to fill in the information in the dialog and press **write**. You must remember to save the image in a non-destructive format like TIFF. If you save it as a JPEG the signature will be destroyed.

ENCRYPT/DECRYPT:

This plug-in makes it possible to **encrypt** your images, so no one besides yourself can see them. To encrypt the image, bring up the filter and type your **password** e.g. ”qwerty”. To **decrypt**, bring up the filter again, type your password and the image will get decrypted.

A warning by the author of this filter; ***Daniel Cotting***: This filter may work for you, but there are no guarantees *(for us at Frozenriver, this filter has worked with fine)*. Remember not to save the image in a destructive file format, e.g JPEG. You can enable backwards compatible with the 1.0.X versions of this plug-in, but I suggest to stick with 2.X and not enable this feature if you haven’t any images encrypted


Parameter Settings
Password:
☑ Show warning message
☐ Enable version 1.0x compatibility
☐ Remember password
OK
Cancel
About...

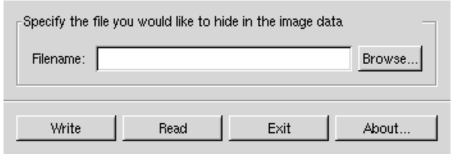
with the 1.0.X version. You can enable the plug-in to remember your password, but don’t forget that passwords are best kept in your own mind where no one can see it.

Encrypted image

The non encrypted original image

STEGANO

This filter lets you **hide** or **read** a *hidden file* inside your image. If you want to *read* the hidden file, press **read**. If you want to *hide* a file, just press **write** and then **browse**. Remember that if you want to hide a big file then you have to use a big image. As with the other encrypt filters you can’t save your image in a destructive file format like jpeg. (This filter is perfect for secret agents like 007 ;-).


Specify the file you would like to hide in the image data
Filename:
Browse...
Write
Read
Exit
About...

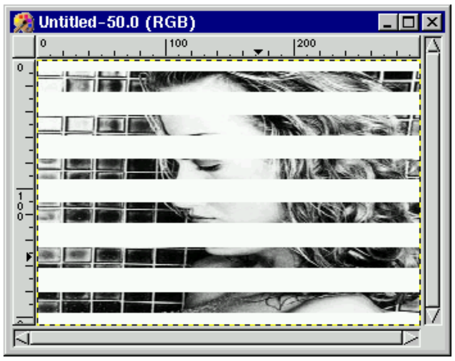
Cryptographic filters

chapter 26

Distort filters

Do you feel that your image lacks a certain something? Here are the ultimate distort tools provided by Gimp. If you so please, you can change your image until it's completely unrecognizable.

BLINDS

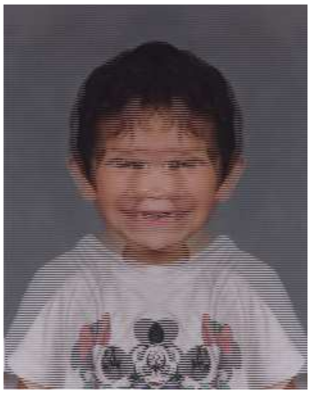


This Plug-in makes **blinds**. It's like slicing your picture in shreds and pasting the slices to the different sections of the blind.

To enable a transparent background, you have to work in **layers** or add an **Alpha channel** to the background, otherwise you'll end up with the background color in the toolbox.

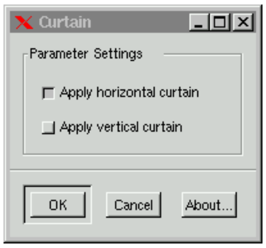
You can create both **vertical** and **horizontal** blinds, just check the right button. **Displacement** refers to the open **angle** of the blind. A zero degree angle makes the blinds shut close and if you set it to 90, you'll open them as much as possible.

CURTAIN



This filter will make it look as if you are looking at your image through a **curtain** (and as it happens, this curtain already has your image printed on it!).

What happens when you apply this filter **vertically**, is that the image is copied, the copy is rotated 180 deg, thinly slashed and then combined with the original image. The same will happen if you choose **Horizontal** but from a horizontal perspective. You can also combine it so it will do both.



Parameter Settings

☑ Apply horizontal curtain

☐ Apply vertical curtain

EMBOSS

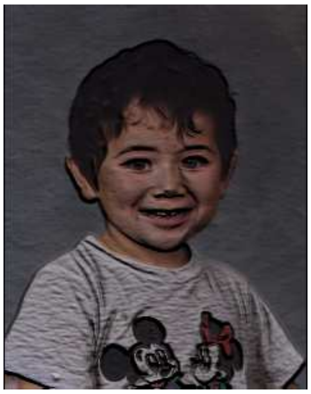
Stamps or **carves** out a three-dimensional look to your image. ***Emboss only works for RGB images.***

The direction of the **stamp** (inwards or outwards) is determined by the original **brightness** value, bright parts will look like they are **raised**, dark parts will look like they are **carved**.

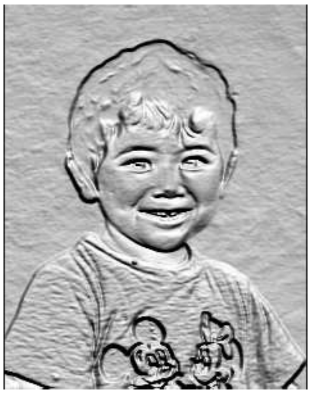
Depth determines the embossed carve depth.

The **Azimuth** slide determines light direction, but so does **Elevation**. The most exact way to describe Azimuth is as *"from which direction the sun rises in the morning"*. Think of Azimuth as a shining satellite, moving around your image - the light in the image will change as it moves. If we thought of Azimuth as the sun in the morning, you might think of **Elevation** as the *"time of the day"*. Think of the sun when it reaches zenith. There will be no shadow. It's the same with Elevation, and if your depth is big, the lowest parts will look like black holes (quite good edge detect). When the sun falls or rises, the shadows will get longer or shorter, and the direction of the shadows will also change.

As you may have noticed, **Emboss** will make your image gray. **Bumpmap** keeps the color information, but the *carving* will not appear as deep as with Emboss. The images below have been embossed respectively bumpmapped with the same values, so you can see the difference. Emboss makes a surface look like metal or rock. Bumpmap just puts more depth to your image, like an impression on paper or leather.

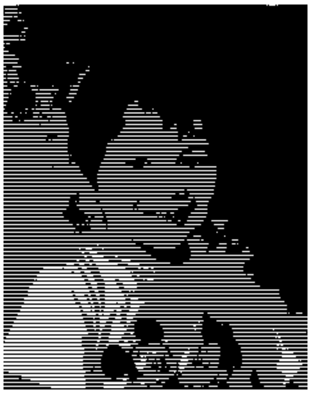


Bumpmap

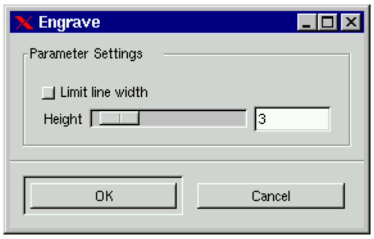


Emboss

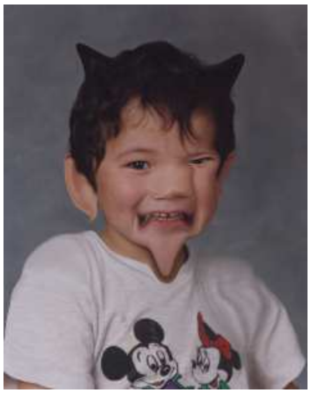
ENGRAVE



Turns your image into something which looks a lot like an **etching**. The slide **More Height** determines the depth of the engraving.You'll get more realistic etchings with low **height** values *(close to 3)*. **Limit line width** is a limit in how many *shades of gray* you want. You will get more contrast if you don't check this button, and the etching will also get more realistic. On the other hand, you can get quite interesting results with this option.



IWARP



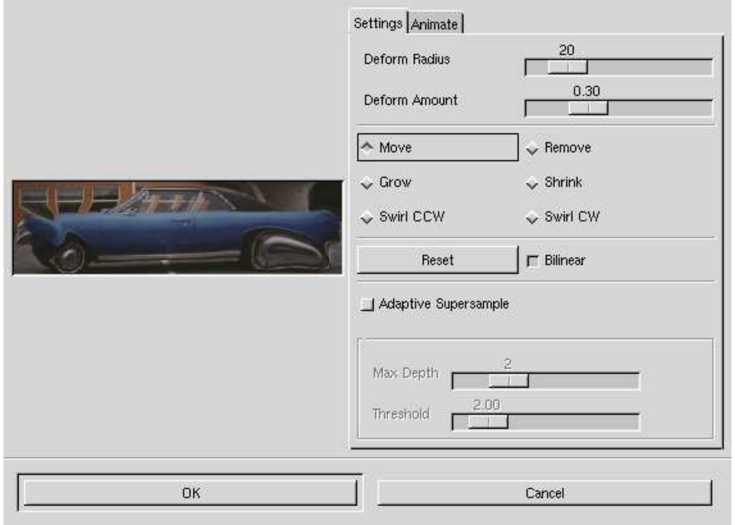
This is Gimp's answer to **Kai's Power Goo**. It's quite an amazing distort filter. You can **grow**, **move**, **shrink**, **swirl** or **remove** parts of an image. You can set the amount of **distortion**, and how large the **deformation zone** should be.

To create a distortion, you just drag the mouse in the preview image and you can watch the distortion take place. If you not are satisfied with a distortion, click **reset** and the image will be back to normal. Like in Power Goo, you can animate the distortions you've just made, and play it like a film.

PARAMETERS

Let’s take a look at what **Warp** can do for you.

- **Deform Radius**: This parameter sets how big/small the distortion area should be when you move the mouse over the preview. It refers to the **radius** (in pixels) from the centre of the mouse pointer.
- **Deform Amount**: Sets the amount of deformation in a scale from 0.00 to 1.00.
- **Move**: Lets you **stretch** parts of the picture. You can for example stretch a tiny little butt of a nose, and turn it into something a witch could be proud of.
- **Grow**: Grows a part of the image.
- **Shrink**: Shrinks a part of your image.
- **Swirl**: Lets you twist a part of the image, either **clockwise** or **counter clockwise**
- **Remove**: Will remove a distortion as a whole, or just partly. This is the perfect tool for adjusting a distortion, i.e you don’t have to press reset if you failed to do what you wanted to do (If you are making a animation this correction will appear in one of your frames).
- **Reset**: Simply resets the image to the original state.
- **Bilinear**: If checked, warping will get more smooth.
- **Adaptive Supersampling**: (see the blend tool).



Deform Radius 20

Deform Amount 0.30

● Move ○ Remove

○ Grow ○ Shrink

○ Swirl CCW ○ Swirl CW

☐ Bilinear

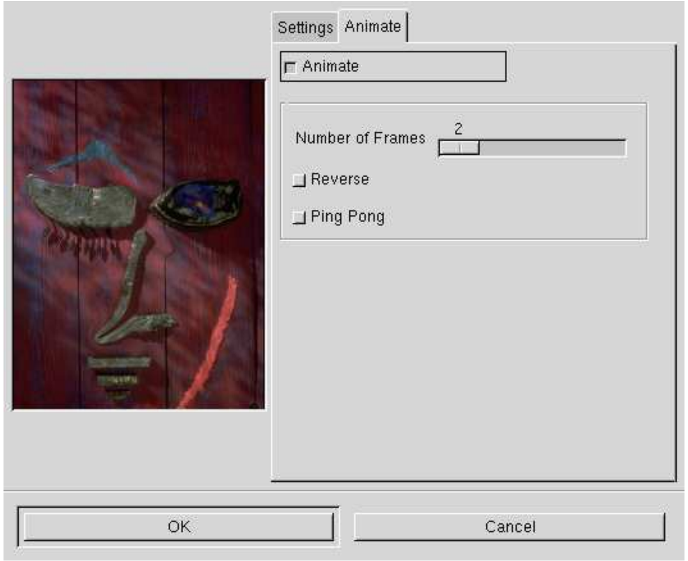
☐ Adaptive Supersample

Max Depth 2

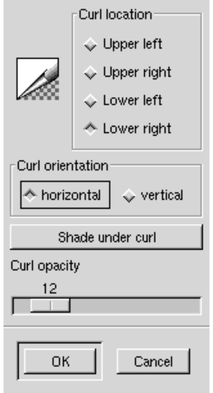
Threshold 2.00

Animation: By checking the **Animate** button, you will get a layered image, ready for making GIF animations. You can set the number of frames, and in what order you want to play the frames of the film.

- **Normal mode**: The first frame is the original image, and the following frames are gradually distorted until the last frame, which displays the fully distorted image you made in the preview window.
- **Reverse** is the opposite of Normal.
- **Ping Pong**: Creates a frame sequence like this: original image..... distorted image... original image
- **Reverse** and **Ping Pong**: distorted image..... original image.... distorted image.
- With **reverse** and **reverse ping pong**, you must remove the background layer after you have applied the filter to make it work properly.



PAGE CURL



Curl location

○ Upper left

○ Upper right

○ Lower left

● Lower right

Curl orientation

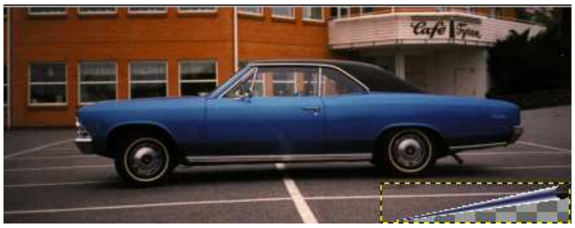
● horizontal ○ vertical

Shade under curl

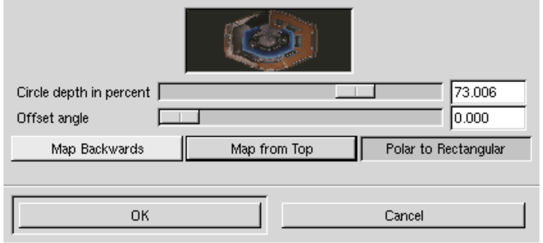
Curl opacity 12

With this plug-in, you can *curl*, or *lift* any corner of your image. To use **Page Curl**, you must have at least one **layer** and a **background** in your image. The top layer to *curl* and a bottom layer to *curl against*. The color of the back of the curled page is controlled by the current colors in the toolbox. The foreground color represents highlight, or the middle part of the gradient, and the background color represents shadow, or the first and last part of the gradient. You can also put a **Shade** under the curl, and set the **Opacity** of the curled page.

If you just want to curl a small part of the image, make a *selection* and curl the selection. This allows you to make a small curl, just as if the glue didn't stick properly to that corner of the image..



POLAR COORDS



Circle depth in percent 73.006

Offset angle 0.000

The main achievement of this plug-in is to make **circular** or **rectangular** shapes of your image or drawable.

. EXAMPLES

To create a powerful, but rather unsophisticated text curve, you can apply this plug-in to a text string. You can also make a target circle out of some lines, or you can make rectangles out of straight lines.

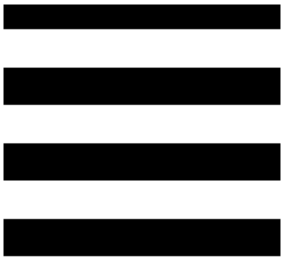
•The slide **Circle depth in percent** controls how round your circle will be. A higher value will make it more circular (Pic 2), a lower value makes it more rectangular (Pic 3).

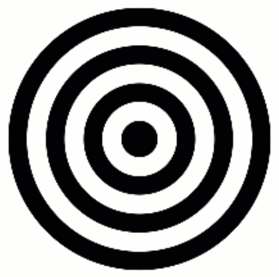
•The **Offset angle** controls where your circle/half-circle will start (in degrees) (pict4)

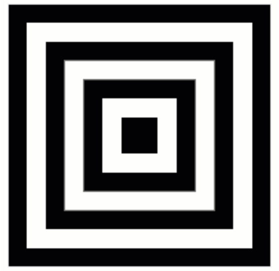
•The button **Map from top** controls if the bottom part (Pic 5) of the image turns up in the middle or not. Press this button and it ends up in the middle (Pic 6), unchecked will put it in the outer regions (Pic 7).

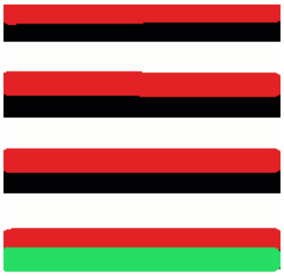
•**Map backwards?** controls whether the image will be mirrored or not.

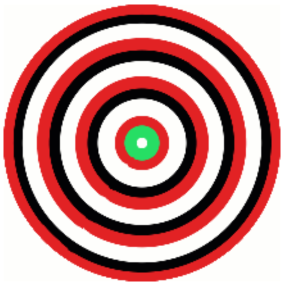
•**Polar to rectangular?** lets you decide whether the image should be mapped to circular or rectangular. Press this button and it will map to rectangular (Pic 8)

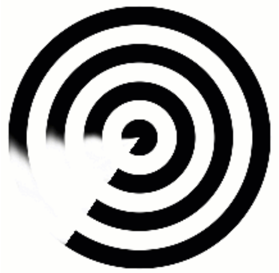

1

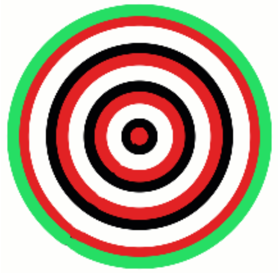

2

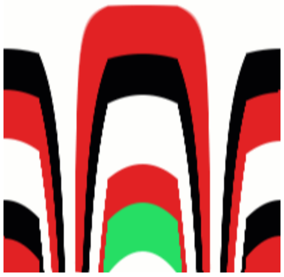

3


5

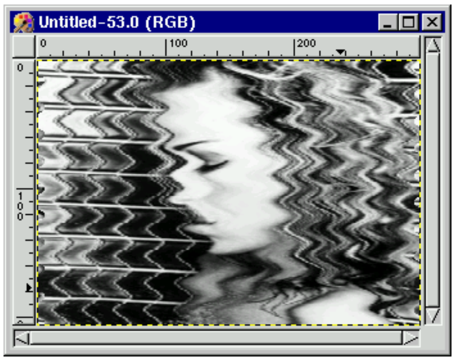

6


4


7

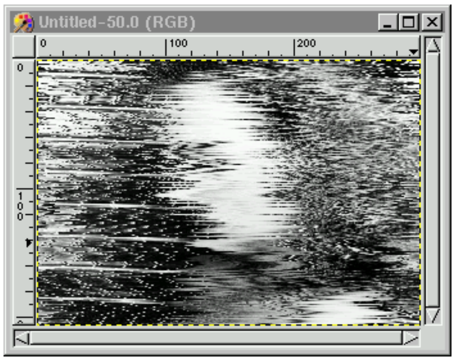

8

RIPPLE



This plug-in displaces the image or selection in waves or ripples so that it looks like a disturbed water surface. You can displace the image in **Sine** or **Sawtooth wave** form. You can also specify the **Period** and **Amplitude** of the wave. **Wrap**, **Smear** and **Black** have the same functions as in the *Displace plug-in, see chapter 32*. If you unchecked **antialiasing** you will get rough edges. Note that you can ripple both **horizontally** and **vertically**.

SHIFT



This filter creates a random displacement of each **pixel row**, horizontally or vertically. **Shift Amount** determines how much a row should be displaced. You can't predict if it will displace left or right / up or down. You just know the amount of displacement.

TWIST

The **Twist** is a plug-in which provides a variety of geometric image distortion functions. The principle of all distortion types provided by the Twist is to map the input image to the output image by translating the original pixels according to a 2D vector field which is determined by the selected distortion function. In order to support the user's creativity, some parameters (up to eight, depending on the function type selected) can be adjusted, resulting in different effects. The output of the Twist plug-in is a geometrically distorted image. The effect of a selected parameter setting can be viewed interactively in a preview image. Each selectable function provides eight predefined effects which can help the unexperienced to get a feeling for the parameter settings. Currently eight different distortion functions are available.

To use it, simply bring it up from the menu. Note that you cannot distort **indexed** images. If you have such an image, it must be converted to an **RGB** or **grayvalue** image.

There are four frames, the 'Preview image' frame, the 'Parameter settings' frame, the 'Functions/Effects' frame and the 'Cutoff function' frame.

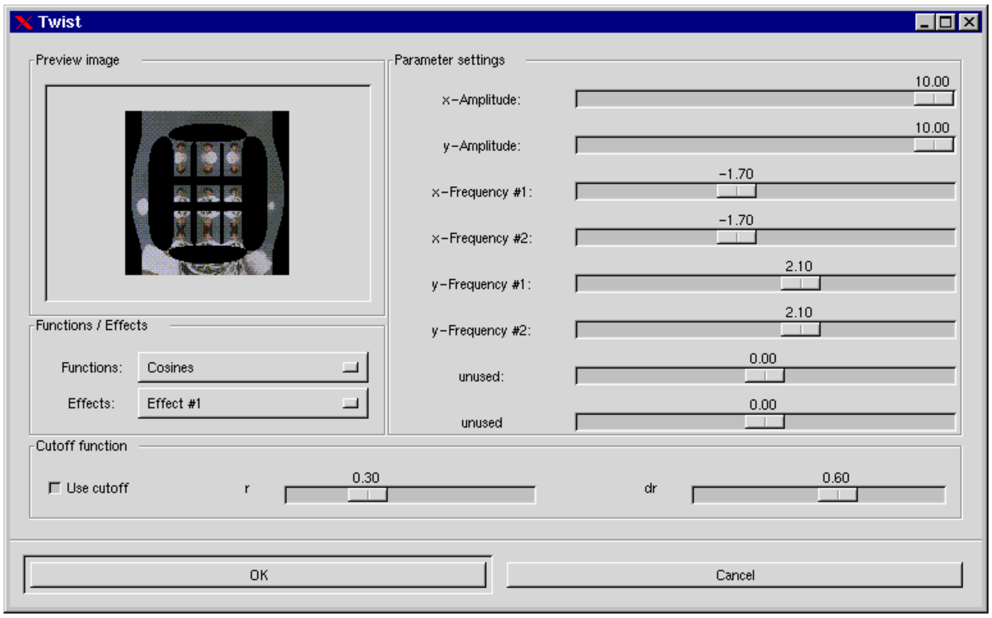
FUNCTIONS EFFECTS

The first frame you'll probably want to use is the **Functions/Effects** frame. This frame contains two menus, the **Functions** menu and the **Effects** menu. You can choose one of several distortion functions by activating the Functions menu. Having chosen a distortion function you must choose a **parameter set** that, in combination with the chosen distortion function, defines the effect you will obtain.

There are two ways to define a parameter set. The first, and easier way is to select one of eight effects from the **Effects** menu in the Functions/Effects frame. The parameters defining the effect will be displayed in the **Parameter settings** frame. The effect on the image can be seen almost immediately in the preview image at the top left corner of the dialog window.

PARAMETER SETTINGS

The second way for choosing a parameter set is to use the **sliders** in the Parameter settings frame. Note that some of the sliders may be labeled ’unused’, indicating that modifying this parameter will not affect the image distortion effect. The labels of the sliders depend on the distortion function.



x–Amplitude: 10.00
y–Amplitude: 10.00
x–Frequency #1: -1.70
x–Frequency #2: -1.70
y–Frequency #1: 2.10
y–Frequency #2: 2.10
unused: 0.00
unused 0.00

Functions: Cosines
Effects: Effect #1

☐ Use cutoff
r 0.30
dr 0.60

In order to acquire a certain feeling for the parameters of each distortion function I recommend that you experiment with the sliders after having chosen an effect. If you feel that the preview image is too small to see what actually will happen to the image, do not hesitate to execute the Twist by pressing the [OK] button at the bottom of the dialog. The Twist will remember all parameters you have selected when it is started again providing you with the possibility of fine-tuning an effect.

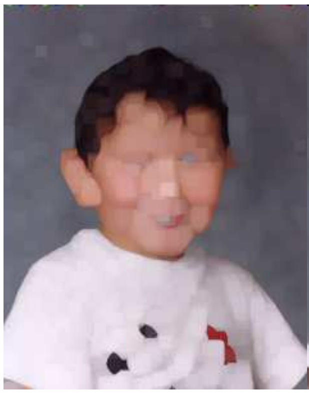
The last frame, which has not been discussed yet, is the **Cutoff function** frame. The cutoff function is a mechanism for damping the image distortion function depending on the distance r from the image’s center. This means that, when the cutoff function is activated, the image distortion is a product of the selected distortion function and of the cutoff function. The toggle button labeled **Use cutoff** is used for switching this modifier on and off. The meanings of the two parameters, r and dr, which can be modified by the sliders are visualized by figure 2 below. Note that some of the predefined effects use the cutoff function. So have an eye on this frame when you want to figure out which parameters yield an effect that you like.

Documentation by Peter Uray

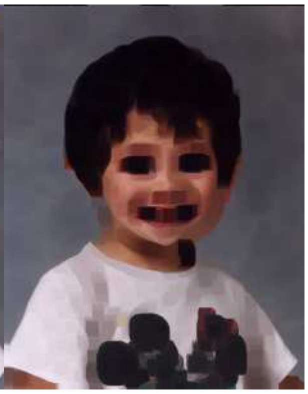
VALUE PROPAGATE

This plug-in works by *spreading* and *increasing* certain value ranges in a specified direction. The first two parameters, **More White** and **More Black**, have a large impact on *scanned photographs* (see pictures). You might describe the effect as a *swelling* of bright (respectively dark) areas where contrast is high, expanding at the cost of the opposite brightness value. When you run the filter several times, those areas will clog together in large square clusters, while middle values will remain relatively unaffected. The result is often quite artistic. The More White effect resembles oil painting, and More Black looks like watercolor. This filter is also very good for drawings. A black and white line drawing made in the computer will look more like a real ink pen drawing if you apply some More White to it, More Black will just thicken the lines.

Original



More White



More Black

TIPS

The next few parameters work best with drawings and computer made images with distinct edges. Scanned photos are either hardly affected, or the result is strange and unpredictable.

Foreground to Peaks draws a fine contour (with the current FG color) around defined objects or shapes in the image, and that includes Propagate cluster formations. If the object border is fuzzy, the FG outline starts where the object color fades out.

Middle Value to Peaks creates (if there isn't one) and propagates the transitional color (blend of object's edge color and background color) and blurs the image. It will blur more for every time you use it, because then the transitional color will be created from the new edge color.

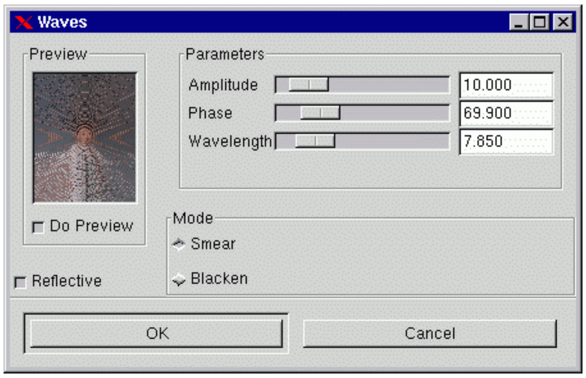
Only Foreground propagates areas which match the exact shade of the foreground color in the toolbox, so soft and fuzzy edges don't propagate well with this option.

Only Background does the same, but for the background color. Sometimes these options causes pixels of the chosen color to spread in an asymmetric way - large lumps or clusters grow. This can occur when you spread soft brush lines, or when you run this filter several times. Don’t worry about it - it’s just the way this filter operates.

More Opaque/More Transparent only applies for images with an **Alpha Channel** (a layer or an alpha enabled background). Those options work exactly the same way as **More White/Black**, but white is replaced with opaque and black with transparent.

Value range can be set differently if you only wish to propagate areas in a certain range, like only very dark areas or only middle values. You can also set the **Propagate Rate** (amount) and the direction of the spreading. The check buttons **Propagate Alpha** respectively **Value Channel** can be used a locking device, if you’re afraid to press the wrong button. (Note, this filter does not work for the kind of Alpha Channels you create in the Channels tab folder).

WAVES

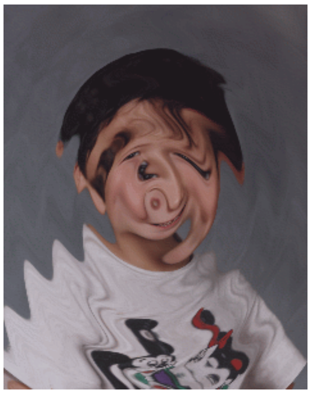


Amplitude 10.000
Phase 69.900
Wavelength 7.850

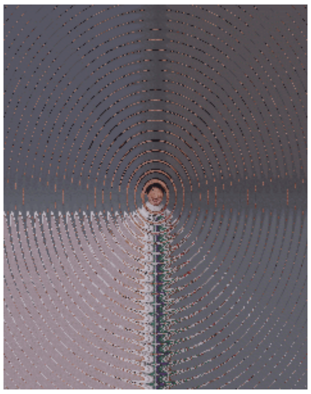
This plug-in is really nice. It simulates the effect that you get when you throw a stone in a pond (if you have unchecked the reflective button).

The **amplitude** is the *height* of your waves. **Phase** is where you are in your wave (it’s like a sinus wave and the phase is where you are in that sinus wave).

Wavelength is how long your (sinus)wave is. The modes **Smear** and **Blacken** have the same function as in the *displace plug-in, see* *chapter 32*. If you check the **Reflective** button you will not get the simple ”throw stone” effect, instead your wave will create an interference pattern.

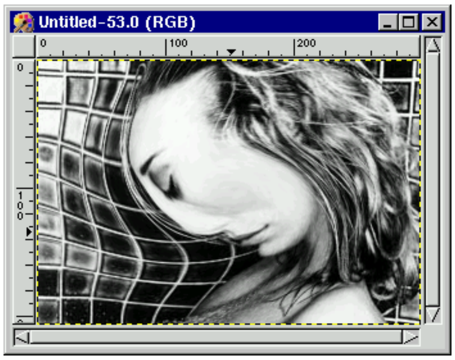


Waves non- reflective



Waves reflective

WHIRL AND PINCH

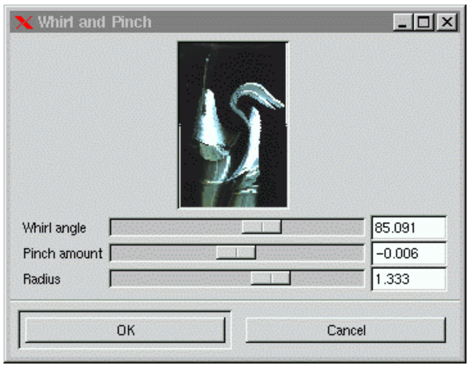


This plug-in will distort your image in a *centre circular way*. **Whirling** distorts it much like the little whirlpool that appears when you empty your bath.

Pinch can be compared to applying your image to a soft rubber surface and pressing it in different ways. If the pinch slide bar has a **negative** value, it will look as if someone had tried to push a round object up toward you from behind the rubber skin. If you use a **positive** value, it looks like someone drags or sucks in the surface from behind, and away from you.

The slidebar for **Whirl** controls how many degrees you want to *turn* your image. The **Pinch** slide bar controls the type and amount of pinch.

Radius determines how much of your image will be affected. If you set radius to ”2”, the whole image will be affected. If you set the radius to ”1”, half the image will be affected, and if it is zero nothing will be affected (just think of it as the radius in a circle. 0 in the center and 1 halfway out...). ***To create a really realistic whirlpool, combine the two distort methods***



Whirl angle 85.091
Pinch amount -0.006
Radius 1.333

chapter 27

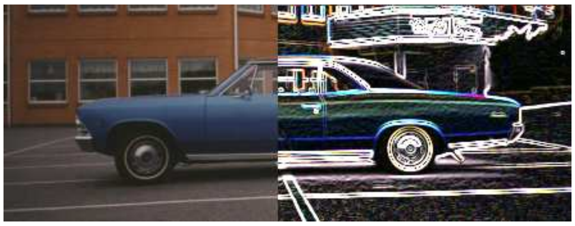
Edge detect filters

Living on the edge? Gimp provides you with a few edge-detect filters to help you find your edge.

INTRODUCTION

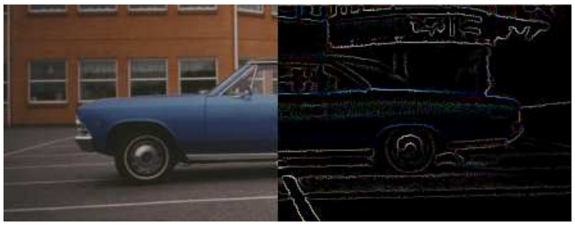
Edge detection filters search out the borders between areas of different color, thereby **tracing** the contours of objects in the image. The background is always black, and the contour is either white, or has the same color as in the original image. It's actually a lot like scrafitto, or artistic scratching techniques where you cover an image with thick black oil/crayon/tempera paint and then scratch the contours with a knife or needle so that the original colors appear in the scratch marks. Edge detect is often used to make **selection** easier, but it can just as well be used for artistic purposes or to achieve certain effects etc. You can of course always invert an edge detected image, and get something that looks like a hand drawn sketch. There are three kinds of edge detect filters:

EDGE



Normal edge detect, meaning a dark image with (mostly) white contours. You can set the amount of Edge-detect: A high value results in a black, high contrast image with thin, sharp edges. A low value will produce thick, coarse edges with relatively low contrast and lots of color in the dark areas. Wrap, Smear and Black doesn't do much good, so you might as well leave those as they are.

LAPLACE

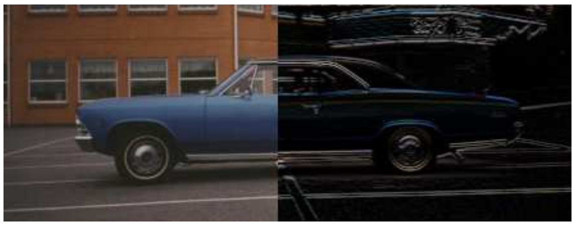


Creates a black image with extremely thin (colored) edge lines (1 pixel).

Note, applied on a layer, Laplace and Sobel edge-detect results in a ***transparent*** *image with thin black edge lines.*

This is of course very useful when you want to put emphasis on the contours of an image, or make a photograph look like an ink drawing. To achieve this effect you duplicate the image and use Laplace/Sobel on the top layer. For the best result, apply some blur to the image before running this filter (Gaussian with a radius of 0.5 to 5.0)

SOBEL



The same effect as Laplace but the edges are a bit broader and also a bit more blurred. If you only apply it in one direction (you can apply it both vertically and horizontally) and check **keep sign of result** you will get something that looks a lot like an emboss effect. If you only apply it **horizontally** you will get more contrast and color in your image. If you only apply it **vertically**, it will get a bit darker and there will be less contrast in the image.

chapter

28

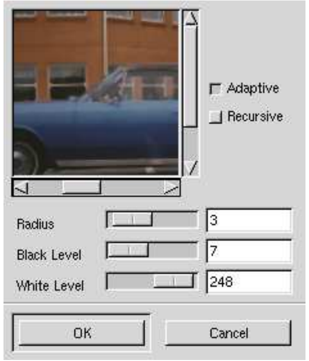
Enhance filters

Do you want to sell your old car? Here is the tool to enhance the advertisement image.

DEINTERLACE

This filter will help you adjust images captured by video cards. Sometimes the even or odd field doesn't get captured correctly, then you can use this filter to fix it..

DESPECKLE

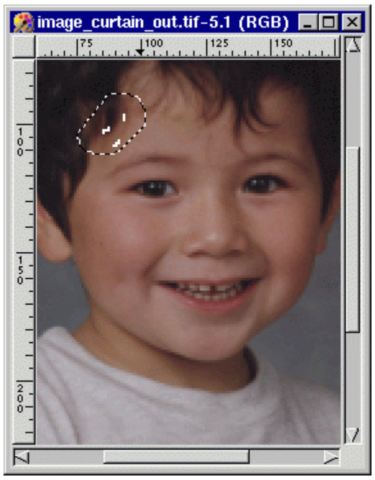


Radius 3
Black Level 7
White Level 248

This is the filter to use if your image is **noisy**, if you scanned an image and there was some **dust** or fibers in the scanner, if you got a **moire** pattern when you scanned a **printed** image from a magazine, or if your image has physical damage, like **scratches**.

If your image is just noisy, or suffers from moire effects, use **Despeckle** over the whole image.

If dust or scratches is your problem, select the damage with the free selection tool, and use Despeckle on that selection.



You have the following choices; **Only Radius**, **Radius and Recursive**, **Adaptive** or **Adaptive and Recursive**.

Radius refers to the *window size* from 1 (3x3 pixels) to 20 (41x41). The image will be chopped up in several windows, with a specified size of for example 3x3 pixels. In each of these windows the filter will try to smooth the color range, and thereby remove unwanted defects like scratches or noisy pixels.

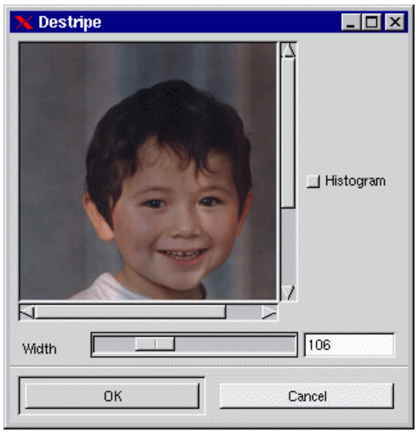
HOW TO USE IT

If you use **Only Radius**, a general algoritm is used to smooth the color range. If you set the radius high, the **blur** level will also be high. If you use **Radius and Recursive** you can use a smaller radius to get the same effect as with a large radius and no recursive. Be careful though, **Recursive** can easily result in unwanted blurring. If you use the **Adaptive** filter, the algoritm will try to calculate the best window-size by itself, and use a general algoritm to smooth the color range in each window. If you use **Adaptive and Recursive**, a recursive algoritm will be used in the windows, calculated by the adaptive algoritm.

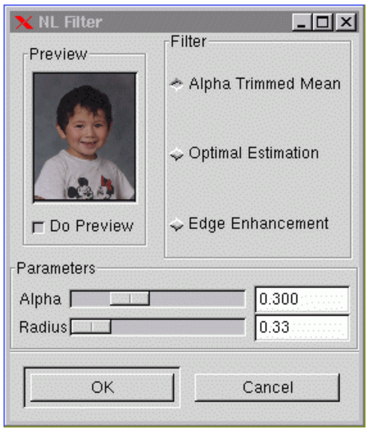
- *If you want to remove a **scratch** or other **defect** by selecting the damaged area and then despeckle it, you can successfully use **Recursive and Radius** to get rid of the scratch.*
- *If you want to remove some **noise**, then **Adaptiv**e or maybe a combination of **Adaptive and Recursive** is the way to go. If you used **Recursive and/or Radius** on the whole image, it will often get too blurred (this is of course no problem when you're only using it on a tiny selection).*
- *You can also set the amount of **black**- or **whiteness**. If the damage you want to repair should be dark, then turn up blackness. If it should be white, turn up whiteness.*

DESTRIPE

This filter corrects badly **scanned** images with stripes on them. As the stripe pattern will be different for each image, you'll just have to experiment to find the setting which will get rid of the stripes in your original.



NL FILTER



☐ Do Preview

● Alpha Trimmed Mean
○ Optimal Estimation
○ Edge Enhancement

Alpha 0.300
Radius 0.33

NL which stands for **non-linear**, is an efficient image enhancing filter. This filter uses a 7 hexagon pixel block that you can adjust with the radius slider (instead of using say, a fixed 3x3 pixel block for its filter algoritms).

There are three filters in this plug-in: **Alpha trimmed mean**, **Optimal estimation smoothing** and **Edge enhancement**.

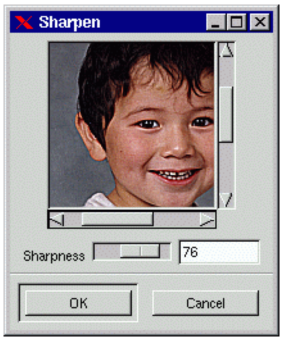
Alpha trimmed mean is the filter to use when removing ”pop” and single noise spots in the image (set alpha to 0.5). *Radius* is the strength of the filter, and *Alpha* determines whether the filter will just smooth out, or reduce noise. Recommended values are between 0.0 and 0.5. If you use values over 0.5 funny things will happen (can be quite artistic).

Optimal estimation smoothing This filter is a bit different from the first, and is very good for reducing dithering noise. Low Alpha values makes the smoothing subtle and high values (1.0) smooths all parts of the image. Radius should be from 0.8 to 1.0 for this filter to work correctly.

Edge enhancement is the opposite of the smoothing filter. It sharpens edges instead of blurring them. Radius stands for the effectiveness of the filter. Useful values range somewhere between 0.5 and 0.9.

SHARPEN

This filter **sharpens** up your image. You can set the amount of sharpness, and you can get a preview. This is a truly amazing tool, and even if it's simple, it is one of the more useful tools in Gimp when it comes to enhancing photographs. Naturally, you'll always run the risk of accentuating noise or blemishes, so if your image is damaged or noisy, try **NL edge enhancement** or the excellent **Unsharp Mask** Script Fu instead.



Sharpness 76

chapter 29

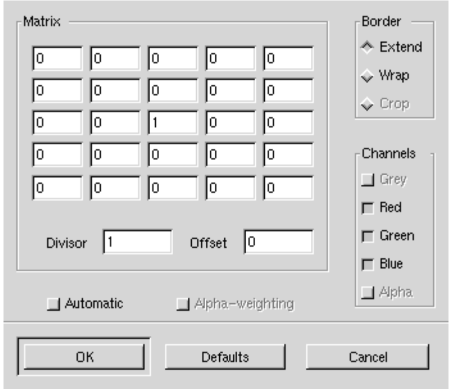
Generic filters

This is the math corner of Gimp. You will find different kind of filters that applies mathematical formulas to your image. Here is also Gimp's equivalent to Filter Factory as well as a clone of the original.

CONVOLUTION MATRIX

*documentation by **Petri Alanko***

Basically, the **Convolution Matrix** allows one to create simple custom filters. It sums together the color values at the 5x5 box around each pixel, multiplying each pixel in the box by the corresponding value from the matrix. The identity matrix does nothing.



Matrix

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

Divisor 1 Offset 0

Border: ● Extend ○ Wrap ○ Crop

Channels: ☐ Grey ☑ Red ☑ Green ☑ Blue ☐ Alpha

☐ Automatic ☐ Alpha-weighting

EXAMPLES

It works like this:

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

Divisor: 1 Offset: 0

The middle value represents the pixel to be modified. Here, the destination value is 1 (the source value, and the surrounding pixels are multiplied by 0 so they don't count). The matrix can be used for offsetting. For example:

0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Divisor: 1 Offset: 1

Now the destination value for a pixel is the source value of the pixel above it, so this offsets the image one pixel downwards. A simple blur is like this:

0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

Divisor: 9 Offset: 0

Now, for each pixel, the value of that pixel and the eight surrounding pixels is taken, added together, and divided by nine. Thus, the resulting pixel is the average of the 3x3 region around it. Similarly, a very strong (and unsophisticated) sharpen filter is like this:

0	0	0	0	0
0	0	-1	0	0
0	-1	5	-1	0
0	0	-1	0	0
0	0	0	0	0

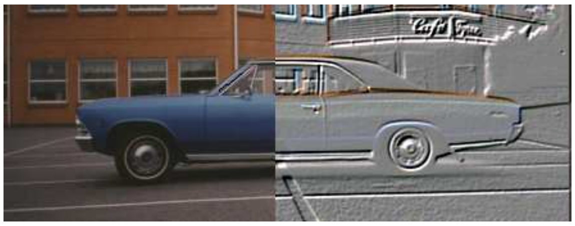
Divisor: 1 Offset: 0

This takes the center pixel, multiplies its value by five, then subtracts the values of the four immediately adjacent pixels from that.This sort of operation enhances differences between colors.

The divisor argument is just a number by which the result is divided, and the offset is added to that. The offset is useful in some cases, such as this:

0	0	0	0	0
0	1	1	0	0
0	1	0	-1	0
0	0	-1	-1	0
0	0	0	0	0

Divisor: 1 Offset: 128



In this case, the values on lower right are subtracted from the values at upper left. This is a basic embossing effect. Since these values could easily be negative, and a picture can't have negative colors, we add 128 everywhere, making this in all likelihood something like a gray bumpmap.The "Automatic" toggle just sets the divisor so that it is the sum of the matrix values. And if this sum is positive, then offset is 0, if it's negative, the offset is set to 255 (for inverting), and if the divisor is 0, the offset is 128 (for embossing).

The channel switches control which channels the plug-in operates on.The "Alpha-weighting" adds an additional factor in the calculations, namely the alpha channel. If this is used, the values of all pixels are weighted both by the matrix values, also by its alpha value. Try this out by making an image with nearly transparent (low alpha) green, and fully opaque red next to it. Now, if you don't have alpha-weighting, and you blur this, then you suddenly see a brownish line appearing between the transparent and red regions, because the transparency (”weakness”, one might say) of the green wasn't taken into consideration. With alpha-weighting on, the blurring shouldn't bring any unexpected artifacts.

MATHMAP

*Documentation by **Mark Probst***

MathMap is a plug-in which allows **distortion** of images specified by mathematical formulas. For each pixel in the generated image, an expression is evaluated which should return a pixel value. The expression can either refer to a pixel in the source image or can generate pixels completely independent of the source.

VARIABLES

In order for the expression to refer to a pixel in the original image, a few variables are set:

- x The first cartesian coordinate of the pixel.
- y The second cartesian coordinate of the pixel.
- r The first polar coordinate of the pixel (0 <= r < 360).
- a The second polar coordinate of the pixel (the distance from the center).

To make it easier to write expressions which depend on the image size, a few additional variables are set:

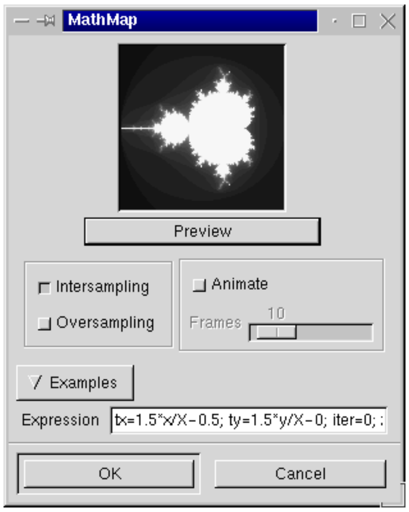
- w The width of the image.
- h The height of the image.
- R The biggest possible value for r for the image.
- X The biggest possible value for x for the image.
- Y The biggest possible value for y for the image.

For the purpose of animations an additional variable is set:

- t The time which is 0 <= t < 1. If animation is disabled, the value of t is 0. If you want to make animations loop, make sure that the images at t == 0 and t == 1 are the same.

Simple Examples: Using the functions origValXY and origValRA it is possible to retrieve values of the origin image. To distort the image into itself, the following expressions can be used:

- origValXY(x,y)
- origValRA(r,a)



Expression: tx=1.5*x/X-0.5; ty=1.5*y/X-0; iter=0; :

•This would, of course, not make a lot of sense, but it should give you a feeling for how it works. The plug-in contains a few examples, (ed note: this makes this plug-in very usable even for non mathematical engineers, it's like the Filter Factory in Adobe Photoshop) so you can get an idea of what is possible with MathMap.
Operators

+ Addition

- Subtraction

* Multiplication

/ Division

% Modulo. This also works with real numbers.

Conditions and Loops

•if condition then consequence end

- Returns the value of consequence if the value of condition is not 0, 0 otherwise.
- if condition then consequence else alternative end
- Returns the value of consequence if the value of condition is not 0, otherwise the value of alternative.
- while invariant do body end
- While invariant is not 0, executes body, then returns 0.
- do body while invariant end
- Executes body until invariant is not equal 0, then returns 0

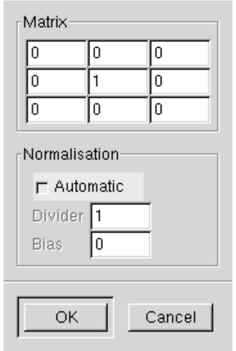
BUILT-IN FUNCTIONS

- sin(x)
- Returns the sine of x.
- cos(x)
- Returns the cosine of x.
- tan(x)
- Returns the tangent of x.
- asin(x)
- Returns the arc-sine of x.

- acos(x)
- Returns the arc-cosine of x.
- atan(x)
- Returns the arc-tangent of x.
- pow(x,y)
- Returns x to the power of y.
- sign(x)
- Returns -1 if x < 0, 0 if x = 0 and 1 if x > 0.
- min(x,y)
- Returns x if x < y, y otherwise.
- max(x,y)
- Returns x if x > y, y otherwise.
- inintv(a,x,y)
- Returns 1 if x <= a <= y, 0 otherwise.
- rand(x,y)
- Returns a random number a with x <= a <= y. Successive random numbers are evenly distributed within the interval.
- red(p)
- Returns the red component of p, which is 0 <= red(p) <= 1.
- green(p)
- Returns the green component of p, which is 0 <= green(p) <= 1.
- blue(p)
- Returns the blue component of p, which is 0 <= blue(p) <= 1.
- alpha(p)
- Returns the alpha component of p, which is 0 <= alpha(p) <= 1.
- gray(p)
- Returns the luminance of p, which is 0 <= gray(p) <= 1.
- origValXY(x,y)
- Returns the pixel value of the pixel at the cartesian position (x,y).
- origValRA(r,a)
- Returns the pixel value of the pixel at the polar position (r,a).
- rgbColor(r,g,b)

- Returns the pixel value of a pixel with red component r, green component g, blue component b and alpha component 1.
- rgbaColor(r,g,b,a)
- Returns the pixel value of a pixel with red component r, green component g, blue component b and alpha component a.
- grayColor(g)
- Returns the pixel value of a pixel with luminance g and alpha component 1.
- grayaColor(g,a)
- Returns the pixel value of a pixel with luminance g and alpha component a.

UNIVERSAL FILTER



0	0	0
0	1	0
0	0	0

☐ Automatic
Divider 1
Bias 0

Documentation by ***Ole Steinfatt***

There are two different types of signal (image) processing; **linear** and **non linear**. An example of a non linear filter is a median algorithm. The **Universal filter** is a linear filter. That means that you can describe a transfer function h which describes the output in relation to the input. In principle, a linear operation is reversible, but in practice a few non linear operations, like clipping and quantization, are involved which limit the reversibility. As for any linear system a frequency response can be calculated. In this description I will skip all mathematics and theory and just describe the basic working algorithm.

At the moment, the Universal filter uses a 3x3 matrix and two further parameters (divider and bias). The extra parameters can be calculated from the matrix in most cases, but you can set them if you want to. The filter works like this: For every pixel of the new image, the original pixel and the surrounding pixels are taken, multiplied with the values in the matrix, added to the bias value and divided by the divider. By changing the matrix values, different kinds of filters can be generated. If you put the value 1 in every matrix field (and leave the divider and bias value as calculated) you'll get a lowpass or blur effect, because you add all nine pixels to calculate the new one. If you want a smaller effect, you can increase the center value so the surrounding pixels will have less effect.To get a highpass filter or sharpening effect, you can use a matrix with -1 in all cells and a 9 in the centre.You can choose to run the filter in just one direction, by only using coefficients in the middle row, or the middle column of the matrix.To invert the picture, just put a -1 in the centre field.

The values for bias and the divider are calculated in such a way, that the resulting image will be in the normal range of 0 to 255 for color or gray values. So the divider is the absolute value of the sum of the matrix. If the sum is 0, then the divider is set to one. The bias value is derived of the sign of the matrix sum. If the sign is positive, the bias is 0, if negative it will be set to 255. In case the sum is 0, the bias value will be 128.

All calculated values are rounded to integers and clipped to the range 0 to 255 after the computation.

USER FILTER (ADOBE PHOTOSHOP FILTER FACTORY)

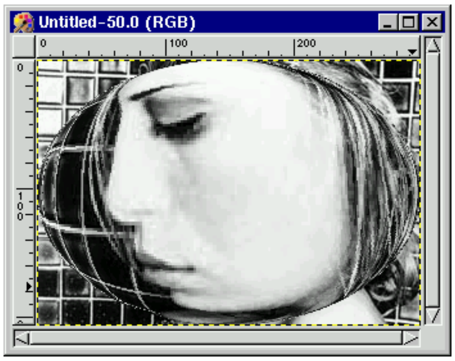
This is a clone of **Photoshop's Filter Factory**. You can load a Photoshop Filter Factory **AFS** file to it or you can create your own filters and save them in AFS format. In order to understand Filter Factory, read the manual that comes with **Adobe Photoshop** (we plan to make some documentation about this later on). The nice thing about this plug-in is that an enormous amount of free filters are now available for Gimp. Many of these filters are precompiled for **Mac** or **Windows** versions of Photoshop, which means that they not are in AFS format and therefore not compatible with Gimp. However, there is a nice little program called **Johann's photoshop Plug-in manager** that decompiles such files to AFS, so you can use them in Gimp. The big drawback is that in order to decompile, you must have Photoshop, so you may have to ask a friend who has Windowz and Photoshop to do it for you. You can take a look in Appendix X to get some info on where to get the program, filters and general info about Photoshop Filter Factory.

chapter 30

Glass effect filters

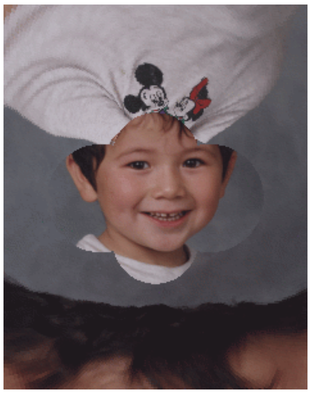
Would you like to create a fish-eye lens or make your image look like it's behind a wall of glass? Then you'll find some interesting filters here.

APPLY LENS



This plug-in places a **lens** (bulb) on top of your image. If you keep the original surroundings, it will look as if you had placed a crystal ball over your image. If you set the surroundings to background color, you will get the background color from the toolbox, transparent surroundings are naturally transparent (nice surrealist button for webpages). The **Lens reflection index** controls how ”fish-eyed” or spherical the lens will be, higher values will be more fishy.

CONICAL ANAMORPHOSE & CENTRAL-REFLECTION



This plug-in as well as the **Central Reflection** filter can best be described by making a parallel to funny mirrors in an amusement park. If we assume that the image you’re working on is a picture of you, it’s like having a conical/tube mirror, which you are looking into from the bottom. What you see is a highly distorted image, and that’s what happens with your image.

Naturally, since you are looking into the mirror-tube, you can’t see yourself undistorted, but if you check **Keep original surroundings** you will be able to see yourself in the middle looking into the mirror.

SETTINGS

You have a bunch of settings in this filter. The most important ones are: **Radius** and **Base angle**, which control the shape of the cone/tube (there is no base angle in Central Reflection since it is a tube).

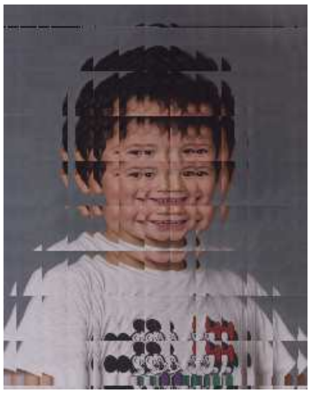
Radius determines the size of the circular bottom of the cone/tube is in pixels.

Base angle refers to how steep the cone is.

The image is flipped **vertically** by default, to make the image easier to understand. If you uncheck **Flip image vertically**, you will see what you would see if you were really looking into such a mirror, but this is probably not what you want - it looks *very* weird. We recommend to use **Antialias** to prevent the image from getting *jaggy*. Take a look at the picture above to get a grip of how the filters works. The

radius is the diameter of the circles in pixels. Using **Initializing** is recommended, because then you will get a much smoother image.

GLASS TILE



With this filter you can make it look like your image object was behind a glass wall. You can set the **height** and **width** of the tiles to a value between 10 and 50 pixels. A perhaps more accurate explanation of what the filter does, is that it splits your image in X*X pixel tiles, where each tile repeats a small part of the previous tile. This behavior will make it look like the object was behind a glass wall or a shower curtain.

REFRACT

This filter **distorts** the image with an imaginary **lens**. The lens shape is created by a second image (the **map image**). The side of the lens that faces the image is flat, and the side facing you is curved. The lens shape is determined by the map image. The map image should be a grayscale image. It will work with an RGB image but that will complicate things, so use a real grayscale image. There are a few parameters that you have to set to make your lens reflect the way you want.

PARAMETERS

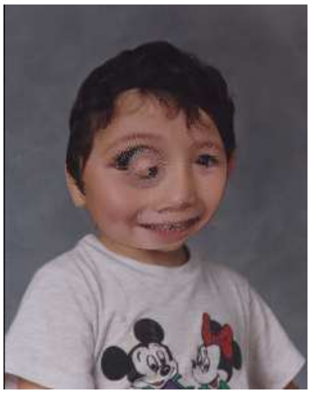
Depth (thickness): This slide controls the thickness of the lens. You can think of it as the thickness of a glass cylinder that we put the curved lens over. The actual lens is in other words built up of a flat, solid cylinder that is X in thickness, and on top of that is a user defined curved lens.

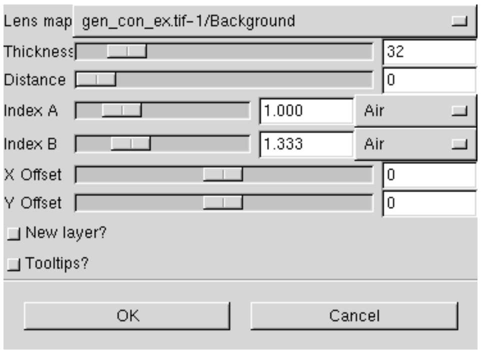
Distance: The distance between the bottom part of the lens and the image (you're holding the lens in the air and this is the distance from the image).

Index A and B: B refers to *the medium the lens is made of* (e.g. glass), and A is for *the medium you are looking through* (e.g. air). So in order to make a lens of **ice**, set B to 1.309 and if you are looking though **air**, set A to 1.0003. If you don't want to set it in numbers you can also pick it from the menu.

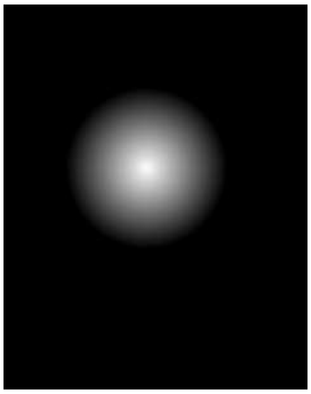
Offset is where you place the lens. I think it a lot simpler to make the map image the same size as the distort image, and make sure you place the map in the right place from the beginning.

The Lens Map: You choose what map you should use as lens in the dropdown dialog. The map should be a grayscale image. **White** represents *maximal lens thickness*, and **black** represents *minimum or no thickness at all*. In order to make it look like you're looking through a magnifying glass: Create a black map image, and apply a radial b/w gradient to it. Below you will see the map and the outcome after applying the filter.





Lens map gen_con_ex.tif-1/Background
Thickness 32
Distance 0
Index A 1.000 Air
Index B 1.333 Air
X Offset 0
Y Offset 0
☐ New layer?
☐ Tooltips?



Here are some values you can use to set a refraction index.

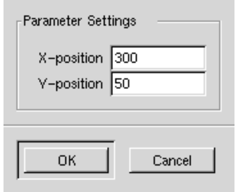
Air	1.0003	Ice	1.309	Fluorite	1.434	Rock Salt	1.544
Flintglass	1.752	Zircon	1.923	Diamond	2.417	Crownglass	1.52
Water	1.333	Ethyl alcohol	1.36	Turpentine	1.472	Glycerine	1.473

chapter 31

Light effect filters

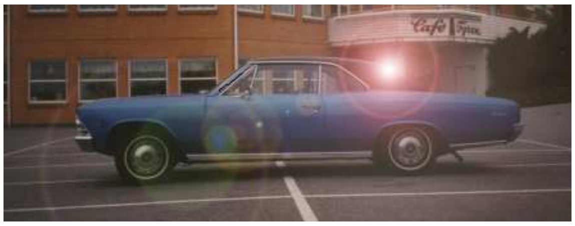
Light effects can add more than the usual flare to an image. The filters you find here are valuable accessories in your Gimp utensil kit.

FLAREFX


Parameter Settings
OK
Cancel

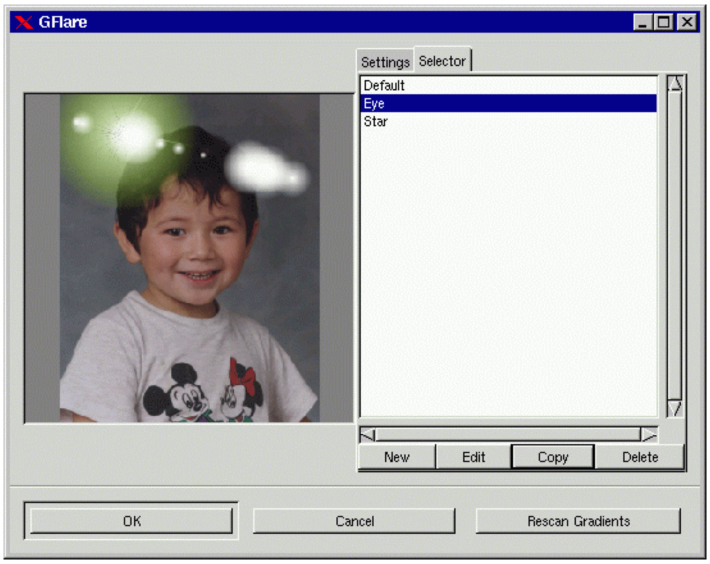
X-position 300
Y-position 50

This a simple and well made **Flare** filter. You type the coordinates in the dialog for example 300/50, which means that the position of your flare will be 300 pixels from the left and 50 pixels from the top in your image. (Note that you can't change the color of the flare, unless you change the parameters in the source code.)



GFLARE

This plug-in is so gigantic, that we're going to discuss what you can do with it in the next ten pages or so. Just kidding, but it's really huge.;-).


GFlare
Settings
Selector
Default
Eye
Star
New
Edit
Copy
Delete
OK
Cancel
Rescan Gradients

MAIN WINDOW

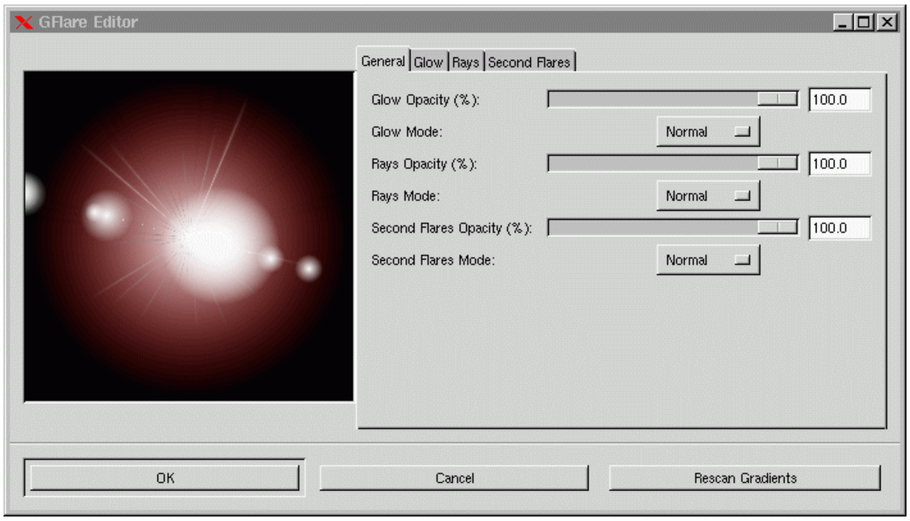
One of the best features in **Gflare** is that you can create different gflares for different situations, and **load** them in the **Selector** tab folder. This is also the right way to work with this plug-in.

When you need a different gflare-pattern, just copy one and edit the copy until you are satisfied. We think the best way to learn how to use Gflare is to first copy -> edit the default pattern. After that, you can learn how to use it in the **Settings** tab folder.

Let's start, press the **Selector** tab folder, **copy** the default pattern and name it something appropriate in the name dialog. Then press **edit**, and a new dialog will pop up - the Gflare editor.

THE GFLARE EDITOR

Yes, as I said, it's big!



Glow Opacity (%): 100.0
Glow Mode: Normal
Rays Opacity (%): 100.0
Rays Mode: Normal
Second Flares Opacity (%): 100.0
Second Flares Mode: Normal

Here we have four tab folders, where you can edit the three **foundations** that the gflare is based on, and a general view where you can set all possible combinations.

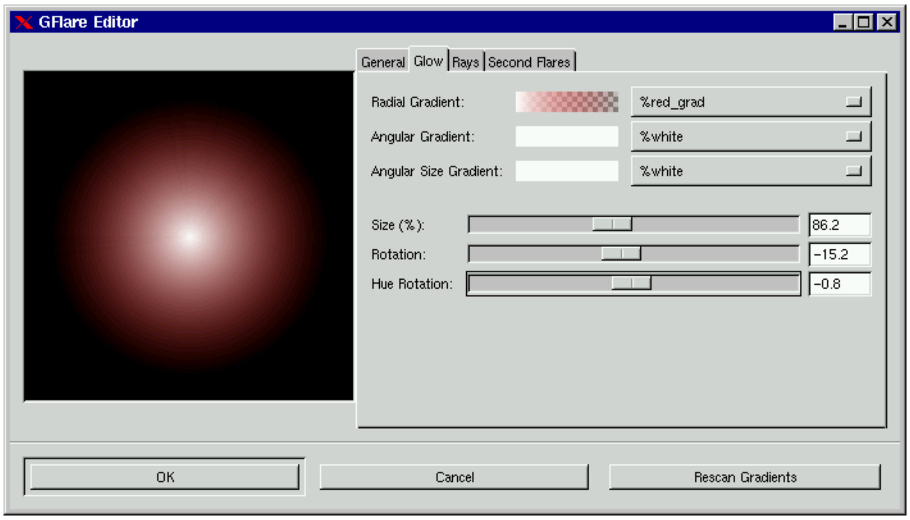
The three *key-stones* of Gflare are **Glow**, **Rays** and **Secondary Flares**.

Glow is the base foundation - the big fireball in the middle, Rays are the spikes that surround the Glow, and Secondary Flares are the attached small novas in front and behind the central Glow.

These three key-stones make up the final gflare. You can see them as three separate layers, where Glow is on top, Rays in the middle and Secondary Flares is at the bottom. In the **General** tab folder, you can set the **opacity** and **mode**, just like in ordinary layers (more info about Modes in chapter 16).

Glow settings

Glow has six different parameters:



GFlare Editor

General
Glow
Rays
Second Flares

Radial Gradient: %red_grad

Angular Gradient: %white

Angular Size Gradient: %white

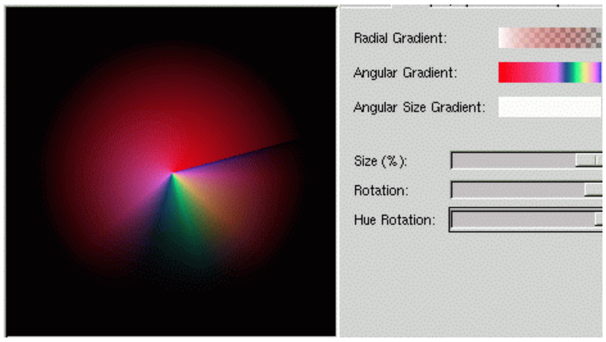
Size (%): 86.2

Rotation: -15.2

Hue Rotation: -0.8

OK
Cancel
Rescan Gradients

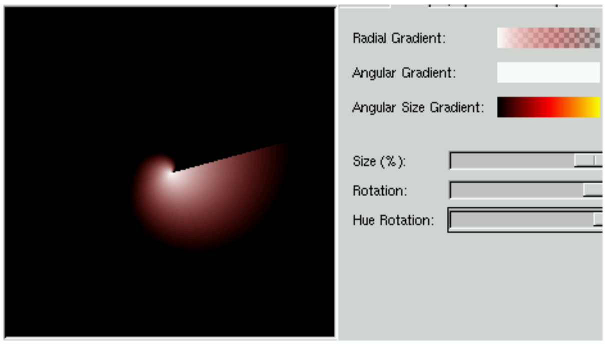
Radial gradient controls what color, shape and tone the glow will get from center to the edge.



Radial Gradient:
Angular Gradient:
Angular Size Gradient:
Size (%):
Rotation:
Hue Rotation:

Angular Gradient controls the circular color, shape and tone. If the **Rotation** option is set to 0, it will start at three o'clock and go counter clockwise. For example, if the Angular gradient fades out to transparency, the glow will also fade out (see fig 1).

The Radial and Angular gradients are **multiplied** (see Modes) and the result is the glow color.


Radial Gradient:
Angular Gradient:
Angular Size Gradient:
Size (%):
Rotation:
Hue Rotation:

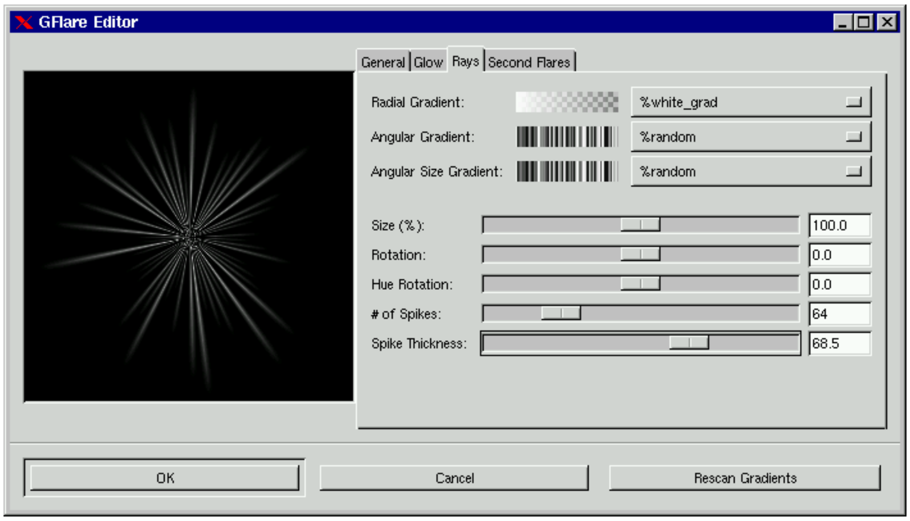
Angular Size Gradient controls the size of the **radius**. It also starts at three o'clock, and goes counter clockwise.

The radius depends on the **luminosity** of the gradient. If the "color" is black, the radius is 0%, and if the "color" is white, the radius is 100%. So, if the gradient goes from white to black, the radius will diminish as you move.

Size controls the size in %. **Rotation** controls where the Angular Size Gradient starts in deg. With **Hue Rotation** you can control the color of the whole glow. To understand the **HUE** color circle, read chapter 12. Note, the gradients starting with a "%", are internal gradients for the editor, the rest comes from the gradient directories. You can add a gradient even when you're inside the editor. To do so, press **Rescan Gradients** to make it available.

Rays

Rays: Has the same controls as **Glow** plus **# of Spikes** and **Spike Thickness**.

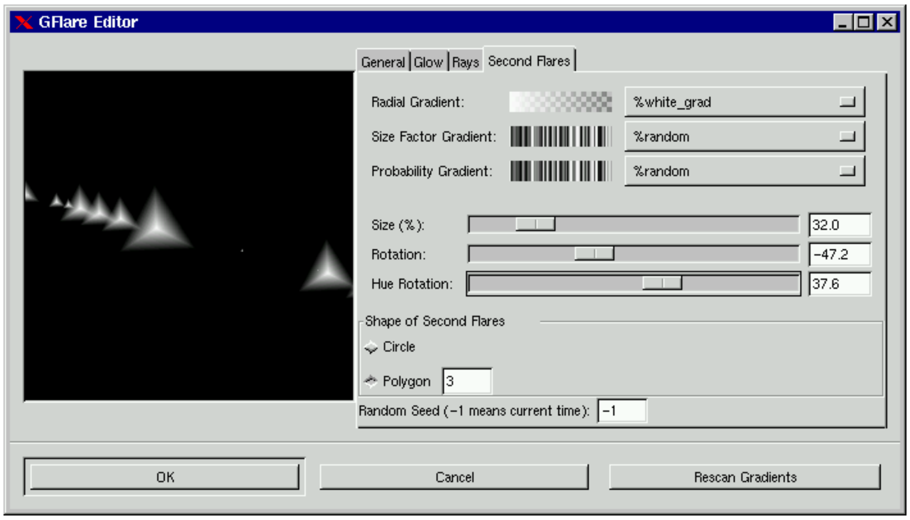

GFlare Editor
General
Glow
Rays
Second Flares
OK
Cancel
Rescan Gradients

Radial Gradient: %white_grad
Angular Gradient: %random
Angular Size Gradient: %random
Size (%): 100.0
Rotation: 0.0
Hue Rotation: 0.0
# of Spikes: 64
Spike Thickness: 68.5

# of Spikes controls the amount of spikes, but this is not the whole truth. Technically, it determines how dense or sparse the "spikeflower" will be.

Second Flares

The Secondary Flares have the same parameters as Glow plus two additional controls: **Shape of Second Flares** and **Random Seed**.



Radial Gradient: %white_grad

Size Factor Gradient: %random

Probability Gradient: %random

Size (%): 32.0

Rotation: -47.2

Hue Rotation: 37.6

Shape of Second Flares

○ Circle

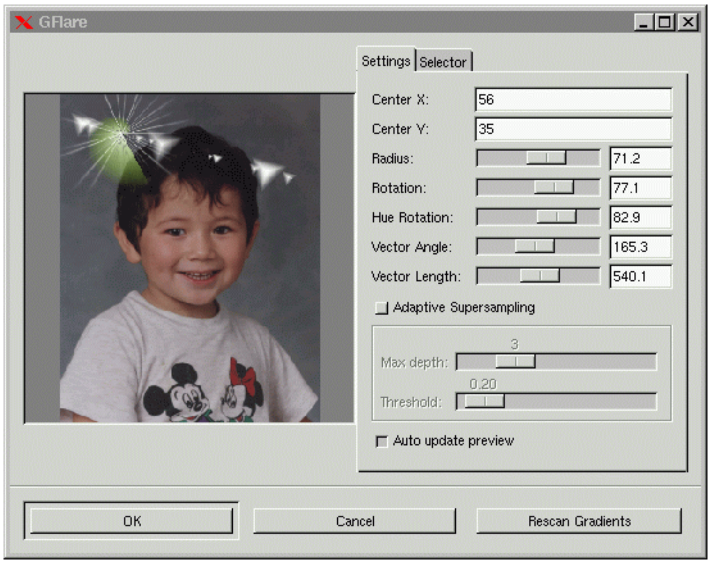
● Polygon 3

Random Seed (-1 means current time): -1

You can control the shape of the flares by choosing **circular** or **polygonal**. You can choose how many sides you want for your polygon, but if you choose a value of 30 or more, it will be the same as **Circle**.

Random seed controls how many flares there will be and where they'll be placed. If you set random seed to "-1" you will use the current time as seed value. This means that you will get a random number and location of your flares, each time you use the gflare-pattern.

BACK TO THE MAIN WINDOW



Center X: 56
Center Y: 35
Radius: 71.2
Rotation: 77.1
Hue Rotation: 82.9
Vector Angle: 165.3
Vector Length: 540.1
☐ Adaptive Supersampling
Max depth: 3
Threshold: 0.20
☐ Auto update preview

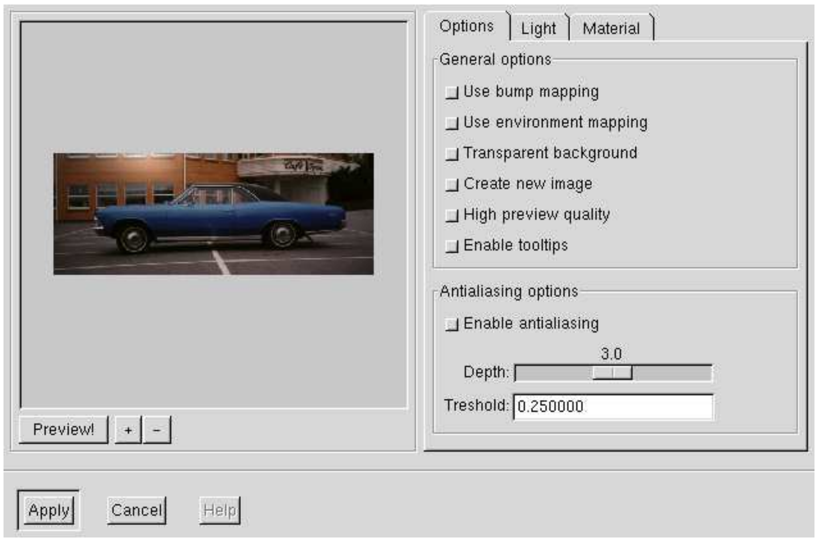
Hopefully, you now have a gflare-pattern which you can use, so let's see what we can do with it in the main dialog. Just press OK, and then press the **Setting** tab folder.

By the way, isn't it wonderful that you can create and save different gflare-patterns and then use them at the appropriate moment, just by choosing them in the **Selector** tab folder? Just remember to create a gflare directory where your gflare-patterns will be saved and make sure you specify the path in your `gimprc` file, otherwise it will fail. (see appendix A)

To place a Gflare, click in the preview window at the place you want to put it, and remember to have **Auto update preview** checked, otherwise you won't see what you are doing. (You can also type the coordinates in the X and Y fields).

- **Radius** controls the size of the Gflare.
- **Rotation** is the angle of the Gflare, and it corresponds to the angle of the gflare key-stones.
- With **Hue Rotation** you can control the color of a Gflare.
- **Vector angle** controls the direction of the **Secondary flares** in degrees.
- **Vector Length** controls the length of the Secondary flares.
- For a explanation of **Adaptive Supersampling**, see the Blend tool in chapter 7. Basically, it makes your gradients more smooth when they go from one color to another.

LIGHT EFFECTS



General options

☐ Use bump mapping
☐ Use environment mapping
☐ Transparent background
☐ Create new image
☐ High preview quality
☐ Enable tooltips

Antialiasing options

☐ Enable antialiasing

Depth: 3.0

Treshold: 0.250000

If you want to try some "real" lighting effects like lighting up a wall with a spotlight, this is the filter to use.

THE MAIN INTERFACE

In the main interface you'll find a **preview** window and a tab folder for **Options**, **Light** and **Material**.

To look at what you're doing, you must first press the **Preview** button. There's no auto-preview because that would slow things down. Every time you do something with this filter, like changing a parameter, you need to press Preview to get a grip of what's happening (don't forget this). You can also **zoom** the preview by pressing **+** or **-**.

Options

Use bump mapping This button turns on the bump map function which will add a 3D effect to the image. When you enable bump map a new tab folder will pop up.

Use environment mapping This option will pop up a new tab folder where you choose the image to "steal" the environment from.

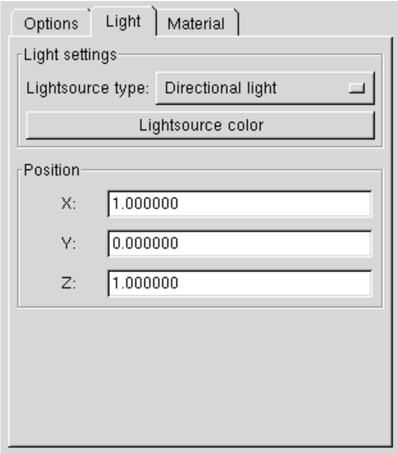
Transparent background If you have selected the bumpmap option, you can make your image transparent where the bump height is zero (bump height is zero in all black areas in the bump map).

Create new image With this option enabled, all changes will appear in a new image instead of the original one. This is nice, since you don't always want to alter the original image.

High preview quality This option is nice, but it slows down the filter. Use it when you have made all changes and want to take a final look at your work before you apply the filter.

Antialiasing Enables you to turn antialiasing on or off. We recommend using it, otherwise the images will look very jagged and ugly. **Depth** refers to the amount of antialiasing, the higher the value, the better the antialiasing, but it will also be conceivably slower. **Threshold** determines the limit of antialiasing, the process is interrupted when the difference between pixels is lower than the value in the input field.

LIGHT



Lightsource type: Directional light

X: 1.000000

Y: 0.000000

Z: 1.000000

You can set the **type of light**, the **light color** and the **position** of the light source.

Type of Light

Point light is a sort of spotlight that shines straight onto your image.

Directional Light is a softer point light (more like a normal lamp in the ceiling).

Spot light is harder and more focused than point light.

Light color

To set the color of the light, press **Light source Color** button to get to the Select light source color dialog.

Position for point light:

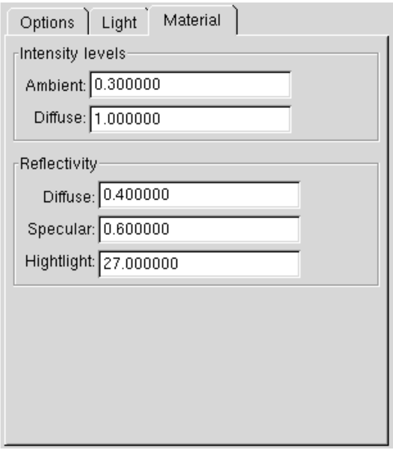
There are three coordinates for controlling light position; **X**,**Y** and **Z**.

The **X -coordinate** moves **horizontally** from -0.5 to 1.5, where -0.5 is the left-hand position, and 1.5 is the right-hand one.

The same goes for the **vertical Y-coordinate**; -0.5 is at the top and 1.5 is at the bottom.

Z is the **depth** of the light, where 0 would be the flat surface of the computer monitor. There is no upper limit for this coordinate, there are no limits for X and Y either - the values are only recommendations from our side.

MATERIALS



Intensity levels

Ambient: 0.300000

Diffuse: 1.000000

Reflectivity

Diffuse: 0.400000

Specular: 0.600000

Hightlight: 27.000000

You can control different materials' **Intensity** and **Reflectiveness**. We have found the following values to be good max/min values:

Intensity:

Ambient is the amount of the original color to show where no light falls (0.1 -> 3.0), **Diffuse** is the intensity of the original color where no light falls (0.5 -> 3.0).

Reflection:

Diffuse is here the amount of light *dark*, *soft* or *plastic* parts of the image will reflect. Naturally, a dark plastic part will reflect little light in real life, so don't set this value to high if you want your image to look good. (0.2 -> 0.9) with a nice value around 0.5.

Specular is the amount of light glossy or metallic parts of the image will reflect, (0.4 -> 0.6).

Highlight refers to how glossy/dull the overall impression of your image should be. Higher values will make the image highlights more focused, and lower values will make it lighter, and more unfocused in the highlight parts. (15 -> 50) but around (20 -> 30)is best most of the time.

Our Chevelle Malibu SS after applying the Lighting effects filter

BUMPMAPPING

Bump map is similar to the bump map filter in chapter 32 so it's wise to check that out before you start. You can only use **grayscale** images as bump maps, so if you want to bump map against the original image, you have to duplicate it and change it to grayscale mode in the image menu.

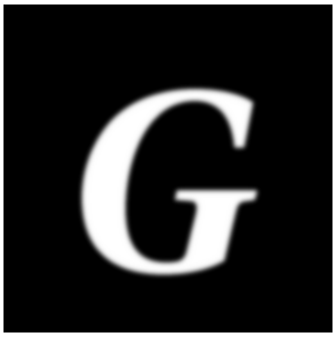
You can set the bump curve to **Linear**, **Logaritmic**, **Sinusoidal** or **Spherical**. Look at the curves in the bump map filter to appreciate the difference. You can also specify the maximum and minimum height/depth of the bump map.

ENVIRONMENT MAPPING

If you use **environment mapping,** you'll get the opportunity to set your object in a setting of your choice. You can choose from all images opened in Gimp, regardless of size and shape. The image you created will look like it's inside a **sphere**, and the inner surface of this sphere is covered with the distorted environment image. Light is reflected from the sphere, and the image inside will reflect the environment mapping.

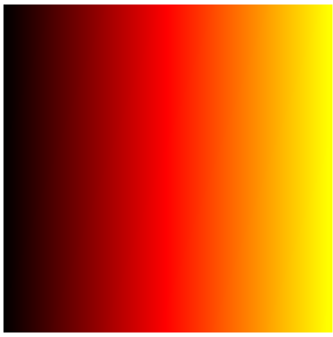
A SIMPLE TUTORIAL

Thanks to ***Tom Bech****, the creator of Lighting effects*



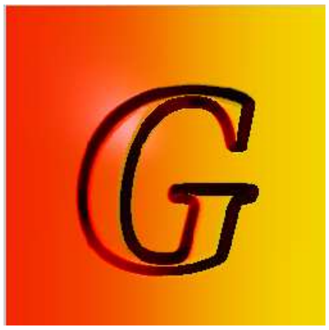
Create a new **grayscale** image (256x256) with a black background. Bring up the text tool and type a 200 pixel ”G” with the Utopia font. Place the white letter in the middle of the grayscale image.

Create a new **RGB** image (256x256). Bring up the **Blend tool** and select to blend with a **custom gradient** from the Gradient Editor. Open the **Gradient Editor** and choose German Flag Smooth (if you haven't changed the settings in `gimprc` the German Flag Smooth is the default gradient). Apply a linear gradient from the left to the right in the RGB image.

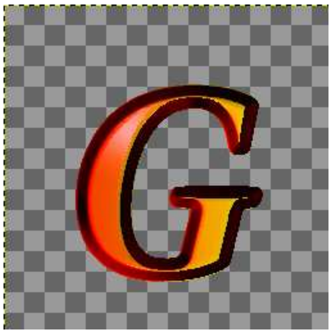


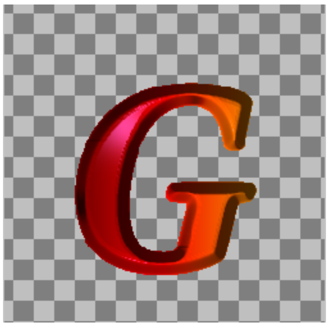
. Bring up the **Lighting Effects** filter from the **RGB** image. Check **Use bump mapping** and **Use environment mapping** in the **Options** tab folder and press **Preview**. You should now have a nice bump mapped and highlighted G.

If there are other opened images in your Gimp session, first make sure that you use the RGB image as environment map and the grayscale image as bump map.



Check **Transparent background**. You will now see your G floating in a transparent surrounding.



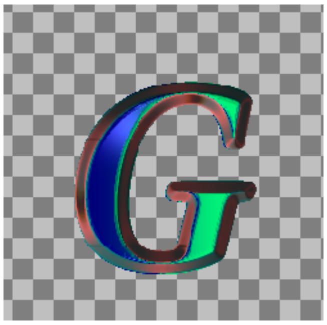


Uncheck the environment mapping, press Preview and look at the difference

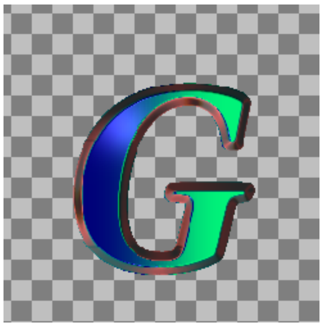
Now, bring up the **Alien Map** filter under Filters/Color from the RGB image. Apply the default values, check **Use environment mapping** and press Preview in the Lighting Effects dialog, and you'll get a brand new color gradient.

A great Gimp feature is that the plug-in will be updated as you change the image. This goes for all Gimp plug-ins.

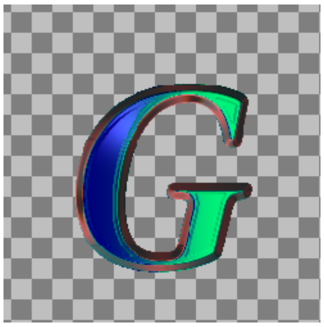
Try different curve settings:



Linear



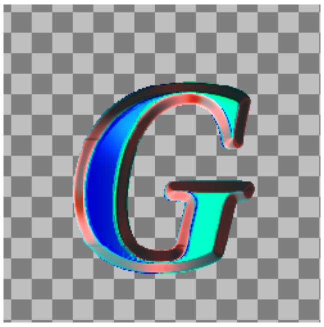
Logarithmic



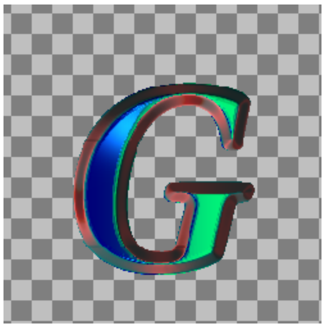
Sinusoidal

Test different types of light sources and light colors (below to the right):

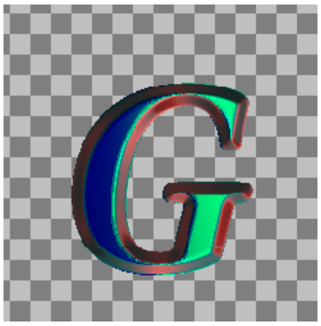
Test different kinds of material reflections and watch the difference, here with Directional light (just remember to press Preview)



Ref: Diffuse 0.8

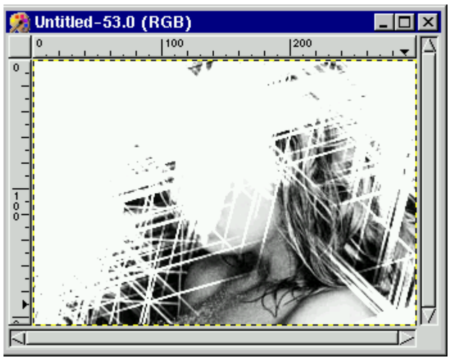


Ref: Specular 0.9



Point Light

SPARKLE



With this tool you create **sparkles**, or get a frosty, glittery feeling to an object. Sparkle selects the brightest point in your picture and puts a sparkle there. This behavior makes it very hard to predict where the sparkles will go. To use it effectively, try it in a small selection first, and you will get more control over where the sparkle will end up. The best way is generally to use a transparent layer (or a black layer + screen mode) where you put tiny white spots. The sparkles will only appear where the spots are.

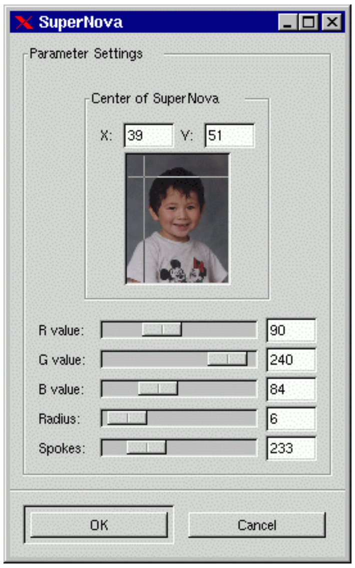
PARAMETERS

- The **Luminance** parameter controls the amount, or density of sparkles. A low value will produce sparkles only at very bright parts of your image.
- **Flare Intensity** refers to the light intensity of the stars.
- **Spike length** controls how long the star spikes will be.
- **Spike points** tells you how many spikes you will get - 1 spike point will get you 2 spikes, 2 will get you 4 spikes and so on.
- **Spike angle** is the angle of the basic spike (1 spike point), and if it's set to zero you will get a horizontal spike, 90 will get you a vertical spike, and so on.

SUPER NOVA

This plug-in creates a big shiny **super nova** in your image. The **R**,**G** and **B** slides determine the color of the supernova. You specify the color just like in the color selection dialog.

The problem is that you can't see what the color looks like, until you apply the plug-in. We suggest that you open the color dialog and choose a color there. When you're happy with the color, type those RGB values in the nova dialog). **Radius** is the radius of the inner part of the nova (the star part). **Spokes** determine how many spokes the nova will get. You place the nova with the cursor grid in the **preview** image (you can also type the coordinates in the X and Y input fields).



X: 39 Y: 51

R value: 90

G value: 240

B value: 84

Radius: 6

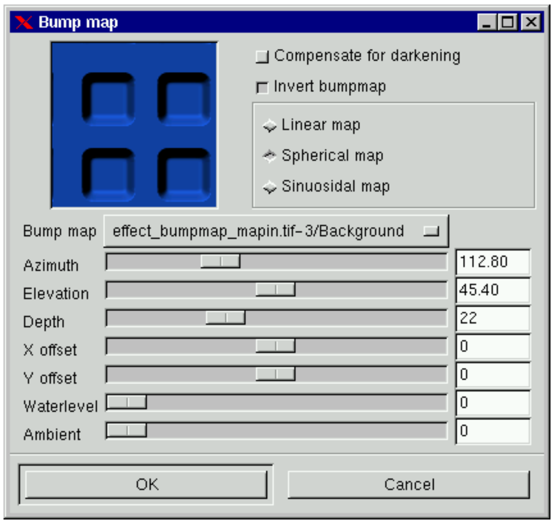
Spokes: 233

chapter 32

Map filters

If you want to bend a text along a curve, you have to use a displacement map. If you want to create 3D effects you use a Bump map, and if you want to make a pattern you map your image to tiles. There are many ways to use maps. All of the Gimp map filters will be discussed here.

BUMP MAP



Bump map

☐ Compensate for darkening

☐ Invert bumpmap

○ Linear map

● Spherical map

○ Sinuosidal map

Bump map: effect_bumpmap_mapin.tif-3/Background

Azimuth: 112.80

Elevation: 45.40

Depth: 22

X offset: 0

Y offset: 0

Waterlevel: 0

Ambient: 0

Bump map works by **embossing** an image, and then **map** it to another image. This will create a 3D effect in the second image, just like if you had modeled your image in clay.

First read about **Emboss** to understand **Azimuth**, **Evaluation** and **Depth**.

There are many differences between Bump map and the bump map in the Emboss plug-in. First of all, in **Emboss bumpmap**, you can’t specify the bumpmap image you want to ”bumpmap with” (picture 2). The Bump map plug-in also has more options. And in this plug-in you can bumpmap any type of image, whereas Emboss only works for RGB images without Alpha.

USAGE

Because you can bumpmap any image, you may have to adjust the position of the background bump map. To do so, use the **X** and **Y offset slides**. You may also want to compensate the loss of luminance resulting from embossing, with the **Compensate for darkening button**. Because Emboss raises light pixels and carves dark pixels, it quite simple to reverse so that otherwise dark and carved parts of your map will turn light and raised. To achieve this, check **Invert bumpmap**.

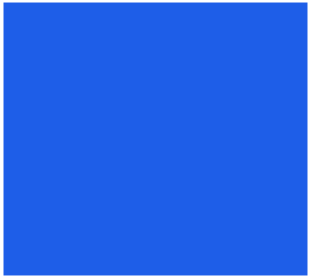
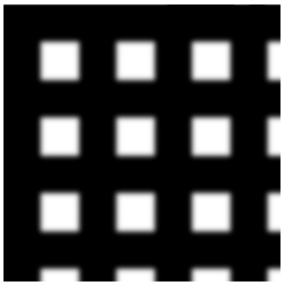
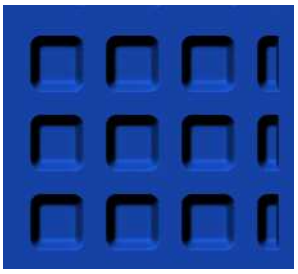


Image to bump map



Map image

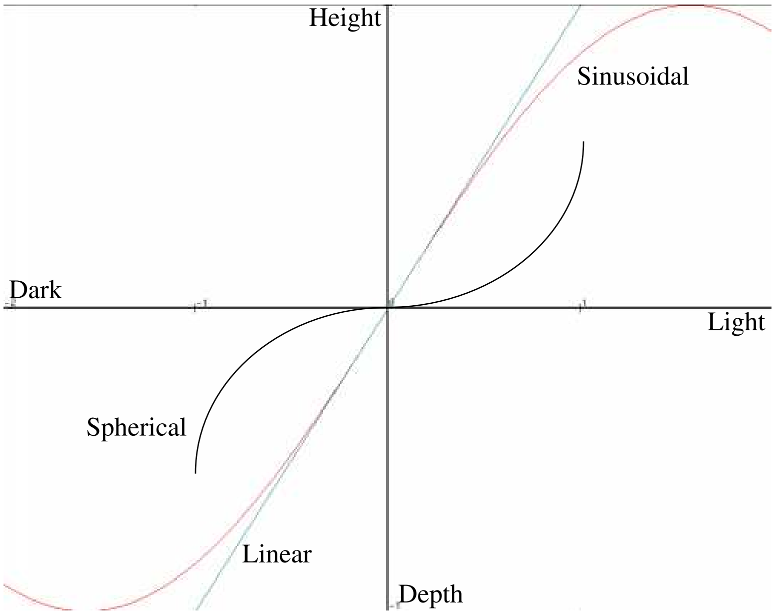


Outcome

The **ambient slide** controls how much ambient light you want in your image. A high level of ambient light will make all shadows disappear, and the raised and carved parts will be less apparent.

Water level? - what the hack is it? Well, you often use **Alpha values** or transparent parts in an image. If you use **transparency** in your map image, those pixels will be treated as dark pixels, and therefore become carved. If you slide the water level up to 255, they will be flattened and turn invisible, just as if you had raised the water level. (If you use **Invert bumpmap** they will be treated as light pixels, and get raised. Using Water Level will flatten that too. So Water Level can be a *water raiser* but also a something of a *bulldozer*).

Linear map, **Spherical map** and **Sinusoidal map** can best be described as the tools you use to create your model. Figure 1 shows the relationship between different map modes, and how they raise/carve in relation to light/dark pixels.



COORDINATE MAP

This filter does the same as the **Displace filter**. The creator of this filter writes that this is easier to use, and maybe he's right, you will have to try it for yourself. There is one limitation though, it will only work with 256x256 pixel images.

Here is how it works: The plug-in will try to map two images, (one in X direction, and one in Y direction) on an original image. The original image will get the architecture from the map images. See pictures and displace plug-in. You can't specify the offset *(just as in the Displace filter)*.

DISPLACE

This filter is a general distort filter, which can be used for nearly all kinds of distortion. You can for example use it to whirl, pinch, shift, spread or melt your image. We will try to give you a short introduction to the hidden secrets of this filter.

DESCRIPTION

It is clear that this filter will **displace** an image or part of an image, the question is how? To put it simple, the Displace filter needs a **displacement map** to tell it how to distort the image. The brightness or darkness values in this map controls how Displace moves the image pixels. To make displacement easy, always use grayscale images as maps. It will work with colored images too, but as displacement depends on the brightness/darkness of the map, the color information is simply not used.

- ***Use grayscale images as displacement maps***
- ***The amount of displacement is based on the brightness/darkness of the map***

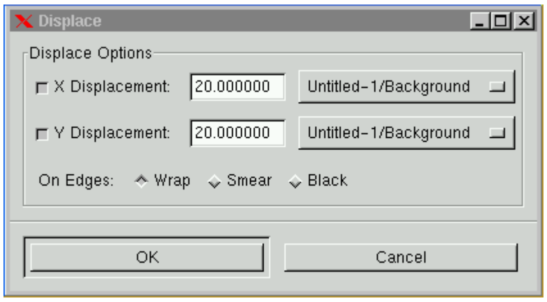
CALCULATIONS

As we know from earlier chapters, you can use 256 shades of gray in a grayscale image. How bright or dark a gray ”color” is, depends on the **Intensity** value, which goes from 1 (black) to 256 (white). In the middle of this range we find ”medium gray” with a value of 128.

One of the fundamental parts of displacement is that the ”dark values” 0 to 127 will be displaced in one direction, the ”medium gray” value 128 will not be displaced at all, and the ”light values” 129 to 256 will be displaced in the opposite direction of the ”dark values”. To make thing easier we can put this range in a new perspective. If you think of -128 as total darkness, and 0 as ”medium gray” and +128 as total lightness, it becomes more clear that displacement will go in different directions, since we have negative and positive intensity in our map. The maximum displacement in either direction happens where the map has a value of either (-128) or (+128).

- ***Displacement is based on a brightness/darkness scale, ranging from -128 to +128***
- ***There will be no displacement where the value is 0***
- ***The displacement goes in two directions. Negative values displace to one direction, and positive to the opposite direction.***

THE USER INTERFACE



☐ X Displacement: 20.000000 Untitled-1/Background

☐ Y Displacement: 20.000000 Untitled-1/Background

On Edges: ● Wrap ○ Smear ○ Black

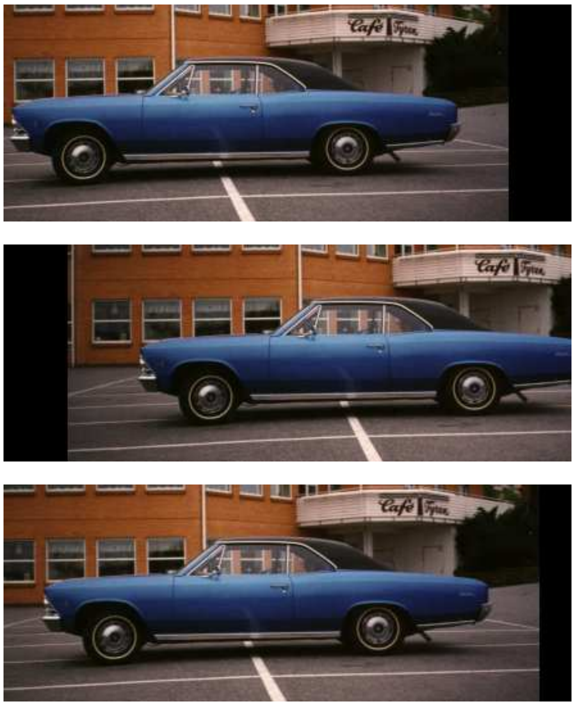
Let’s take a closer look at the **user interface**. As you can see, you can displace in both X -and Y directions. You can also set how much you want to displace, and which map you want to use. To make it easy to understand, we will stick with only one direction (X in this case).

The first thing you have to do is create, or open the image to be displaced (We will render a grid system so you can see what’s happening).

The second thing you have to do is to make a displacement map. The easiest way to do this is to **duplicate** your image, because the map must have the same size or proportions as the image you want to displace. Convert the duplicate to Grayscale and clear it.

- ***The map must have the same proportions as the image or selection that is going to be displaced***

EXAMPLES



Example 1; basic displacing

- To make it easy; copy, grayscale and clear the image that is going to be displaced, and use that as the base for a map. Now, bring up the Displacement plug-in from the image which is going to be displaced.
- Uncheck ”Y” and make sure that ”X” is checked and check Black.
- Set the X displacement to 50, and bring up the map image.
- Fill the map with white (+128), choose the name of the map image in the displacement dialog, and press OK. The image has now been displaced 50 pixels to the left (pic. X).
- Now press `Ctrl -Z` and do it again, but this time with a map which is totally black (-128) The image has now been displaced 50 pixels to the right (pic X).
- Let’s do this with a map with a value of +64 (a gray with the Intensity value 191). This will displace the image 25 pixels to the left because (64 *50)/128 is 25, which leads to the following algoritm: (value (of gray) * (displacement value))/ 128 equals the amount of displacement. (pic X).

Positive values will displace to the left, and negative to the right. More examples: Gray value: -64 (dark) Displacement value: +50 -> (-64 *50)/128 = -25 -> 25 pixels displacement to the right. Gray value: +64 and Displacement value: -50 -> (64 *-50)/128 = -25 -> 25 pixels displacement to the right.

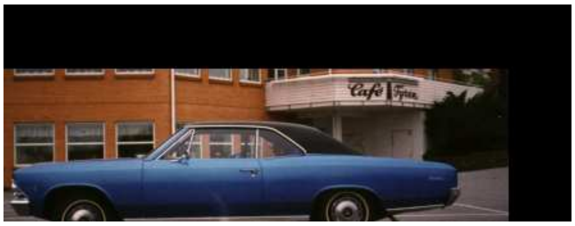
- ***If you have positive ”X” displacement values, a bright map will displace to the left and a dark to the right***

- ***The amount of displacement is based up on this algoritm: (value *displacement)/128= displacement in pixels, positive to the left and negative to the right.***

The above is also true for Y direction displacement, you just have to switch left to top and black to bottom.

- ***If you have positive "Y" displacement values, a bright map will displace to the top of the image and a dark to the bottom of the image***
- ***The amount of displacement is based on this algoritm: (value*displacement)/128= displacement in pixels, positive to the top and negative to the bottom.***

Example 2; displacing in two directions

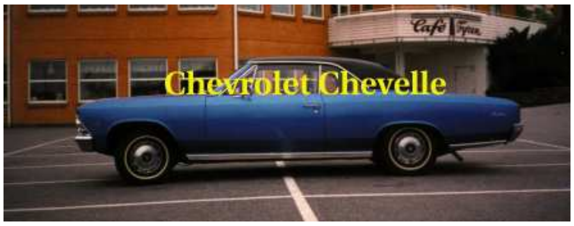


Now we move to more complex displacement. As you see in the dialog box, you can displace in both X and Y direction. This means that you can e.g. displace the image towards the bottom left corner. To achieve this, you have to make **two maps**, one white and one black. Map the black one to Y and the white one to X. Your image will now be displaced toward the bottom left corner (pic X).

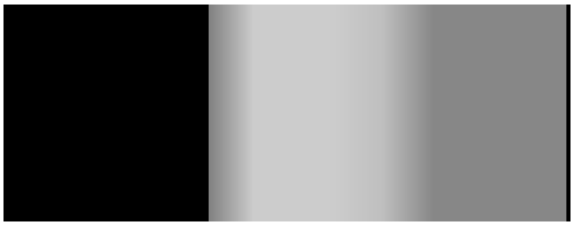
Example 3 and 4; spreading and curving

We said earlier that you can achieve any kind of distortment with this plug-in, so we will give you two more examples:

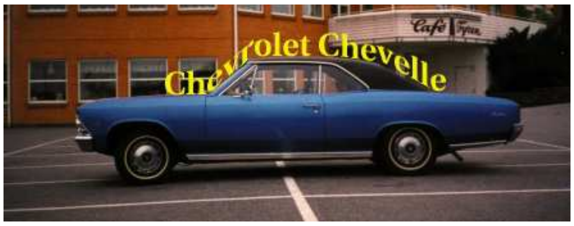
Spread horizontal: Create a map made of horizontal black, white and gray stripes. The different intensity in the stripes will make the displaced image look as if you have spread it in horizontal ribbons. You have to experiment by yourself to figure out what you can do, but trust me - you can do a lot with this plug-in.



Two-layered image, the Chevelle in the background layer and the text in layer 1.



The Y displacement map, the displacement value is 50. Note! the black ”color” is transparent.



The final outcome after a few calculations.

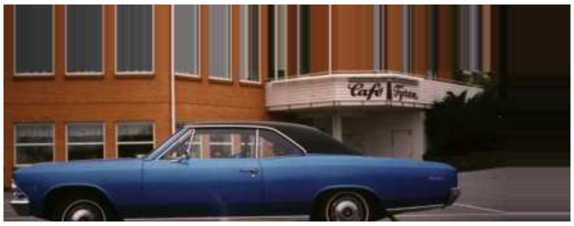
Curving Text: The obvious thing is to bend text over something circular, e.g a terrestrial globe. To achieve that, you’ll have to create a suitable gradient (dark in both ends and light in the middle) with the **Gradient Editor**. Here we have created a displacement map that will make the text flow around the shape of the car.

TIPS & TRICKS

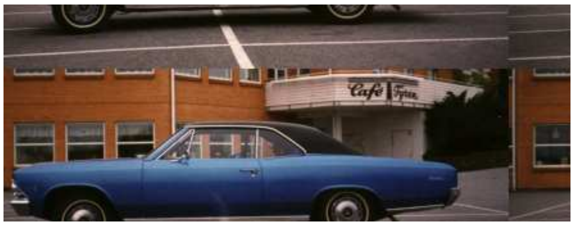
Here is some additional information. Sometimes it’s hard to create the kind of map you want, for example when you only want to displace part of your image. When you experiment with different maps, you’ll often want to darken/lighten them, but this will inevitably alter the ”medium gray” value. The perfect 0 will be transformed to something which will displace your image. The way to come around this problem is to make the static parts of your map **transparent**, because fully transparent pixels are considered to have a value of 0. Black pixels which are (almost) transparent are considered to have a value of a bit under 0, or somewhere around (-5). The same goes for semi-transparent white pixels, but they will get a positive value around (+5), so **Alpha values** should play a very important part in your map making.

WHAT’S THE DIFFERENCE BETWEEN BLACK, SMEAR AND WRAP?

Say that you’re using a white map, a displace value of 50, and that you only want to displace in the X direction. When the image is displaced 50 pixels to the left, there will be 50 pixels missing at the right side of the image. If you check **Black**, a black color will fill this part for you. If you press **Smear**, those 50 pixels will be stretched from the right part of the image (pic X). If you press **Wrap**, the 50 pixels you pushed out of the frame at the left side, will appear at the right side of the image to cover for the missing pixels.



Smear



Wrap

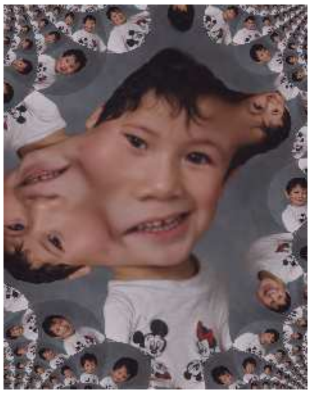
MORE CALCULATIONS

How to calculate where a pixel will end up: Let's choose a pixel which is positioned at 50(x)and 50(y) in the image coordinate system.We go to our **maps** and check the **Intensity** value of the equivalent pixel in that position. In this case we presume that the Intensity value we got from map X (the horizontal displacement map), was 230, which means (230 -128)= 102 in the displacement range. Map Y (the vertical displacement map) had an Intensity value of 55, which means (55-128) = (-73) in the displacement range. We also choose a displacement value of 50 for X, and 30 for Y. If we put these values in our algoritm.

- (102*50)/128=39.84 which is the displacement of map X (the pixel will displace 40 pixels to the left)
- (-73*30)/128=-17.11 which is the displacement of map Y (the pixel will displace 17 pixels downwards)

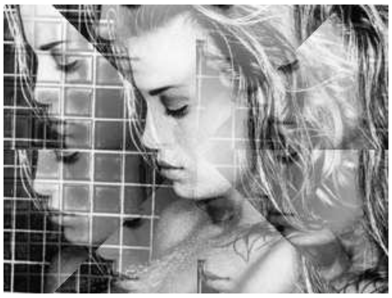
These calculations tell us that our pixel will end up at x=10 and y=67, i.e. the pixel will be moved a bit closer to the bottom left corner.

FRACTAL TRACE



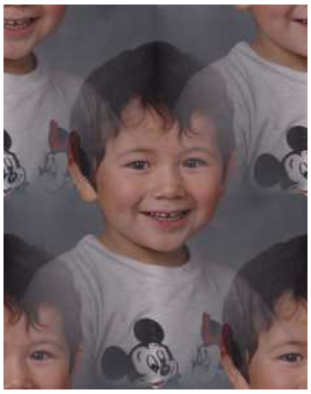
Fractal Trace distorts your image using a **Mandelbrot** fractal. It does it in an unusual way by mapping your image to the fractal (see picture). There are five parameters that control the mandelbrot fractal. For a better understanding of the parameters, read about the **Mandelbrot Fractal filter** in the **Render** menu. **Outside Type** lets you choose between four different backgrounds (as in the Displace filter).

ILLUSION



Illusion duplicates your image X times, and puts them in a ring around the centre of the original image. The surrounding images are superimposed on the original, so it looks a bit blurry or ghostlike. The only option in this filter is how many (X) copies you want to apply to your image. Another word for this filter is **kaleidoscope** which is a quite accurate description of what it does.

MAKE SEAMLESS



This plug-in prepares an image for **tiling** by creating **seamless** edges. This filter does away with all those ugly edges, and it's an easy way to create good-looking patterns.

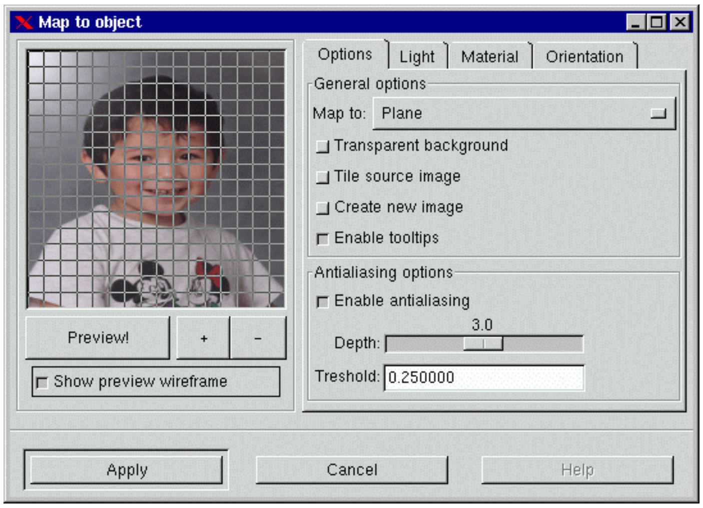
If you want to create a **seamless pattern** as a background for your desktop or in a webpage, this is (often) the best tool to use. Note that the centre of the original image will constitute the only focused part of the new image. Mirrors of the original image are placed in the centre and in all four corners of the Seamless image, and those mirrors blend together at the edges, causing it to look a bit blurred or double-exposed in those areas.

If an important part of your image (like a face) is somewhere else than in the middle, you may have to do some correction work by copying parts of the original image, and pasting them into the seamless image. If you feel that the result of this filter is to soft or blurred, try to correct your image with the Offset plug-in in the Image menu.

MAP OBJECT

This filter will **map** your image to a **sphere** or a **plane**. You can also set different l**ighting effects** on the mapping object to make it even more convincing. You can even set how different ”materials” will appear in your sphere or plane. This is a very nice plug-in for creating 3D-effects in Gimp. You can for example create a perfect football (soccer, not american). There are of course many more applications for this excellent plug-in, the only limit is your imagination.

MAIN INTERFACE



General options

Map to: Plane

☐ Transparent background

☐ Tile source image

☐ Create new image

☑ Enable tooltips

Antialiasing options

☑ Enable antialiasing

Depth: 3.0

Treshold: 0.250000

☑ Show preview wireframe

In the main interface there is a **preview** window and a tab folder for **Options**, **Light**, **Material** and **Orientation**. To look at what you're doing, you must first press the Preview button (there's no auto-preview because that would slow things down).

A hint is to use **Show preview wireframe**, then your object will preview as a wireframe, and you can follow what you are doing in real time. Whenever you need to take a good look, just press **Preview**. You can also **zoom** the preview by pressing **+** or **-**.

Options

You can choose to map to a **plane**, or to a **sphere**. Checking **Transparent background** results in a transparent surrounding.

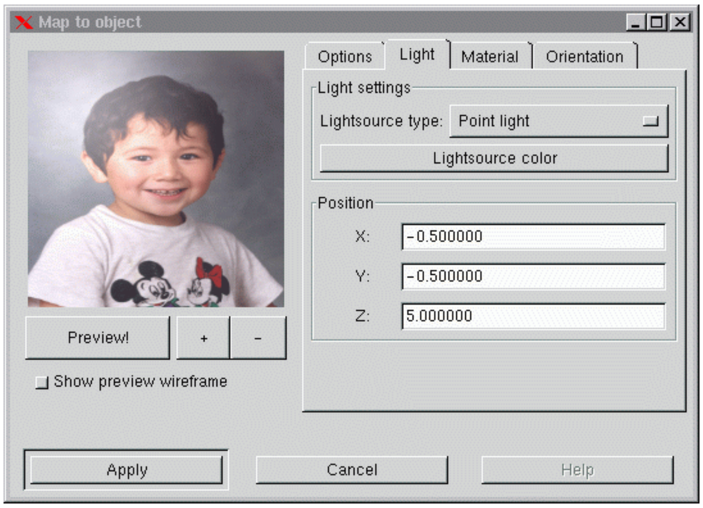
Tile Source Image means that the plane or sphere will be repeated or tiled, where it would otherwise end. For example if you map to a plane and this plane is tilted in some direction, there will be a lot of space around the output image. Instead of having this space filled by a background color or transparency, the plane will tile itself where it ends (see pic X).

Create new image preserves the original image, instead it creates a duplicate image where the filter takes effect.

Enable tooltips is a nice option. When you get to know the filter and already know what the tooltips say, you can disable the tooltips messages by unchecking this option.

The **Antialiasing** option enables you to turn antialiasing on or off, but we recommend to use it, otherwise the images will look very jagged and ugly (there can of course be moments when a non-antialiased image can be useful). **Depth** refers to the amount of antialiasing, the higher the value, the better the antialiasing, but it will also be conceivably slower. **Threshold** determines the limit of antialiasing, the process is interrupted when the difference between pixels is lower than the value in the input field.

LIGHT


Map to object
Options
Light
Material
Orientation
Light settings
Lightsource color
Position
Preview!
+
-
☐ Show preview wireframe
Apply
Cancel
Help

Lightsource type: Point light

X: -0.500000

Y: -0.500000

Z: 5.000000

You can set the **type** of **light**, the **light color** and the **position** of the light source, and if you have chosen **Directional Light**, you can also set the **angle** of the light source.

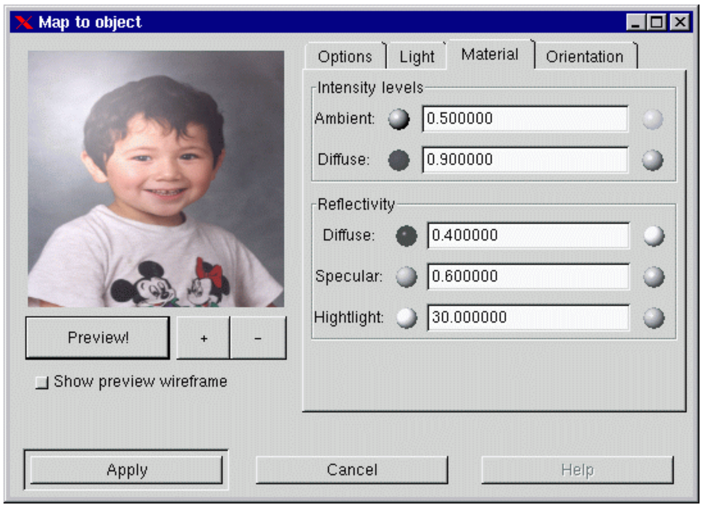
Point light is a sort of spotlight which shines straight onto your image.

Directional Light is a softer point light (more like a normal lamp in the ceiling).

To set the color of the light, simply press **Lightsource Color**.

Position for pointlight: There are three coordinates; X,Y and Z. The X -coordinate moves horizontally from -0.5 to 1.5, where -0.5 is the left-hand position, and 1.5 is the right-hand one. The same goes for Y; -0.5 is at the top and 1.5 is at the bottom. Z is the depth of the light, where 0 would be the flat surface of the computer monitor. There is no upper limit for this coordinate, but if you exceed a value of 5, a little **blue dot** shows up on the image. You can grab this dot with the cursor and change the position of the light source by dragging (there are no limits for X and Y either - the values are only recommendations from our side).

MATERIALS



Intensity levels

Ambient: 0.500000

Diffuse: 0.900000

Reflectivity

Diffuse: 0.400000

Specular: 0.600000

Hightlight: 30.000000

You can control different materials' **Intensity** and **Reflectiveness**. The little spheres represent the properties of the material.

Low values makes the "material" look like the left sphere and high values makes it look like the sphere to the right. We have found the following values to be good max/min values:

Intensity

- **Ambient**, which is the amount of the original color to show where no light falls (0.1 -> 3.0), **Diffuse**: the intensity of the original color where no light falls (0.5 -> 3.0).

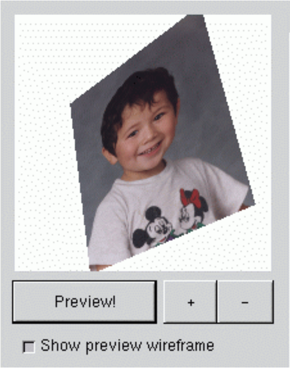
Reflection

Diffuse: which is the amount of light the dark/soft/plastic parts of the image will reflect. Naturally a dark/plastic part will reflect little light in real life, so don't set this value to high if you want your image to look good. (0.2 -> 0.9) with a nice value around 0.5.

Specular: which is the amount of light the gloss/metallic parts of the image will reflect, (0.4 -> 0.6).

Highlight: this is how much of your image that is considered glossy. Higher values will make the image highlights more focused, and lower values will make it overall lighter, and more unfocused in the highlight parts. (15 -> 50) but around (20 -> 30)is the best.

ORIENTATION


Preview!
+
-
☐ Show preview wireframe

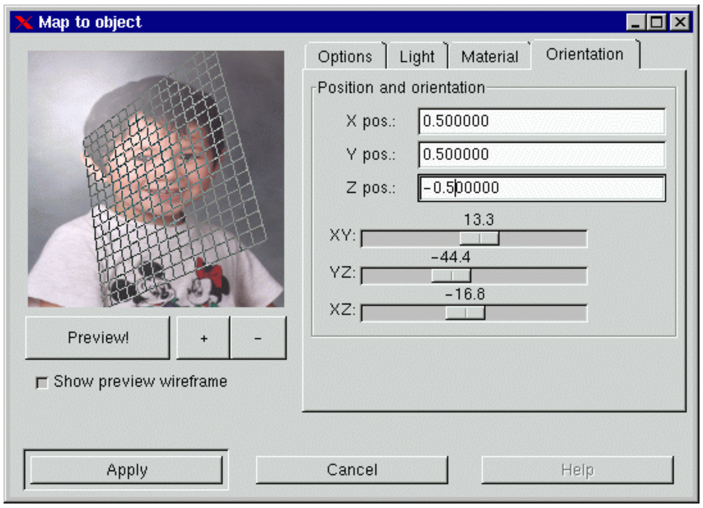
The best way to understand this folder is to bring up the **wireframe preview,** and play around. It will take you less than a minute to learn it.

X,Y,Z pos. moves the drawable, just like **Offset** in the **Image menu**. The option name stands for the position of the centre of the image.

The default centre value of the **X-** and **Y** directions is 0.5.

For the **Z-**direction, 0 is the centre value (1 is maximal zoom, and negative values is ”away from you”) there is no limit for the negative scale (but for values under -60 you will probably not see anything).

XY controls the **Y** rotation around the **Z** axle, **YZ** controls the **Z** rotation around the **Y** axle and **XZ** controls the **Z** rotation around the **X** axle.


Map to object
Options
Light
Material
Orientation
Preview!
+
-
☐ Show preview wireframe
Apply
Cancel
Help

Position and orientation

X pos.: 0.500000

Y pos.: 0.500000

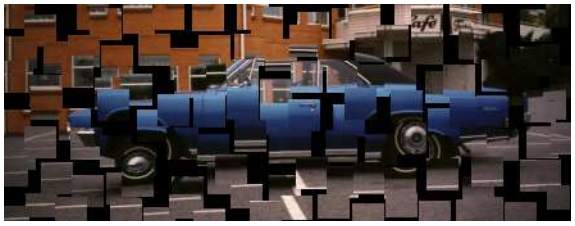
Z pos.: -0.500000

XY: 13.3

YZ: -44.4

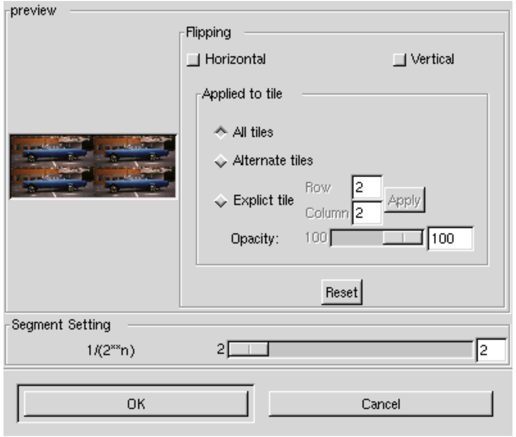
XZ: -16.8

PAPER TILE



Paper Tile makes your image look like it had been cut into small paper tiles, and then put together in a rather sloppy way, so that a variable gap is formed between each paper tile. This filter is easy to use, and there are only a few options. You set the **size** and **shape** of the tiles (in pixels) with the width and height slide bars. **Slide** determines the size of the largest gap (also measured in pixels), and you can choose between a black or white background.

SMALL TILES



preview
Flipping
☐ Horizontal
☐ Vertical
Applied to tile
All tiles
Alternate tiles
Explict tile
Apply
Reset
Segment Setting
OK
Cancel

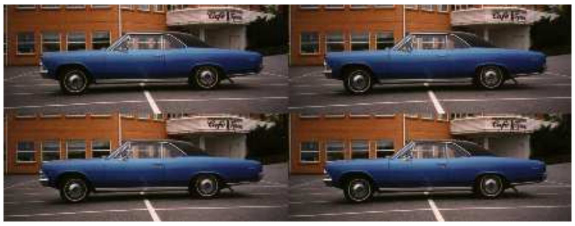
Row 2
Column 2
Opacity: 100
1/(2**n) 2

The function of this filter is the same as in **Tile**, but it tiles the image within its original size, so the tiled pictures will get smaller.

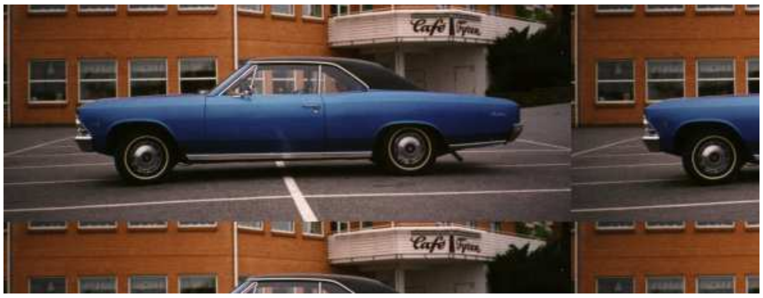
You can set how many segments/tiles you want with the **Segment** slide.

You can also **flip** tiles both vertically and horizontally. You can flip a single tile with **Explicit tile**, but you have to press **Apply** to execute the flip. You can also flip every odd tile (counting from the left) with **Alternate tiles**. If you check **All tiles** the flip will affect all tiles.

If you have an **Alpha channel** in your image, you can also set the opacity of your tile.



TILE



This plug-in **tiles** your image according to the size you set *(the new size must be larger than the old for this to work)*.

It is wise to check **New Image**, because this spares the original image. Instead it creates a new, tiled image.

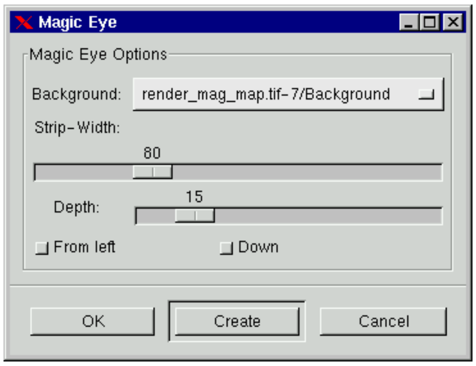
An example may clarify how it works: Say that you have a 287x425 pixel image. If you set the new size to 574x850 you will get four original images in the new image. If you don't make sure that the new image matches the old one, the result may look like the image above.You can choose **Constrain Ratio** which will make it easier to make ”cleanly” tiled images.

chapter 33

Misc. Filters

Assorted candy, here are all the filters that don't fit anywhere else. You can for example find filters that makes stereographic images here.

MAGIC EYE

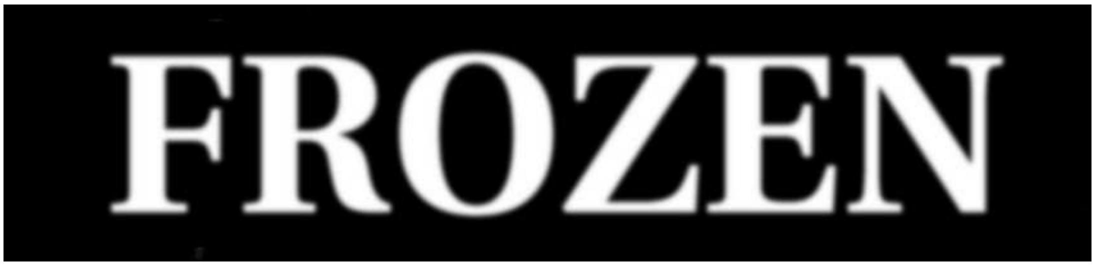


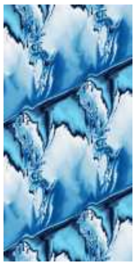
Background: render_mag_map.tif-7/Background
Strip-Width: 80
Depth: 15
☐ From left
☐ Down

Magic Eye is a way to create 3D stereo images in Gimp. I think everybody have seen 3D images in magazines. A strange-looking image that you have to look at in a certain way, and up pops a flower or something. Now you can do the same thing in Gimp. Here is how it works:

EXAMPLE AND PARAMETERS

- First create a **grayscale map image**, (the image containing the 3D thing that will popup)

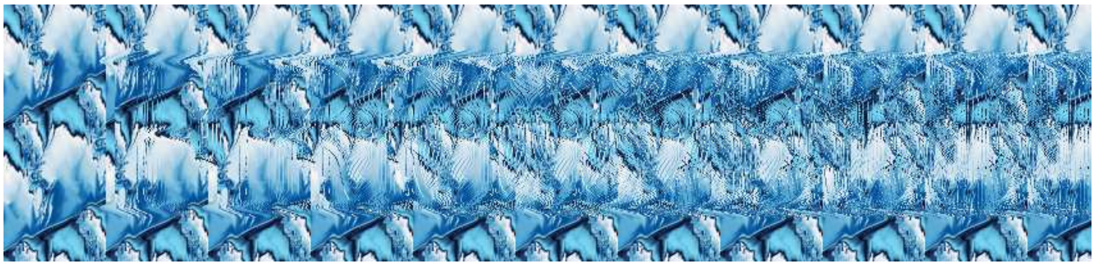




- Create a **mask image** (this is the image which will hide the map image). The image must have several colors in it (a repeated texture is great), if this is going to work properly.

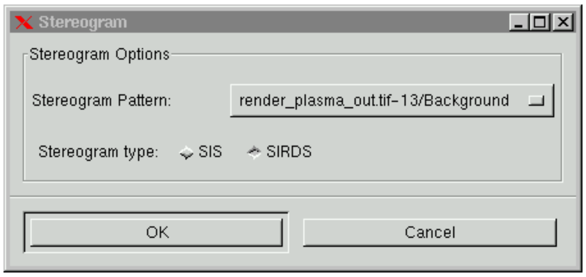
- Bring up the filter from the map image.
- Select the mask image in the **Background** menu
- In the **Strip** slide bar, set how many columns (a column is one pixel wide) of the **mask** image you want to use for the background pattern clone strip (preferably between 50 and 100). The columns are counted from the left.

- Remember that you must leave the first and last part of the **map image** black (about half the size of the background strip).
- **Depth** is the amount of 3D you want *(or how much you want the thing to popup).*
- **From left** will build your image from the left instead of from the middle of the image. *(I have noticed that a 3D image is often easier to see if you create it from the left.)*
- **Down** *carves* the image instead of *raising* it.
- You have now hopefully chosen **Depth**, **Mask** and **Strip**.
- Press **OK** or **Create**, and the new 3D image will appear



Remember that the brighter the object is in the map image, the higher it will pop up in the stereo image.

STEREOGRAM



Stereogram

Stereogram Options

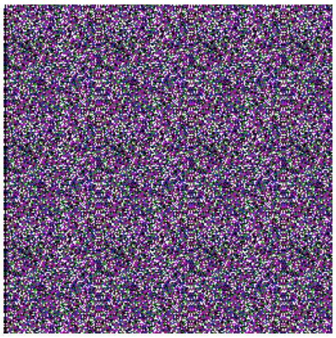
Stereogram Pattern: render_plasma_out.tif-13/Background

Stereogram type: ○ SIS ● SIRDS

Creates a 3D stereogram of a **grayscale** image (similar to Magic Eye). It works like this:

HOW TO:

- Create a simple **grayscale** image (the popup).
- Bring up the plug-in
- Choose a **background** (an RGB image of equal size)
- Choose stereogram type: **SIS** or **SIRDS**
- press **OK**

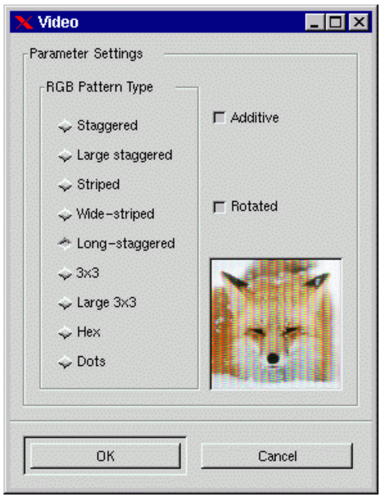


Now you have a **stereogram**. Take a good look at it with the focus slightly behind the image, until the 3D representation of your image appears in the stereo image.

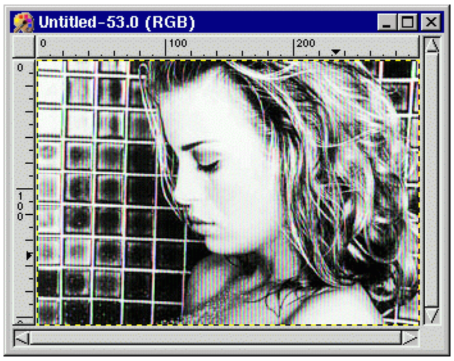
PARAMETERS

If you check **SIS** *(Single Image Stereogram)*, the background will be used as a pattern. In **SIRDS** *(Single Image Random Dot Stereogram)*, you'll only get a stereo noise for background.

VIDEO


Video
Parameter Settings
RGB Pattern Type
○ Staggered
○ Large staggered
○ Striped
○ Wide-striped
● Long-staggered
○ 3x3
○ Large 3x3
○ Hex
○ Dots
☐ Additive
☐ Rotated
OK
Cancel

This plug-in creates the illusion that the image is an ordinary low-res/dot pitch **video monitor**. You can achieve this with different patterns, which means that you can create a whole lot of ”bad monitors”. By default the pattern is horizontal, but you can by checking **Rotated** let it be vertical instead. Use **Additive** to get a realistic look. For an explanation of Additive, see the **Modes** chapter. If you don't use Additive, the pattern will just get on top of the image and darken it a lot.


Untitled-53.0 (RGB)
0
100
200
0
100

chapter 34

Noise filters

If your image is oversharp, it may some times be necessary to add some noise to improve it. This is the place to find such a filter.

NOISIFY

Noisify adds random noise to an image.

The scale goes from 0 - no noise to 1 - full noise level. You can add noise to each **RGB** channel independently. Channel #0 is **red** (or gray in a grayscale image). Channel #1 is **green** (or Alpha in a grayscale image). Channel #3 is **blue** and Channel #4 is **Alpha**. This filter will not work with indexed images.

*If you have used a lot of **Sharpen** to fix an unfocused photo, you can add a little noise to it to make it look more natural.*

RANDOMIZE

This plug-in causes **random displacement**. The random level can be adjusted from 0% to 100%. You can also specify if you want the filter to be repeated only once, or up to 100 times. There are four types of randomizing.

You can set the r**andom seed** to the current time, or you can set it yourself in the input field. All the modes works for all kinds of images, with the exception of **Blur**, which only works for RGB and grayscale images. A warning is in place for **Hurling** since a too high value, or if you repeat it too many times will get you an unrecognizable image.

RANDOMIZATION TYPES

Blurring

This type blurs your image by adapting the pixel color to the color of pixels close to it.

For each pixel a random number is picked. A pixel will be blurred if the random number is in the range of the randomize percent determined by you. It's the same for the other randomize modes.

Hurling

Hurling will change a pixel to a random color, if it's in the range of randomization.

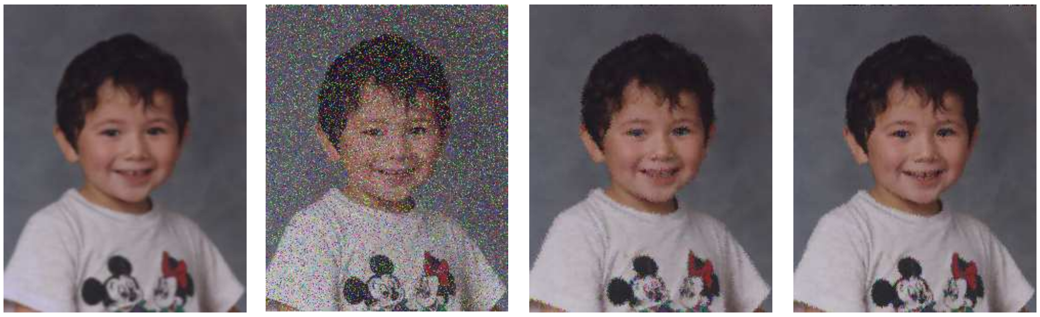
Picking

Picking will apply the color it picked from a random neighbor pixel.

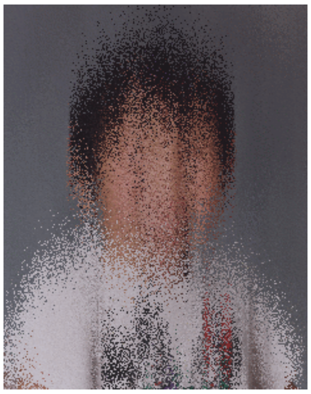
Slurring

Slurring will distort you image downwards. If a pixel is determined to be slurred it's an 80% chance that the pixel straight above is used, otherwise a random neighbor above is used.

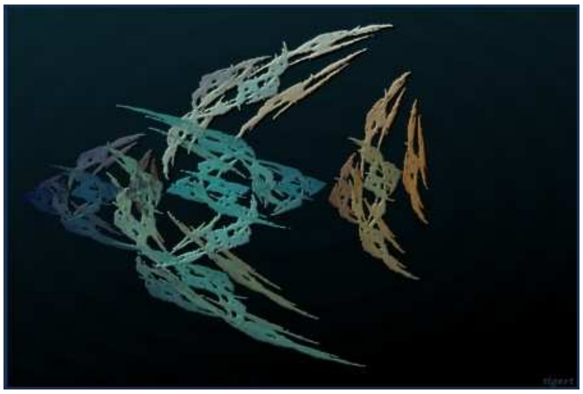
The pictures below represents from the left: Blur, Pick and Slur, all with 50 randomize. and 3 repeats Since Hurl is easy to over do. You have to lover the values. Here is Hurl with 20 randomize and 1 repet.



SPREAD



This plug-in **spreads** the pixels in your image. It will move a pixel to a random location. This location will be in the range of your setting in the slidebars. This makes it possible to spread the image vertically, horizontally or in both directions.

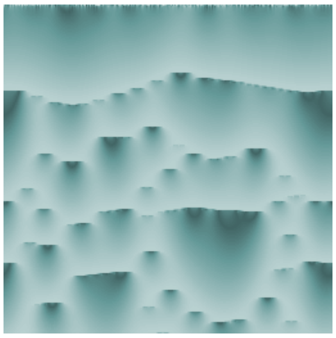


Made by Tuomas Kuosmanen in ifs compose chapter 35

Render filters

Would you like to create wonderful lifelike trees, or perhaps make an interesting texture? This is a real texture factory. You can create the most sophisticated images and textures here. There is even a drawing plug-in.

CML EXPLORER



Well, I suppose you might call this the Swiss army texture maker. You can compose an abstract pattern based on 12 different mathematical functions, 14 different ways of composing those functions and 10 arrangement variations. As if that wasn't enough, you can set also set 10 control parameters.

Now, if you think this is a lot - that's just the choice of settings for the Hue values of this plug-in. You also have those options for Saturation and Value, as well as a variety of additional options... Writing full documentation for this plug-in would be too extensive for this manual, but we'll give you a few guidelines.

GENERAL SETTINGS

Functions

- The first function type is called **Keep Image's Value**. This means that the **Hue**, **Saturation** or **Value** of the image you opened this plug-in from will stay *unchanged*, so if you have chosen this function, there is no point of changing any of the other settings in that tab folder.
- **Keep first Value** doesn't have anything to do with your image - it just sets the initial colors to standard cyan ”shower curtain” with a little spilloff from the surrounding colors in the HSV color circle (Sat. and Value also get a standard curtain).
- **Fill with parameter K**, sets a quite smooth surface, which is controlled by the **K** slide bar. The other functions which contain K, are variations of that function, but they create very interesting brocade like patterns when you raise the K-value, though high values always end up with colored noise.
- **Delta** creates similar patterns. The **Sinus** function creates wavelike shapes (with the right setting), like northern light or curtain folds.

Composition and Arrangement

You can experiment with **Composition/Misc. Arrangement** as you like, but the effect varies so much with the other parameters that you can compare it to Forrest Gump's chocolates: you never know what you're gonna get...

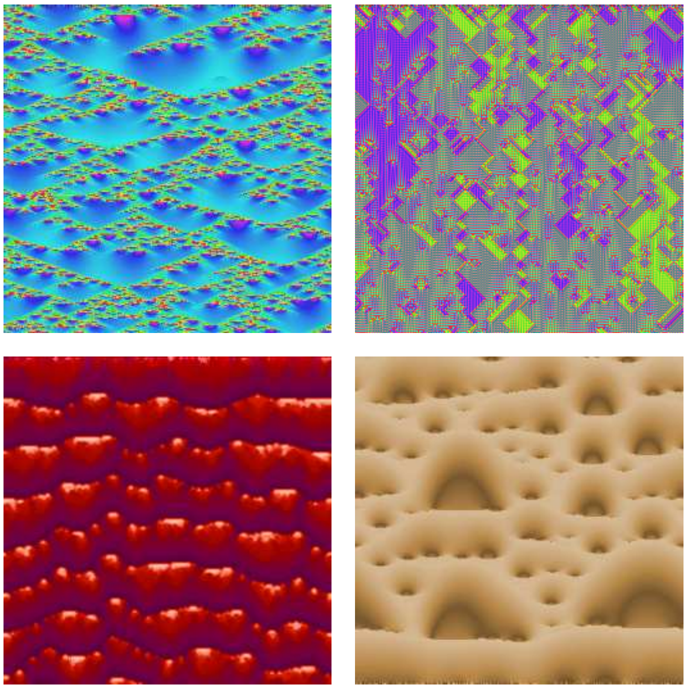
Still, it is often the case that **Composition** functions starting with **Max**, produce denser, darker patterns than **Min** functions.

Random Arrangement usually result in striped patterns, and **gradient Arrangement** causes shifts or blends from one side of the pattern to the other, so you can't use a gradient arrangement for tileable patterns.

The Slide Bars

The slide bar settings are equally hard to predict, but this is generally true:

- **Moderation Rate** goes from vertical stripes to rounder shapes.
- **Environment Sensitivity** has a similar effect, but breaks up the pattern more.
- **Diffusion distance** changes the sense of size and direction.
- **Number of Subranges** increases complexity in the pattern.
- **Parameter K** and **P** affect functions containing those values.
- **Low/High Range** control the Value ranges of each HSV tab.



ADVANCED SETTINGS

The **Advanced** settings tab allows you to experiment with **Channel** sensitivity and **Mutations** for random seed. Use it with care, or you'll just end up with colored noise.

Other options

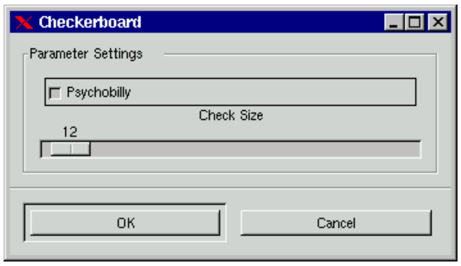
This tabfolder lets you set a **channel independent** initial value for the pattern you want to create. Note that this option often locks the pattern in a **horizontal** direction. You can also **Zoom** or **Offset** your pattern in this folder.

Misc. options

These options include the possibility to **copy** and **change** the settings from one channel to another. You have probably seen that you can **save** and **load** patterns you have created in CML explorer. Here's an option to just load the settings of a certain channel instead of the whole pattern.

Tip: You'll often get a better looking pattern by changing it to grayscale, and then back to RGB. After that, you can set any color you like with the Image/Color controls.

CHECKERBOARD



☐ Psychobilly

Check Size 12

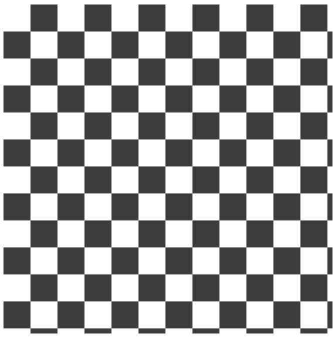
This plug-in creates **Checkerboards** - what else?

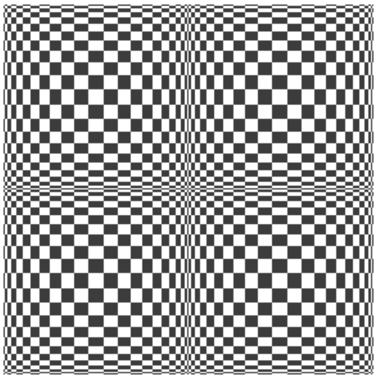
The **Size** slide controls the size of the checkers in pixels (X*Y pixels). If you check **Psychobilly**, you'll get tiled 3D "pouting" checkerboards... (see Pic 2). **Size** now represents the biggest check in a tile.

Say that you set **Size** to 4 pixels, then the middle check (which is always the biggest one) will be 4x4, the next check will be one pixel thinner, until you get to the outer check which is always one pixel wide. The check size follows the following algoritm: 1,2,3,..."check size" ..,3,2,1. From this follows that each Psychobilly tile will be 16x16 (because 4x4=16).

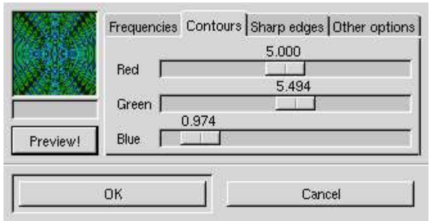
EXAMPLE

If you for example choose 12 as check size, for an image that is 288x288, you will end up with four Psychobilly tiles, because 12x12 gives you tiles which are 144x144 pixels big, and there can only be four such tiles in a 288x288 image.





DIFFRACTION PATTERNS

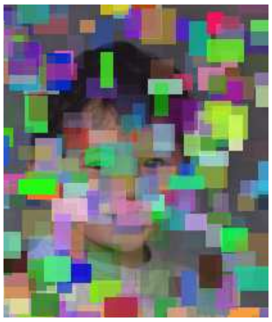


Red 5.000
Green 5.494
Blue 0.974

Lets you make **diffraction** or **wave interference** textures. You can change the **Frequency**, **Contours** and **Sharp Edges** for each of the RGB channels.

You can also set **Brightness**, **Scattering** and **Polarization** of the texture. There is no automatic preview, so you must press the preview button to update. This is a very useful filter if you want to create intricate patterns, e.g it's perfect for making psychedelic, batik-like textures, or for imitating patterns in stained glass (as in a church window).

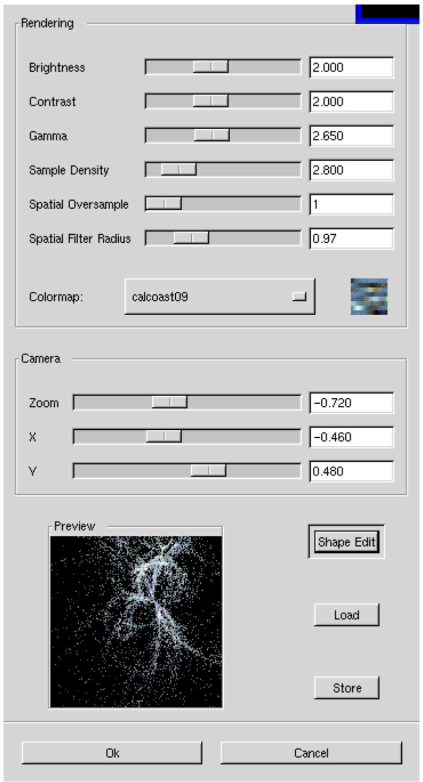
FIGURES



This plug-in adds a random number of **rectangles** of different size and color to your image. You can constrain the average rectangle size to **Min/ Max Width** and **Height**.

Density controls the number of rectangles. A low density value results in a low amount of rectangles, and a high value produces a whole lot of them. This plug-in can with success be used as a texture maker.

FLAME



Rendering
Brightness 2.000
Contrast 2.000
Gamma 2.650
Sample Density 2.800
Spatial Oversample 1
Spatial Filter Radius 0.97
Colormap: calcoast09

Camera
Zoom -0.720
X -0.460
Y 0.480

With this filter, you can create stunning, randomly generated **fractal patterns**. You can't control the fractals as in the Ifs Compose filter, but you can steer the random generator in a certain direction, and choose from variations of a theme you like.

MAIN INTERFACE

In the **Main** window dialog, you can set **Render** and **Camera** parameters. The three first parameters in the Render display are **Brightness**, **Contrast** and **Gamma**. The result of these options are visible in the Preview window, but it's generally better to stick to the default values, and correct the rendered image later with Image/Colors.

Render and Camera parameters

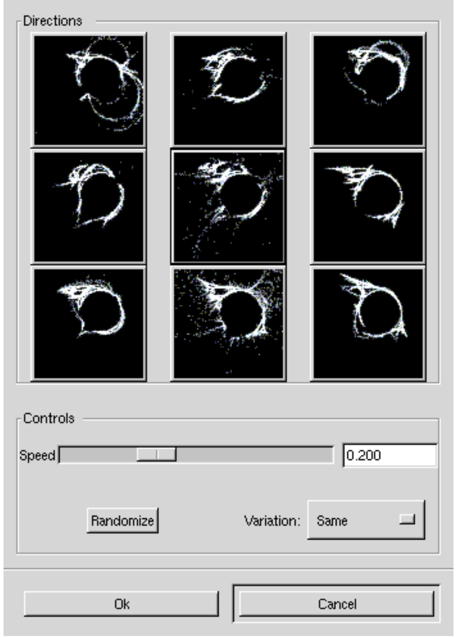
The other three parameters affect the rendering process and don't show in the preview window. **Sample Density** which controls the resolution of the rendered pattern, is the most important of these. A high sample density results in soft and smooth rendering (as a spider's web), while low density rendering resembles spray or particle clouds. The **Camera** parameters allows you to **zoom** and **offset** the flame pattern, until you're happy with what you see in the preview window. Flame also offers the possibility to **store** and **load** your favorite patterns.

Colormap controls

The **Colormap** controls the color **blend** in the flame pattern. You can set Colormap to:

- The current gradient from the **Gradient Editor**
- A number of **preset** colormaps
- The colors from images which are presently open in Gimp. You can use the **Smooth Palette** filter in the Filters/Colors menu to create suitable colormaps from your images.

THE EDIT DIALOG



Speed 0.200

Variation: Same

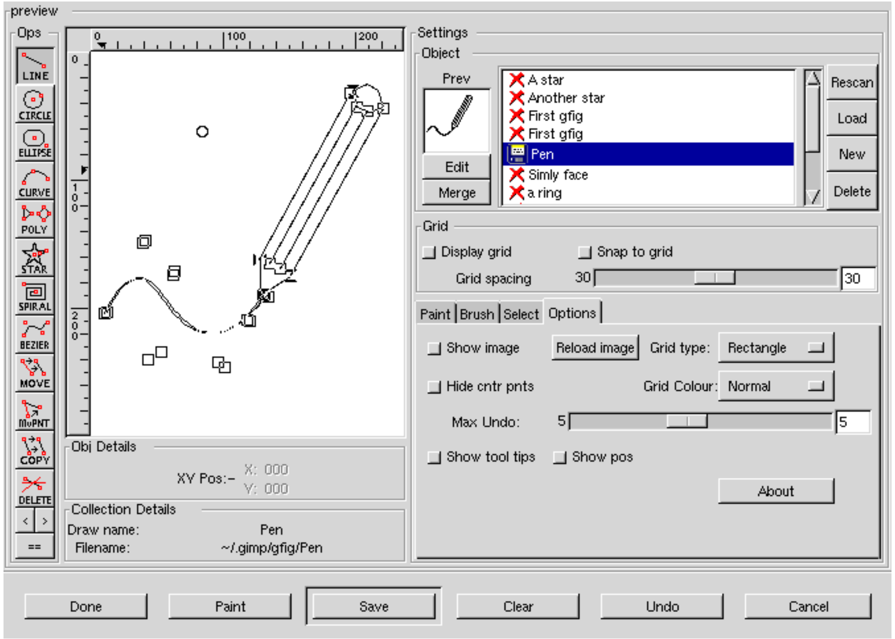
Pressing the **Shape Edit** button switches to the **Edit Dialog:**

The Edit dialog shows nine different windows. The pattern displayed in the *centre* is the *current* pattern, and the eight windows surrounding it are random variations of that pattern.

Clicking on the central image creates eight new variations, which can be adjusted with the **Speed** control. You select a variation by clicking on it, and it instantly replaces the image in the middle.

To pick a certain character or theme for the variations, you can choose from nine different themes in the **Variations** menu. You can also use **Randomize** which replaces the current pattern with a new random pattern.

GFIG



preview
Ops
LINE
CIRCLE
ELLIPSE
CURVE
POLY
STAR
SPIRAL
BEZIER
MOVE
MvPNT
COPY
DELETE
0
100
200
Settings
Object
Prev
Edit
Merge
A star
Another star
First gfig
First gfig
Pen
Simly face
a ring
Rescan
Load
New
Delete
Grid
Display grid
Snap to grid
Paint
Brush
Select
Options
Show image
Reload image
Hide cntr pnts
Show tool tips
Show pos
About
Obj Details
Collection Details
Done
Paint
Save
Clear
Undo
Cancel

Grid spacing 30

Grid type: Rectangle

Grid Colour: Normal

Max Undo: 5

XY Pos:- X: 000 Y: 000

Draw name: Pen

Filename: ~/.gimp/gfig/Pen

This is a wonderful plug-in, which adds basic **drawing** capabilities to Gimp. You can draw circles, lines, curves, ellipses, X- sided polygons (X >=3), stars, spirals and bezier curves. The drawing can then be *rendered* into your image. The drawing objects have **control points** which can be moved/edited to adjust the shape of the object. There is now also a **Selection** option in this plug-in, which transforms a Gfig drawing to a selection in your Gimp image, and a **Select+Fill** option which enables you to fill a selection directly from Gfig.

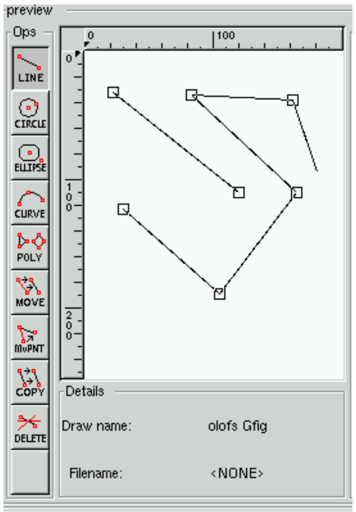
USER INTERFACE

The user interface is divided into a **Preview** area with a *drawing* area and *user tools*, a **Settings** area with *object manipulation* options, **Grid settings** to control grids and a few tabbed **folders** to control *paint* options.

PREVIEW AREA

In **Ops** you'll find the all the drawing tools. You can **draw**, **delete**, **move**, **edit**, and **copy** objects.

Lines


preview
Ops
LINE
CIRCLE
ELLIPSE
CURVE
POLY
MOVE
MvPNT
COPY
DELETE
0
100
0
100
200
Details

Draw name: olofs Gfig

Filename: <NONE>

To draw a *straight line*, simply **click** on the spot where you want your line to begin, **drag** and **release** the mouse button where you want it to end.

If you want a *crooked line* (built of several control points): hold `Shift`, this will make the control points attach to each other. E.g. hold `Shift`, click first point, move, click second point, move, click third point, release `Shift`, move, and click to get the final point.

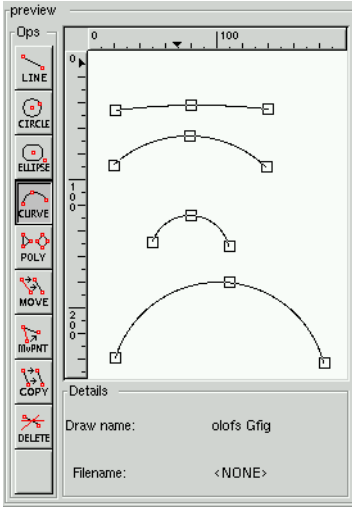
Circle

Draws a **circle**. The start *click* becomes the centre of the new circle.

Ellipse

Draws an **ellipse**. The start *click* becomes the centre of the new ellipse.

Curve


preview
Ops
LINE
CIRCLE
ELLIPSE
CURVE
POLY
MOVE
MvPNT
COPY
DELETE
0
100
0
100
200
Details

Draw name: olofs Gfig

Filename: <NONE>

Draws an **arch** or a **semi-circle**. The object has three points (two end points and a middle point).

To clear things up, follow this example: Start by drawing a curve where the curve points are in placed in a horizontal row. This will result in a **straight line** (1), you can compare this to a tiny segment of a huge circle. Now, if you raise the middle point a bit, the curve will become a low **arch**. It now represents a larger part of the imaginary circle (2). Finally, reduce the distance between the two end points and you will find that the curve gets even more **circular** (3).

Poly(gon)

The default polygon is a **triangle**, but you can change this by double-clicking on the icon. This brings up a dialog where you can choose how many sides you want for your polygon.

Star

The default star has three spikes, but as with Poly, you can double-click to adjust the number of spikes. There are three controls points for further modification with **MvPnt**.

The **central** control point allows you to change the relative position of the star by rotating it around the outer control point. The **outer** control point controls the external radius of the star, and the **middle** control point controls the size of the core radius, which determine the relative length & sharpness of the spikes.

Spiral

Draws a spiral. You can set the number of **twists** by double-clicking on the spiral icon. This brings up a dialog where you can also set the direction to clockwise or counter clockwise.

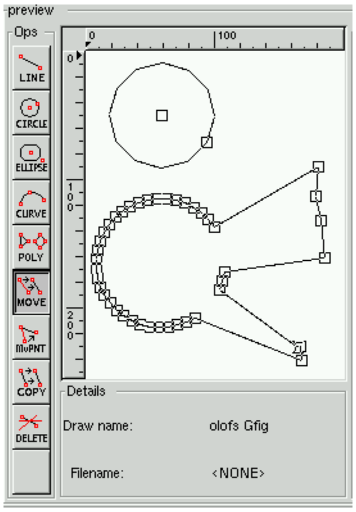
Bezier

Lets you draw **bezier curves**. They're not as easy to modify as the bezier curves in the Toolbox since there are no handles to pull, but it works quite nice for simple drawings. You'll have practise a bit to learn how to control them. As usual, you end the curve with a `Shift-click`.

Move

Moves a single object or all objects in a drawing. To move a single object, just click at a control point and drag. To move all objects hold `Shift`, click somewhere in the drawing area and drag, this will cause all objects in the drawing area to move.

MvPNT



Draw name: olofs Gfig

Filename: <NONE>

Lets you move a single control point. This function lets you alter the shape of a object by **stretching** and **moving** the lines that it's made of. As you may have noticed, you can't change the shape of a circle - you can only make it smaller or move it. A way around this problem is to use polygons. If you use a polygon with a lot of sides, you will get something very close to a circle. If you simply choose the **MvPNT** tool, you'll only change it the usual way, but if you hold `Shift` and then click at a control point with the MvPNT tool active, you'll break up the polygon into a lot of lines. Now you can drag at the "circle" control points any way you like.

Copy

Will copy an object. To use it, **click** and **hold** an object's control point, then **drag** the copy to where you want it.

Delete

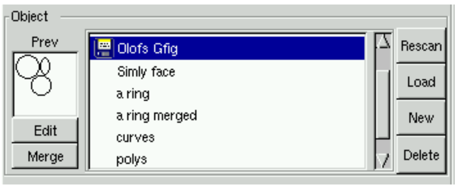
Deletes an object (the entire object, not just a control point)

Misc.

< > and ==: These little signs let you **browse** your drawing as if every line of your drawing was a *frame in a movie* **<** = backwards **>** = forward and **==** show all lines in the drawing.

SETTINGS

Objects



One of the nicest things with **gfig** is that you can **save** a whole bunch of drawings on disk. This makes it possible to create *standard figures* for your special needs.

To create a new drawing, just click **New** and name it in the **naming dialog** and an empty drawing area will appear. As you see, the new drawing name is highlighted in blue, and there is a little floppy symbol at the left side of the name, indicating that the drawing hasn't been saved yet.

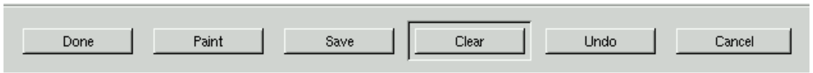
If you want to **rename** your drawing, just `double-click` its name. If you `right-click` at a name, you'll get a menu where you can **Save**, **Save as**, **Copy** and **Edit** the drawing.

To **Edit** a drawing, choose the name and either `right-click` and select **edit** or press the Edit button. If you got some Gfig drawings from the Internet or from a friend, you can copy them to the gfig directory (`~/.gimp/gfig`), press **Rescan** and they will appear in the name list. If you want to load a drawing outside of your Gfig directory, press **Load**. This brings up an ordinary open file dialog where you can choose a Gfig drawing. You can browse your drawings by highlighting them, and look at them in the **Prev** window. **Delete** pops up a dialog asking if you want to delete the drawing (both the file and the name in the browser).

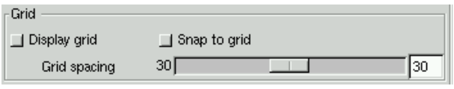
The possibility to **merge** Gfig drawings is a very nice option. To do so, select a drawing and press **Edit**, then highlight the drawing that you want it to be merged with, and press **Merge**. The selected drawing will now merge with the highlighted drawing. Only the Edit drawing will be altered - the highlighted drawing will not be changed.

Command bar

There's also a command bar at the bottom of Gfig. **Done** is for when you want to exit Gfig after you're done with it. **Paint** executes your command, i.e. paints your Gfig drawing into a Gimp image according to your settings. **Save** saves a gfig drawing. **Clear** will erase everything in the drawing area. **Undo** will undo your last operation. **Cancel** lets you exit Gfig, and drop all that you have done.



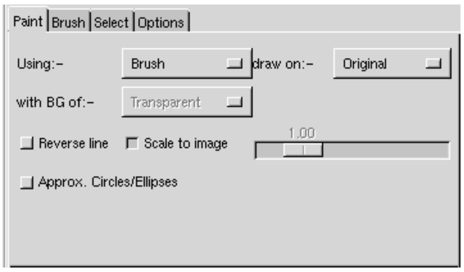
Grid



Controls the **support grids** in the Gfig drawing. If you have selected **Grid Type**: Rectangle in the **Option** tab-folder, you can set the grid **size** in X*X pixels with the Grid spacing slide. You can also choose to *Display* the grid, and to make your objects *Snap to* the grid. **Snap to grid** is an excellent tool if you work with precision drawings. If you move objects with Snap, the anchor point you drag at will snap to the grid. If you move all your objects with Snap enabled, an invisible middle point will be calculated and that point will snap to the grid system.

THE TAB FOLDERS

Paint



Using:– Brush
draw on:– Original
with BG of:– Transparent
☐ Reverse line
☑ Scale to image 1.00
☐ Approx. Circles/Ellipses

The **Using** menu controls where paint is, or can be applied. Use **Brush** to paint the outline of the drawing, **Selection** if you want to transform the drawing to a selection, or **Selection + Fill** if you want to make a selection and then fill it with a color or pattern (see Select tab).

Draw on determines where the drawing is placed. **Original** puts the drawing on the original image. **New** creates a new layer for the drawing, and **Multiple** creates a new layer for each object in your drawing. If you select New or Multiple, you can also set what type of **Background** you want for the new layer. There are four options: **Transparent**, **Background** (fills with the current background color in the toolbox), **White** and **Copy** (copies the original image to the new layer), if you choose Multiple, the previous layer will be copied to the new layer.

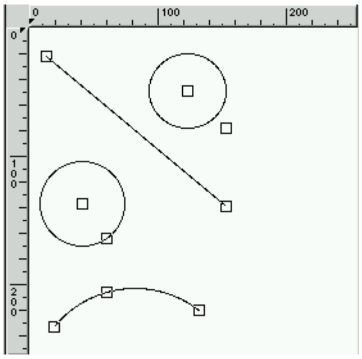
Example:

- Input image `julius.tif` and a Gfig drawing with 3 objects. **Draw on Multiple** with **Bg Transparent**: This setting results in: Background Julius, layer 0 the first draw object, layer 1 the second draw object, layer 2 the third draw object.
- Input image `julius.tif` and a Gfig drawing with 3 objects. **Draw on Multiple** with **Bg Copy**: This setting results in: Background Julius, Layer 0 Julius plus object 1, Layer 1 Julius plus object 1 and 2, Layer 2 Julius plus object 1,2 and 3.

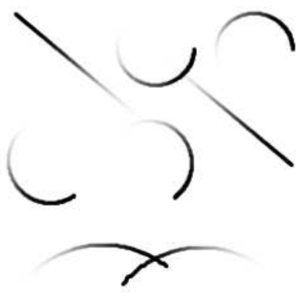
This obviously fits like a glove when you want to make GIF animations! (see chapter 20)

Reverse line: With this option you can control the direction of brush strokes. Normally, the *rendering* of your stroke goes from the first control point to the last one. If you set **brush fade out** to 20 in the brush tab folder, the stroke will fade out after 20 pixels (counted from the first control point). If you check *Reverse line* it will fade out from the last control point instead of the first. This option works with lines, curves and polygons. When it come to polygons, fading goes clockwise from the *first* control point if Reverse line is unchecked.

Approx. Circles/Ellipse: If you check this button, fading also applies to circles and ellipses. The fading goes clockwise from the *first* control point.


0
100
200
0
100
200

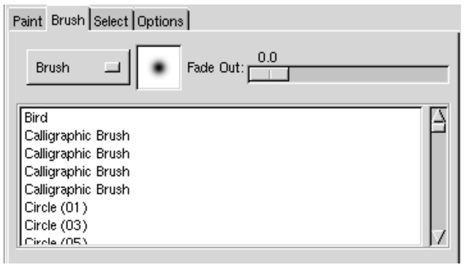
Original Figures



Drawn with Reverse line checked and unchecked

Scale to image: The drawings can be set to fit the image, or just get bigger or smaller. Just uncheck and drag the slider to *scale up/down*. This is nice option that makes it easy to **magnify** your drawings and work with small details. When you're done, check **Scale to image** to *zoom out*.

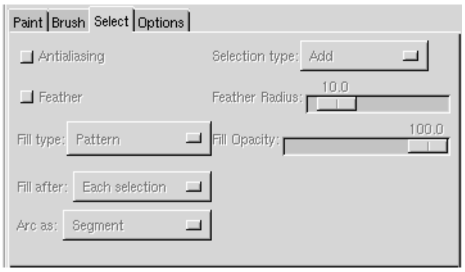
Brush


Paint
Brush
Select
Options
Brush
Fade Out:
0.0
Bird
Calligraphic Brush
Calligraphic Brush
Calligraphic Brush
Calligraphic Brush
Circle (01)
Circle (03)

In this tab folder you choose a suitable **paint brush**. A tip is to bring up the **Select brush dialog** so you can set the spacing and opacity of the brush, otherwise Gfig will use the default values of the brush you select. To select a brush, highlight it, and it will be displayed in the preview.

You can choose from four different types of brushes. **Brush** is an ordinary brush, with a fade out control which controls where the stroke will start to fade out (if fade out is set to 0.0, there will be no fading). **Airbrush** is like a spray can with pressure control, **Penci**l is like a brush with hard edges, and **Pattern** allows you to paint with a pattern (much like the Clone Tool does).

Select



☐ Antialiasing
Selection type: Add
☐ Feather
Feather Radius: 10.0
Fill type: Pattern
Fill Opacity: 100.0
Fill after: Each selection
Arc as: Segment

This folder is only enabled when painting with **Select** or **Select + Fill**. In order to understand the different Selection types, you have to have a fairly good grasp of basic selection (see chapter 6).

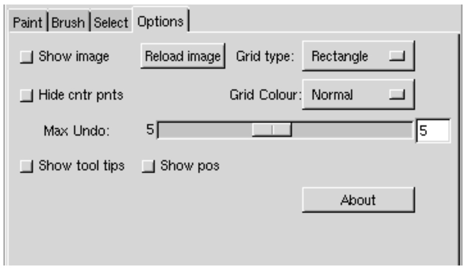
An example: Suppose that you have made two (overlapping) star objects, and you have selected **Paint with Select** in the Paint folder. If you now choose **Add** as Selection type, you'll get a selection that is a *combination* of the two objects (a merged double star).

If you choose **Sub**, you'll get nothing, (because there is no selection to subtract from) but if there had been a previous selection, the shape of the twin stars would be **cut** from this selection. If you choose **Replace**, the previous selection will be **replaced** with the star that you painted last in the drawing area. **Intersect** means that the intersection of the two star objects will become a selection.

You can fill the selection with a **pattern**, **foreground** or **background** color (Fill Opacity controls the transparency of the fill). You must bring up the **Pattern dialog** to set a pattern, otherwise the last or default pattern will be used. The **Fill after** button controls the fill order; e.g. if you choose **All selections** and **Replace** as selection type, only the last drawing object will be filled. If you had chosen **Each selection** then all objects would get filled.

Feather controls the gradual transparency or edge softness of the selection. **Arc as** determine how curved objects will be treated. If you set this parameter to **Sector**, curves will be treated as circle sectors (pie slices), and if you set it to **Segment** they are filled as a circle segments (half-moons). The last option is antialiasing, which is generally good to enable.

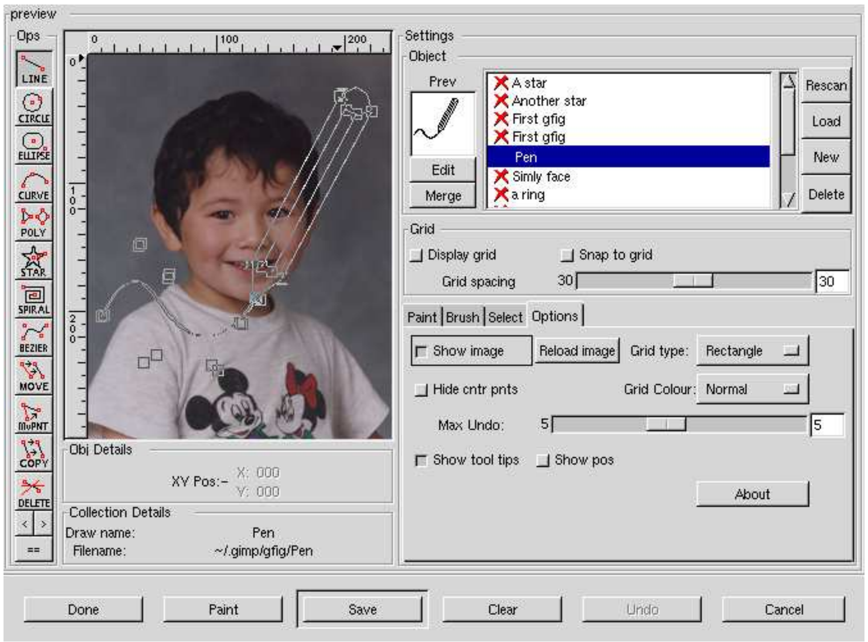
Options



☐ Show image
Grid type: Rectangle
☐ Hide cntr pnts
Grid Colour: Normal
Max Undo: 5
☐ Show tool tips
☐ Show pos

This folder contains various options, like the possibility to make an image appear in the **preview** drawing area, or to **reload** it after applying a drawing. You can also **hide** the centre points in drawing objects, you can get **tool tips**, set a different level of **undo**, or change the color and shape of the **grid** system. You can also choose to show the X/Y position of the mouse in the **Obj Details** window. If you want to achieve a high level of precision, this option is very useful.

.


preview
Ops
LINE
CIRCLE
ELLIPSE
CURVE
POLY
STAR
SPIRAL
BEZIER
MOVE
MvPNT
COPY
DELETE
0
100
200
Settings
Object
Prev
Edit
Merge
A star
Another star
First gfig
First gfig
Pen
Simly face
a ring
Rescan
Load
New
Delete
Grid
Display grid
Snap to grid
Paint
Brush
Select
Options
Reload image
About
Obj Details
Collection Details
Done
Paint
Save
Clear
Undo
Cancel

Grid spacing 30

☒ Show image

Grid type: Rectangle

☐ Hide cntr pnts

Grid Colour: Normal

Max Undo: 5

☒ Show tool tips ☐ Show pos

XY Pos:– X: 000 Y: 000

Draw name: Pen

Filename: ~/.gimp/gfig/Pen

EXAMPLE

In order to understand this nice plug-in, we will give you an example:

How to make a star with sparkles around it and a gradient fill:

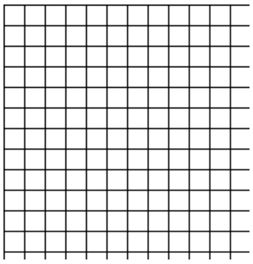
- 1: Bring up a new image, invoke **Gfig** and choose **New Object**.
- 2: Choose the **star** object, set it to seven spikes and draw it.
- 3: Choose **Paint with Brush** and a Pattern/Small Galaxy brush. Bring up the **Pattern dialog** and choose Ice as pattern (you may have to refresh the pattern in the brush tab folder, to do so choose Pencil and then Pattern again).
- 4: Press **Paint** to paint the sparkling outline of your star.
- 5: Now switch to **Paint with Selection**:
- 6: Press **Paint**, which will produce a star shaped **selection**
- 7: Now, bring up the **Gradient editor** and choose Cold_Steel as gradient
- 8: `Double-click` on the **Blend tool** (in the Toolbox) and set Blend to Custom and Gradient to Conical (asymmetric) in the dialog

- 9: Set the Blend tool in the centre of the star and blend.
- 10: You will now have a frosty star.

The same as above but with a pattern fill:

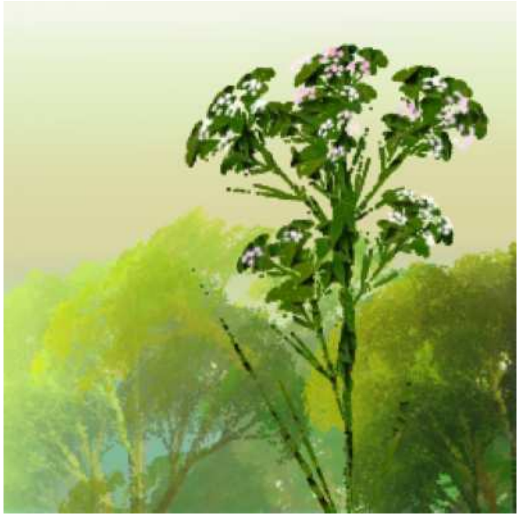
- 1: Repeat step 1 to 4
- 2: Now switch to **Paint** with **Selection+Fill** and press paint
- 3: You will now have a frosty star.

GRID



With this filter you can easily create a **grid** system. **Size** obviously controls the size of the grids in pixels. **Offset** controls where (in pixels) the first unbroken square will be drawn.

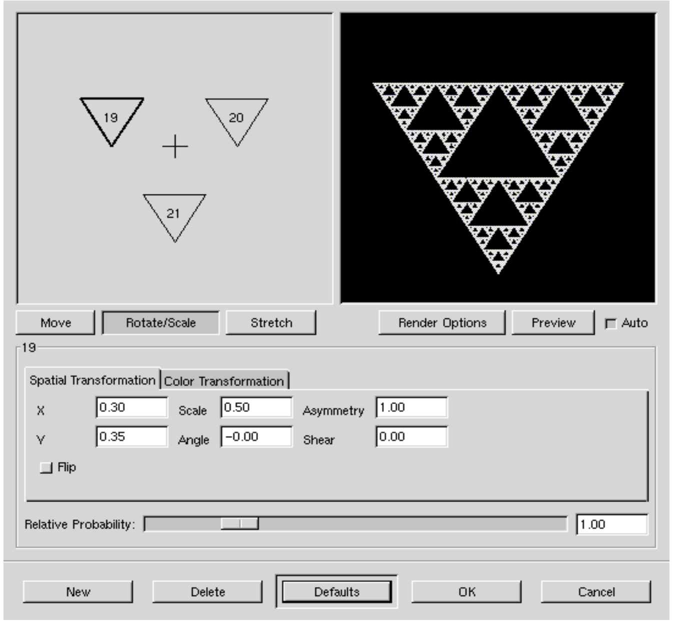
IFS COMPOSE



This **Fractal** based plug-in is truly wonderful! With this versatile instrument, you can create amazingly naturalistic organic shapes, like leaves, flowers, branches, or even whole trees.

Usage

The key to using this plug-in lies in making very *small* and *precise* movements in fractal space. The outcome is always hard to predict, and you have to be extremely gentle when you change the pattern. If you make a fractal triangle too big, or if you move it too far (even ever so slightly) the preview screen will black out, or more commonly, you'll get stuck with a big shapeless particle cloud. A word of advice: When you have found a pattern you want to work with, make only small changes, and stick to variations of that pattern. It's all too easy to lose a good thing



X 0.30 Scale 0.50 Asymmetry 1.00

Y 0.35 Angle -0.00 Shear 0.00

☐ Flip

Relative Probability: 1.00

The plug-in interface consists of the **compose area** to the left, a **preview** screen to the right, and some tab folders and option buttons at the bottom of the dialog.

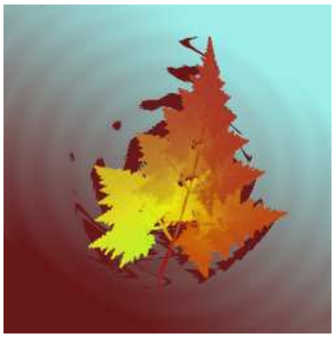
Settings

The **Default** setting (to the right) are three equilateral triangles. To create a new pattern, you **Move**, **Rotate**, **Scale** and **Stretch** these triangles. You either use the three buttons under the compose area, or press the right mouse button to access a **popup menu**.

Contrary to what you might believe, it's really much easier to create a leaf or a tree with Ifs Compose, than to make a defined geometrical pattern (where you actually know what you're doing, and end up with the pattern you had in mind).

HOW TO USE IFS COMPOSE

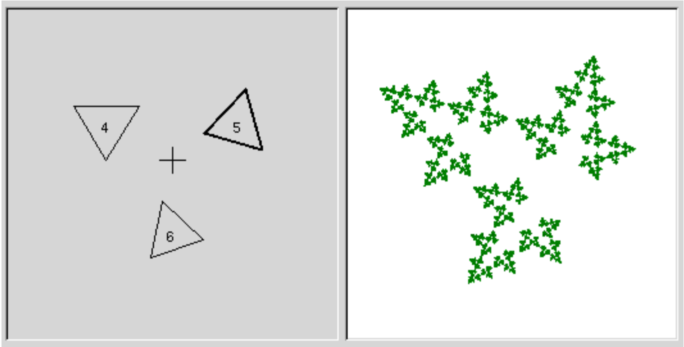
This is a rather complex plug-in, so to help you understand it, you'll be guided through an example:



Making a leaf or a branch:

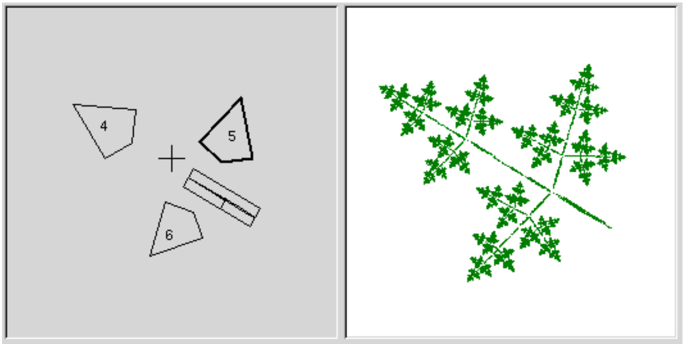
Many forms of life, and especially plants, are built like mathematical **fractals**, i.e a shape that reproduces or repeats itself indefinitely into the smallest detail. You can easily reproduce the shape of a leaf or a branch, by using four (or more) fractals. Three fractals make up the tip and sides of the leaf, and the fourth represents the stem.

Before opening Ifs Compose: Make a **New image**, and create a **transparent layer**. Set the FG color in the toolbox to a nice green, and set BG to white.

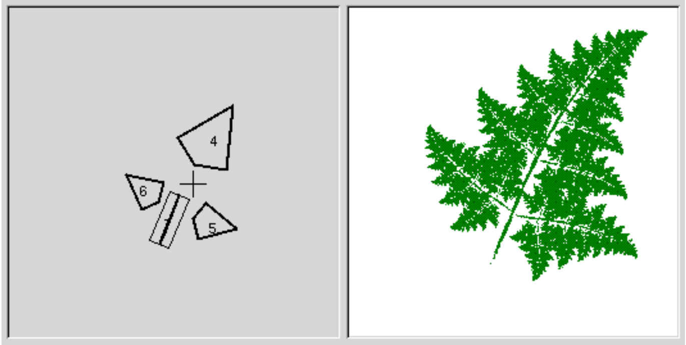


Start by **rotating** the right and bottom triangles, so that they point *upwards*. You'll now be able to see the outline of what's going to be *tip* and *sides* of the leaf.

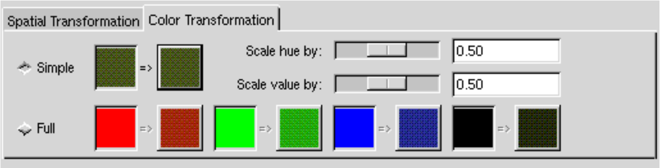
To make the leaf **symmetrical**, adjust the *bottom* triangle to point slightly to the *left*, and the *right* triangle to point slightly to the *right*.



Press **New** to **add** a fractal to the composition. This is going to be the stem of the leaf, so we need to make it long and thin. Press **Stretch**, and *drag* to stretch the new triangle. Don't be alarmed if this messes up the image, just use **Scale** to adjust the size of the overlong triangle. You'll probably also have to move and rotate the new fractal to make it look convincing.

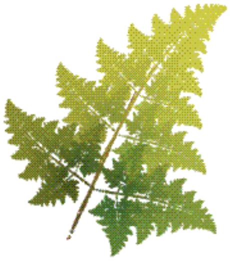


You still have to make it look more leaf-like. *Increase* the **size** of the top triangle, until you think it's thick and leafy enough. *Adjust* all fractals until you're happy with the shape. `Right-click` to get the **popup** menu, and choose **Select all**. Now all fractals are selected, and you can scale and rotate the entire leaf.



Scale hue by: 0.50
Scale value by: 0.50

The final step is to adjust **color**. Press the **Color Transformation** tabfolder, and choose a different color for each fractal. To do this, check **Simple** and press the *right* color square. A color circle appears, where you can click or select to choose a color.



Press **OK** to apply the image, and voila, you've just made a perfect fractal leaf! Now when you've got the hang of it, you'll just have to experiment and make your own designs. All plant imitating fractals (be it oak trees, ferns or straws) are more or less made in this fashion, which is *leaves* around a *stem* (or several stems). You just have to twist another way, stretch and turn a little or add a few more fractals, to get a totally different plant.

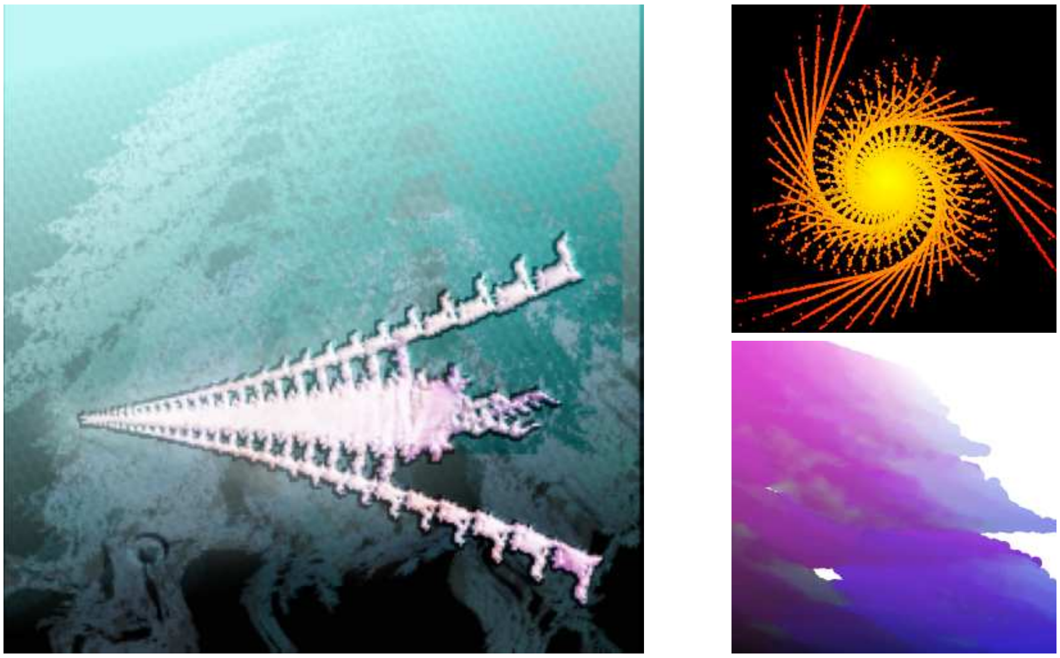
MAIN OPTIONS

- **Relative Probability**: Determines influence or total impact of a certain fractal.
- **Spatial Transformation** gives you information on the active fractal, and allows you to type a value instead of changing it manually. Changing parameters with the mouse isn't very accurate, so this is a useful option when you need to be exact.
- **Simple color transformation** changes the current fractal color (which is the FG color in the toolbox) to a color of your choice.
- **Scale Hue/Value** When you have many fractals with different colors, the colors *bleed* into each other. So even if you set "pure red" for a fractal, it might actually be quite blue in some places, while another "red" fractal might have a lot of yellow in it. Scale Hue/Value changes the color strength of the active fractal, or how influential that fractal's color should be.
- **Full color transformation** is used to change color influence from the *other (inactive) fractals*. You can for example in the red fractal set Full transformation to red in all of the swatches, and you won't get any influence at all in that part of the fractal composition.

RENDER OPTIONS:

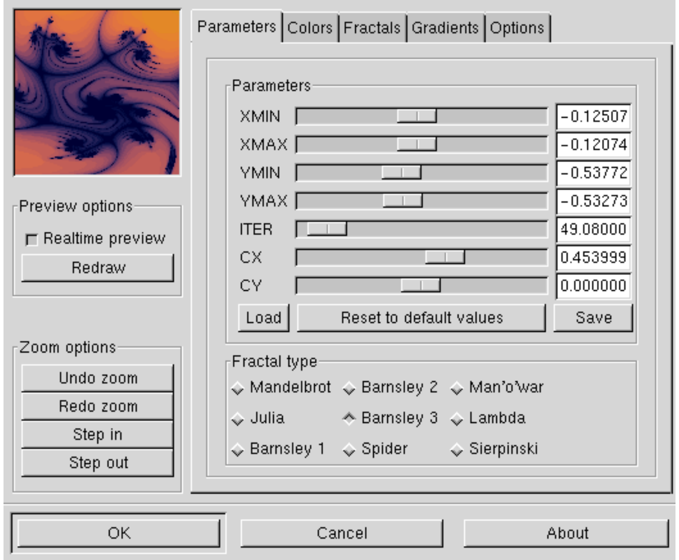
- **Spot Radius** determines the density of the ”brushstrokes” in rendered image. A low spot radius is good for thin particle clouds or spray, while a high spot radius produces thick, solid color strokes much like watercolor painting (see pic. bottom/right). Be careful with using too much spot radius - it takes a lot of time to render.
- **Subdivide** controls the level of detail, and **Iterations** determine how many times the fractal will repeat itself (a high value for subdivide and iterations is for obvious reasons a waste of process time unless your image is very large).
- **Max. Memory** enables you to speed up rendering time. This is especially useful when working with a large spot radius, just remember to use even multiples of the default value: 4 096, 8192, 16 384...

.



There is no end to what you can create with this plug-in and a little imagination

FRACTAL EXPLORER



Parameters
Colors
Fractals
Gradients
Options
Preview options
Redraw
Zoom options
Undo zoom
Redo zoom
Step in
Step out
Load
Reset to default values
Save
OK
Cancel
About

☐ Realtime preview

Parameters

XMIN -0.12507
XMAX -0.12074
YMIN -0.53772
YMAX -0.53273
ITER 49.08000
CX 0.453999
CY 0.000000

Fractal type

○ Mandelbrot ○ Barnsley 2 ○ Man'o'war
○ Julia ● Barnsley 3 ○ Lambda
○ Barnsley 1 ○ Spider ○ Sierpinski

We will try to explain this *fractal generator* in a non-mathematical way, so that people who are unfamiliar with advanced mathematics will understand what it does. These fractal images can be used as **patterns**, **textures** etc. or simply as interesting images.

The interface is divided into several tab folders.

- **Parameters** is where you select the fractal type and its parameters. In this description we will focus on the *Mandelbrot* fractal.
- **Color** is where you set color *functions* and *parameters*.
- **Fractal**, here you can select a custom fractal with *predefined* settings.
- **Gradient** is an option for mapping a gradient over your fractal. **Options** allows you to specify the interface language: English, German or French.

The **preview window** allows you to **zoom** an interesting part of the fractal. To zoom in, simply choose an area with the hair cross mouse cursor (as with the toolbox Zoom tool) or use the button **Step in**. To zoom out, press **Step out**. You can also **undo** and **redo** zooms.

PARAMETERS

Min and Xmax

Xmin controls how much the fractal will stretch to the left, and Xmax controls the stretch to the right.

Ymin and Ymax:

Ymin controls how much the fractal will stretch *upwards* (positive Y values in a coordinate system) and Ymax controls how much it will stretch *downwards* (negative Y values).

ITER

stands for *iterations*, and controls the level of *detail* in the fractal. It's not always best to have a high level of detail. Often a fractal will get more interesting when created with lower values. We have found that in the range between 16 and 70 you will find some very nice fractal patterns.

CX and CY

CX and CY have no impact on Mandelbrot and Sierpinski fractals. CX values between -5 and 0 control the *horizontal* distance of the two fractal parts. Low values (-5) takes them apart totally, and high values (0) *merges* them totally. Values between 0 and +5 will do the same, but in a *vertical* direction. CY has a similar control mechanism, but not in a horizontal/vertical direction. CY works in a 45 /225 and135/315 deg direction.

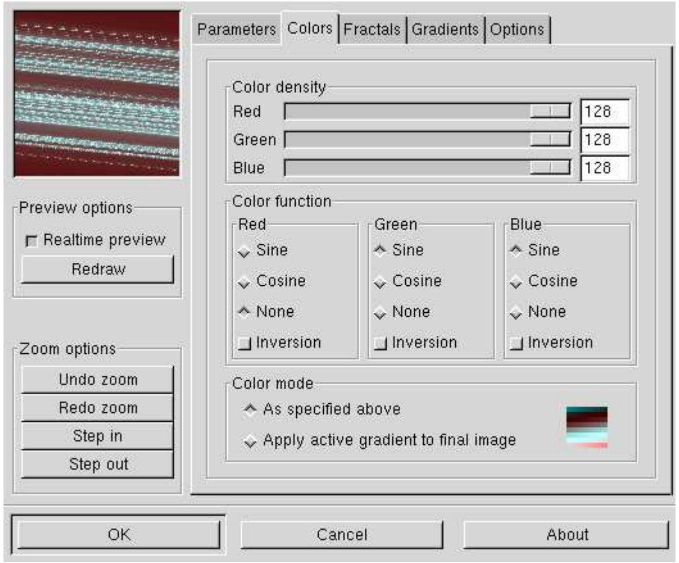
Load, Reset and Save

These buttons lets you load a specific setting from your personal fractal directory, or save a nice setting that you have done. **Reset** will naturally reset the values to the default values.

Fractal type;

Here can you choose between well known fractal types like *Mandelbrot*, *Julia* or *Sierpinsky* but you'll also find more exotic fractal types like *Man´o war* or *Spider*.

COLORS



☐ Realtime preview

Color density

Red 128

Green 128

Blue 128

Color function

Red	Green	Blue
○ Sine	● Sine	● Sine
○ Cosine	○ Cosine	○ Cosine
● None	○ None	○ None
☐ Inversion	☐ Inversion	☐ Inversion

Color mode

● As specified above

○ Apply active gradient to final image

A fractal consists of *three* parts. An **outer** part, an **inner** part and the "**fractal**" border between them.

Color Function

The **Sine** function controls the color of the inner and outer parts, while the **Cosine** function controls the fractal part. **None** means that you'll use linear instead of trigonometrical mapping for a certain color channel.

You can by checking and unchecking the three color functions choose suitable colors for the different parts of your fractal.

The **Inversion** button will invert low color values to high color values and vice versa. E.g checking Inversion in the green channel will make areas previously low on green, blaze with strong green color. To fully understand it, take a quick look at the **Channels** tab in the **Layers & Channels** dialog. Deselect all channels but the green channel, apply the `Image/Colors/Invert` command, and see what happens in the image and in the thumbnail channel representation.

Color Density

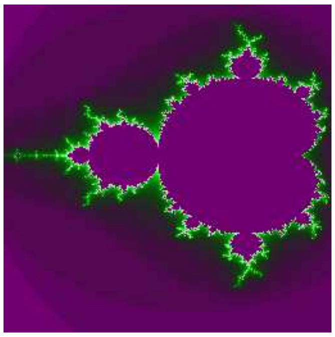
You can also set the *intensity* of the color channels with the slides in **Color density**. This makes it possible to set any color you want to your fractal.

Color mode

Either you use the color mode you have specified in Color options, which is what you see in the preview window, or you use a **gradient** from the **gradient folder**. This option will map the gradient over your fractal. You can achieve quite fantastic looking fractals this way.

GRADIENTS

Here you specify the gradient to use in the **colors** tab folder, a tip is to bring up the **gradient editor** so you can take a look at the gradients.

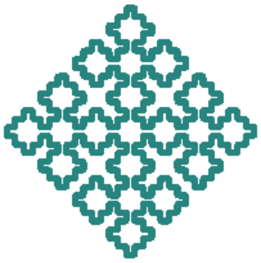


FRACTALS

Here you'll find some nice predefined fractals that you can use straight off, or use as a backbone or point of departure for your own experiments.

L-SYSTEM

documentation by Michal Gomulinski



Lindenmayer Systems, called L-Systems, is a mathematical formalism designed primarily for concise description of natural shapes of plants. You can find out more about history and roots of L-Systems in the publications mentioned at the end of this description.

The whole concept of L-Systems consists of two main parts:

- Preparing the L-Systems description
- Applying this formula to render a graphic representation.

During the preparation stage, the program iteratively applies a set of rewriting rules onto a string. The process starts with an initial string (called axiom) and for every iteration this string grows longer. The preparation is finished when the given number of iterations have been completed.

A SIMPLE EXAMPLE

To clarify this description, let's take a look at a simple example:

```
Axiom = A
Rules = A -> BA and
        B -> A
```

What does it mean?

It means that we start with a **string** of **length 1** which contains the letter **A**. For each iteration, we replace every **A** with a string **BA** and every **B** with string **A**. This is how it goes:

```
Iteration 0   A
Iteration 1   BA
Iteration 2   ABA
Iteration 3   BAABA
Iteration 4   ABABAABA
...
```

Note that we can use many different letters in both the axiom and the rules. If a string should contain a letter for which no rule has been defined, it will be ignored and passed to the resulting string. An example here below:

```
Axiom = A
Rules = A -> BA
        B -> AC

Iteration 0  A
Iteration 1  BA
Iteration 2  ACBA
Iteration 3  BACACBA
```

As you can see, by using a *small* and *concise* description we can obtain quite complex and long character strings. This is the first important lesson about L-Systems. You need only define the **seed** and **rules** which govern growth to generate large and complicated, yet interesting data.

GRAPHIC REPRESENTATION

- To **convert** the string generated during the *first phase* of the process into its **graphical representation**, we must come to an agreement on some assumptions. You can compare the **rendering** process to an imaginary *turtle* that draws a line while wandering over our image. What the turtle does is to ”*eat*” consecutive characters, and then it decides what to do next based on that information. Any character that has no meaning for the turtle is simply ignored. The following three characters have **fixed meanings**.
- The **F** character means **go forward** one step, drawing a line.
- The **f** character means **jump forward** one step.
- The **(and)** characters will cause our turtle to save its state (position, heading, parameters of its drawing tool), decrease the step length by a given factor and to draw a sub-system until it encounters a matching closing brace.
- Traditionally, the characters **+** and **-** mean turn left and turn right, but in my plug-in they are defined as so called **special rules**. Users are encouraged to keep the original meaning of these two signs for the sake of readability.

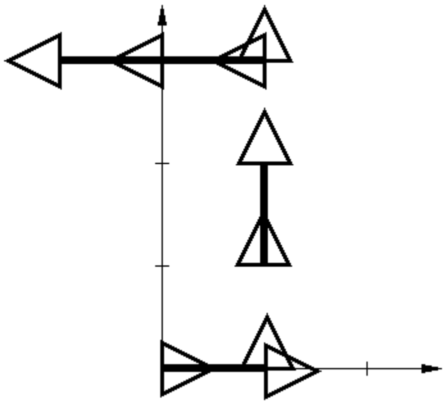
As an illustration for the render process, walk through the following example:

String to be rendered: **F+fFf+AFAF**. Let’s assume that the initial position of the turtle is (0,0), initial heading is 0 degrees *(the turtle is ”looking” to the right)*, step length is 1 and the turn angle associated with the + and - characters is equal to 90 degrees.

Interpreted character	Position (x, y)	Heading	Result
	0, 0	0	

`F`	`1, 0`	`0`	`a line from (0,0) to (1,1)`
`+`	`1, 0`	`90`	
`f`	`1, 1`	`90`	
`F`	`1, 2`	`90`	`a line from (1,1) to (1,2)`
`f`	`1, 3`	`90`	
`+`	`1, 3`	`180`	
`A`	`1, 3`	`180`	`(character ignored)`
`F`	`0, 3`	`180`	`a line from (1,3) to (0,3)`
`A`	`0, 3`	`180`	`(character ignored)`
`F`	`-1, 3`	`180`	`a line from (0,3) to (-1, 3)`

Also have a look at the arrow picture below. It represents the turtle movements as they occur during interpretation of the simple string from the example above.



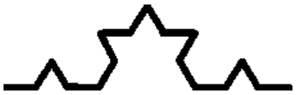
`F+fFf+AFAF`

It's when you **combine** the possibilities offered by a concise description of long strings, by **rewriting** the rules with a drawing turtle that lets the characters describe its way, that you get a really powerful tool.

This idea is called L-Systems.

Take a look at the final example before we move on to some implementation details. This is **Koch's** curve. Below, you have its definition as an L-System. First three iterations, and a picture drawn on the basis of the third string.

```
Axiom = F
Rules = F -> F-F++F-F

Iteration 0   F
Iteration 1   F-F++F-F
Iteration 2   F-F++F-F-F-F++F-F++F-F++F-F-F-F++F-F
...
```



As you can tell from the picture, the angle added to the current turtle heading by + and - characters equals 60 degrees.

USING THE L-SYSTEMS PLUG-IN

What can you do with the L-Systems Plug-in?

- You can create a new L-System, give it a unique name and a unique file name in which the definition will be stored. You can also use an existing L-System as a base of a newly defined one, but be sure not to forget to change the file name or you will lose either the new or the old one! All this is done within the top-left part of the plug-in's GUI. There is a list with all defined L-Systems and a set of buttons to perform various operations on them.
- You can define your L-System by entering an **axiom**, a set of **rules** and the number of **iterations** which should be performed before rendering. Use the first page of the notebook at the bottom of the GUI. Adding rules is as simple as pressing the **Add** button and filling in the pop-up dialog window. The same goes for **editing** and **removing** rules.
- You can define a set of **additional** rules applied once only after the last iteration. You'll learn more about such rules later. To manipulate the set of last rules you should switch to the second page of the notebook widget.
- Once you've defined the L-System, you have to move on to the *third page of the notebook* at the bottom of the GUI, and specify the correct information about at least two *special symbols*: **+** and **-**. This is because the special rules for these characters are always included in the set of special rules. The only thing you will probably have to do is to provide correct information about the turn angles.
- Here I should explain something. What does these *magic* special rules entries mean? That's quite simple, the rule:
  ```
  + -> 60.00:       <<CURRENT>>:   -1: -1%:(NO CHANGE)
  ```

means exactly the following: The **plus** character will cause the turtle to increase its heading angle by 60 degrees, not to change its brush (i.e. keep the CURRENT one), not to change brush spacing (the first **-1**, sorry about these meaningless expression :), not to change the opacity either (the second **-1**) and not to change the drawing color.

- In the end you can change various rendering parameters like **initial position** and **heading**, **step length**, **turn** and **step** randomness and others.

ADVANCED FEATURES

Using Braces

As I explained before, you can introduce **matched braces** in both the **axiom** and in all **rules**. They are significant during the rendering process because they cause the turtle to draw part of the L-System in a separate context. The most common use of braced L-Systems is to draw plant-like shapes. You use braces to denote that the turtle after drawing for example *a branch* should return to the position where the branch started and continue drawing the *main trunk*. You can change **brush shapes**, **colors**, **spacing** and so on while drawing the branch, but all of these changes will have no effect on the visualization of the trunk and - possibly - the consecutive branches.

As an example, look carefully at the definition of L-Systems with names that start with `Plant/`

The Purpose of Last Rules

An additional set of rules applied once, just before rendering, serves several purposes. First, it can be used to translate abstract objects (with no turtle meaning) used in main rules into something visible. You often do this when designing plants or flowers. In this case, every character in main rules and axiom would have a specific meaning, like root, trunk, leaf and so on. Every such abstract symbol should be transformed into something visible with a given color, brush shape, length.

Secondly, you can use last rules to modify the generated shape to make it look more interesting. This is quite common when the L-System looks like a kind of a grid. It would be more interesting if you could see the way the turtle has drawn the shape but that's impossible since many lines are connected in a single point. To avoid that, one should define a set of three simple last rules, which will make turns less sharp:

```
+ -> +F+

- -> -F-

F -> FF
```

Be sure to adjust the definition of + and - in the special rules section. They must use half the angle they had before the introduction of the last rules. Probably, also the length of a single step will have to be cut in half.

Effective Use of Special Rules

Last rules are not only turns! You can use them quite extensively to make your L-Systems look more lively and realistic.

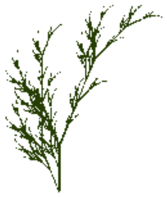
- First comes **brushes** which will provide you with various textures and line styles.
- Second is **color**. What the purpose is of using color in the image you are painting is something you probably know better than me!
- Third is **opacity**. It works great in plants, where for example leaves can be made somewhat translucent to make the picture more interesting. Painting with a semi-opaque paint can also produce nice effects when drawing patterns with intersecting lines. The final color of various parts of the image will then depend on how many times the pixel has been painted - and with what color!
- Fourth and the last is **spacing** which probably has only one use: to make the program draw just a single *print* of a brush instead of an awful *line* painted with a brush which was not designed for it.

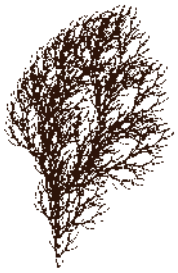
Where to Look for More Information

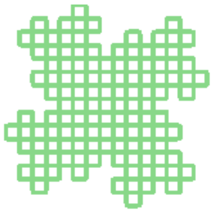
If you are interested in the concept of L-Systems look for the following books/articles. For more references look in Prusinkiewicz's book.

- P. Prusinkiewicz, J. Hanan Lindenmayer Systems, Fractals and Plants in series ”Lecture Notes in Biomathematics”, Springer Verlag 1989
- A. R. Smith ”Plants, Fractals and Formal Languages” in ACM Computer Graphics vol. 18, no 3, July 1984

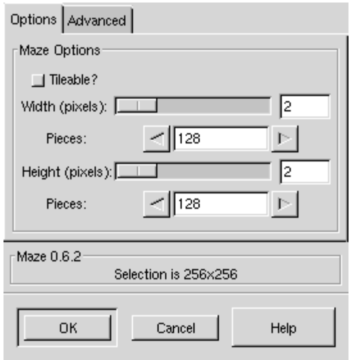
Examples







MAZE



☐ Tileable?

Width (pixels): 2

Pieces: 128

Height (pixels): 2

Pieces: 128

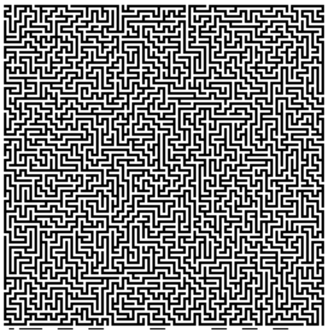
Maze 0.6.2

Selection is 256x256

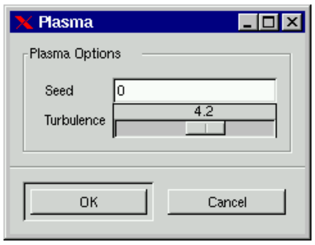
Generates a **maze**. If you want to use it in a pattern, you can get the maze **tileable** if you check that button.

Width and **height** controls how many pathways the maze should have. The *lower* values for width and height, the *more* paths you will get. The same happens if you *increase* the number of **pieces**.

You can specify random **Seed** or use the **time** as seed. Using random seed means that the maze pattern will be different each time you invoke the filter.



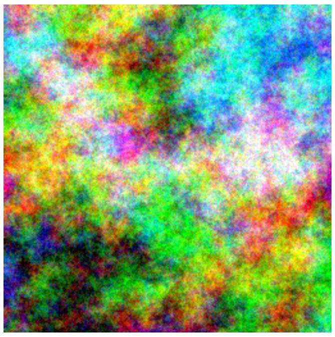
PLASMA



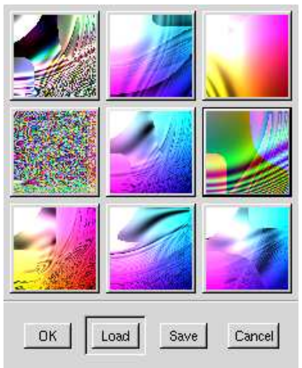
Seed: 0

Turbulence: 4.2

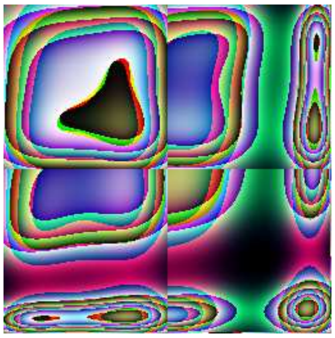
Plasma generates colorful *clouds*, which can be used for textures etc... You can control the **turbulence** in the plasma cloud with the **Turbulence** slide. High values give a hard and cold feeling to the cloud, low values produce a softer and warmer cloud. You can also generate **Seed** for random variations in the plasma cloud.



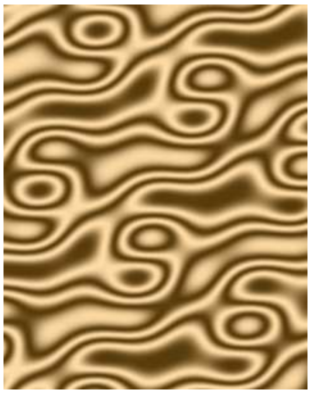
QBIST



This filter generates *random* textures, much like **KPT Texture Explorer**. The original texture is displayed in the middle square, and different variations surround it. If you like one of the alternative textures, click on it. The chosen texture now turns up in the middle, and variations on that specific theme are displayed around it. When you have found the texture you want, click on it and then click OK. The texture will now appear on your drawable.You can **save** and **load** your textures. This is quite handy since it's almost impossible to recreate a good pattern by just clicking around.



SINUS

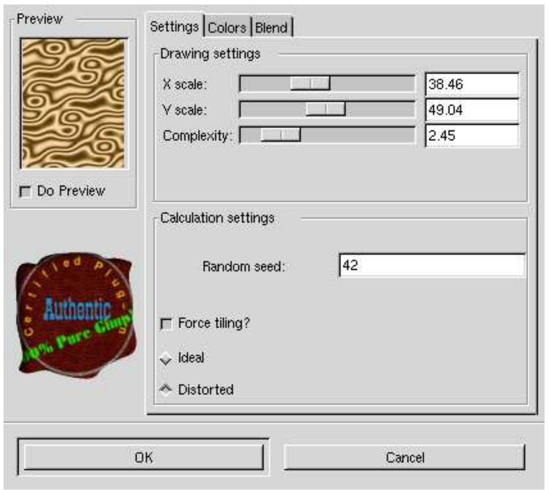


Lets you make **sinusoidally** based *textures*, which looks rather like watered silk or plywood. This plug-in works by using two different colors that you can define in the **color tab folder**. These two colors then create wave patterns based on a sine function.

You can set the **X** and **Y scale** which determine how *stretched* or *packed* the texture will be. You can also set the **complexity** of the function, a high value creates more interference or repetition in the pattern. You can also specify a random **seed** for greater variation in the pattern.

FUNCTIONS

Let's discuss the functions in more detail.



Drawing settings

X scale: 38.46

Y scale: 49.04

Complexity: 2.45

Calculation settings

Random seed: 42

☐ Force tiling?

○ Ideal

● Distorted

X/Y scale

A low X/Y value will maximize the horizontal/vertical stretch of the texture, while a high value will compress it.

Complexity and Random Seed

Complexity controls how the two colors interact with each other (the amount of interplay or repetition). Random seed increases variation in the pattern.

Force tiling

If you want to use the texture as a tiled background in a webpage, this option creates better looking tiles (though not quite seamless). Tip: To get a perfect result; flip the tiles in the **Make Seamless** plug-in in the **Filters/Map** menu so that they mirror each other.

Ideal/Distorted

This option gives additional control of the interaction between the two colors. Distorted creates a more distorted interference between the two colors than Ideal.

THE COLOR TAB FOLDER

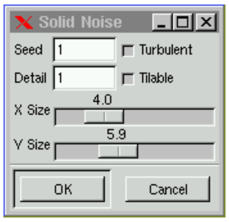
Here you set the two colors that make up your texture. You can use **Black and white** or the **foreground/ background** colors in the toolbox, or you can choose a color with the **color icons**.

The **Alpha channel** controls the transparency of your texture, just remember that you need an alpha enabled background or a layer.

THE BLEND TAB FOLDER

In this folder you can control the blend of the two colors. You can choose between three functions to set the blend - **Linear**, **Bilinear** and **Sinusoidal**. The **Exponent** controls which color is dominant. If you set the exponent to -7.5 the *left* color will dominate totally, and if you set it to + 7,5 it will be the other way around, a zero value is neutral.

SOLID NOISE



Seed 1 ☐ Turbulent

Detail 1 ☐ Tilable

X Size 4.0

Y Size 5.9

This is a great texture maker. Note that this noise is always gray, even if you applied it to a very colorful image (it doesn't matter what the original image looks like - this filter doesn't use background information). This is also the filter you use for creating displacement maps for the Twist plug-in.

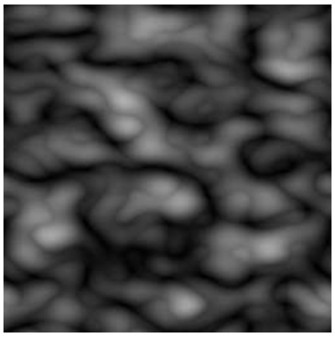
X and Y Size controls size and proportion of the noise shapes in X and Y direction.

Seed generates a random variation in your texture.

Detail controls the amount of detail in the noise texture, higher values gives higher level of detail, and the noise seems to be made of small particles, which makes it feel *hard*. A low value makes it more *soft* and *cloudy*. I

f you check **Turbulent**, you'll get very interesting effects, often something which looks much like oil on water (or living tissue).

If you check **Tileable**, you'll get a noise which can be used as tiles, e.g you can use it as a background in a HTML page, and the tile edges will be invisible (seamless).



part VII

Animations

- *How to use AnimFrames*

chapter

36

Advanced animation with Gimp or how to use AnimFrames

Gimp has many excellent tools for making animations. The far most advanced tool of these is Anim Frames. We will unleash the power of it in this chapter.

BASIC CONCEPT

You will find the **AnimFrames** tool under the Image menu **<image>/AnimFrames**. As the name indicates, AnimFrames works with image frames. A frame is an ordinary Gimp image saved with a special extension. When you make an ordinary GIF animation you create a multi-layered image, where the object that you want to animate moves a little bit for every new layer. If you want to use more than one animation object, you have to put all of the objects in the same layer. *(If you want to know more about ordinary Gimp animations read Chapter 20)* This makes it hard to create advanced animations like in an animated film. If you want to work with several objects which can move independently of each other, the ordinary Gimp animation ”tools” do not suffice. AnimFrames overcomes this problem in a wonderful way. Instead of using a single multi-layered image, AnimFrames works with several images, where each image can have several layers with different objects in them. This makes every object independent of the other animation objects (just like in a cartoon frame where several people move around independently, because each character was painted on a separate plastic sheet).

HOW TO CREATE AN ANIMATION WITH ANIMFRAMES

MAKING A FRAME

AnimFrames is built up of frames. The first thing you have to determine when you create an animation is the size of the background image. When you’ve made up your mind, you continue like you always do in Gimp: `File -> New`. To turn your image into a frame you must save it. Choose `Save as` and save it as animationname_0001.xcf (the special extension is _0001.xcf). Now it’s time to create the Frames. Choose `AnimFrames/Duplicate Frame`. This brings up a dialog asking you to specify how many duplicates you want, or how may frames you want for your animation (the slide only goes as far as to 50 frames, but you can easily change that by typing an optional number of frames in the field). The format must be XCF, but if you are low on disk space, you can choose a compressed XCF by adding .gz or .bz to the end of the frame name. To make this work, you must of course have either gzip or bzip2 installed on your system. Don’t worry about saving, all frames are stored on disk immediately without explicit save. The same happens when you work with other tools in AnimFrames, so only save when you exit AnimFrames.

HOW TO NAVIGATE OUR FRAMES

We have now created a few basic frames, say for example 10 frames. We are in frame number 1 (name_0001.xcf.) First we must make something clear, ***you must never open two frames in Gimp at the same time!*** If your animation is called My_anim, make sure you don’t open My_anim_0001.xcf and My_anim_0002.xcf at the same time. ***If you let this happen, things can get really bad and in the worst scenario destroy your animation and crash Gimp!*** This is not a good idea, so we suggest you use the special menu commands for navigating the frames instead. You can navigate in the following ways:

- `Goto First` which will bring you to frame My_anim_0001.xcf
- `Goto Next` -> My_anim_0002.xcf
- `Goto Prev` -> My_anim_0001.xcf
- `Goto last` -> My_anim_0010.xcf, and you can also try
- Goto Any which will bring up a dialog asking you which frame to go to.

The author of the plug-in suggests that you assign hotkeys to the ”Goto” commands, because you will move around a lot in your frames. You can choose any keybindings you like, but here are a few suggestions:

```
Goto First  Ctrl-Alt-1
Goto Prev   Alt-1
Goto Next   Alt-2
Goto Any    Alt-3
Goto Last   Ctrl-Alt-2
```

Read more about keybindings in chapter 2.

YOUR FIRST ANIMATION

Or how to make a **Move Path**. The main tool in AnimFrames is the Move Path tool. This tool allows you to move your animation object along a specified path. You need at least two things to make an animation:

- A set of frames, i.e your ten My_anim images
- An object to animate.

The animation object must be of the same type as the frame. You can’t mix indexed images with RGB images, but an RGB and an RGB Alpha will work just fine. The frame and the object layer don’t need to be of the same size.

MOVING ALONG

To make the **animation object**, create a new transparent image and paint a simple object in the centre of the image - a big, red dot for example. Open the first frame (My_anim_0001.xcf.), select the `Move Path` option in the AnimFrames menu, and the **Move Path** window will appear. This window is where you control the animation sequence. This simple example will help you get a basic understanding of how Move Path works. Later in this chapter, we’ll return for a more profound description of the ”Move Path” window.

- Set `SourceImage/Layer` menu to your animation object. Set the **Handle** menu to `Center`, the **Stepmode** menu to `Loop` and the **Mode** menu to `Normal`.

- `Mouse-click` somewhere in the preview window, press the `Next Point` button and click somewhere else in the preview window.
- Now, set the `Start Frame` to your first frame and the `End Frame` to your last frame, leave the Layerstack and Preview Frame as they are and press OK.

You have now created your first animation. The red dot will now move from the first click-point to the second, and it will use ten frames to complete the movement. In order to view the animation, you must select `Frames Flatten` and include all of your frames in the `Frame Range` dialog. The second step is to make your frames turn into a multi-layered image. Do this by selecting `Frames to Image` and choose all your frames. Now you can watch your animation with the **Animation Playback** filter. If you want to save it as a GIF animation, you will have to index your image first. We suggest that you stick to RGB during the whole animation process, and don't convert to Indexed until you're done. At the time of this writing you can only save your animations as GIF:s, but it's probably only a matter of time before other animation formats will be supported by Gimp.

MOVING BEYOND THE BASICS

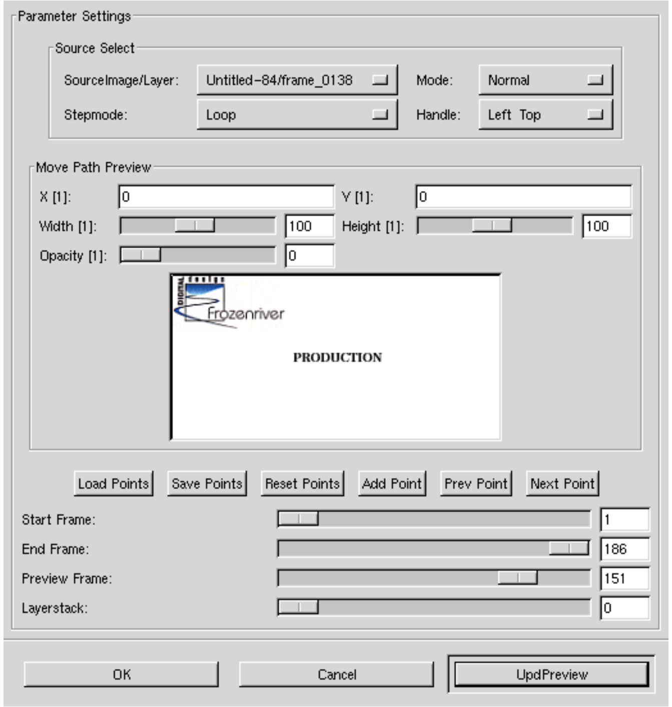
HOW FAR CAN ANIMFRAMES TAKE ME?

There are nearly no limits to what you can achieve with AnimFrame, but some animations are harder to make than others. To create certain effects, you may have to edit frames by hand; for example if you want to rotate an object at the same time as it moves. We've made an animation demo where you can see a few examples of what AnimFrames is capable of; like ***paning*** the background, making an ***aftertext*** that ***rolls*** over the frame, a ***perspective intro text***, moving objects ***away*** from you and ***towards*** you, setting ***different speeds*** to objects, etc. Later in this chapter you'll find a gallery with a description of how the demo animation was created.

THE MOVE PATH TOOL

The **Move Path** window is the heart of Anim Frame, so let's have a closer look at what it can do and what the parameter settings mean. The dialog is divided into three different parts; **Source Select**, **Move Path Preview** and a bottom part where you set some slides and buttons.

Source select



Parameter Settings

Source Select

SourceImage/Layer: Untitled-84/frame_0138

Mode: Normal

Stepmode: Loop

Handle: Left Top

Move Path Preview

X [1]: 0

Y [1]: 0

Width [1]: 100

Height [1]: 100

Opacity [1]: 0

Start Frame: 1

End Frame: 186

Preview Frame: 151

Layerstack: 0

The SourceImage/Layer

This menu is where you select animation object, i.e the layer or image where your animation object is. You can only open AnimFrames/Move Path from a Frame image, so you must keep a frame image (just one!) open as well as one or more object images.

Mode

Here you set the color/brightness relation between the imported layer/image and the rest of the layers in the frame. *You can read more about modes in chapter 16.*

Handle

The Handle determines which part of the imported image/layer should be used as point of departure for the move path. We will discuss handles more in detail when we come to Points a bit further down the text.

Step mode

Determines the nature of the animation. **Loop** creates a continuous animation from the first control point to the last. **Loop reverse** does the same but goes the other way around; from the end point to the first control point. (Once and Once reverse is currently the same as Loop, but that will change with the development of this plug-in). **Ping Pong** makes your animation go to the first point -> last point -> back to the position after first point -> position before last point and so on. **None** cancels the animation. We recommend that you stick to Loop and Loop reverse, and control the output by animating different sequences separately, or by exchanging frames.

The preview window

Move path preview and control points

Here's where the actual setting of the animation takes place. By placing a number of control points along a certain track, you create a linear moving path for the object. Note that the path and the points are invisible. You can only see them by stepping your way forwards or backwards with the **Next Point/Prev Point** buttons.

By default, two control points are placed in the upper left corner. The animation object will move from the first to the second control point over the frames you set by the slides **Start Frame** and **End Frame**. You can add more controls points by pressing the **Add Point** button. As the animation object's move path is determined by the control points, you will have to use a lot of control points to create a circular path. The maximum number of control points are 256, but that should be quite sufficient.

To start adding/changing control points, click in the preview window to set the first control point. Then press **Next Point** to set the second control point. If you want to adjust the position of the first control point, simply press **Previous Point** (the status of the point controls will now change from 2 to 1, e.g X[2] to X[1] so you know what control point that you are dealing with).

If you want to add a control point, press the **Next Point** button to arrive at the last control point (in this case number 2), press Add Point and click at the place where you want to put it. Note that you can't add a control point to adjust the path between two existing points, you can only add to the prolong the path. Also remember to be sure to stand at the last control point when you add a new point, otherwise it will be quite hard to follow the path you're creating.

A very nice option is the possibility to **save** and **import** control points. This makes it possible to let different objects use the same move path (you can also save your favorite move path for later use).

You can create a very exact moving path by using the control point **position commands**; X and Y. It's normally better to create a path with the mouse in the preview window, but the values here give you the necessary numerical information about every point's position.

Note that the position cross is not restricted to the area inside the preview. Dragging it outside of the preview window will for example enable you to pane the background. The **handle** mentioned in the earlier paragraph is for locking a certain corner or the midpoint of the import layer/image to the move path. If you select top left handle, the control point will stick to the top left corner of the import layer, forcing the layer's position to the bottom right side of the control point.

The **Width** and **Height** controls allows you to you resize the layer/image at a certain control point, This is of course ideal for zooming.

Opacity controls transparency of the object image. You can for example use this control to fade in and out of images. As you see, with these controls you can achieve many of the common effects you've seen in ordinary movies.

Control slides

The Frame Slidebars

These slidebars control where and how the import layer should be placed in your frame. The **Start Frame** and **End Frame** sliders determine which frames the image/layer (with the current move path) should be imported to. The **Preview Frame** slider controls which frame you'll see when you press the update preview button. Note that this only works for the first and last control point. This tool is for viewing the beginning and end of an animation sequence

The Layerstack

The **Layerstack** slidebar controls in which position the imported layer/image will be placed in relation to the other layers in the Frame. In order to understand this you must think of the layers as a stack of cards. If the layerstack is set to zero, then the layer will end up on top of all other layers (cards). If the layerstack is set to 1, then the layer ”card” will be inserted between the first and second layer ”card” etc.

THE ANIMFRAME MENU

Undo and preview?

Besides the preview in the move path tool, there is no preview which lets you see your move while it’s under production (it would be to slow to load and unload images from disk, especially if there are a lot of layers). If you want a preview you can either step forward with **Next Frame** or copy all your frames to a new directory and create a multi-layer image.

There is no undo in AnimFrame because all frames are saved immediately. A way around this is to always keep your layers in the frames. If you aren’t satisfied with a certain frame layer sequence, then you can easily delete them all with the Frames Layer Delete command.

Frames LayerDel

This command will bring up a dialog asking you to specify the frame range and position in the layer stack (see above). Be careful here, because when you press OK, there’s no way back.

Frames Convert and Exchange

If you want to export your frames to different image formats, you can use the **Frames Convert** command. The drop down menu lets you select whether you want to keep the image type (RGB) or convert it to a different format (Gray, Indexed or RGB). This is a must because if you want to save your frames to GIF format, you have to convert it to Indexed first. The field beside the menu is for specifying image format by typing the suffix e.g.gif,.tiff,.jpeg etc. To decide what frames to convert, use the **From Frame** and **To Frame** slidebars. The last slide will flattens the frames before you save them. If you save them as GIF:s, there’s no problem, since GIF handles layers. However, if you want to save them as TIF images, you have to flatten the images first, because tif doesn’t support layers.

Exchange Frame is the tool to use if you want to rearrange your frames. The dialog will ask you what frame you want to swap the current frame with. For example. if you invoke the command from frame 3 and sets the slide to 5, then frame 3 will end up as frame 5, and frame 5 will end up as frame 3.

Frames Flatten and Frames to Image

Frames Flatten and Frames to Image are the tool to use when your animation is finished and you want to save it as a GIF animation. In order to make a multi-layered image, you must first flatten all layers, and then convert them with **Frames to Image**. We recommend that you do this on a copy, and not the original frames (e.g `cd ”your frame directory” && mkdir framecopy && cp * frame-`

`copy` in a terminal window). If you flatten the original frames, you'd better make sure that everything's OK, because *once the frames are flattened there is no way to edit the frame layers*.

Frame Duplicate

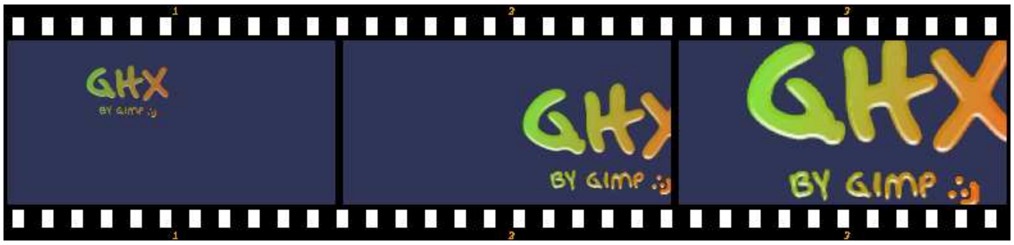
As we've mentioned before, this tool is used for creating base frames in an animation. Note that if you are in a middle of an animation and want to duplicate a frame, the duplicate frames will be inserted after the original frame.

ANIMATION GALLERY/TUTORIAL

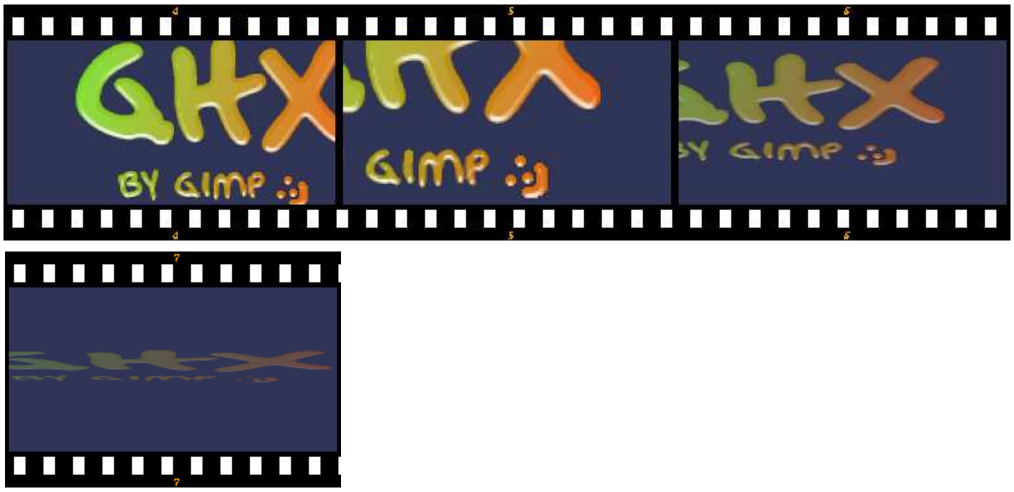
We have made a little animation to show you some of the things you can do with Anim Frame. The animation is 186 frames long.

The animation begins with a logo that moves toward you before it bounces back, stretches out and finally fades away. Those effects were quite easy to achieve:

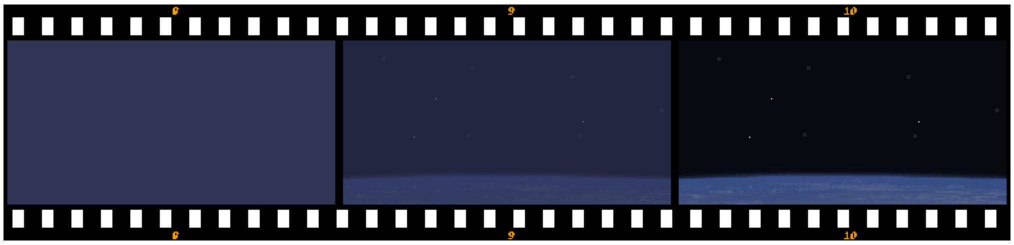
We started by specifying a low value for **Width** and **Height** in the first point in the logo's move path, so that the logo would appear to move towards the camera.



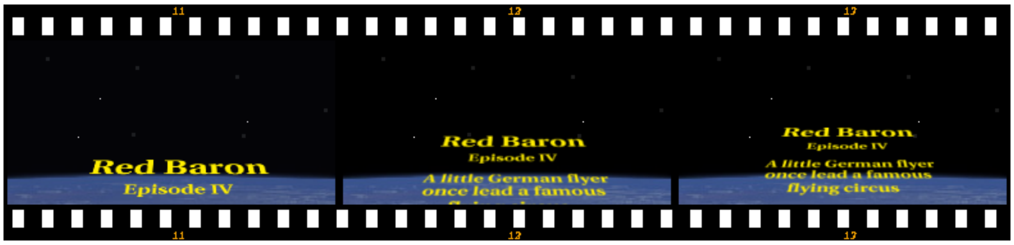
The fade away/stretch effect was created by setting the **opacity** in the final point to 0%, the **X scale** to 200 and the **Y scale** to 0.

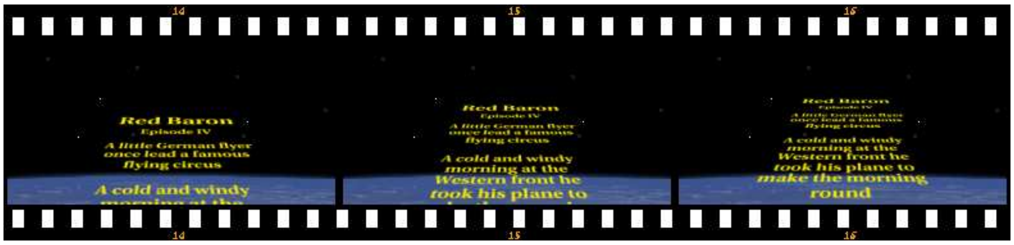


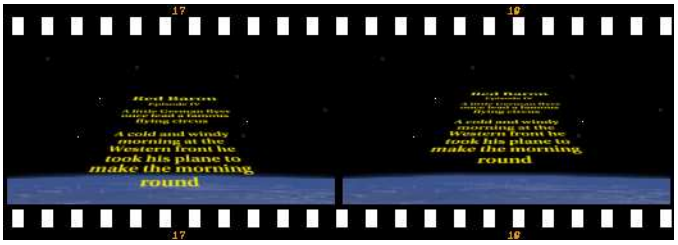
The next phase in the animation was to fade over from the background in the bouncing logo scene to the space background in the next scene. This was done by placing the space background in the lowest layer in a series of 10 frames, then we made a **move path** with the blue logo background with 100% opacity in the first control point and 0% opacity in the final control point.



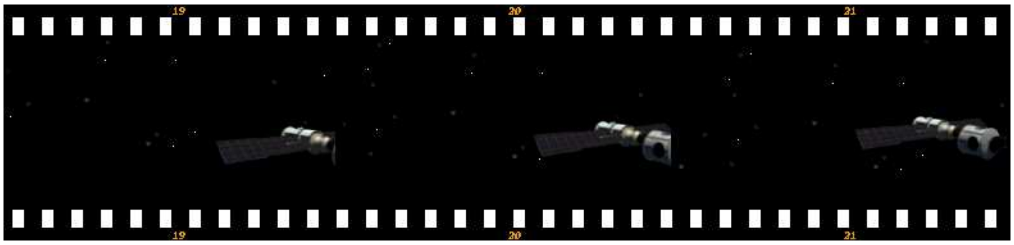
Then we amused ourselves by creating a paraphrase of the rolling perspective text in a movie we assume you've never heard of. (Don't read the text, it's quite meaningless) For the perspective effect we used an image which was higher than the frame, and then we skewed the text with the **Transform/Perspective** tool. The move path (center Handle) begins a bit below the actual frame, and the second and final point was placed a bit below the centre of the image. The final point also scaled down the import image to about a 1/3 of its original size. This made the text float away, just like in the movie. Afterwards, we adjusted the time each frame was shown to keep the text from accelerating as it moves away.

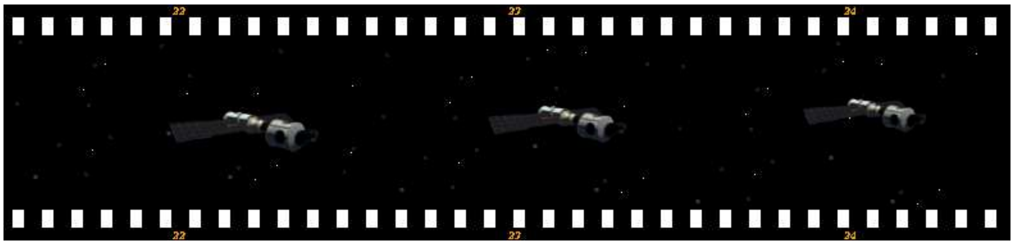

11
12
13
Red Baron
Episode IV
A little German flyer
once lead a famous
flying circus

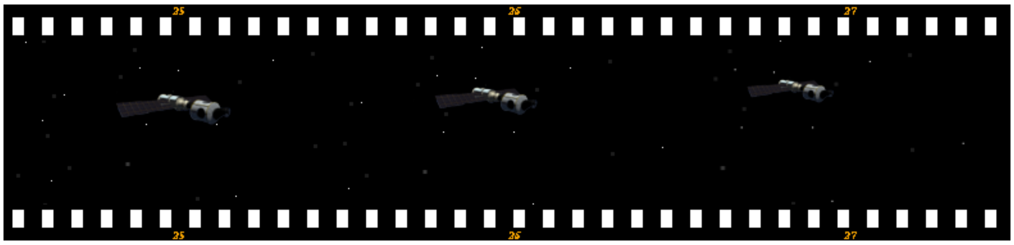

14
15
16
Red Baron
Episode IV
A little German flyer
once lead a famous
flying circus
A cold and windy
morning at the
Western front he
took his plane to
make the morning
round


17
18
Red Baron
Episode IV
A little German flyer
once lead a famous
flying circus
A cold and windy
morning at the
Western front he
took his plane to
make the morning
round

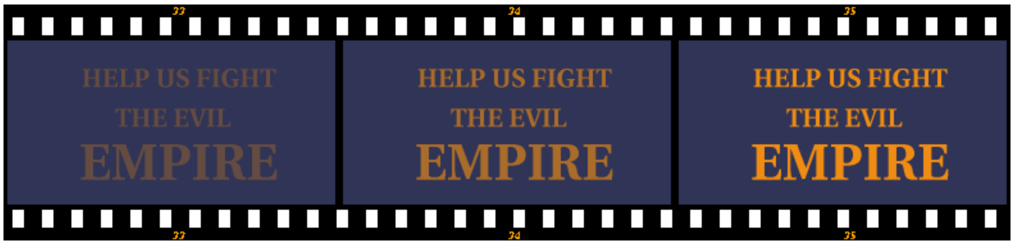
The space scene displays an orbiting space craft or satellite. The space craft accelerates as it moves away from you and around the Earth. As background setting we used a large image of the Earth seen from space. The space part of the background was panned with the **move tool**, and the last stage in the scene was a pan and zoom down towards Earth. Then we added the space craft image with a move path from the right side of the ”screen” to the left. As the space craft was moving along the path it was simultaneously scaled down. This created the acceleration effect, and the feeling than you move away from the space craft as it disappears beyond the horizon.



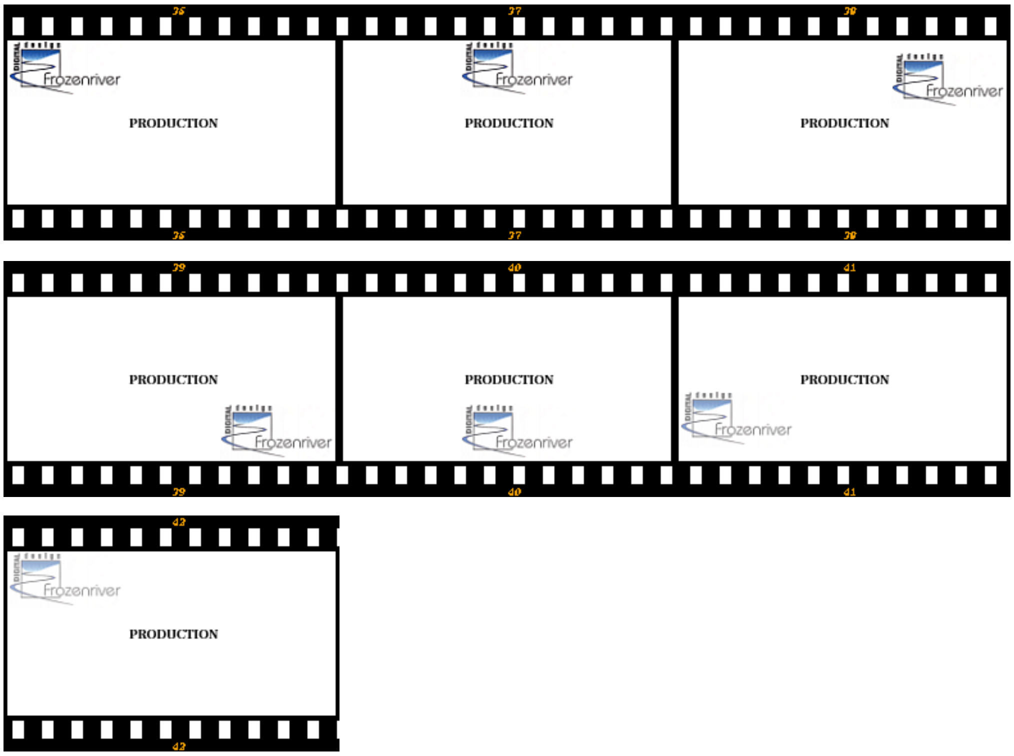




Now it’s time for some propaganda in the Help us fight the evil $Empire fade out. This text was faded in the same fashion as the bouncing logo background, only in reverse direction.


33
34
35
HELP US FIGHT
THE EVIL
EMPIRE
HELP US FIGHT
THE EVIL
EMPIRE
HELP US FIGHT
THE EVIL
EMPIRE

The final scene displays the production team behind this animation, and this book. The Frozenriver logo was made to circulate around the word ”Production” and fade away at the same time. This was done by using ”production” as a background. The logo was then made to move around the text in a five point move path, where the opacity was lowered for each point.


36
37
38
Frozenriver
PRODUCTION
39
40
41
42

This animation demo comes both in GIF and Anim Frame format, and can be downloaded from `ftp.gimp.org/pub/gimp/manual`.

part

VIII

Script-Fu

- SCRIPT-FU
- DAVE'S SCRIPT-FU LESSONS
- MIKE'S BLACK BELT SCHOOL OF SCRIPT-FU

chapter

37

Script-Fu; description and function

In this chapter will we take a look at what's under the Xtns/Script-Fu menu in the Toolbox and in the Script-Fu menu in the image menu.

SCRIPT FU?

Script Fu is what the Windoze world would call a ”macro”. But Script Fu is more powerful than that. Script Fu is based on an interpreting language called scheme, and works by using querying functions to the Gimp ”data base”. You can do all kinds of things with Script Fu, but an ordinary Gimp user will probably use it for automating things that:

- he/she wants to do frequently
- or/and is really complicated to do, and hard to remember.

Remember that you can do a whole lot with a script fu. The scripts that comes with Gimp can be quite useful, but they can also serve as models for learning script fu, or at least as a framework and source of modification when you make your own script. Read Chapter 38 and 39 if you want to learn how to make scripts.

INSTALLING SCRIPT-FU:S

The great thing about Script-Fu:s is that you can share your script with all your Gimp friends. There are several scripts that come with Gimp by default, but there are also scripts that are available for download all around the Internet.

If you have downloaded a script, copy or move your new script to your .gimp/scripts directory and make a refresh.

The script will now appear in one of your menus. If you don’t find it, look for it under the root file menu filters. If it doesn’t appear at all, something was wrong with the script.

Note that you can’t use more than one script-fu dialog at a time. So don’t open a script and one more after that. The last one will never be opened and displayed.

DON’T AND DO:S

A common error when you are dealing with script Fu:s is that you simply bring them up and press the OK button. When nothing happens, you probably think that the script is broken or buggy, but there is most likely nothing wrong with it. Think again. Did you really read the information in the dialog, or did you just press the button? If you forgot an input the script needs, or if you gave it the wrong input, the script will fail. One of the most common errors is that the font specified in the script dialog hasn’t been installed in your system. So please, check the information in the dialog before blaming the script.

DIFFERENT KINDS OF SCRIPT FU:S

There are two kinds of Script Fu:s - stand alone scripts and image dependent scripts. You will find the stand alone variants under Xtns/Script Fu/Type of Script, and the Image dependent scripts are placed under <image>/Script Fu/Type of Script.

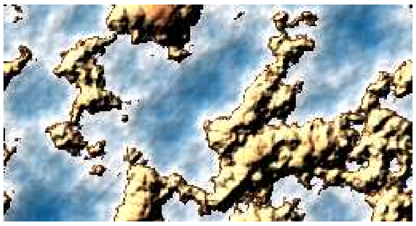
STAND ALONE SCRIPTS

We will not try to describe every script in depth, as we did with the filters. Most Script Fu:s are very easy to understand and use. At the time of this writing, the following types are installed by default.

- Patterns
- Web page themes
- Logos
- Buttons
- Utils
- Make Brush
- Misc.

Patterns

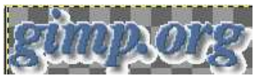
You will obviously find all kinds of pattern generating scripts here. Generally, they are quite useful since you can add many arguments to your own patterns.



.We'll take a look at the Land script. In this script you have to set the image/pattern size, and specify what levels of random to use for your land creation. The colors used to generate the land map are taken from the currently selected gradient in the gradient editor. You must also supply values for the level of detail, land and sea height/depth, and the scale. Scale refers to the scale of your map, just as in an ordinary road map. 1:10 will be typed as 10.

Web page themes

Here is clearly a practical use for scripts. By creating a script for making your custom text, logos, buttons arrows etc. for your web site, you will give them all the same style and shape.You will also be saving a lot of time, since you don't have to create every logo, text or button by hand. You will find the Gimp.org theme under the Web page theme submenu. If you want to create your own theme, this script will serve as an excellent start to modify (remember to share your modified script's with the rest of the Gimp community).



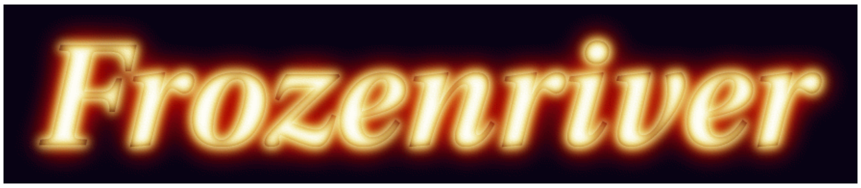
The scripts are quite self-explaining but here are some hints:

- Leave all strange characters like ' " intact
- Make sure that your pattern exists
- Padding refers to the amount of space around your text
- A high value for bevel width gives the illusion of a higher button.
- If you type TRUE for "Press", the button will look pressed down.
- Choose transparency if you don't want a solid background. If you choose a solid background, make sure it is the same color as your webpage background

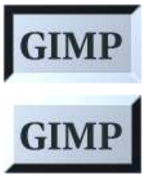
Logos

Here will you find all kinds of logo-generating scrips. This is nice, but use it with care as people might recognize your logo as made by a known Gimp script. You should rather regard it as a base that you can modify to fit your purpose. The dialog for making a logo is more or less the same:

- In the "Text String" field you type your logo name, like Frozenriver.
- In the "Font Size" text field, type the size of your logo in pixels.
- In the "Font" text field, type the name of the font that you want to use for your logo.
- To choose the color of your logo, just click on the color button. This brings up a color dialog.
- If you look at the current command field, you can watch the script run.



Make Buttons



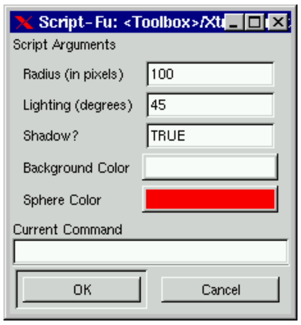
Under this headline you'll find a script that makes beveled buttons. The script has a dozen parameters or so, and most of them are similar to those in the previous scripts. You have to experiment with different settings to come up with a button you like.

Utils

utopia
times

Under Utils will you find small but nice script. We'll take a look at the Fontmap script, which makes an image of your fonts. You will have to type the names of the fonts you want displayed in the "Fonts" text field. A tip is to use xfontsel to see what fonts are installed. The Custom gradient script creates an image of the current custom gradient in the gradient editor. This can be useful if you want to pick colors from a gradient as in a a palette.

Misc.



Script Arguments

Radius (in pixels) 100

Lighting (degrees) 45

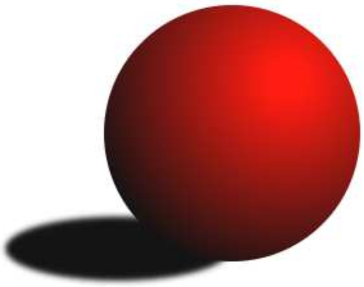
Shadow? TRUE

Background Color

Sphere Color

Current Command

Under Misc. you'll find scripts which can be quite useful, but aren't suitable for the other submenus. An example is the Sphere script. You will have to set the radius in pixels to determine the sphere size. The lighting angle is where at the sphere you point the spotlight. This value also has an impact on the sphere shadow. If you don't want a shadow, you will have to type FALSE. The last thing you have to select is background color, and the color of your sphere



Make Brush

Brushes lets you make your own custom rectangular/circular brushes, with or without feathered (blurred) edges. To ensure full control over the parameters you will have to look in chapter 11. The script will automatically store your brush in your personal brush directory. You just have to press refresh in the brush selection dialog to use your newly created brush.

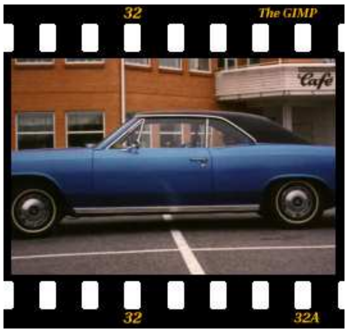
IMAGE DEPENDENT SCRIPTS

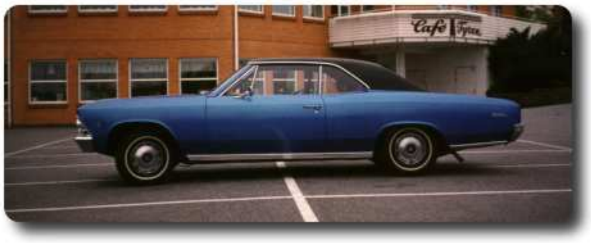
The image dependent script are scripts that will do certain things to an already existing image. These scripts are more like the plug-ins in the filters submenu. At the time of this writing, the following script groups are installed by default.

- Decor
- Modify
- Animators
- Stencil Ops
- Alchemy
- Shadow
- Render
- Utils
- Selection

Decor

Scripts that add different kinds of borders to an existing image or selection.





As you see in this example, you must take care to be a bit more exact than usual with scripts than with filters. Our Malibu was sadly chopped in both the front and the rear. The other script was a lot kinder when adding a nice border shadow to our Malibu image.

Modify

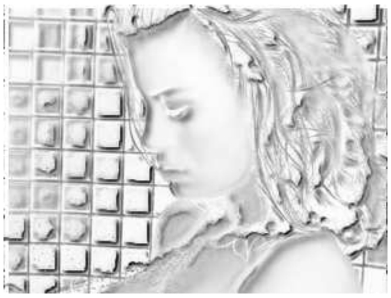
Here will you find the add border script, which is nearly perfect if you want to make a passe-partout, or simply add a beveled border.

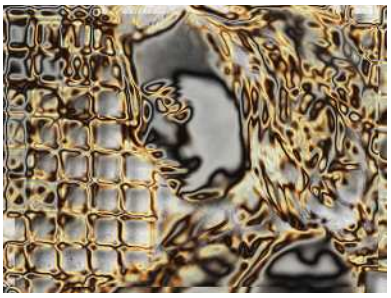
Animators

These scrips will do all kinds of automated animations. Try for example the Waves script, which creates a pond ripple effect.

Stencil Ops

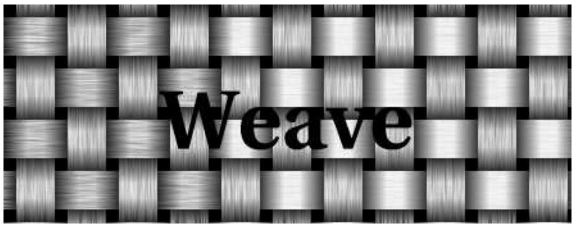
Here you'll find two scripts: ”Carve it” and ”Chrome it”, which can create some truly nice artistic effects on grayscale images.





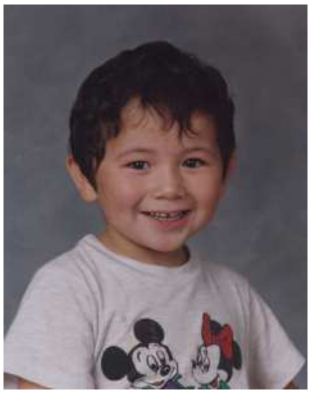
Alchemy

This is the script equivalent to the artistic filters and enhance filters in the filters submenu.

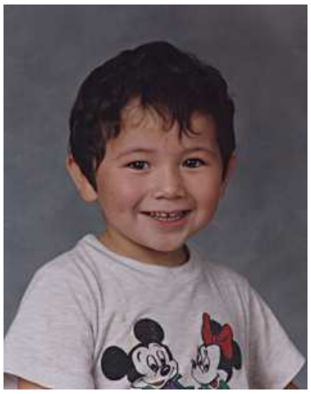


Unsharp Mask

One of the most useful Script Fu's is **Unsharp Mask** which *sharpens* an image. Unsharp Mask is often a better alternative than **sharpen**. Sharpen will accentuate all of your image, including scratches, noise and other imperfections that you don't want in your image. Unsharp Mask works by increasing the contrast of edges and nearby pixels, while other areas are left pretty much untouched. This makes Unsharp mask ideal for enhancing scanned images. The *mask value* is the amount of sharpening you want, or more accurately; how wide the edge areas should be.



Before Unsharp Mask



After Unsharp Mask

If you apply image effect functions like Scale, Rotate, Perspective etc. try using Unsharp mask afterwards, since those type of filters use *interpolation* and thereby softens or blurs your image. If you just want to enhance a small part of your image, then ordinary sharpen is an adequate tool (you can't use Unsharp Mask on selections).

Tips

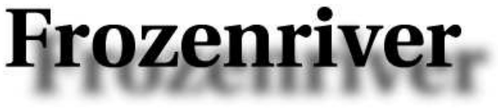
It's a good idea to run Unsharp mask twice with half the mask value instead of just once with the whole value. This will sharpen more smoothly (don't forget to flatten the image).

Shadow

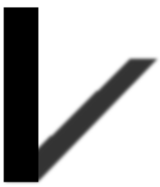
Here are two really useful scripts that you will probably use quite frequently. These scrips are called Drop Shadow and Perspective Shadow.

Drop Shadow

Drop Shadow will cast a shadow behind your selected object. It has three important parameters; X and Y offset which is where the shadow will be placed in relation to the selected object, and it’s measured in pixels. High values makes the shadow look like it’s far away, and low values will make it look closer to the object. The blur value is also important since a shadow that is cast far from the object has a higher blur level.



Perspective Shadow



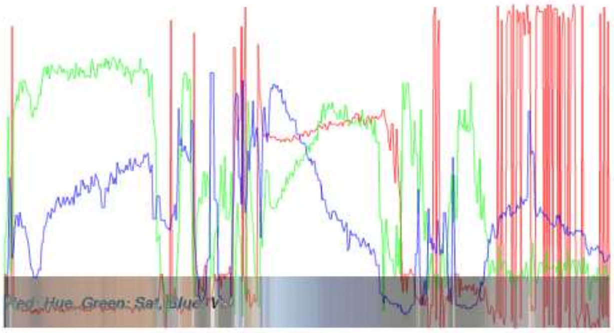
Perspective Shadow has a very important parameter; and that is the perspective angle. if this angle is set to 0 or 180 there will be no shadow, because the script assumes that the object has no thickness. This also means that this script looks fine in certain angles, but unnatural in others. The other parameters are quite self-describing. You’ll get more blur if the horizon is far away, and the shadow length is the length in relation to the selected object.

Render

Rendering scripts are not very different from what you find in the filter menu

Utils

Here are two scripts that display image structure, the example below shows the HSV curve of the Malibu image.



Selection

These scripts can be applied on your current selection. There isn't much to say, they are quit-self explaining.

chapter

38

Writing a Script Fu

A Scheme Tutorial for Gimp Users: Author Dov Grobgeld; Copyright 1997, 1998 by Dov Grobgeld License GDPCL.

INTRODUCTION

One of the wonderful features of Gimp is that it all its functionality may be accessed through **scripting**. The major scripting language for Gimp that has been attached to it today is **Scheme**. This document will try to give a brief introduction to Scheme, just teaching the essentials in order to write script-fu scripts, without getting into the programming language theory that is so typical of other Scheme references. If you want you can visit Dov's online version of this lesson at http://imagic.weizmann.ac.il/~dov/gimp/scheme-tut.html he has also a lesson about the Perl scripting and Gimp if you want to use Perl in sted of Scheme it's at http://imagic.weizmann.ac.il/~dov/gimp/perl-tut.html.

EXPRESSIONS

Scheme is a **Lisp** variant and all expressions are surrounded by parenthesis. E.g. a list which will calculate the *sum* of 3 and 4 is written:

```
(+ 3 4)
```

The **+** sign is the **addition** function and 3 and 4 are the first and second **parameters** to this function. Expressions may be **nested**, so the expression (3+4)*(5/6) would in Scheme be written:

```
(* (+ 3 4) (/ 5 6))
```

White space has no importance so the above expression may as well be written:

```
(*
   (+ 3 4)
   (/ 5 6))
```

FUNCTIONS

Besides the four arithmetic functions that are represented through the symbols + - * / there are lots of other functions built into the language. All of them have the form:

```
(foo param1 param2 ...)
```

Additional functions may be defined by the user through the **define** keyword. E.g. a function that calculates the *square value* of its single argument may be declared like this:

```
(define (square x) (* x x))
```

and this function may be called through:

```
(square 5)
```

Lisp and its variants make heavy use of **lists**. Script-fu is no exception and it uses e.g. a list of three elements to write an RGB color. E.g. the color orange would be written:

```
'(255 127 0)
```

The **'** sign is necessary to tell Scheme that this is a **literal** list. If the **'** was omitted Scheme would try to look up a function with the name 255 and send it the two parameters 127 and 0, which is obviously not what we want.

To create a variable called *orange* with the above value, and then set the background color to it, we can write:

```
(set! orange '(255 127 0))
(gimp-set-background-color orange)
```

CAR, CDR AND FRIENDS (*)

A list in Scheme is always composed of a **head** and a **tail**. The head is the first entry in the list, and the tail is the list of the rest of the elements. This means that the list (255 127 63) really means (255 (127 (63 ()))) but Scheme allows the previous form as a short cut. The **car** function is used to return the head of the list and the **cdr** (usually pronounced cudder) is used to get the tail of the list.

[The following is a test of the above functions which may interactively be conducted in the Script-Fu console.]

```
=> (set! color '(255 127 63))
(255 127 63)
=> (car color)
255
=> (cdr color)
(127 63)
```

To get the blue component of a color it is necessary to apply the **cdr** function twice and then the **car** function.

```
=> (car (cdr (cdr color)))
63
```

This is very inconvenient to write. Therefore there are defined abbreviations of the form: **cadr**, **cddr**, **caddr**, etc. that concatenate the operations described above. The previous expression may therefore much more conveniently be written:

```
=>(caddr color)
63
```

For the Script-Fu writer one of the most important uses of the **car** function is to *access* the returned values from the built-in gimp functions. All gimp-functions return a *list*, and even if the list contains only one element it must be accessed by car. This is e.g. the case for the important functions `gimp-new-image` and `gimp-new-layer` used below.

LOCAL VARIABLES (*)

More experienced Scheme programmers mostly use local variables instead of the global variables described above. This is considered better programming practice and this construct should be recognized in order to be able to read others ScriptFu scripts.

Local variables are declared through the **let** keyword as in the following example:

```
(let* ((a 3)
   (b 4))
   ((* a b)))
```

Here **a** and **b** have a local scope and retain their values only up to the closing parenthesis matching the one before let* .

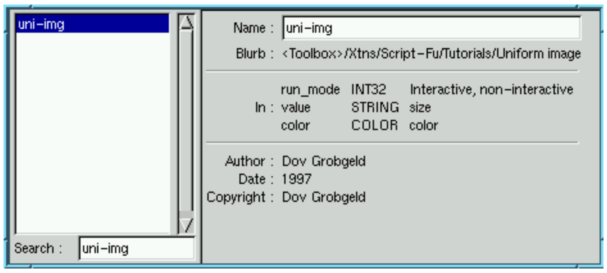
THE GIMP PDB

All functionality in GIMP is available through the *procedural database* (PDB). Each procedural database function has a corresponding scheme function mapping. E.g:

```
(gimp-image-new 100 150 RGB)
```

produces a new gimp image of type RGB and size 100x150.

In Gimp, a browser for all the functions in the PDB is included. This browser is available from the main menu through `Xtns>DB BROWSER`. E.g, the DB Browser entry for **uni-img**, which we will define in the example below looks like this



Name : uni-img

Blurb : <Toolbox>/Xtns/Script-Fu/Tutorials/Uniform image

```
     run_mode  INT32   Interactive, non-interactive
In : value     STRING  size
     color     COLOR   color
```

Author : Dov Grobgeld
Date : 1997
Copyright : Dov Grobgeld

Search : uni-img

For the Scheme programmer this information shows that **uni-img** may be called with three parameters of the types `INT32`, `STRING` and `COLOR`. The different types will be explained below.

REGISTERING THE SCRIPT WITH SCRIPT-FU

After a function has been written, it has to be registered with script-fu before it can be used. This is done through the Scheme function script-fu-register. The registering has following purposes:

- Choose the place of the script in the Script-Fu pulldown menus.
- Tell script-fu the type of parameters the script takes and give these parameters default values.
- Register the script as a command in the PDB.

The last point above actually means that a script is from Gimp's viewpoint in no way different from a built-in command or a plugin command. As long as a command is registered in the PDB it can be called by any script or plugin.

The parameters of script-fu-register may be divided into two groups. The first group of seven parameters must always be given. These are:

- The name of the Lisp function.
- The position of the script in the Gimp menus.
- A help string describing the function of the script.
- The script author.
- The script copyright.
- Script date.
- List of valid image types for the script. This only has a meaning on scripts operating on images that already exist.

After these seven parameters have been given, follows a list of the parameters required by the script. Each parameter is given as a group of three items:

- The type of the parameter. Valid types are

  `SF-COLOR`

  `An RGB color.`

  `SF-TOGGLE`

  `A true or false value.`

  `SF-IMAGE`

  `SF-DRAWABLE`

  `SF-VALUE`

  `Any scalar value, string, integer, or floating point.`

- A label for script-fu to display when querying for the parameter.
- A default value.

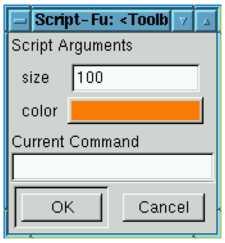
A COMMENTED SCRIPT

The following script **uni.scm** receives two parameter from the user, the **size** of the image, and a **color** and goes on to produce a uniform image of the requested size and the requested color. Not very useful, but it shows the essential steps in producing a script-fu script. (";" is a mark that the following is a comment)

```
; Define the function of the script and list its parameters
; The parameters will be matched with the parameters listed
; below in script-fu-register.
(define (uni-img size color)
   ; Create an img and a layer
   (set! img (car (gimp-image-new size size RGB)))
   (set! layer (car (gimp-layer-new img size size
            RGB "layer 1" 100 NORMAL)))

   ; The following is done for all scripts
   (gimp-image-disable-undo img)
   (gimp-image-add-layer img layer 0)

   ; Here is where the painting starts. We now have an image
   ; and layer and may paint in the layer through the PDB functions
   (gimp-palette-set-background color)
   (gimp-edit-fill img layer)

   ; The following is also done for all script
   (gimp-display-new img)
   (gimp-image-enable-undo img)))

   ; Finally register our script with script-fu.
```

```
(script-fu-register ”uni-img”
          ”/Xtns/Script-Fu/Tutorials/Uniform image”
          ”Creates a uniform image”
          ”Dov Grobgeld”
          ”Dov Grobgeld”
          ”1997”
          ””
          SF-VALUE ”size” ”100”
          SF-COLOR ”color” ’(255 127 0)))
```

To test it copy it to `$HOME/.gimp/scripts/uni.scm` and press **Refresh** in `Xtns/Script-Fu`. The script Uniform image should now appear in the pulldown menu `Script-Fu>Tutorials>Uniform image`. Selecting this script gives the following popup:



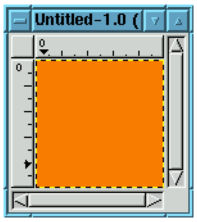
Script Arguments

size 100

color

Current Command

Accepting these default parameters through the OK button gives us the following new image:



It is also possible to access this script through the Script-Fu console by typing the command

```
(uni-img 100 ’(0 255 127))
```

HANGING A SCRIPT IN THE IMAGE MENU

In the uni-img script it was placed under `Xtns>` in the main Gimp window. This is done to create a new image that is independent of earlier images. It is also possible to create a script which works on an already existing image. If in script-fu-register as the second argument is written:

```
<Image>/Script-Fu/...
```

then the script will be available through the Gimp menu that is launched by the right mouse button over an image.

Here is an example script which **copies** the current layer to a **new layer**, **blurs** it and **inverts** it.

```
(define (script-fu-copy-blur  img
                              drawable
                              blur-radius)
   ; Create a new layer
   (set! new-layer (car (gimp-layer-copy drawable 0)))

   ; Give it a name
   (gimp-layer-set-name new-layer "Gauss-blurred")

   ; Add the new layer to the image
   (gimp-image-add-layer img new-layer 0)

   ; Call a plugin to blur the image
   (plug-in-gauss-rle 1 img new-layer blur-radius 1 1)

   ; Invert the new layer
   (gimp-invert img new-layer)

   ; Flush the display
   (gimp-displays-flush)
)

(script-fu-register "script-fu-copy-blur"
            "<Image>/Script-Fu/Tutorial/copy-blur"
            "Copy and blur a layer"
            "Dov Grobgeld"
            "Dov Grobgeld"
            "1998"
"RGB*, GRAY*"
SF-IMAGE "Image" 0
SF-DRAWABLE "Layer to blur" 0
SF-VALUE "Blur strength" "5")
```

PAINTING AREAS WITH SELECTIONS

In uni-img we called the procedure `gimp-edit-fill` to fill the whole image. Looking at the info for `gimp-edit-fill` in the DB browser we find the following:

NAME:	gimp-edit-fill		
BLURB:	Fill selected area of drawable		
IN:	image	IMAGE	the image
	drawable	DRAWABLE	the drawable to fill

Thus, if we have a selection active when the `gimp-edit-fill` is called, only the **selection** is painted. There are lots of ways of choosing a selection as can be seen when searching for a ``select'' in the PDB. We will use `gimp-rect-select`, whose entry in the PDB looks as follows:

NAME:	gimp-rect-select		
BLURB	Create a rectangular selection over the specified image		
IN:	image	IMAGE	the image
	x	FLOAT	x coordinate of upper-left corner of rectangle
	y	FLOAT	y coordinate of upper-left corner of rectangle
	width	FLOAT	the width of the rectangle: width > 0
	height	FLOAT	the height of the rectangle: height > 0
	operation	INT32	the selection operation: {ADD (0), SUB (1), REPLACE (2), INTERSECT (3)}

feather	INT32	feather option for selections (0=FALSE, 1=TRUE)
feather_radius	FLOAT	radius for feather operation

A simple use of this function which selects the rectangle (x,y,width,height)=(0,25,100,50), paints this region blue, and releases the selection looks as follows:

```
(gimp-rect-select img 0 25 100 50 REPLACE 0 0)
(gimp-palette-set-background ’(0 0 255))
(gimp-edit-fill img layer-one)
(gimp-selection-none img)
```

LOOPS

The only **looping** construct that exists in Script-Fu is **while**

[Note: this constraint is due to the current scheme interpreter SIOD used for Script-Fu. Once the scheme interpreter as planned is changed to Guile, more looping constructs will probably be added.] The while loop looks as follows:

```
(while      (condition)
            (statement1)
            (statement2)
            :
            )
```

Here’s an example which draws horizontal lines, 16 pixels high, on an image:

```
(set! y 0)
(while (< y size)
            (gimp-rect-select img 0 y size 16 REPLACE 0 0)
            (gimp-edit-fill img layer-one)
            (set! y (+ y 32)))
```

FLOATING SELECTIONS

When pasting an image from the clipboard, or when creating text in a a drawable, the result is not put directly in the drawable. Instead it is put into a special temporary layer known as a **floating selection**.

The floating selection may be manipulated in several ways, and finally it is merged into its associated layer, a process known as **anchoring**.

HELLO WORLD - WRITING TEXT IN AN IMAGE

When creating text through the `gimp-text` command, the text is always put into a temporary layer. This temporary layer then has to be anchored. Here is an example of creating some text which is pasted into the current drawable:

```
(define (script-fu-hello-world img drawable)
   ; Start an undo group. Everything between the start and the end
   ; will be carried out if an undo command is issued.
   (gimp-undo-push-group-start img)

   ; Create the text. See DBbrowser for parameters of gimp-text.
   (set! text-float (car (gimp-text
                         img
                         drawable
                         10 10
                         "Hello world"
                         0
                         1
                         30
                         1
                         "*" "Helvetica" "medium" "r" "*" "*")))

   ; Anchor the selection
   (gimp-floating-sel-anchor text-float)

   ; Complete the undo group
   (gimp-undo-push-group-end img)

   ; Flush output
   (gimp-displays-flush))
```

```
(script-fu-register ″script-fu-hello-world″
            ″<Image>/Script-Fu/Tutorial/Hello World″
            ″Write Hello World in the current image″
            ″Dov Grobgeld″
            ″Dov Grobgeld″
            ″1998″
            ″RGB*, GRAY*″
            SF-IMAGE ″Image″ 0
            SF-DRAWABLE ″Layer″ 0)
```

This script shows another feature we haven't mentioned before. The possibility of creating an **undo** group. All the commands between the commands `gimp-push-undo-group-begin` and `gimp-push-undo-group-end` are undone together if the undo command is issued.

COPYING A SELECTION

To **copy** a selection, the command `gimp-edit-copy` is used. It places a copy of the selection contents in the **cut-buffer**. The contents of the cut-buffer may then be pasted into a layer, the same layer or another one, and it is then pasted as a floating layer.

In the following example the selection is copied, pasted into the same layer, offset a fixed distance, finally anchored. Try it by drawing a small blob in the middle of the image, select the blob, and then call this script.

```
(define (script-fu-sel-copy img
                            drawable)

   (gimp-undo-push-group-start img)

   (gimp-edit-copy img drawable)
   (set! sel-float (car (gimp-edit-paste img drawable FALSE)))
   (gimp-layer-set-offsets sel-float 100 50)

   ; Anchor the selection
   (gimp-floating-sel-anchor sel-float)
   ; Complete the undo group
   (gimp-undo-push-group-end img)

   ; Flush output
   (gimp-displays-flush))


(script-fu-register ”script-fu-sel-copy”
            ”<Image>/Script-Fu/Tutorial/Selection Copy”
            ”Copy the selection into the same layer”
            ”Dov Grobgeld”
            ”Dov Grobgeld”
            ”1998”
            ”RGB*, GRAY*”
            SF-IMAGE ”Image” 0
            SF-DRAWABLE ”Layer” 0)
```

chapter

39

Mike Terry's black belt school of Script-Fu

Author Mike Terry; Copyright 1998 by
Mike Terry License, GDPCL.

THE ROAD TO SCRIPT-FU MASTERY

So, little grasshopper, you have found Gimp, and you want to learn of its secrets?

More specifically, you wish to learn of its fantastic scripting abilities, no? You are perhaps tantalized at the prospect of automating image-editing drudgery, or maybe you seek the kind precision in your work that only a well-written script can achieve...

Well, you have come to the right place, my friend, as *Mike Terry's Black Belt School of Script-Fu* can train you in the not-so-ancient art of Script-Fu.

COURSE OUTLINE

In this training course, we'll introduce you to the **fundamentals** of **Scheme** necessary to use Script-Fu, and then build a handy script which you can add to your toolbox of scripts. The script prompts the user for some text, then creates a new image sized perfectly to the text. We will then enhance the script to allow for a *buffer* of space around the text.

Meet your instructor

Let me first confess that I am currently only a yellow-belt of this art, and as such, can only take you so far. However, together, we can press on and reach new heights. If I err, omit some important detail in this training, or am just plain wrong about something, please email me so I may correct it. Similarly, if you have tips or suggestions on how to improve your training, please forward them to me.

I hope you benefit from this training, and may you soon become a Master of **Script-Fu**!

Audience

These training sessions are intended for the beginning Script-Fu'er. When I heard that Gimp was scriptable, I got very excited, and wanted to dive right in. Unfortunately, the tutorials were scant and incomplete, especially if you knew no **Scheme** (like I didn't). After about two days of trying to force my square peg of C/C++ knowledge into the round hole of Scheme, I reckoned a tutorial from the ground-up, chock-full of demos, would do the new Script-Fu'er a lot of good.

Currently, then, the tutorial is really aimed at the beginner, but as I learn more, I will expand it so we can all be Script-Fu Masters! Your suggestions and complaints are welcome.

`Michael Terry`

`mterry@soulfry.com`

LESSON 1: GETTING ACQUAINTED WITH SCHEME

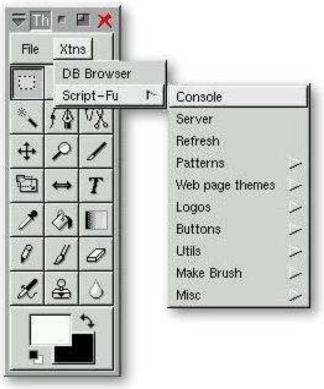
Let's start Scheme'ing

The first thing to learn is that every statement in Scheme is surrounded by **parentheses** - ().

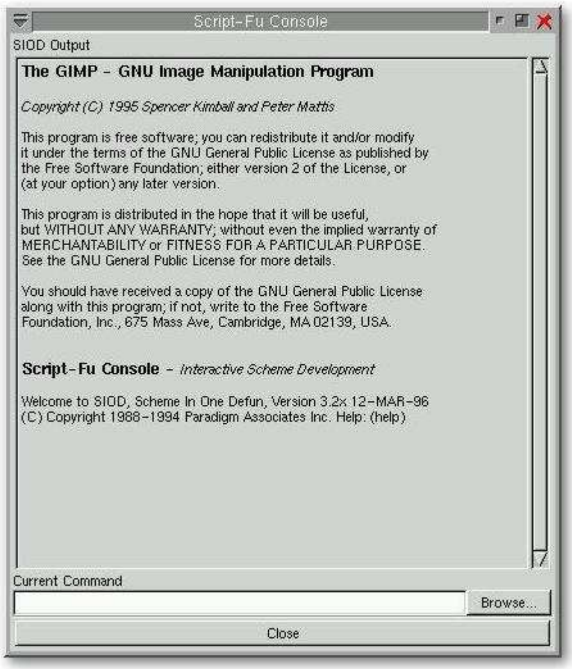
The second thing you need to know is that the **function** is always the *first item* in the parent and the rest of the items are **parameters** to the function. (However, not everything enclosed in parentheses is a function - they can also be items in a list, but we'll get to that later).

The third thing to understand is that **mathematical operators** are also considered **functions**, and thus are listed first when writing mathematical expressions. If you're familiar with **post-fix notation**, or own a calculator that uses **Reverse Polish Notation** (such as most HP calculators), you should have no problem adapting to formulating expressions in Scheme.

Now, young grasshopper, let's practice what we have just learned. Start up Gimp, if you have not already done so, and choose `Xtns->Script-Fu->Console`. This will start up the **Script-Fu Console** window, which allows us to work interactively in Scheme.



In a matter of moments, the **Script-Fu Console** will appear:



```
The GIMP - GNU Image Manipulation Program

Copyright (C) 1995 Spencer Kimball and Peter Mattis

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Script-Fu Console - Interactive Scheme Development

Welcome to SIOD, Scheme In One Defun, Version 3.2x 12-MAR-96
(C) Copyright 1988-1994 Paradigm Associates Inc. Help: (help)
```

At the bottom of this window is an entry-field entitled **current command**. Here, we can test out simple Scheme commands interactively.

Let's start out easy, and add some numbers.

```
(+ 3 5)
```

Typing this in and hitting **Return** yields the expected answer of *8* in the center window.

Now what if we wanted to add more than one number? The **+** function can take 2 or more arguments, so this is not a problem:

```
(+ 3 5 6)
```

This also yields the expected answer of *14*.

So far, so good - we type in a Scheme statement and it's executed immediately in the Script-Fu Console window. Now for a word of caution...

Watch out for extra parens.

If you're like me, you're used to being able to use extra parentheses whenever you want to - like when you're typing a complex mathematical equation and you want to separate the parts by parentheses to make it clearer when you read it. In **Scheme**, you have to be careful and not insert these extra parentheses incorrectly. For example, say we wanted to add 3 to the result of adding 5 and 6 together:

```
3 + (5 + 6) = ?
```

You might be tempted to translate that into the following Scheme statement:

```
(+ 3 (5 6) )
```

However, this is *incorrect* - remember, every statement in Scheme **starts** and **ends** with parens, so the Scheme interpreter will think that you're trying to call a function named **5** in the second group of parens, rather than summing those numbers before adding them to 3.

The correct way to write the above statement would be:

```
(+ 3 (+ 5 6) )
```

Practice a bit with simple mathematical equations in the Script-Fu Console until you're totally comfortable with these initial concepts.

LESSON 2: OF VARIABLES AND FUNCTIONS...

So, my student, you are curious and want to know about variables and functions? Such vigor in your training - I like it.

Variables

Now that we know that every Scheme statement is enclosed in parentheses, and that the function is listed first, we need to know how to create and use variables, and create and use functions. We'll start with the variables.

Declaring global variables with "set!"

Variables can have either **local** or **global** scope. To declare a global variable, use the **set!** function:

```
(set! myVar 5)
```

Now you can use `myVar` as you'd expect:

```
((+ myVar 8)
```

Go ahead and try declaring and using a variable in the Script-Fu Console.

Declaring local variables with "let"

Variables can also have **local** scope. This is achieved by using the **let** statement:

```
(let ( (x 5) (y 6) ) (...) )
```

You'll probably notice an abundance of parens here - all I can say is, get used to it.

In the let statement, after the *let* keyword, we have a list of initialized values. We haven't gotten into lists yet, but the syntax shouldn't be too hard to grasp - we open a parens to contain all the variable declarations, then we enclose each variable declaration in its own set of parens.

After all the variable declarations, we can then start using our local variables, up until the final closing parens (the statements would go where the ellipsis (...) is).

The variables declared within the let statement have scope only within the enclosing parens. An example will help clarify this.

```
(let
(
(x 5)
(y 6)
)
(+ x y)
)
```

As you might expect, this produces the answer of *11* when used within a script. However, if we were to follow this with the following statement:

```
(* x y)
```

we'd get an error because x and y are now *out of scope* - they only are *usable* within the **let** statement in which they're declared.

White space

If you notice above, the **let** statement is written across **multiple lines** - this is not a problem, as white space can be liberally applied to help clarify and organize the code within a script. (However, if you're working in Script-Fu's Console window, you'll need to enter everything on one line.)

Assigning a new value to a variable

Once you've initialized a variable, you'll more than likely need to change its value later on in the script. Use the **set!** statement for both local and global variables to change the variable's value:

```
(set! gCount 15)
(let ( (theNum 10) ) (set! theNum (+ gCount theNum) ) (set! gCount
theNum) )
(* gCount gCount)
```

Try to guess what the above statements will do, then go ahead and enter them in the Script-Fu Console window.

Functions

Now that you've got the hang of variables, let's get to work with some **functions**.

You declare a function with the following syntax:

```
define (TheFunctionName param1 param2) (...) (...) )
```

where `TheFunctionName` is the function's name, and any and all param names follow it. Notice that the parameter's don't have any *types* - Scheme is a type-less language.

The **(...)** characters represent the function's **code**. Thus, to find the square of a number, we could write the following function:

```
(define (square inNumber) (* inNumber inNumber) )
```

If you type this into Script-Fu's Console window, you'll get a message about **Closure**, then you'll be ready to use the function:

```
(square 5)
```

Typing this in will yield the expected result.

But how do you know what will be returned? Basically, the result of the last statement executed within the function is the result it returns.

LESSON 3: '(LISTS LISTS AND MORE LISTS)

We've trained you in variables and functions, young Script-Fu'er, and now we must enter the murky swamps of Scheme's **lists**. Are you ready for the challenge?

Defining a list

Before we talk more about lists, it is necessary that you know the difference between **atomic values** and **lists**.

You've already seen **atomic values** when we initialized variables in the previous lesson. An atomic value is *a single value*. So, for example, we can assign the variable **x** the single value of *8* in the following statement:

```
(set! x 8)
```

Try typing both statements into the Script-Fu Console and notice how it replies. When you type the first statement in, it simply replies with the result:

```
8
```

However, when you type in the other statement, it replies with the following result:

```
(1 3 5)
```

When it replies with the value *8* it is informing you that **x** contains the atomic value *8*. However, when it replies with (1 3 5), it is then informing you that **x** no longer contains a single value, but a list of values. Notice that there are no *commas* in our declaration or assignment of the list, nor in the printed result.

The syntax to define a list is:

```
'(a b c)
```

where a, b, and c are **literals**. We use the **apostrophe** (') to indicate that what follows in the parentheses is a list of literal values, rather than a function.

An empty list can be defined as such:

```
'()
```

or simply:

```
()
```

Lists can contain atomic values, as well as other lists:

```
(set! x '("The GIMP" (1 2 3) ("is" ("great" ()) ) ) )
```

Notice that after the first apostrophe, we no longer need to use an apostrophe when defining the inner lists. Go ahead and copy the statement into the Script-Fu Console and see what it returns.

You should notice that the result returned is not a list of single, atomic values - rather, it is a list of a literal ("The GIMP"), a list of the values (1 2 3), etc.

Concatenating variables to lists

To **concatenate** variables to a list, use the **cons** function:

```
(cons x (cons y () ) )
```

This concatenates the empty list to the variable **y**, then concatenates that list to the variable **x**.

Accessing values in a list

To access the values in a list, we use the functions **car** and **cdr**, which return the first element of the list, and the rest of the list, respectively.

car

car returns the *first element* of the **list** (also known as the **head** of the list). The list needs to be non-null. Thus, the following returns the first element of the list:

```
(car '("first" 2 "third"))
```

(which returns:)

```
"first"
```

cdr

cdr returns the *rest of the list* after the first element (also known as the **tail** of the list). If there is only one element in the list, it returns an empty list.

```
(cdr '("first" 2 "third"))
```

returns:

```
(2 "third")
```

while the following:

```
(cdr '("one and only"))
```

returns:

```
()
```

Accessing other elements a list

OK, great, we can get the first element in a list, as well as the rest of the list, but how do we access the second, third, or other elements of a list? Well, there exist several *convenience* functions to access, for example, the head of the head of the tail of a list (**caadr**), the tail of the tail of a list (**cddr**), etc.

The basic naming convention is easy - the **a**'s and **d**'s represent the **heads** and **tails** of lists, so

```
(car (cdr (car x) ) )
```

could be written as:

```
(cadar x)
```

To view a full list of the list functions, refer to the **SIOD** home page (or appendix D) which lists the available functions for the version of Scheme used by Script-Fu.

To get some practice with list accessing functions, try typing in the following and using different variations of car and cdr to access the different elements of the list:

```
(set! x '( (1 2 (3 4 5) 6) 7 8 (9 10) ) )
```

Try accessing the number *3* in the list using only two function calls. If you can do that, you're on your way to becoming a Script-Fu Master!

LESSON 4: YOUR FIRST SCRIPT-FU SCRIPT

Do you not need to stop and catch your breath, little grasshopper? No? Well then, let's proceed with the 4th lesson in your training - your first Script-Fu Script.

Creating a text box script

One of the most common operations I perform in Gimp is creating a box with some text in it for a web page, a logo, or whatever. However, you never quite know how big to make the initial image when you start out - you don't know how much space the text will fill with the font and font size you want.

The Script-Fu Master (and student) will quickly realize that this problem can easily be solved and automated with Script-Fu.

We will therefore create a script, called **Text Box**, which creates an image correctly sized to fit snug around a line of text the user inputs. We'll also let the user choose the **font**, **font size**, and **text color**.

GETTING STARTED

Editing and storing your scripts

Up until now, we've been working in the **Script-Fu Console**. Now, however, we're going to switch to editing script text files.

Where you place your scripts is a matter of preference - if you have access to Gimp's default script directory, you can place your scripts there. However, I prefer keeping my own personal scripts in my own script directory, to keep them separate from the *factory-installed* scripts.

I copied the script to `~/.gimp/scripts` which is the my personal script directory (made by Gimp when it was installed).

Now, whenever Gimp starts up, or I refresh the Script-Fu database (`Xtns->Script-Fu->Refresh`), it will add my personal scripts to the procedural database.

The bare essentials

Every Script-Fu script defines at least one **function**, which is the script's main function. This is where you do the work.

Every script must also *register* with the **procedural database**, so you can access it within Gimp.

We'll define the main function first:

```
(define (script-fu-text-box inText inFont inFontSize inTextColor))
```

Here we've defined a new function called **script-fu-text-box** which takes four parameters, which will later correspond to some text, a **font**, the font **size**, and the text's **color**. The function is currently empty and thus does nothing. So far, so good - nothing new, nothing fancy.

Naming conventions

Scheme's naming conventions seem to prefer **lowercase** letters with **hyphens**, which I've followed in the naming of the function. However, I've departed from the convention with the parameters - I like more descriptive names for my parameters and variables, and thus add the **in** prefix to the parameters so I can quickly see that they're values passed into the script, rather than created within it. I use the prefix **the** for variables defined within the script.

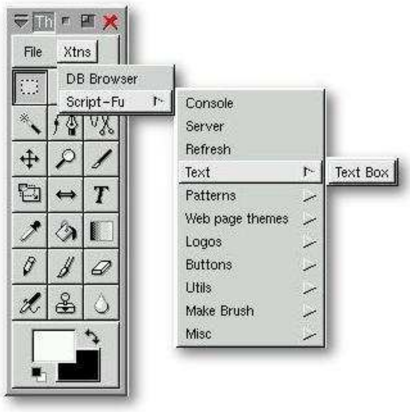
It's Gimp convention to name your script functions **script-fu-abc**, because then when they're listed in the procedural database, they'll all show up under **script-fu** when you're listing the functions. This also helps distinguish them from plug-ins when listed.

Registering the function

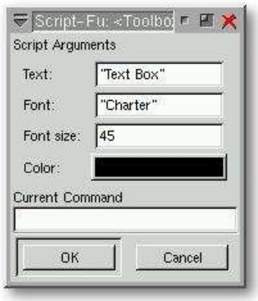
Now, let's **register** the function with Gimp - when Gimp reads in a script, it will search for this function and use it to register the script with the procedural database. These lines of code come after the function definition listed above.

```
(script-fu-register
   "script-fu-text-box"
   "<Toolbox>/Xtns/Script-Fu/Text/Text Box"
   ”Creates a simple text box, sized to fit around the user’s
choice of text, font, font size, and color.”
   ”Michael Terry”
   ”copyright 1997, Michael Terry”
   ”October 27, 1997”
   ””
   SF-VALUE ”Text:”       ”\”Text Box\””
   SF-VALUE ”Font:”       ”\”Charter\””
   SF-VALUE ”Font size:”  ”45”
   SF-COLOR ”Color:”      ’(0 0 0)
)
```

If you save these functions in a text file with a **.scm** suffix in your script directory, then choose `Xtns->Script-Fu->Refresh`, this new script will appear as `Xtns->Script-Fu->Text->Text Box`:

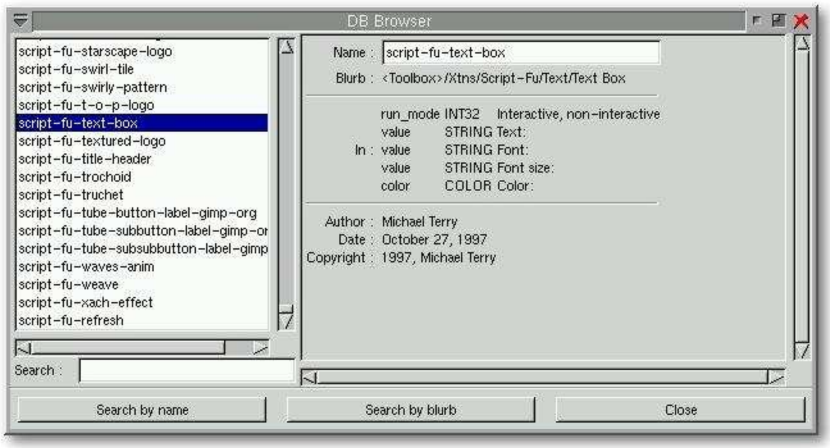

File
Xtns
DB Browser
Script-Fu
Console
Server
Refresh
Text
Text Box
Patterns
Web page themes
Logos
Buttons
Utils
Make Brush
Misc

If you *invoke* this new script, it won't do anything, of course, but you can view the **prompts** you created when registering the script (more information about what we did is covered next).



Text: "Text Box"
Font: "Charter"
Font size: 45
Color:

Finally, if you invoke the **DB Browser** (the procedural database browser - `Xtns->DB Browser`), you'll notice that our script now appears in the database:



script-fu-starscape-logo
script-fu-swirl-tile
script-fu-swirly-pattern
script-fu-t-o-p-logo
script-fu-text-box
script-fu-textured-logo
script-fu-title-header
script-fu-trochoid
script-fu-truchet
script-fu-tube-button-label-gimp-org
script-fu-tube-subbutton-label-gimp-or
script-fu-tube-subsubbutton-label-gimp
script-fu-waves-anim
script-fu-weave
script-fu-xach-effect
script-fu-refresh

```
     Name : script-fu-text-box
    Blurb : <Toolbox>/Xtns/Script-Fu/Text/Text Box

            run_mode INT32    Interactive, non-interactive
            value    STRING Text:
       In : value    STRING Font:
            value    STRING Font size:
            color    COLOR Color:

   Author : Michael Terry
     Date : October 27, 1997
Copyright : 1997, Michael Terry
```

Steps for registering the script

To register our script with Gimp, we call the function **script-fu-register**, fill in the 7 required parameters, and add our scripts' own parameters, along with a description and default value for each parameter.

The required parameters

- The **name** of the function we defined above. This is the function called when our script is invoked (i.e., the entry-point into our script). This is necessary because we may define additional functions within the same file, and Gimp needs to know which of these functions to call.In our example, we only defined one function, **text-box**, which we registered.

- The **location** in the menu where the script will be inserted. The exact location of the script is specified like a **path** in Unix, with the root of the path being either **<Toolbox>** or **<Image>**. If your script does not operate on an existing image (and thus creates a new image, like our Text Box script will), you'll want to insert it in the **<Toolbox>** menu - this is the menu in Gimp's **main window** (where all the *tools* are located - the selection tools, magnifying glass, etc.). If your script is intended to work on an image being **edited**, you'll want to insert it in the **<Image>** menu - this menu appears when your `right-click` on an open image. The rest of the *path* to the menu lists menus and sub-menus. Thus, we registered our **Text Box** script in the **Text** menu of the **Script-Fu** menu of the **Xtns** menu of the **Toolbox** (`Toolbox->Xtns->Script-Fu->Text->Text Box`). If you notice, the **Text** sub-menu in the **Script-Fu** menu wasn't there when we began - Gimp automatically creates any menus not already existing.
- A **description** of your script. I'm not quite sure where this is displayed.
- Your **name** (the author of the script).
- **Copyright** information.
- The **date** the script was made, or the last **revision** of the script.
- The **types** of images the script works on. This may be any of the types: RGB, RGBA, GRAY, GRAYA, INDEXED, INDEXEDA, or it may be none at all - in our case, we're creating an image, and thus don't need to define the type of image on which we work.

Registering the script's parameters

Once we have **listed** the required **parameters**, we then need to list the parameters which correspond to the parameters our script needs. When we list these params, we give *hints* as to what their types are - this is for the dialog box which pops up when the user selects our script. We also provide a **default** value.

This section of the registration process has the following format:

```
Param-type ”Prompt text” ”default value”
```

The different parameter types, plus examples, are listed below:

Param type	**Description**	**Examples**
SF-VALUE	Accepts numbers and strings.Note that quotes must be escaped for default text	SF-VALUE ”Text:” ”\”Some text\”” SF-VALUE ”A number:” ”34”
SF-COLOR	Indicates that a color is requested in this parameter	SF-COLOR ”Color:” ’(0 0 0)
SF-TOGGLE	A checkbox is displayed, to get boolean value	SF-TOGGLE ”Resize?” TRUE

Param type	**Description**	**Examples**
SF-IMAGE	If your script operates on an open image, this should be the first parameter after the required parameters. The GIMP will pass n a reference to the image in this parameter.	SF-IMAGE ”The image” 0
SF-DRAWABLE	If your script operates on an open image, this should be the second parameter after the SF-IMAGE param. It refers to the active layer. The GIMP will pass in a reference to the active layer in this param.	SF-DRAWABLE ”The layer” 0

Now, young student, this was a lot of information, so take a break.

LESSON 5: GIVING OUR SCRIPT SOME GUTS

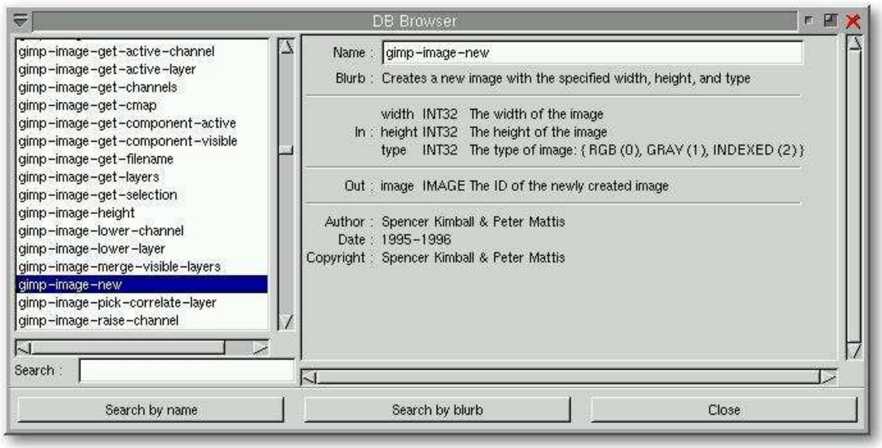
You show great dedication to your studies, my student. Let us thus continue with your training and add some functionality to our script.

CREATING A NEW IMAGE

In the previous lesson, we created an empty function and registered it with Gimp. In this lesson, we want to provide **functionality** to our script - we want to create a new image, add the user's text to it, and resize the image to fit the text exactly.

Once you know how to **set variables**, **define functions**, and access list members, the rest is all downhill - all you need to do is familiarize yourself with the functions available in Gimp's procedural database and call those functions directly. So fire up the DB Browser and let's get cookin'!

Let's begin by making a new image. We'll create a new variable, **theImage**, set to the result of calling Gimp's built-in function `gimp-image-new`:



gimp-image-get-active-channel
gimp-image-get-active-layer
gimp-image-get-channels
gimp-image-get-cmap
gimp-image-get-component-active
gimp-image-get-component-visible
gimp-image-get-filename
gimp-image-get-layers
gimp-image-get-selection
gimp-image-height
gimp-image-lower-channel
gimp-image-lower-layer
gimp-image-merge-visible-layers
gimp-image-new
gimp-image-pick-correlate-layer
gimp-image-raise-channel

Name : gimp-image-new
Blurb : Creates a new image with the specified width, height, and type

In :
width INT32 The width of the image
height INT32 The height of the image
type INT32 The type of image: { RGB (0), GRAY (1), INDEXED (2) }

Out : image IMAGE The ID of the newly created image

Author : Spencer Kimball & Peter Mattis
Date : 1995-1996
Copyright : Spencer Kimball & Peter Mattis

As you can see from the **DB Browser**, the function `gimp-image-new` takes three parameters - the image's **width**, **height**, and the **type** of image. Since we'll later **resize** the image to fit the text, we'll make a 10x10 RGB image. We'll store the image's width and sizes in some variables, too, as we'll refer to and manipulate them later in the script:

```
(define (script-fu-text-box inText in(set! theImageWidth 10)
   (set! theImageHeight 10)
   (set! theImage (car (gimp-image-new theImageWidth theImage-
Height RGB) ) )
    )Font inFontSize inTextColor)
)
```

You should notice that we used the value **RGB** to specify that the image is an RGB image. We could have also used **0**, but **RGB** is more descriptive when we glance at the code.

You should also notice that we took the **head** of the result of the function call - this may seem strange, because the database explicitly tells us that it returns only one value - the ID of the newly created image. However, all Gimp functions return a list, even if there is only one element in the list, so we need to get the head of the list.

Adding a new layer to the image

Now that we have an image, we need to add a layer to it. We'll call the `gimp-layer-new` function to create the layer, passing in the ID of the image we just created. (From now on, instead of listing the complete function, we'll only list the lines we're adding to it.)

```
(set! theLayer (car (gimp-layer-new theImage theImageWidth theIm-
ageHeight RGB_IMAGE ″layer 1″ 100 NORMAL) ) )
```

Once we have the new layer, we need to **add** it to the image:

```
(gimp-image-add-layer theImage theLayer 0)
```

Now just for fun, let’s see the fruit of our labors up until this point, and add this line to show the new, empty, image:

```
(gimp-display-new theImage)
```

Save your work, select `Xtns->Script-Fu->Refresh`, run the script, and a new image should pop up. It will probably contain garbage (random colors), because we haven’t erased it. We’ll get to that in a second.

Adding the text

Go ahead and **remove** the line to display the image (or comment it out with a **;** as the first character of the line).

Before we add text to the image, we need to set the **background** and **foreground** colors so that the text appears in the color the user specified. We’ll use the `gimp-palette-set-back/foreground` functions:

```
(gimp-palette-set-background ′(255 255 255) )
(gimp-palette-set-foreground inTextColor)
```

With the colors properly set, let’s now clean out the garbage currently in the image. We’ll select everything in the image, and call **clear**:

```
(gimp-selection-all theImage)
(gimp-edit-clear theImage theLayer)
(gimp-selection-none theImage)
```

With the image cleared, we’re ready to add some **text**:

```
(set! theText (car (gimp-text theImage theLayer 0 0 inText 0 TRUE
inFontSize PIXELS ″*″ inFont ″*″ ″*″ ″*″ ″*″)))
```

While a long function call, it’s fairly straight-forward if you go over the parameters while looking at the function’s entry in the DB Browser. Basically, we’re creating a new text layer and assigning it to the variable **theText**.

Now that we have the text, we can grab its **width** and **height** and resize the image and the image’s layer to the text’s size:

```
(set! theImageWidth  (car (gimp-drawable-width theText) ) )
(set! theImageHeight (car (gimp-drawable-height theText) ) )
```

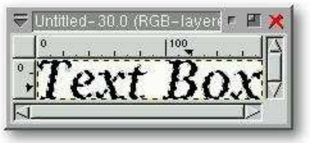
```
(gimp-image-resize theImage theImageWidth theImageHeight 0 0)
(gimp-layer-resize theLayer theImageWidth theImageHeight 0 0)
```

If you're like me, you're probably wondering what a **drawable** is when compared to a layer - A drawable is the *paintable* area in an image, like a **layer**, a **background**, or a **selection** (floating or in a layer).

With the image ready to go, we can now re-add our **display** line:

```
(gimp-display-new theImage)
```

Save your work, refresh the database, and give your first script a run! You should get something like the following:



Clearing the "dirty" flag

If you try and close the image created without first **saving** the file, Gimp will ask you if you want to save your work before you close the image. It asks this because the image is marked as *dirty*, or unsaved. In the case of our script, this is a nuisance for the times when we simply give it a test run and don't add or change anything in the resulting image - that is, our work is easily reproducible in such a simple script, so it makes sense to get rid of this *dirty* flag.

To do this, we can clear the dirty flag after displaying the image:

```
(gimp-image-clean-all theImage)
```

This will dirty count to 0, making it appear to be a "clean" image.

Whether to add this line or not is a matter of personal taste - I use it in scripts that produce new images, where the results are trivial, as in this case. If your script is very complicated, or if it works on an existing image, you will probably not want to use this function.

Enabling and Disabling undo

If your script works on an image, you'll probably want to call `gimp-image-disable-undo` at the beginning of the script, and `gimp-image-enable-undo` at the end of the script - these functions turn **undo-recording** off and on, respectively. If you are writing a complex script, a user will not want to have to hit undo a million times after invoking your script, to undo all the actions your script took to do its job.

LESSON 6: EXTENDING THE TEXT BOX SCRIPT

Your will and determination are unstoppable, my eager student. So let us continue your training.

The game plan

Now that we have a very handy-dandy script to create text boxes, let's add two features to it:

- Currently, the image is resized to fit exactly around the text - there's no room for anything, like drop-shadows, or special effects (even though many scripts will automatically resize the image as necessary). Let's add a **buffer** around the text, and even let the user specify how much buffer to add as a percentage of the size of the resultant text.
- This script could easily be used in other scripts that work with text - let's extend it so that it returns the image and the layers, so that other scripts can call this script and use the image and layers we create.

Modifying the parameters and the registration function

To let the user specify the amount of buffer, we'll add a parameter to our function and the registration function:

```
(define (script-fu-text-box inText inFont inFontSize inTextColor
inBufferAmount)
...
)
(script-fu-register
   "script-fu-text-box"
   "<Toolbox>/Xtns/Script-Fu/Text/Text Box"
   "Creates a simple text box, sized to fit around the user's
choice of text, font, font size, and color."
   "Michael Terry"
   "copyright 1997, Michael Terry"
   "October 27, 1997"
   ""
   SF-VALUE "Text:"       "\"Text Box\""
   SF-VALUE "Font:"       "\"Charter\""
   SF-VALUE "Font size:"  "45"
   SF-COLOR "Color:"      '(0 0 0)
   SF-VALUE "Buffer amount (0 - 100% height of text):" "35"
)
```

Adding the new code

We’re going to add **code** in two places - right before we **resize** the image, and at the **end** of the script (to return the new image, the layer, and the text).

After we get the text’s height and width, we need to **resize** these values based on the buffer amount specified by the user. We won’t do any error checking to make sure it’s in the range of 0-100% because it’s not life-threatening, and because there’s no reason why the user can’t enter a value like ”200” as the percent of buffer to add.

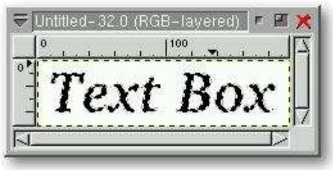
```
(set! theBuffer (* theImageHeight (/ inBufferAmount 100) ) )
(set! theImageHeight (+ theImageHeight theBuffer theBuffer) )
(set! theImageWidth  (+ theImageWidth theBuffer theBuffer) )
```

All we’re doing here is setting the buffer based on the height of the text, and adding it twice to both the height and width of our new image. (We add it twice to both dimensions because the buffer needs to be added to both sides of the text.)

Now that we have resized the image to allow for a buffer, we need to **center** the text within the image. This is done by moving it to the (**x, y**) coordinates of (`theBuffer, theBuffer`). I added this line after resizing the layer and the image:

```
(gimp-layer-set-offsets theText theBuffer theBuffer)
```

Go ahead and save your script, and try it out after refreshing the database. You should now get a window like the following:



All that is left to do is to return our image, the layer, and the text layer. After displaying the image, we add this line:

```
(cons theImage (cons theLayer (cons theText () ) ) )
```

We use the function **cons** to create a list of values. This is the last line of the function, making this list available to other scripts that want to use it.

To use our new text box script in another script, we could write something like the following:

```
(set! theResult (script-fu-text-box ”Some text” ”Charter” ”30” ’(0
0 0) ”35”) )
```

```
     (gimp-image-flatten (car theResult) )
```

Congratulations, my student, you are on your way to your Black Belt of Script-Fu!

part IX

Adv installations

- *How to provide fonts to Gimp*
- *To make or not to make, plug ins with Gimp*

chapter 40

How to get fonts to Gimp

In this chapter you'll find some basic information about how to make fonts available to Gimp in Xwindow.

HOW FONTS WORK IN GIMP

All of the fonts you use in Gimp come from the **X server**. There is no internal font render in Gimp, so Gimp needs X to render the fonts. This is also the reason why the text tool dialog looks a bit like the **xfontsel** program in X.

SCALABLE FONTS

First of all, use **scalable** fonts (Type 1 or Speedo) in Gimp. As you can tell from the name, these fonts can be **scaled** up and down without losing in quality. There are also **bitmapped** fonts in X. When you scale a bitmap font, it will lose its shape and get very *jaggy* and ugly.

WHERE ARE THE FONTS AND FONT PATH

To find out where in your system X keeps its fonts, type: `xset -q <enter>` in a shell. The last line in the output tells your **font path**. If you are an XFree86 user, and use the default XF86 configuration, you have to make some editing to make sure that the **Type 1** and **Speedo** font are the first to come up when X is looking for a certain font. It's very simple, just place the Speedo and Type 1 directories first in your `Fontpath`.

An other way is to add a string `:unscaled` after your bitmap fonts. If you add this string the scalable i.e the type 1 fonts will be used if you need to scale up or down your font. The font path in the XF86Config file may the look like this.(Note this may only work with XFree 86)

```
Section ″Files″
   RgbPath    ″/usr/X11R6/lib/X11/rgb″
   FontPath   ″/usr/X11R6/lib/X11/fonts/misc:unscaled″
   FontPath   ″/usr/X11R6/lib/X11/fonts/75dpi:unscaled″
   FontPath   ″/usr/X11R6/lib/X11/fonts/100dpi:unscaled″
   FontPath   ″/usr/X11R6/lib/X11/fonts/MY_NEW_FONTS″
   FontPath   ″/usr/X11R6/lib/X11/fonts/Type1″
   FontPath   ″/usr/X11R6/lib/X11/fonts/Speedo″
   FontPath   ″/usr/X11R6/lib/X11/fonts/misc″
   FontPath   ″/usr/X11R6/lib/X11/fonts/75dpi″
   FontPath   ″/usr/X11R6/lib/X11/fonts/100dpi″
EndSection
```

If you can stand that your fonts in Xwindow will look a bit odd, i.e when an unscaled font would normally be used, instead a scalable type 1 font will be used the set all your type 1 fonts before the bitmap fonts.

To find out what kind of **Type 1** fonts you have; change directory to the Type 1 dir, and read the `fonts.dir` file (e.g `cd /usr/X11/lib/X11/fonts/Type1 && more fonts.dir`) in a terminal window and all your available Type 1 fonts will be displayed. One of the most common reasons for a Script-Fu to fail, is that the specified Script-Fu font isn't available. This makes the script *bug* and there will be no output.

INSTALLING FONTS

TYPE 1 FONTS INSTALLATION & THE TYPE1INST PROGRAM

We will only cover Type 1 fonts (*postscript fonts*) other font types are beyond the scope of this chapter. The first thing you have to do is to download a program called `type1inst` (written by James Macnicol) from `ftp://sunsite.unc.edu/pub/Linux/X11/xutils` or one of its mirrors.

Preparing for installation

At the time of this writing, the file is called `type1inst-0.6.tar.gz.` Unpack the program like you did with the Gimp archive (see chapter 4). Copy the files `type1inst` and `t1embed` to a directory in your PATH, for example `/usr/local/bin`.

To see your PATH, execute the following in a shell: `echo $PATH`. If you aren't the system administrator, then you probably can't install files in system directories. To overcome this problem, create a **bin** directory in your home directory and install the `type1inst` program there. To include the program in your PATH, do the following if you run **Bash**, **Sh** or **Ksh** as your shell:

```
export PATH=$PATH:$HOME/bin
```

Copying the fonts to the fonts dir

Now copy The Type 1 fonts that you want to install in a directory of your own choice. A word of advise: If you have a large font collection, and you want easy font management in Gimp, install an **xfontserver**, or install the fonts in different directories with, say 20 fonts in each.

Font management

Here's how we manage fonts at Frozenriver: Karin needs a lot of fonts, but if you load all fonts into X and Gimp, it becomes hard to choose and find the right font. We have solved this by making *different directories for each type of font*, e.g we have one directory for artistic fonts and one for strict business fonts and so on. This makes it easy for Karin to load and unload the fonts depending on what she wants to do.

Running type1inst

When you have copied the fonts, `cd` to the directory where you installed the fonts, and run `type1inst`. This will create the necessary `font.dir` and `font.scale` file in this directory.

Loading the fonts into X

Now you can load the fonts into X by applying `xset +fp full path to the font directory`. The fonts are now installed and you can use them in Gimp, but if you installed them while you had Gimp running, you must **restart** Gimp.

Please notice that the **+** is placed in front of **fp**. If you put it in the wrong place, the all the default font directories will be queried of the font name first, and if there is a font with the same name as yours this font will be used instead of your own.

The *Helvetica* font is a good example. The standard Helvetica font that comes with X is of quite poor quality, but if you download Acrobat Reader from Adobe, it will also include some new fonts including a very nice Helvetica font.

If you install these fonts with `xset fp+` you will never see them, because the standard Helvetica font that comes with X, is placed in front of the font you installed when X is looking for its font directories. So make sure that you do it right, otherwise you will be wasting valuable fonts (and maybe money too).

INSTALLING TYPE 1 FONTS BY HAND

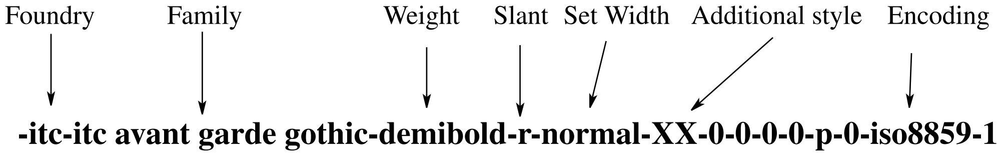
It's not always possible to use the `type1inst` program. In that case, you'll have to do it the hard way and install the `fonts.*` files by hand.

The font file

Generally you have to load the fontfile to a **text editor** and look in the file **header** for name, type etc.... The header of the file ends with the `eexec` command, and everything after that is binary font data and of little interest. Note that you can only do this with ps font files. Such files normally end with `.pfb` and have `%!PS-AdobeFont` typed in the first line of the file.

THE FONT FIELD IN THE FONT FILE

The first thing you need to understand is how to code the font *field* in the `font.scale` file. Here is an example line:


Foundry
Family
Weight
Slant
Set Width
Additional style
Encoding
-itc-itc avant garde gothic-demibold-r-normal-XX-0-0-0-0-p-0-iso8859-1

Extracting data

All you have to do now is to load the `pfb` in a file editor (like `vi file.pfb`) in a terminal such as **xterm**, and search for data. Here's what to look for:

- **Foundry**: The registered **name** of the font foundry - usually a company. This was written in the ps font file: "usage: 24954 31846%% ITC Avant Garde Gothic is a registered trademark of International Typeface" This means that the font will use **itc** as Foundry.
- **Family**: The font **family** the font belongs to. There are a usually a lot of font files in a font family, all with different characteristics such as **bold**, **thin**, **condensed** etc. Here is what was in the font file readonly: "def/FamilyName (ITC Avant Garde Gothic)" which will be **itc avant garde gothic** in the family field.
- **Weight**: The **weight** of the font, for example **medium**, **bold**, **thin** etc. Here is what was in the ps file "readonly def/Weight (Demi)" which will put **demibold** in the weight field.
- **Slant**: The **posture** of the of the font. If there is no slant info for the font then it's **Roman** or *upright* posture, and will have an **r** in the slant field. If there is info like *Gothic Demi Oblique* the slant is **oblique**, and it will have an **o** in the slant filed
- **Set Width**: This is the **horizontal** width. Here you have to search the font name. **Condensed** is an example of a width.
- **Additional style**: This is a seldome used option for additional styles, it's mostly never used in ps fonts
- **Encoding**: This is what type of langs the font supports. Here's an example from the ps file "def/Encoding StandardEncoding" which will end up with iso8859-1, for example isolatin1 in the encoding field
- The rest is the same as in the example line. It's the standard ps font line.

We know that this is probably not the best description of font decoding, but if we were to describe it in depth we could make a book out of it.

In the end of this chapter you'll find some tables that can help you with your decoding. They are taken from the source of the type1inst program.

Generally, you don't have to worry, because type1inst will make the job for you. But sometimes it will not get the Foundry information, and then you can take a look in the font file or the log file which type1inst creates. Most of the time, it's quite easy to figure out what's missing.

TABLES

FOUNDERY TABLE

Companie	**Foundery**
Adobe	adobe
Publishers Paradise	paradise
Bigelow & Holmes	b&h
Bitstream	bitstream
International Typeface Corporation	itc
IBM	ibm
LETRASET	letraset
Monotype Corporation	monotype
SoftMaker	softmaker
URW	urw
Jonathan Brecher	brecher
Brendel Informatik	brendel
A. Carr	carr
FontBank	fontbank
Hershey	hershey
A.S.Meit	meit
Andrew s. Meit	meit
S.G. Moye	moye
D. Rakowski	rakowski
David Rakowski	rakowski
Reasonable Solutions	reasonable
Southern Software	southern
Title Wave	titlewave
ZSoft	zsoft

WEIGHT TABLE

Weights	**Weight**
book	book
demibold	demibold
semibold	demibold
demi	demibold
semi	demibold
extrabold	extrabold
boldface	bold
bold	bold
heavyface	heavyface
heavy	heavy
ultrablack	ultrablack
extrablack	extrablack
ultra	ultra
black	black
extralight	extralight
light	light
thin	thin
super	super
normal	medium
regular	regular
roman	regular

SLANT TABLE

Slants	**Slant**
italic	i
roman	r
regular	r
cursive	i
kursiv	i
oblique	o

Slants	Slant
obl	o
slanted	o
upright	r
inclined	i

SET WIDTH TABLE

Widths	Set Width
extracondensed	extracondensed
condensed	condensed
cond	condensed
sans	sans
wide	wide
cn	condensed
narrow	narrow
extracompresse	extracompresse
compressed	compressed
extraextended	extraextended
extended	extended
expanded	expanded
normal	normal

ADDITIONAL STYLE TABLE

Styles	Style
alt	alternate
beginning	beginning
display	display
dfr	dfr
ending	ending
ep	expert
exp	expert

Styles	**Style**
ornaments	ornaments
osf	oldstylefigures
outline	outline
sc	smallcaps
shaded	shaded
shadowed	shadowed
stencil	stencil
swash	swash
sw	swash
one	one
two	two
three	three
four	four
a	alternate

chapter 41

Compiling plug-ins

In this chapter will we try to explain how to compile plug-ins so they can be used in Gimp.

WHAT IS A PLUG-IN?

A plug-in is a program that can't run on its own, it needs to be started by Gimp.

COMPILE?

Most of the time, plug-ins are distributed in the **source code**. All programs have a source code (if they aren't written directly in machine code). The source code is written in a language humans can understand, like C, C++ etc...

The computer doesn't understand C or C++, so the code needs to be translated to machine code which the computer understands and can execute. This translation is called **compiling**.

WHAT WAY TO GO WHEN YOU WANT TO COMPILE

There are three common ways to compile; **Make**, **Configure** and **Plain cc**. This is not totally correct, but it's good enough for now. You can also use gimp-tool to build and install a plugin if it's only one .c file see appendix B.

Make is a program which takes a specification file, commonly called **Makefile**, and by running this file it *compiles* the program with the help of the **compiler**.

Configure is a program that generates a **Makefile**. After doing this you will have to run **Make** in order to compile.

The most simple way is to use the compiler directly to compile the code.

We will try to explain how to handle these three types of compiling. But before we start, we must tell you that we can't cover every angle or error that can arise when you compile. We will cover the basics, but not extras like special compiler flags, debugging etc.

We really want you to try to compile a plug-in even if you aren't a programmer or hacker. If you have a friend who is a Unix programmer, we still think you should read this chapter and try to compile, and if it goes totally wrong anyway, call in your mate.

HOW TO OBTAIN AND INSTALL THE SOURCE CODE

Most of the time, you will go to the **plug-in registry** or the `Gimp.org ftp` site to obtain/download the source code. Other sources of information are **Gimp News** and the Gimp developer / Gimp user **mailing lists**.

UNPACKING THE SOURCE CODE

When you have **downloaded** the source code, you must **unpack** it. Usually, the code ends with `xxx.c.gz`, `xxx.tgz` or `xxx.tar.gz`. The `xxx.c.gz` can be decompressed with `gunzip xxx.c.gz`, and the `xxx.tgz` and `xxx.tar.gz` archives can be unpacked with `gunzip xxx.tgz && tar xvf xxx.tar`. Before you unpack the source, do the wise thing and move it to a new directory. Then unpack and build the plug-in. For example write: `mkdir build; mv xxx.tgz build; gunzip xxx.tgz && tar xvf xxx.tar` in an Xterm window.

If the source ends with `xxx.c`, then everything's fine. You don't need to decompress it, because it has already been done. Just save it if you use **Netscape**. One thing to remember if you use Netscape is that Netscape often tries to **decompress** the archive, and it may also fail to do this. If this is the case, check the source file `xxx.c` with a file viewer like **More** (for example `more xxx.c`). If you can't read what's in the file, then it's not decompressed. To fix it, write: `mv xxx.c xxx.c.gz && gunzip xxx.c.gz`.

COMPILING THE CODE

We will use **GNU** tools to compile the code. If you are using a system that doesn't have GNU tools installed, then you have two options: either download and install GNU tools (beyond the scope of this book) or use the system's own tools.

There is no big difference between GNU and system tools. Some of the GNU tools can be recognized by the fact that their names start with the letter **g**. For example, the GNU compiler is called **gcc** and the system's own compiler is called **cc**. We think that this won't make any difference here, but we can't be sure because there are a lot of different Unix systems out there, and we cant cover them all.

FINDING OUT HOW TO COMPILE THE PLUG-IN

When you have downloaded and unpacked the plug-in source, you have to find out how to compile it. The plug-in are often packed with several other files. Sometimes there is a `README` or `INSTALL` file. If you find one of these files, always read it (write `more README`).

Most of the time, you will find information on how to compile and install the plug-in in one of these files. You can also take a look in the **head** of the **C code file**. If there is no information, then you have to follow our examples here, and even if you do find some information, you are advised to read on.

The `README` file will also hopefully tell you whether you need some **additional libraries** to compile the plug-in. For example, you may need the `libtif` library and header files in order to compile the plug-in. If you don't have these libraries or header files, the building of the plug-in will fail.

You can most likely find out if you have this libraries, or other required programming by reading the chapter about installing the source distribution of Gimp in Chapter 4.

Compiling a library is a bit different than compiling a plug-in., but the start is basically the same, so keep on reading.

If the plug-in you've downloaded is a single file, or if the archive you just unpacked has no Makefile (it can be named Makefile, Makefile.classic, MAKEFILE etc.) or configure file, then you have to compile it directly with **gcc** or you have to create a Makefile or configure script.

If there is a Makefile then you can use make to compile the plug-in. But before you begin, you have to check and perhaps edit the Makefile to find out whether it is correct for your system.

If there is a **configure** script, then use it to generate a Makefile

USING GCC TO COMPILE THE PLUG-IN STRAIGHT OFF

A first try

If the file is named `plugin.c`, then compile it with `gcc -o plug-in plugin.c`. The `-o` flag tells the compiler that the final output (the plug-in) should be named **plug-in**. If your C-compiler (GCC) is configured to search the standard directories for the include and library files, and those files are indeed in the standard directories, then the plug-in will compile cleanly. But most of the time this is not the case since you probably will need specific Gimp libraries and include files.

Here is an example of what happens if you compile the waterselect.c file this way: (only some of the lines)

```
/tmp/cca017741.o(.text+0xc): undefined reference to `gimp_main'

/tmp/cca017741.o(.text+0x87): undefined reference to
`gdk_input_list_devices'

/tmp/cca017741.o(.text+0xc5): undefined reference to
`g_strcasecmp'

/tmp/cca017741.o(.text+0x626): undefined reference to
`gtk_preview_get_type'

/tmp/cca017741.o(.text+0x632): undefined reference to
`gtk_object_check_cast'

/tmp/cca017741.o(.text+0x63d): undefined reference to
`gtk_preview_draw_row'

/tmp/cca017741.o: In function `update_buckets':

/tmp/cca017741.o(.text+0x682): undefined reference to
`gtk_widget_draw'

/tmp/cca017741.o: In function `update_gimp_color':

/tmp/cca017741.o(.text+0x6e2): undefined reference to
`gimp_palette_set_foreground
```

Libraries

This happens when the libraries needed aren't linked to the executable that you are trying build. A **library** is a set of **functions**, for example a function which creates a *window* or a *scrollbar.* The plug-in will call different functions in these libraries, like the function `gtk_object_check_cast`. That's why you have to tell the executable where to find these libraries so they can be called by the plug-in.

Another try

Now let's try to compile it with `gcc -o plug-in plugin.c -L/usr/local/lib -lgtk -lgimp`. Now the plug-in compiles cleanly on the system. We work on Linux, but if I had been on my Sun Solaris system, I would have to add `-I/usr/local/include` before even starting to think about libraries.

Include file

The error messages telling you that an include (header file) is missing goes like this:

```
waterselect.c:39: gtk/gtk.h: No such file or directory
```

What does this message really mean? It says that a file named `gtk.h` is missing, but this is not always the case. A more correct answer is that the compiler (more accurately the preprocessor compiler) can't find it in the directories it was configured to search.

We have to tell the compiler where to search for the file. We can do that by adding a `-I/where/to/search` Because the file is located in `/usr/local/include/gtk`, and the error was can't find `gtk/gth.h`, this tells us that the compile line will be:

```
gcc -o plug-in plugin.c -L/usr/local/lib -lgimp -lgtk -I/usr/
local/include
```

The `-I/usr/local/include` that we added tells the compiler to search that directory for include files, in our case `gtk/gtk.h`. The compiler will find the file because it will add `gtk/gtk.h` to `/usr/local/include`. This will give us `/usr/local/include/gtk/gtk.h` which is where the file is located.

The -L flag and how to find out which libs to link

The **-L** flag works the same way. It tells the compiler where to search for libraries, in this case the libraries `libgimp.so` and `libgtk.so`. The `-lgimp` is just a short cut so we don't need to write `libgimp.so`.

How do we know which libraries to add and which directories to include? We have to take a look inside the code to get a hint about what to add (generally located in the beginning of the file). Here's an example from the `waterselect.c` file.

```
#include <stdlib.h>

#include <stdio.h>
```

```
#include <math.h>
#include <gtk/gtk.h>
#include "libgimp/gimp.h"
```

This code tells us that the compiler will include five files when it compiles the code. The first three are ordinary standard C include files, and the other ones are special Gimp and gtk files.

The -I flag

The standard **C header files** are in located in a **standard include** directory, which the compiler will search. However, the `gtk.h` and `gimp.h` files aren't in a standard directory, so we need to tell the compiler (`-I flag`). If the `-I flag` is `/usr/local/include`, the compiler will try to get the `gtk.h` files in `/usr/local/include/gtk/gtk.h`.

The source code lines can also give us a hint on which libraries to link with; in this case `libgimp` and `libgtk` (the standard C library and Math library are linked by default on most systems)

What to do when there are several source files

Now we've hopefully learned how to compile a plug-in that is made up of a single C source code file. However, a plug-in can be contain more that one C source code file. How do we compile such plug-ins?. If the extra file is a **header file**, (named `plugin.h`) then we can compile the plug-in just as we did before. The header file is only a file which specifies functions, and how you should call these functions. We only have to look at the files (or just the header file) to find out what include directories to include, and which libraries we need to link.

What should you do if the plug-in is built up of several files? (like this plugin:`.h pluginmain.c pluginpart1.c pluginpart2.c` etc.) If this is the case, we will have to attack the compiling a bit differently. First, we need to compile each part separately, and then compile (or more exactly link) them to an executable plug-in.

What we need to do is to make **object** files out of each part. These object files will then be **linked** together, thus creating the final executable plug-in. We will give you an example:

```
gcc -c pluginmain.c -I/usr/local/include
gcc -c pluginpart1.c -I/usr/local/include
gcc -c pluginpart2.c -I/usr/local/include
gcc -o plug-in pluginmain.o pluginpart1.o pluginpart2.o -L/usr/
local/lib -lgtk -lgimp -lgimpui -lmpeg
```

The first three lines is where we make the object files. We tell the compiler to make object files by adding the `-c` flag. When we make object files we don't need to worry about libraries, we only have to make sure that the compiler finds the include files. That's why we only need to use the `-I` flag when we create the object files.

The last line is where we make the final plug-in. Here, we don't need to worry about include files. We only have to make sure that we **link** the libraries needed by the plug-in. We can look at it this way: when we make the object files, we include descriptions of library function calls by including header files. The plug-in object file will happily use them, because it assumes that it will be linked to these libraries later on.

This is what happens in the last line: Here we tell the compiler that the final outcome will be a plug-in. We do this by adding `-o plug-in`, then we add all the object files we made in the first three lines. Because we have used a lot of library functions in the object files, we have to add the libraries to provide these functions to the plug-in. We do this by `-lgimp -lgimpui` etc. We also tell the compiler where to search for the libraries by adding `-L/usr/local/lib` (the directory `/usr/lib` and `/lib` are also searched by default).

A big program or plug-in can consist of several parts. If your dealing with such a plug-in, compiling the different parts manually is often the wrong way to go. The reason for this is that if something goes wrong, you will have to type all your commands again. The solution is to create a **Makefile**. A Makefile is a specification which the Make command uses as an input to compile the entire plug-in with the help of the compiler.

HOW TO CREATE A MAKEFILE AND HOW TO USE IT

Now we'll take a look at how to create a simple Makefile. This will help us understand the structure of a Makefile, so that we'll know how to edit a Makefile which comes with a plug-in. We will take a quick look at a generic Makefile

```
CC = gcc
INCDIR = -I/usr/local/include -I/usr/local/tiff/include
CFLAGS = -O2
LIBDIR = -L/usr/X11/lib -L/usr/local/lib
LFLAGS = -lgimp -lgtk -lX11 -ltiff -lgimpui
HEDERS = plugin.h plugin2.h
SOURCES = pluginmain.c pluginpart1.c pluginpart2.c
OBJECTS = pluginmain.o pluginpart1.o pluginpart2.o

plug-in: $(OBJECTS) $(HEADERS)
        $(CC) $(CFLAGS) -o $@ $(OBJECTS) $(INCDIR) $(LIBDIR)
$(LFLAGS)
```

This Makefile will first build the object files, and then link the plug-in. All you have to do if you want to use it is to replace the **header**, **object**, and **source** files with the correct ones. You can also change what libs to link, and what directories to search for libs and includes. The `CFLAG -O2` stands for optimization of the executable. Notice the tab at the last line. If you forgot the tab you will get this error message:

```
Makefile:11: *** missing separator. Stop.
```

A Makefile example

Here's how the Makefile looked when we built the **Guash** plug-in at our Linux system (there is a Makefile included in the distribution of Guash, this example is only for training)

```
CC = gcc
INCDIR = -I/usr/local/include
CFLAGS = -O2
LIBDIR = -L/usr/local/lib
LFLAGS = -lglib -lgdk -lgtk -lgimp
HEADERS = guash-directory.h guash-banner.h
SOURCES = guash.c
OBJECTS = guash.o

guash: $(OBJECTS) $(HEADERS)
	$(CC) $(CFLAGS) -o $@ $(OBJECTS) $(INCDIR) $(LIBDIR)
$(LFLAGS)
```

The important thing is to change, and maybe add **include** and **library** directories so it will fit your system. We hope you're now able to edit a Makefile so it will fit your system. You have to remember that you can write a Makefile in several different ways, this is only one of many.

Variables

When you edit another makefile it may not look like this, but with the basic knowledge and the knowledge of how to compile by hand, you should be able to edit it. As you see, the first eight lines are variables which are called later in line 10 and 11 with `$(VARIABLE)`. The special `$@` variable is an internal variable which will expand to the first word in the line above it (i.e guash in this example).

To build the plug-in, simply invoke Make by typing `make` or `make -f WhatYouCallYourMakefile`. Make will now invoke **gcc** to build your object files and to link your executable.

A common error when you compile a plug-in which comes as an archive with a Makefile, is that the Make-program complains about **dependencies**. If so, remove the.deps directory (`rm -rf .deps`) in the building directory. Dependencies can also be included in the Makefile. If so, you will find them in the end of the file. Remove them and everything should work fine.

CONFIGURE A WAY TO AUTOMATE THE BUILDING PROCESS

The **configure script** that comes with the plug-in is a way to **automate** the process of making a Makefile. This script will try to find the include files and libraries that the plug-in needs.

If you are on a common UNIX platform like **Solaris**, **Linux** etc. then it's generally quite simple to use a configure script. To build your plug-in, type the following in the directory where you unpacked the plug-in: `./configure && make` Now the plug-in will be compiled and ready to use. Do make sure that you are in the directory where the configure script is located. Other wise you will get this error message: `bash: ./configure: No such file or directory`.

If you don't have a certain file or lib, or the script can't find it, then you have to install it or tell the script where to find it.

To find out how to tell the script where to find include files and libraries etc. invoke the script like this: `./configure --help` This will print several lines of flags that you can supply the script with. Here's an extract of the most important flags that you can include in your script:

```
--libdir=DIR
```

```
--includedir=DIR
```

With these two flags you can add an **include/library dir** like this `--includedir=/your/includedir`. If you want to add more than one include dir, you have to do it this way if you work in an ordinary shell:

```
export CFLAGS="-I/one/includedir -I/another/includedir"
```

If you are working in a C shell, then you have to do it like this:

```
setenv CFLAGS "-I/one/includedir -I/another/includedir"
```

If you want to add more than one libdir, type the same as for the CFLAGS, but write LDFLAGS instead.

Configure is not bullet-proof, and you may run into several problems, especially if you are using a UNIX dialect which is different from what the configure authors were working on. If you run into this kind of problems, first try to add all the lib and include dirs to the script with the FLAGS, or to add them by hand in the generated Makefile. If this still doesn't solve the problem, try writing a mail to the Gimp mailing list.

appendix

A

Appendix

- MAN PAGES
- INITIATION FILES AND COMMAND LINE SWITCHES
- COMMERCIAL SUPPORT
- SIOD REFERENCE
- LINKS AND REFERENCES

appendix A

Gimp start flags and rc-files

Most of the time Gimp is probably started by an icon or a menu in your favorite window manager but you can also start it by hand. There are also several initiation files that control the behavior of Gimp.

GIMP COMMAND LINE SWITCHES AKA FLAGS (OPTIONS)

A excellent source of information about the **command line switches** and **environments** that Gimp support is found in the Gimp **man page**. It's located in the `doc dir` in your Gimp source distribution and normally you only have to type `man gimp` after you have installed Gimp.

We will make a short description of each flag here in GUM. All of this is done in a shell, for example in an xterm window or rxvt window. As we mentioned earlier, you have probably hidden all this in your Window manager so you only have to click on a icon or drag at a menu. If you want to test **flags** and different **environments** or even run gimp without Xwindow you must do this in a shell.

Typing `gimp --help` or `gimp -h` and press `enter` will result in a short description of available flags that you can give Gimp.

```
[olof@olof olof]$ gimp --help
Usage: gimp [option ...] [files ...]
Valid options are:
  -h --help              Output this help.
  -v --version           Output version info.
  -b --batch <commands>  Run in batch mode.
  -n --no-interface      Run without a user interface.
  --no-data             Do not load patterns, gradients, palettes,
brushes.
  --verbose              Show startup messages.
  --no-splash            Do not show the startup window.
  --no-splash-image     Do not add an image to the startup window.
  --no-shm              Do not use shared memory between GIMP and
its plugins.
  --no-xshm              Do not use the X Shared Memory extension.
  --display <display>    Use the designated X display.
[olof@olof olof]$
```

The `-h` and `--help` flags obviously printed out the above message. The `-v` and `--version` flags shows what version of Gimp you are running (this information is also available in the about dialog and the splash window that shows up when you start Gimp). It will typically look like this:

```
[olof@olof olof]$ gimp -v
GIMP version 1.0
```

Always visit `www.gimp.org or ftp.gimp.org` or one of it’s mirrors to get the newest stable version of Gimp.

Batch mode and ”no-interface”

The **-b** and **--batch** option allows you to execute Gimp with arguments to run. This is ideal if you want to execute a lot commands to a lot of files. Then its can be quite annoying to have a GUI to do it in (i.e. **open** image, **apply** commands, **save** image, **open** another image, apply command etc., you see it will be a lot of mouse training if you want to do this). The `<command>` is a script-fu that will do the actual work for you (even if you can execute ordinary `gimp pdb` command directly)

The **-n** and **--no**-interface are suitable if you run Gimp in batch mode since you most of the time don’t want to have a user interface if you are running a batch. This will also save some memory and system resources, so it would be quite pointless to fire up the user interface. Here is an example of two batch commands

```
gimp -n -b ′(gimp-procedural-db-dump ″pdb_dump.tmp″)′ ′(gimp-quit
0)′
```

which will dump Gimp’s pdb database to a file called `pdb_dump.tmp` in your working directory. Here is a example of a custom script that is invoked by Gimp (`my-script` is your own personal custom script).

```
gimp -n -b ′(my-script 1 ″\″Sample text.\″″)′ ′(gimp-quit 0)′
```

The (gimp-quit 0) is so Gimp quits gracefully and returns the command prompt to you.

If you don’t have Xwindow up and running (i.e you are running your UNIX session in a console that has no graphic capabilities or you have a modem connection to your UNIX host with a vt100 terminal only), then you still can run Gimp in batch mode, just do it like this:

```
Xvfb :1 -screen 0 10x10x8 -pixdepths 1 &
```

```
gimp --display :1.0 -n -b ′<commandos> ′ ′(gimp-quit 0)′
```

This will fire up an invisible X server, which you run Gimp in.

More options

As you saw above, we introduced a new flag **--display**, since X lets you run Gimp on one host and displaying it on another, you have to specify the display to run Gimp’s user interface on.

Normally you run and display Gimp on the same host, and you don’t have to barter about display settings. We will give you an example on how to use the display option in Gimp. Say that you have a ”Supercomputer” at your campus running UNIX and this computer has Gimp installed. Then it would be more than efficient to edit huge Gimp images at this computer while displaying them at your local workstation or X-terminal. Here is a quick how to do it. At your workstation, you will have to execute a command allowing the Supercomputer to display Gimp on your workstation.

```
[olof@olof olof]$ xhost niceriver.frozenriver.com (my local super-
computer ;)

niceriver.frozenriver.com being added to access control list
```

The second line tells us that we have enabled the ”supercomputer” to access our X-server at our workstation. All we have to do now is to `telnet` or `rlogin` or `rsh` to the ”supercomputer”

```
[olof@olof olof]$ rlogin niceriver.frozenriver.com

[olof@niceriver olof]$
```

Now we are logged in to the ”supercomputer”. All we have to is to fire off Gimp and tell it where to display, i.e at `olof.frozenriver.com,` and which display to use at olof.frozenriver.com. Since we only have one display at `olof` we will use display number 0

```
gimp --no-xshm --display olof.frozenriver.com:0.0
```

Now Gimp will display at your workstation and you can work with it just like you would have if you had run Gimp at your workstation. When you are finished with Gimp you just have to logout from the supercomputer and tell your workstation that you don’t want the supercomputer accessing your X-server

```
[olof@olof olof]$ xhost - niceriver.frozenriver.com

niceriver.frozenriver.com being removed from access control list
```

The last line tells us that the supercomputer no longer can access our X-server. There are of course better ways to handle remote display, i.e better security, automatic transfer of the display environment etc. but this is beyond the scope of this book.

As you saw above, we introduced a new option `--no-xshm`, which tells Gimp not to try to use X shared memory. We have to do this because we ran Gimp on a different host. If we run Gimp at the same host that we displayed Gimp at, we can use X shared memory to speed things up a little bit, and also so we will be lean onsystem resources. There is another shared memory flag `--no-shm` which tells Gimp not to share memory with its plug-ins. It’s generally good to let Gimp do this but if you encounter problems it can be wise to turn it off.

The `--no-splash` tells Gimp to not show the splash when it starts. If you just tell Gimp `--no-splash-image` then the splash will be shown, but without the image.

`--verbose` will start up Gimp+ a little more verbose and you will se in the shell how it’s phrasing the different initiations files.

```
[olof@olof olof]$ gimp --verbose

parsing "/home/olof/.gimp/gtkrc"

parsing "/usr/local/share/gimp/gimprc"

parsing "/home/olof/.gimp/gimprc"

parsing "/home/olof/.gimp/pluginrc"
```

```
writing "/home/olof/.gimp/pluginrc"
parsing "/home/olof/.gimp/menurc"
Starting extensions: extension_script_fu
```

The last option is `--no-data` which you can use if you run Gimp in batch mode and don't need brushes, gradients, palettes or patterns. The start up time for Gimp will then be minimized.

INITIATIONS FILES AKA RC FILES

Gimp has a lot of initiations files which controls the behavior of Gimp. Most of the options that you set in different rc files are done by the **preference** dialog see chapter 5. We will only take a look at the options that you can't set from the dialog.

GIMPRC AND ~/.GIMP/GIMPRC

The **system-wide** `gimprc` and the **personal** `gimprc` located in your gimp directory (i.e usually `~/.gimp/gimprc`) control nearly all of Gimp's options. To change things in your **personal** `gimprc` file (`~/.gimp/gimprc`) you have to bring it up in an editor and edit the file.

```
[olof@olof olof]$ xemacs ~/.gimp/gimprc
```

There won't be much in it, because most of the options are written in the system-wide `gimprc` file (usually `/usr/local/share/gimp/gimprc`). If you want to change a system-wide setting, then copy it from the system-wide `rc` file, paste it into your personal `rc` and then change the setting of it. Since the system-wide `rc` file is phrased before the personal `rc` file, everything written in the personal file will override what's written in the system-wide file. It's a good idea to open the system-wide `rc` file to get a glimpse of what you can change. If you are a system administrator it's wise to change the system-wide `rc` file to fit your site's needs. We will take a look at the `rc` file.

As you can see, the file is more or less self-explaining, and we will only comment on things that you can't set in the **preference** dialog.

```
# This is the system-wide gimprc file. Any change made in this file
# will affect all users of this system, provided that they are not
# overriding the default values in their personal gimprc file.
#
# Lines that start with a '#' are comments.
# Blank lines are ignored.

# The variable gimp_dir is set to either the interned value
# .gimp or the environment variable GIMP_DIRECTORY. If
# the path in GIMP_DIRECTORY is relative, it is considered
# relative to your home directory.

(prefix "/usr/local")
(exec_prefix "${prefix}")
(gimp_data_dir "${prefix}/share/gimp")
(gimp_plugin_dir "${exec_prefix}/lib/gimp/0.99")
```

You ***shouldn't change*** these preferences, but if you feel you must, you can only do it in an **editor**

```
# Set the temporary storage directory...files will appear here
# during the course of running the gimp. Most files will disappear
# when the gimp exits, but some files are likely to remain,
# such as working palette files, so it is best if this directory
# not be one that is shared by other users or is cleared on machine
# reboot such as /tmp.
(temp-path "${gimp_dir}/tmp")

# Set the swap file location. The gimp uses a tile based memory
# allocation scheme. The swap file is used to quickly and easily
# swap files out to disk and back in. Be aware that the swap file
# can easily get very large if the gimp is used with large images.
# Also, things can get horribly slow if the swap file is created on
# a directory that is mounted over NFS.  For these reasons, it may
# be desirable to put your swap file in "/tmp".
(swap-path "${gimp_dir}")

# Set the brush search path...this path will be searched for valid
#  brushes at startup.
(brush-path "${gimp_dir}/brushes:${gimp_data_dir}/brushes")

# Specify a default brush. If none is specified it defaults to the
#  "1circle.gbr" brush which is just a single pixel sized brush.
#  The brush is searched for in the brush path.
(default-brush "19fcircle.gbr")
```

Not adjustable in the pref. dialog

```
# Set the pattern search path...this path will be searched for
valid
#  patterns at startup.
(pattern-path "${gimp_dir}/patterns:${gimp_data_dir}/patterns")

# Specify a default pattern.
#  The pattern is searched for in the specified pattern paths.
(default-pattern "wood2.pat")
```

Not adjustable in the pref. dialog

```
# Set the palette search path...this path will be searched for
valid
#  palettes at startup.
(palette-path "${gimp_dir}/palettes:${gimp_data_dir}/palettes")

# Specify a default palette.
#  The pattern is searched for in the specified pattern paths.
(default-palette "Default")
```

Not adjustable in the pref. dialog

```
# Set the gradient search path...this path will be searched for
valid
#  gradients at startup.
(gradient-path "${gimp_dir}/gradients:${gimp_data_dir}/gradients")

# Specify a default gradient.
#  The gradient is searched for in the specified gradient paths.
```

```
(default-gradient ”German_flag_smooth”)
```

Not adjustable in the pref. dialog

```
# Set the plug-in search path...this path will be searched for
#  plug-ins when the plug-in is run.
(plug-in-path ”${gimp_dir}/plug-ins:${gimp_dir}/plug-ins/script-fu:${gimp_plugin_dir}/plug-ins”)

# Set the path for the script-fu plug-in.  This value is ignored by
#  the GIMP if the script-fu plug-in is never run.
(script-fu-path ”${gimp_dir}/scripts:${gimp_data_dir}/scripts”)
```

Not adjustable in the pref. dialog

```
# The tile cache is used to make sure the gimp doesn’t thrash
# tiles between memory and disk. Setting this value higher will
# cause the gimp to use less swap space, but will also cause
# the gimp to use more memory. Conversely, a smaller cache size
# causes the gimp to use more swap space and less memory.
# Note: the gimp will still run even if `tile-cache-size’ is
# set to 0. The actual size can contain a suffix of ’m’, ’M’,
# ’k’, ’K’, ’b’ or ’B’, which makes the gimp interpret the
# size as being specified in megabytes, kilobytes and bytes
# respectively. If no suffix is specified the size defaults to
# being specified in kilobytes.
(tile-cache-size 10m)

# Speed of marching ants in the selection outline
#  this value is in milliseconds
#  (less time indicates faster marching)
(marching-ants-speed 300)
```

```
# Set the number of operations kept on the undo stack
(undo-levels 5)

# Set the color-cube resource for dithering on 8-bit displays
#  The 4 values stand for Shades of red, green, blue and grays
#  Multiplying the # of shades of each primary color yields
#  the total number of colors that will be allocated from the
#  gimp colormap.  This number should not exceed 256.  Most of the
#  colors remaining after the allocation of the colorcube
#  will be left to the system palette in an effort to reduce
#  colormap ”flashing”.
(color-cube 6 6 4 24)
```

Not adjustable in the pref. dialog

```
# Install a GIMP colormap by default -- only for 8-bit displays
# (install-colormap)

# Specify that marching ants for selected regions will be drawn
#  with colormap cycling as opposed to redrawing with different
stipple masks
#  this color cycling option works only with 8-bit displays
# (colormap-cycling)

# Tools such as fuzzy-select and bucket fill find regions based on
#  a seed-fill algorithm.  The seed fill starts at the initially
# selected pixel and progresses in all directions until the
# difference of pixel intensity from the original is greater than a
# specified threshold ==> This value represents the default
# threshold
(default-threshold 15)
```

Not adjustable in the pref. dialog

```
# There is always a tradeoff between memory usage and speed.  In
most
#  cases, the GIMP opts for speed over memory.  However, if memory
is
#  a big issue, set stingy-memory-use
# (stingy-memory-use)

# When zooming into and out of images, this option enables the
#  automatic resizing of windows
# (allow-resize-windows)

# Context-dependent cursors are cool.  They are enabled by default.
#  However, they require overhead that you may want to do without.
#  Uncomment this line to disable them.
# (no-cursor-updating)

# Layer preview sizes:
#  none:    no previews in layers dialog/layer selector
#  small:   32x32
#  medium:  64x64
#  large:   128x128
#  #:       #x#
(preview-size small)

# Tooltips
#  Comment this out to disable the tooltips in the toolbox
# (dont-show-tool-tips)

# Controlling ruler visibility
#  The default behavior is for rulers to be ON
#  This can also be toggled with the View->Show Rulers command or
shift+control+r
# (dont-show-rulers)
```

Not adjustable in the pref. dialog

```
# Ruler units
#  The units of rulers can be one of: (pixels inches centimeters)
#  The default is pixels
(ruler-units pixels)
```

Not adjustable in the pref. dialog, but don't change it unless you know what you are doing.

```
# Disable auto saving
#  Just uncomment the line below...
# (dont-auto-save)
```

Not adjustable in the pref. dialog, it doesn't do anything at the moment

```
# Disable confirmation before closing an image without saving
#  Just uncomment the next line
# (dont-confirm-on-close)
```

Not adjustable in the pref. dialog

```
# Setting the level of interpolation
#  Uncommenting this line will enable cubic interpolation.
#  By default, GIMP uses linear interpolation, which is faster, but
has
# .poorer quality
# (cubic-interpolation)

# Set the gamma correction values for the display
#  1.0 corresponds to no gamma correction.  For most displays,
#  gamma correction should be set to between 2.0 and 2.6
#  Run the utility ”gamma_correct” to determine appropriate values
#  for your display.
#
# One important item to keep in mind:  Many images that you might
#  get from outside sources will in all likelihood already be
#  gamma-corrected.  In these cases, the image will look washed-out
#  if the gimp has gamma-correction turned on.  If you are going
#  to work with images of this sort, turn gamma correction off
#  by removing this line, or setting the values to 1.0.
#  gamma-correction 1.0
#  gamma-correction 2.0
#                 ___
(gamma-correction 1.0)
```

Not adjustable in the pref. dialog, see the chapter 13 about how to make a gamma correction.

```
# Set the manner in which transparency is displayed in images
#  Transparency type can be one of:
#    0: Light Checks
#    1: Mid-Tone Checks
#    2: Dark Checks
#    3: White Only
#    4: Gray Only
#    5: Black Only
#  Check size can be one of:
#    0: Small
#    1: Medium
#    2: Large
(transparency-type 1)
(transparency-size 2)
```

The rest is paths for different plug-ins. Quite often you have to add a line like this to specify the path to auxiliary files for some new plug-ins. These lines are ***not adjustable*** in the pref. dialog

```
(fractalexplorer-path "${gimp_data_dir}/fractalex-
plorer:${gimp_dir}/fractalexplorer")
```

```
(gfig-path "${gimp_data_dir}/gfig:${gimp_dir}/gfig")
```

```
(gflare-path "${gimp_dir}/gflares:${gimp_data_dir}/gflares")
```

MENURC

This is a **personal only** file located in your `.gimp` directory. This is where all your dynamically changed **key-bindings** end up. The easiest way to edit this file is to do the key-binding in Gimp. The altered short cuts will be written to this file as soon as you quit Gimp. If your key-bindings are totally screwed, then remove the file.

```
[olof@olof olof]$ rm ~/.gimp/menurc
```

There is also a key-binding file that will make Gimp use Photoshop's key-bindings. It's located in the **system-wide** directory and it's called `ps-menurc`. If you want to use it, just copy it to your `.gimp` directory

```
[olof@olof olof]$ cp /usr/local/share/gimp/ps-menurc ~/.gimp/
```

But why use Photoshop key-bindings when there are Gimp key-bindings?

PLUGINRC

This file holds information about all the plug-ins available to Gimp. ***Do not edit this file!*** If Gimp starts to act spooky when it comes to plug-ins, then you can delete this file and Gimp will write a new one for you.

```
[olof@olof olof]$ rm ~/.gimp/pluginrc
```

GTKRC

This file is also a **personal only** file. It controls the behavior of the **GTK tool kit** that Gimp uses for its menus, tabfolders etc.... One of the few reasons to edit this file is to change the font that Gimp uses in the menus. You maybe want to make it bigger or smaller. Here is an extract:

```
# style <name> [= <name>]
# {
#   <option>
# }
#
```

```
# widget <widget_set> style <style_name>
# widget_class <widget_class_set> style <style_name>

style ”ruler”
{
  font = ”-adobe-helvetica-medium-r-normal--*-80-*-*-*-*-*-*”
}

style ”default”
{
  font = ”-adobe-helvetica-medium-r-normal--*-100-*-*-*-*-*-*”
}

#style ”lsystem_rules”
#{
#  font = ”-*-courier-medium-r-normal--*-100-100-100-m-*-*-*”
#}



widget_class ”*Ruler*” style ”ruler”
widget_class ”*” style ”default”
```

To understand how to change the font line, please read chapter 10 and 40. This file is otherwise quite self-explaining.

INSTALLING A NEW GIMP

If you are installing a new version of Gimp, please remember to remove your personal `rc`files. We mostly do this by renaming our `.gimp` directory to `.gimp.old.` Then we can always open our old files and cut and copy special file modifications.

appendix B

Gimp man pages

There are two Gimp man pages one for Gimp and one for a util called gimp-tool.

GIMP MAN PAGE

NAME

Gimp - an image manipulation and paint program.

SYNOPSIS

gimp [–h] [—help] [-v] [—version] [–b] [—batch <*commands*>] [–n] [—no–interface] [—no–data] [—verbose] [—no–shm] [—no–xshm] [—display *display*] [—no–splash] [—no–splash–image] [—debug–handlers]

DESCRIPTION

The *gimp* is the GNU Image Manipulation Program. It is used to edit and manipulate images. It can load and save a variety of image formats and can be used to convert between formats.

Gimp can also be used as a paint program. It features a set of drawing and painting tools such as airbrush, clone, pencil, and paint brush. Painting and drawing tools can be applied to an image with a variety of paint modes. It also offers an extensive array of selection tools like rectangle, ellipse, fuzzy select, bezier select, intelligent scissors, and select by color.

Gimp offers a variety of plugins that perform a variety of image manipulations. Examples include bumpmap, edge detect, gaussian blur, and many others.

In addition, Gimp has several scripting extension which allow for advanced non-interactive processing and creation of images.

OPTIONS

.l
The *gimp* accepts the following options:

–h, —help
Display a list of all commandline options.

–v, —version
Output the version info.

–b, —batch <*commands*>
Execute the set of <*commands*> non-interactively. The set of <*commands*> is typically in the form of a script that can be executed by one of the Gimp scripting extensions.

–n, —no-interface
Run without a user interface.

—no–data
Do not load patterns, gradients, palettes, or brushes. Often useful in non-interactive situations where startup time is to be minimized.

—verbose
Show startup messages.

—no–shm
Do not use shared memory between GIMP and its plugins. Instead of using shared memory, GIMP will send the data via pipe. This will result in slower performance than using shared memory.

—no–xshm
Do not use the X Shared Memory extension. If GIMP is being displayed on a remote X server, this probably needs to be enabled. Also useful for any X server that doesn't properly support the X shared memory extension. This will result in slower performance than with X shared memory enabled.

—display *display*
Use the designated X display.

—no–splash
Do not show the splash screen.

—no–splash–image
Do not show the splash screen image as part of the splash screen.

—debug–handlers
Enable debugging signal handlers.

ENVIRONMENT

DISPLAY
to get the default host and display number.

XENVIRONMENT
to get the name of a resource file that overrides the global resources stored in the RESOURCE_MANAGER property.

FILES

Most gimp configuration is read in from the users init file, **$HOME**/.gimp/gimprc. The system wide equivalent is in **$PREFIX**/share/gimp/gimprc. The system wide file is parsed first and the user gimprc can override the sytem settings. **$PREFIX**/share/gimp/gimprc_user is the default gimprc placed in users home directories the first time gimp is ran.

$HOME/.gimp/gtkrc - users set of GTK config settings. Options such as widget color and fonts sizes can be set here.

$PREFIX/share/gtkrc - sytem wide default set of GTK config settings.

$HOME/.gimp/menurc - user's set of keybindings.

$PREFIX/share/menurc - system wide set of keybindings.

$HOME/.gimp/plug-ins - location of user installed plugins.

$HOME/.gimp/pluginrc - plugin initialization values are stored here. This file is parsed on startup and regenerated if need be.

$HOME/.gimp/tmp - default location that gimp uses as temporary space.

Gimp's data files are stored in **$PREFIX**/share/gimp where **$PREFIX** is set on install, but is typically /usr/local.

$PREFIX/share/gimp/brushes - system wide brush files.

$HOME/.gimp/brushes - user created and installed brush files. This files are in the .gbr (gimp brush) format.

$PREFIX/share/gimp/palettes - the system wide palette files. The files are copied to the user palettes directory when gimp is first ran to allow the user to modify the palettes. This directory is not searched for palettes by default.

$HOME/.gimp/palettes - copies of the system palette files as well as user created and modified palette files.

$PREFIX/share/gimp/patterns - basic set of patterns for use in gimp.

$HOME/.gimp/patterns - user created and installed gimp pattern files. This files are in the .pat format.

$PREFIX/share/gimp/gradients - standard system wide set of gradient files.

$HOME/.gimp/gradients - user created and installed gradient files.

$PREFIX/share/gimp/palettes - system wide palette files.

$HOME/.gimp/palettes - user created and installed palette files.

$PREFIX/share/gimp/scripts - system wide directory of scripts used in Script-Fu and other scripting extensions.

$HOME/.gimp/scripts - user created and installed scripts.

$PREFIX/share/gimp/gflares - system wide directory used by the gflare plug-in.

$HOME/.gimp/gflares - user created and installed gflare files.

$PREFIX/share/gimp/gfig - system wide directory used by the gfig plug-in.

$HOME/.gimp/gfig - user created and installed gfig files.

$PREFIX/share/gimp/gimp_splash.ppm - graphic file used for the gimp splash screen.

$PREFIX/share/gimp/gimp_logo.ppm - graphic file used in the gimp about dialog.

$PREFIX/share/gimp/gimp_tips.txt - list of tips displayed in the "Tip of the Day" dialog box.

SEE ALSO

X(1)

COPYRIGHT

Copyright © 1995 Spencer Kimball and Peter Mattis

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

SUGGESTIONS AND BUG REPORTS

Any bugs found should be reported to the Gimp Developer mailing list at gimp-developer@scam.xcf.berkeley.edu or you may want to make use of the online bug-tracking system available on the web at http://www.wilberworks.com/bugs.html.

Before reporting bugs, please check to see if the bug is mentioned in the FAQ's or the mailing list archive. See the section on Other Info for locations of these.

When reporting Gimp bugs, it is important to include a reliable way to reproduce the bug, version number of Gimp (and probably GTK), OS name and version, and any relevant hardware specs. It is also very important to include as much info as possible about the Xserver the problem was found on including at least server name, the visual, and the bit depth.

If a bug is causing a crash, it is very useful if a stack trace can be provided. And of course, patches to rectify the bug are even better.

OTHER INFO

The canonical place to find GIMP info is at http://www.gimp.org. Here you can find links to just about every other gimp site, tutorials, data sets, mailing list archives, and more.

There is also a Gimp User Manual available at http://www.dtek.chalmers.se/~d95olofs/manual/ that goes into much more detail about the interactive use of Gimp. (Check www.gimp.org for a upto date location, this location is not permanent)

The latest version of Gimp and the gtk libs is always available at ftp://ftp.gimp.org.

AUTHORS

Spencer Kimball and Peter Mattis.

With patches, fixes, plugins, extensions, scripts and more from lots and lots of people including but not limited to Lauri Alanko, Shawn Amundson, John Beale, Zach Beane, Tom Bech, Marc Bless, Edward Blevins, Roberto Boyd, Seth Burgess, Brent Burton, Ed Connel, Andreas Dilger, Larry Ewing, David Forsyth, Jim Geuther, Scott Goehring, Heiko Goller, Michael Hammel, Christoph Hoegl, Jan Hubicka, Simon Janes, Ben Jackson, Tim Janik, Tuomas Kuosmanen, Peter Kirchgessner, Karl LaRocca, Jens Lautenbacher, Laramie Leavitt, Raph Levien, Adrian Likins, Ingo Luetkebohle, Josh MacDonald, Ed Mackey, Marcelo Malheiros, Ian Main, Torsten Martinsen, Federico Mena, Adam D. Moss, Shuji Narazaki, Sven Neumann, Stephen Robert Norris, Erik Nygren, Miles O'Neal, Jay Painter, Mike Phillips, Raphael Quinet, James Robinson, Mike Schaeffer, Tracy Scott, Manish Singh, Nathan Summers, Mike Sweet, Eiichi Takamori, Tristan Tarrant, Owen Taylor, Ian Tester, James Wang, Kris Wehner.

GIMP TOOL MAN PAGE

NAME

gimp-tool - script to perform various Gimpy functions

SYNOPSIS

gimp-tool [—prefix*[=DIR]*] [—exec–prefix*[=DIR]*] [—version] [—libs] [—cflags] [—build *plug-in.c*] [—install *plug-in.c*] [—install-admin plug-in.c]

DESCRIPTION

gimp-tool is a tool that can, among other things, build plug-ins and install them if they are distributed in one .c file.

gimp-tool is also used by programs that need to know what libraries and include-paths *Gimp* was compiled with. .m4 macros for use with GNU autoconf are also included, to make detection of these libraries et cetera easy for the upstream maintainer.

OPTIONS

.l
gimp-tool accepts the following options:

—build *plug-in.c*
Compile and link *plug-in.c* into a Gimp plug-in.

—install *plug-in.c*
Compile, link, and install *plug-in.c* into the user's personal Gimp configuration directory (for example, /home/che/.gimp/plug-ins/)

—install–admin *plug-in.c*
Compile, link, and install *plug-in.c* into the system-wide Gimp plug-ins directory (for example, /usr/lib/gimp/0.99/plug-ins/)

—version
Display the currently installed version of Gimp.

—libs
Display the libraries Gimp was compiled with.

—cflags
Display the flags that were passed to the compiler when Gimp was compiled.

ENVIRONMENT

GTK_CONFIG
to get the location of the gtk-config program.

CC to get the name of the desired C compiler.

CFLAGS
to get the preferred flags to pass to the C compiler.

SEE ALSO

gimp(1), **gtk-config**(1)

COPYRIGHT

Copyright © 1995 Spencer Kimball and Peter Mattis

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

appendix C

Commercial Gimp support

At the moment, two companies provide support to Gimp; WilberWorks and Frozenriver.

WILBERWORKS

WilberWorks, Inc.
3121 Kingsley Dr.
Bloomington, IN 47404
USA
Tel: +1 812 332-8375
Email: wilber@wilberworks.com
Web: www.wilberworks.com

FROZENRIVER

Frozenriver
Norra Dragsplesgatan 12
S-421 43 VÄSTRA FRÖLUNDA
SWEDEN
Tel +46 (0)31 47 43 56
Email: karin@frozenriver.ale.se

From summer -98
Email: karin@frozenriver.com
Web: www.frozenriver.com

appendix D

SIOD: Scheme in One Defune, reference appendix

Author George J Carret; Copyright 1996-1997

REFERENCE SECTION FOR BUILT-IN PROCEDURES

Note that the arguments to built-in procedures are always optional and default to (). Many of these procedures call a C library function of the same orsimilar name, in the obvious way. Therefore you can refer to the unix manual page for more detailed information about function behavior. Such proceduresare indicated with a bold **U**.

(%%%memref address)

This is a lowlevel routine which should not be invoked in normal code. References a byte of memory at address. Used mostly to cause a core dump fordebugging purposes by referencing address 0 or -1.

(%%closure env code)

This is a lowlevel routine which should not be invoked in normal code. If code is a cons of the form (arlist . body) then env is a list of frames, and the application of the closure will invoke the interpreter. Otherwise code should be of type tc_subr_X and the application of the closure will pass the env as thefirst argument to the C procedure implementing the subr.

(%%closure-code closure)

This is a lowlevel routine which should not be invoked in normal code. Returns the code passed to %%closure.

(%%closure-env closure)

This is a lowlevel routine which should not be invoked in normal code. Returns the env passed to %%closure.

(%%stack-limit amount silent)

If amount is non-null it sets the runtime stack check pointer to allow for that number of bytes. If silent is non-null the resulting (or current) stack size is returned, otherwise a message is printed.

(* x1 x2 ...)

Returns the product of all its arguments, or 1 if no arguments.

after-gc

A variable, the value is an express evaluated after the gc has done its work. For example:

```
(set! *after-gc* '(if (< (gc-info 4) 5000) (allocate-heap)))
```

`*args*`

A variable, bound to the list of arguments passed to the main program siod.

`(*catch tag body ...)`

A special form. Tag is evaluated and kept in a special location while all the forms in the body are evaluated. Normally returns the value of the last form except if a *throw is encountered within the dynamic scope of the evaluations. Errors may be caught by using a tag of 'errorobj.

`*env*`

A variable, bound to the list of environment values passed to the main program siod.

`*eval-history-ptr*`

A variable, default (), but if set to a list (possibly circular) then each call to eval will cause the car of the list to receive a pointer to the form being evaluated,and then the variable will be set to the cdr of the list. Useful for writing a retrospective trace debugging capability.

`*pi*`

A variable, value 3.1416.

`*plists*`

A variable, internal to the implementation of get and putprop.

`(*throw tag value)`

Locates an active *catch for which the tag is identical and then forces the *catch form to return the value.

`*traced*`

A variable, value is a list of procedures that have been traced.

`(+ x1 x2 ...)`

Returns the sum of its arguments.

`(- x1 x2 ...)`

With one argument returns the negation, returns the difference of the first argument and the sum of the rest.

`(/ x1 x2 ...)`

With one argument returns the inverse, otherwise returns the quotiont of the first argument and the product of the rest.

(< x y)

Returns true if x is numerically less than y.

(<= x y)

Returns true if x is numerically less than or equal to y.

(= x y)

Returns true if x is numerically equal to y.

(> x y)

Returns true if x is numerically greater than y.

(>= x y)

Returns true if x is numerically greater than or equal to y.

(F_GETLK fd ltype whence start len)

The fd may be an integer or file. The function fcntl **(U)** is called on the file descriptor and an appropriate struct flock constructed from the ltype, whence,start and len arguments, and the lock operation F_GETLK. The ltype may be F_RDLCK,F_UNLCK, or F_WRLCK. Whence may be SEEK_CUR, SEEK_END or SEEK_SET.

(F_SETLK fd ltype whence start len)

Same as F_GETLCK but with lock operation F_SETLK. **U**.

F_SETLKW fd ltype whence start len)

Same as F_GETLCK but with lock operation F_SETLKW. **U**. For a good example see the command script cp-build.

(abs x)

Returns the absolute numerical value of x.

(access-problem? filename method)

Invokes the access function **(U)** on the filename and flags created from the method string which should contain one or more of the characters ”rwxf”returning non-null if there is a problem with accessing the file in that way. For example:

```
(if (access-problem? ″x.y″ ″r″) (error ″can′t read x.y″))
```

(acos x)

Returns the inverse cosine of x.

(alarm seconds flag)

Invokes the alarm function (U). The handling of which will causes an error to be signaled in so many seconds. But if flag is false then the error will not be signaled if the alarm took place inside a system call or other critical code section.

(allocate-heap)

Attempts to allocate (call the C library malloc procedure) to obtain an additional heap. The size of the heap and the maximum number of heaps aredetermined at startup time. Returns non-null if successful.

(and form1 form2 form3 ...)

A special form which causes the evaluation of its subforms in order, from left to right, continuing if and only if the subform returns a non-null value.

(append l1 l2 l3 l4 ...)

Returns a list which the result of appending all of its arguments. Example:

```
(append '(a b) '(c d)) => (a b c d)
```

(apply function arglist)

Applies the function to the argument list arglist.

(apropos substring)

Returns a list of all symbols containing the given substring.

(aref array index)

Returns the element of the array at the given index.

(array->hexstr string)

Takes a string or byte array and returns a string in representing the values of the elements in hex.

(aset array index value)

Stores the value at the given index in the array.

(ash value bits)

Arithmetic shift of value a given number of bits to the left (positive) or right (negative).

(asin x)

Returns the inverse sin of x.

(ass key alist function)

Returns the first element of the alist such that the function applied to car of the element and the key returns a non-null value. For example:

```
(define (assq x alist) (ass x alist eq?))
```

(assoc key alist)

Same as (ass key alist equal?).

(assq key alist)

Same as (ass key alist eq?).

(assv key alist)

Same as (ass key alist eql?).

(atan x)

Returns the inverse tagent of x.

(atan2 x y)

Returns the inverse tagent of x/y.

(base64decode x)

Given a string X in base64 representation returns a string with bytes computed using the base64 decoding algorithm. See rfc1521.txt.

(base64encode x)

Returns a string computed using the base64 encoding algorithm.

(begin form1 form2 ...)

A special form which evaluates each of its subforms one after another, returning the value of the last subform.

(benchmark-eval nloops exp env)

A zero-overhead way of evaluating the exp n times.

(benchmark-funcall1 nloops f arg1)

A zero-overhead way of calling the function f n times on arg1.

(benchmark-funcall2 nloops f arg1 arg2)

A zero-overhead way of calling the function f n times on arg1 and arg2.

(bit-and x y)

Returns the bitwise logical ”and” (C language & operator) of numerical arguments x and y.

(bit-not x)

Returns the bitwise logical complement (C language ~ operator) of numerical argument x.

(bit-or x y)

Returns the bitwise logical ”or” (C language | operator) of numerical arguments x and y.

(bit-xor x y)

Returns the bitwise logical ”xor” (C language ^ operator) of numerical arguments x and y.

(butlast x)

Returns a new list which has all the elements of the argument x except for the last element.

(bytes-append x1 x2 ...)

Returns a new byte array by appending its arguments which may be strings or byte arrays.

(caaar x)

Same as (car (car (car x))).

(caadr x)

Same as (car (car (cdr x))).

(caar x)

Same as (car (car x)).

(cadar x)

Same as (car (cdr (car x))).

(caddr x)

Same as (car (cdr (cdr x))).

(cadr x)

Same as (car (cdr x)).

(car x)

If x is the result of (cons a b) then (car x) is the same as a.

(cdaar x)

Same as (cdr (car (car x))).

(cdadr x)

Same as (cdr (car (cdr x))).

(cdar x)

Same as (cdr (car x)).

(cddar x)

Same as (cdr (cdr (car x))).

(cdddr x)

Same as (cdr (cdr (cdr x))).

(cddr x)

Same as (cdr (cdr x)).

(cdr x)

If x is the result of (cons a b) then (cdr x) is the same as b.

(chdir path)

Changes default directory to path. **U**.

(chmod path mode)

Changes the file mode of path. **U**. For example, to add execute access permission to the file f:

```
(chmod f
   (encode-file-mode (append '(XUSR XGRP XOTH)
            (cdr (assq 'mode (stat f))))))
```

(chown path uid gid)

Changes file ownership. **U**.

(closedir stream)

Closes a directory stream. **U**.

(cond clause1 clause2 ...)

A special form where each clause is processed until the predicate expression of the clause evaluates true. Then each subform in the predicate is evaluatedwith the value of the last one becoming the value of the cond form:

```
(predicate-expression form1 form2 ...)
```

(cons x y)

Allocates a list object with x as the car and y as the cdr. For example:

```
(cons 1 (cons 2 (cons 3 ())))
```

evaluates to

```
(1 2 3)
```

(cons-array dimension kind)

Allocates an array (currently limited to one dimension). The kind may be string, byte, double, or lisp (default).

(copy-list x)

The toplevel cons objects of x are copied, returning a new list.

(cos x)

Returns the cosine where x is in units of radians.

(cpu-usage-limits soft-limit hard-limit)

Invokes getrlimit if the arguments are null or otherwise setrlimit. U.

(crypt key salt)

A form of string hash. **U**.

(current-resource-usage kind)

Kind is the symbol SELF or CHILDREN, calls getrusage, **U**.

(datlength data ctype)

Returns the dimension of the data as if viewed as an array by the datref function.

(datref data ctype index)

References the data as if it were an array of C data type ctype, at the given index. The ctype may be CTYPE_CHAR, CTYPE_DOUBLE, CTYPE_FLOAT, CTYPE_LONG, CTYPE_SHORT, CTYPE_UCHAR, CTYPE_ULONG, or CTYPE_USHORT. The data may be a string or byte array.

(decode-file-mode x)

Returns a list of symbols given a numerical file mode.

(define subform1 subform2)

A special form used to assign a value to a variable in one of two ways:

```
(define variable value)
```

or to create a procedure

```
(define (procedure-name arg1 arg2 ...)
      form1
      form2
      ...)
```

(delete-file path)

Deletes the file specified by path.

(delq element list)

Deletes the elements of the list which are eq to its first argument. Possibly modifying the list using the set-cdr! operation.

(encode-file-mode list)

Takes a list of file mode symbols and returns the numerical value. SUID, SGID, RUSR, WUSR, XUSR, RGRP, WGRP, XGRP, ROTH, WOTH, XOTH.

(encode-open-flags list)

Takes a list of open **(U)** flag symbols and returns a numerical value. NONBLOCK, APPEND, RDONLY, WRONLY, RDWR, CREAT, TRUNC, EXCL.

(endpwent)

See **U**.

(env-lookup indentifier environment)

Returns an object such that the car is the location where the value of identifier is stored.

(eof-val)

Returns the object returned by read upon encountering and end of file condition.

(eq? x y)

Returns true if x and y are the same object.

(equal? x y)

Returns true if x and y are equal objects.

(eqv? x y)

Returns true if x and y are the same object or numerically equal.

errobj

This variable is assigned to the offending object when the error procedure has been invoked. Useful mainly during interactive debugging.

(error message object)

Prints the error message then aborts the current execution by invoking *throw using the symbol errobj as the tag and the cons of the message and the object as the value. Equivalent to:

```
(define (error message object)
   (if (> (verbose 0))
      (writes nil ″ERROR: ″ message ″\n″))
   (set! errobj object)
   (*throw ′errobj (cons message object)))
```

(eval expression environment)

Evaluates the expression in the context of the environment. This is not a special form. For example:

```
(eval (read-from-string "(+ 1 2)"))
```

evaluates to 3.

(exec path args env)

Calls execv or execve **U**.

(exit status)

Calls exit **U**.

(exp x)

Computes the exponential function of x.

(fast-load path noeval-flag)

Loads a file of binary format expressions, if noeval-flag is true returns a list of the forms instead of evaluating them.

(fast-print object state)

Outputs a fast (binary) format representation of object, where the state is a list of (file hash-array index).

(fast-read state)

Inputs a form which had been output in fast (binary) format.

(fast-save filename forms nohash-flag comment-string)

Creates a file by using fast-print to output each of the forms. A true value for the nohash-flag will cause symbol names to be output each time they are encountered. Otherwise a more optimal index representation is used. The comment-string is used as the first line of data in the file.

(fchmod filedes mode)

The filedes may be an number or an open file object. **U**.

(fclose stream)

Closes the open file stream. **U**.

`(fflush stream)`

See **U**.

`(file-times path)`

Returns a list of the st_ctime and the st_mtime returned by the stat function. **U**.

`(first x)`

Returns the first element (car) of the list x.

`(fmod x y)`

Floating point mod. **U**.

`(fnmatch pattern string flags)`

Returns true if the string matches the pattern. **U**.

`(fopen path mode)`

Opens the file and returns a file stream. U.

`(fork)`

Create a child process. Returning a numerical pid in the parent, () in the child, or call error if the child cannot be created. **U**.

`(fread size-or-buffer stream)`

Returns a new string of size bytes by calling fread **U**. Or uses the buffer (a string or a byte array) instead and returns the number of bytes read. Returns () on end-of-file.

`(fseek file offset direction)`

The direction is SEEK_CUR, SEEK_END or SEEK_SET. **U**.

`(fstat stream)`

Calls fstat **U** and returns an alist with elements dev, ino, mode, nlink, uid, gid, rdev, size, atime, mtime, ctime, blksize, blocks, flags, and gen.

`(ftell stream)`

Calls ftell **U** to return the current offset into a file.

`(fwrite data stream)`

Write the data, a string or byte-array to the stream. Or data can also be a list of a string or byte-array and a numerical length.

`(gc)`

Invokes the garbage collector.

`(gc-info item)`

ITEM	VALUE
0	true if copying gc, false if mark and sweek
1	number of active heaps
2	maximum number of heaps
3	number of objects per heap
4	amount of consing of objects before next gc

`(gc-status [flag])`

If flag is not specified prints information about the gc. Otherwise flag can be used to turn on or off gc messages or turn on or off the gc itself when in stop and copy mode.

`(get object key)`

Returns the key property of the object.

`(getc stream)`

Reads a character from the stream, returns () for end of file. **U**.

`(getcwd)`

Returns the current working directory. **U**.

`(getenv name)`

Returns the value of the environment variable named, or (). **U**.

`(getgid)`

Returns the group id of the process. **U**.

`(getgrgid gid)`

Returns a list of members of the specified numerical group. **U**.

`(getpass prompt)`

Prompts the user and reads a line with echoing turned off. **U**.

`(getpgrp)`

Returns the process group ID of the calling process. **U**.

`(getpid)`

Returns the process ID of the calling process. **U**.

`(getppid)`

Returns the parent process ID of the calling process. **U**.

`(getpwent)`

Returns an alist representing the next item in the /etc/passwd file. **U**.

`(getpwnam username)`

Returns the /etc/passwd file entry for the given username. **U**.

`(getpwuid)`

Returns the /etc/passwd file entry fo the given user id. **U**.

`(gets stream)`

Reads a line from the stream, () on end-of-file.

`(getuid)`

Returns the uid of the current process. **U**.

`(gmtime value)`

Decodes the value into an alist. The value defaults to the current time. **U**.

`(hexstr->bytes str)`

Decodes the hex representation into a byte array.

(href table key)

The hash table is a one dimensional array of association lists.

```
(define (href table key)
   (cdr (assoc key
      (aref table (sxhash key (length table))))))
```

(hset table key value)

Stores the value into the hash table slot indicated by key.

(html-encode str)

If str contains any special html characters (<>&) a new string is returned with these replaced by their cooresponding representations &lt; &gt; &amp;.

(if predicate-form true-form false-form)

A special form that evaluates the true-form or the false-form depending on the result of evaluating the predicate form.

(intern str)

Looks up a string in the symbol table or enters a new symbol.

(kill pid sig)

Calls the kill function U. With sig defaulting to SIGKILL.

(lambda (arg1 arg2 ...) form1 form2 ...)

Returns an applicable procedure object (CLOSURE) with the given argument list and expression subforms. For example:

```
(mapcar (lambda (x) (* x x)) '(1 2 3))
```

evaluates to:

```
(1 4 9)
```

Also used by the define special form.

(larg-default list index default-value)

Reference the list according to index, but skipping over strings that begin with a colon or a dash. If the list is not long enough it returns the default-value instead. Most useful when used with the *args* variable inside a main program.

(last list)

Returns the last cons in a list.

(last-c-error)

Returns the value of the C library strerror(errno) **U** interned as a symbol.

(lchown path owner group)

Changes the ownership of a symbolic link **U**.

(length object)

Returns the length of an object which may be a string (acts like strlen) or a list, or an array.

(let (binding1 binding2 ...) form1 form2 ...)

A special form where each binding is a (variable value) pair. It works by computing the values, establishing the bindings, and then evaluating the forms, returning the value of the last one. For example the following evaluates to 30:

```
(let ((x 10)
   (y 20))
   (+ x y))
```

(let* (binding1 binding2 ...) form1 form2 ...)

A special form where each binding is a (variable value) pair. It works by sequentially computing each value and then establishing a binding. For examplethe following evaluates to 30:

```
(let* ((x 10)
   (y (+ x 10)))
   (+ x y))
```

(letrec (binding1 binding2 ...) form1 form2 ...)

Useful when the value forms of the bindings are lambda expressions with which you desire to program mutually recursive calls.

(link existing-file entry-to-create)

Creates a hard link **U**.

(list item1 item2 ...)

Conses up its arguments into a list.

(lkey-default list index default-value)

Returns the substring on the right hand size of the equal sign of the first element of the list of the form index=value, or the default-value if none are found. Useful when processing the *args* value inside a main program.

(load fname noeval-flag search-flag)

If search-flag is true it looks for fname in the current directory and then in the SIOD_LIB directory. The forms from the file are parsed according to the ”parser:xxx” directive at the begining of the file (default ”parser:read”). If the neval-flag is true then a list of the forms is returned otherwise the forms are evaluated.

(load-so fname init_fcn)

Loads the dynamic library specified by fname, invoking the init_fcn if specified (default init_fname).

(localtime value)

Returns an alist representing the value as a localtime. **U**. Value defaults to the current time.

(log x)

Computes the natural logarithm of x.

(lref-default list index default-fcn)

Returns the index element of the list or the result of calling the default-fcn if the list is not long enough.

(lstat path)

Returns the stat information of a logical link. **U**.

(make-list length element)

Creates a list of the given length filled with the element specified.

(mapcar fcn list1 list2 ...)

Returns a list which is the result of applying the fcn to the elements of each of the lists specified.

(max x1 x2 ...)

Returns the maximum of x1, x2, etc.

(md5-final state)

Returns a byte array computed from the state, derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm. See rfc1321.txt. Example:

```
(define (md5 str)
   (let ((s (md5-init)))
      (md5-update s str)
      (array->hexstr (md5-final s))))
```

`(md5-init)`

Returns an md5 algoritm state as a byte array.

`(md5-update state string length)`

Performs the update step of the md5 algorithm using data from the string up to length, or length can be an open file object, in which case the data from the file is used to perform the update.

`(member key list)`

Returns the portion of the list where the car is equal to the key, or () if none found.

`(memq key list)`

Returns the portion of the list where the car is eq to the key, or () if none found.

`(memv key list)`

Returns the portion of the list where the car is eqv? to the key, or () if none found.

`(min x1 x2 ...)`

Returns the numerical minimum of its arguments.

`(mkdatref ctype ind)`

Creates a closure functionally equivalent to (lambda (x) (datref x ctype ind)).

`(mkdir path mode)`

Creates a directory with the specified numerical access mode. **U**.

`(mktime alist)`

Returns the numerical time value cooresponding to the alist in the same format as returned by the localtime function. **U**.

`(nconc l1 l2)`

Makes the cdr of the last cons of l1 point to l2.

`(nice increment)`

Changes the priority of the current process by the increment. **U**.

`nil`

Do not change the value of this variable which is bound to the empty list.

`(not x)`

Returns the reverse truth sense of x.

`(nreverse list)`

Destructive reversal of the elements of a list using set-cdr!.

`(nth index list)`

Reference the list using index, with the first element being index 0.

`(null? x)`

Returns true of x is the empty list.

`(number->string x base width precision)`

Formats the number according to the base, which may be 8, 10, 16 or the symbol e or f. The width and precision are both optional.

`(number? x)`

Returns true of x is a number.

`(opendir path)`

Returns a directory stream. Not that in unix path is the name of a directory, but in WIN32 path is a wildcard pattern. **U**.

`(or form1 form2 ...)`

A special form which causes the evaluation of its subforms in order, from left to right until a form evaluates to a non-null value.

`(os-classification)`

Returns unix, win32, vms.

`(pair? x)`

Returns true if x is a pair (created by cons).

`(parse-number str)`

Convers a string to a number.

`(pclose stream)`

Used to close a stream opened using popen. Makes sure the associated process is killed. **U**.

`(popen command type)`

Executes the command in a child process and opens a stream connected to the process standard output if type is r, or the standard input if type is w. **U**.

`(pow x y)`

Computes the result of x raised to the y power.

`(prin1 object stream)`

Outputs the standard readable representation of the object to the stream, which defaults to the standard output.

`(print object stream)`

Same as prin1 followed by output of a newline.

`(print-to-string object string no-trunc-flag)`

Puts the readable representation of the object into the string, starting at the first character unless the no-trunc-flag is true, in which case the representation starts at the current length of the string.

`(prog1 form1 form2 form3 ...)`

A special form which evaluates all its subforms but returns the value of the first one. A useful shorthand to employ instead of using a let.

`(putc char stream)`

Outputs the character to the stream. **U**.

`(putenv setting)`

With a setting is of the form ”key=value” makes a new environment binding available to the getenv function of the current and subsequent child processes, or updates an old one. U.

(putprop object value key)

Not implemented.

(puts string stream)

Outputs the string to the stream. **U**.

(qsort list predicate-fcn access-fcn)

Implements the recursive quicksort algorithm on elements of the list compared by using the predicate-fcn on the results of invoking the access-fcn.

Example	Result
(qsort '(3 1 5 4 2) <)	(1 2 3 4 5)
(qsort '((3 a) (2 b)) < car)	((2 b) (3 a))

(quit)

Cause the read-eval-print loop to return, usually resulting in an exit from the main program of siod, but may not when other C programs are utilizing the libsiod functionality.

(quote x)

A special form that returns x without evaluating it. Commonly written in abbreviated format as 'x.

(rand modulus)

Computes a random number from 0 to modulus-1. Uses C library rand.

(random modulus)

Computes a random number from 0 to modulus-1. Uses C library random.

(read stream)

Inputs characters from the stream returns the parsed standard expression, or (eof-val).

(read-from-string string)

Performs a read operation on the characters in the string.

(readdir directory-stream)

Returns the name of the next entry in the directory stream or () of none left.

(readline stream)

Reads a line of characters from the stream, returning () on end of file. The terminating newline is not included in the string, which is usually more convenient. For example, this procedure for loading a tab-delimited spreadsheet file:

```
(define (load-spread-sheet filename)
   (if (>= (verbose) 2)
      (writes nil ";; loading spread sheet " filename "\n"))
   (let ((result nil)
      (line nil)
      (f (and (not (equal? filename "-")) (fopen filename "r"))))
   (while (set! line (readline f))
      (set! result (cons (strbreakup line "\t") result)))
   (and f (fclose f))
   (nreverse result)))
```

(readlink path)

Returns the contents of the symbolic link at path. **U**.

(realtime)

Returns a double precision floating point value representation of the current realtime number of seconds. Usually precise to about a thousandth of a second.

(rename from-path to-path)

Renames a directory or file within a file system. **U**.

(require path)

Computes a variable name by concatenating "*" + path + "-loaded*" and then calling (load path nil t) if and only if the variable is not bound to true. After the file is loaded the variable is set to true. This is the correct way of making sure a file is only loaded once.

(require-so path)

Computes a variable name by concatenating "init_" + path, and calling (load-so path) if and only if the variable is not bound to true. After the shared library has been loaded the variable is set to true. The correct way of making sure a shared library is only loaded once is:

```
(require-so (so-ext 'name))
```

`(rest x)`

Returns the rest of the list x, in other words the cdr.

`(reverse x)`

Returns a new list which has elements in the reverse order of the list x.

`(rld-pathnames)`

Returns a list of the pathnames which represent shared libraries that have been loaded by the the current process.

`(rmdir path)`

Removes the directory entry specified by path. **U**.

`(runtime)`

Returns a list of containing the current cpu usage in seconds and the subset amount of cpu time that was spent performing garbage collection during the currently extant read-eval-print loop cycle.

`(save-forms filename forms how)`

Prints the forms to the file, where how can be ”w” (default) or ”a” to append to the file.

`(sdatref spec data)`

Used as the %%closure-code by mkdatref.

`(set! variable value)`

A special form that evaluates the value subform to get a value, and then assigns the variable to the value.

`(set-car! cons-cell value)`

Changes the car of the cons-cell object to point to the value.

`(set-cdr! cons-cell value)`

Changes the cdr of the cons-cell object to point to the value.

`(set-eval-history length circular-flag)`

Creates a list of the specified length and establishes bindings for *eval-history-ptr* and *eval-history*. The list is circular if the flag is specified true. Try the following:

```
(define (fib x) (if (< x 2) x (+ (fib (- x 1)) (fib (- x 2)))))
(set-eval-history 200)
```

```
(fib 10)
(mapcar (lambda (x) (if (pair? x) (car x) x)) *eval-history*)
```

(set-symbol-value! symbol value env)

Finds the location of the value cell for the specified symbol in the environment env and sets the value.

(setprop obj key value)

Not implemented.

(setpwent)

Resets the pointer into the /etc/passwd file. **U**.

(setuid x)

Sets the userid of the process. **U**.

(sin x)

Computes the sine function of the angle x in radians.

(siod-lib)

Return the setting of the siod library directory.

(sleep n)

Sleep for n seconds, where n may be fractional on some systems.

(so-ext path)

Append the path with the file extension for shared libraries.

(sqrt x)

Compute the square root of x.

(srand seed)

Reset the algorithm seed for the rand function. **U**.

(srandom seed)

Reset the algorithm seed for the random function. **U**.

(stat path)

Return an alist describing file status information, or () if the path cannot be accessed, (last-c-error) may be used to return the reason.

(strbreakup string sep)

Return a list of the portions of string indicated by the separator.

```
(strbreakup ″x=y&z=3″ ″&″) => (″x=y″ ″z=3″)
```

(strcat str1 str2)

Copies the string str2 into str1 starting at the current active end of str1, which is determined by the location of a 0 byte, calling error if there is not enough room left in str1. **U**.

(strcmp str1 str2)

Returns 0 if str1 and str2 are equal, or -1 if str1 is alphabetically less than str2 or 1 otherwise. **U**.

(strcpy str1 str2)

Copies str1 into str1 or calling error if there is not enough room. **U**.

(strcspn str indicators)

Returns the location of the first character in str which is found in the indicators set, returns the length of the string if none found. **U**.

(strftime format-string alist)

Uses the format-string to compute a string using broken-up time/data information from the alist (defaults to the current time) **U**, for example:

```
(strftime ″%B″ ′((mon . 3))) => ″April″
```

(string->number str radix)

Converts the string to a number assuming the specified radix.

(string-append str1 str2 str3 ...)

Returns a new string which contains the concatenation of all its string arguments.

(string-dimension str)

Returns the maximum possible length of a string array.

(string-downcase str)

Return a new string converting all the characters of str to lowercase.

(string-length str)

Returns the active string length of str.

(string-lessp str1 str2)

Return true if str1 is alphabetically less than str2.

(string-search key str)

Locate the index of the key in the specified string. Returns () if not found.

(string-trim str)

Return a new string made by trimming whitespace from the left and right of the specified string.

(string-trim-left str)

Like string-trim but only the left hand side.

(string-trim-right str)

Like string-trim but only the right hand side.

(string-upcase str)

Returns a new string with all the lowercase characters converted to uppercase.

(string? x)

Returns true if x is a string.

(strptime str format alist)

Parses str according to format and merges the values with the alist. **U**

```
(cdr (assq ′mon (strptime ″March″ ″%B″))) => 2
```

(strspn str indicators)

Returns the location of the first character in str which is not found in the indicators set, returns the length of the str if none found **U**. For example:

```
(define (string-trim-left x)
   (substring x (strspn x ″ \t″)))
```

(subset pred-fcn list)

Return the subset of the list such that the elements satisify the pred-fcn. For example:

```
(subset number? '(1 b 2 c)) => (1 2)
```

(substring str start end)

Returns a new string made up of the part of str begining at start and terminating at end. In other words, the new string has a length of end - start.

(substring-equal? str str2 start end)

An efficient way to determine if the substring of str2 specified by start and end is equal to str1.

(swrite stream table form)

This is the same as the write-smart-html procedure described in ftp://ftp.std.com/pub/gjc/www95-paper.html.

(sxhash data modulus)

Computes a recursive hash of the data with respect to the specified modulus.

(symbol-bound? symbol env)

Returns true of the symbol is bound in the environment.

(symbol-value symbol env)

Returns the value of the symbol in the environment.

(symbol? x)

Returns true if x is a symbol.

(symbolconc arg1 arg2 ...)

Slightly more efficient than calling intern on the result of using string-append on the arguments. This procedure actually predates the availability of the string data type in SIOD.

(symlink contents-path link-path)

Creates a directory entry link-path pointing to the contents-path. **U**.

(system arg1 arg2 ...)

Appends the string arguments to form a command to be executed by the operating system. **U**.

t

Please do not change the global value of this variable, bound to a true value.

(tan x)

Computes the tagent of the angle x specified in radians.

(the-environment)

A special form which returns the interpreter environment structure for the current lexical scope.

(trace fcn1 fcn2 ...)

Traces the specified interpreted procedures by modifying the closure objects.

(trunc x)

Returns the integer portion of x.

(typeof x)

Returns a symbol describing the type of the object x, or the integer type code.

(unbreakupstr list sep)

The reverse of strbreakup. The following example saves a list of lists as a tab delimited spreadsheet:

```
(define (save-spread-sheet filename data)
   (if (>= (verbose) 2)
      (writes nil ";; saving spread sheet " filename "\n"))
   (let ((result data)
      (f (and (not (equal? filename "-")) (fopen filename "w"))))
   (while result
      (writes f (unbreakupstr (car result) "\t") "\n")
      (set! result (cdr result)))
   (and f (fclose f))))
```

(ungetc char stream)

Puts the char back into the stream for the next call to getc.

(unix-ctime x)

Converts the integer time x into a string. **U**

`(unix-time)`

Returns the current number of seconds since 1-JAN-1970 GMT. **U**

`(unix-time->strtime x)`

Returns a string of the form ”YYYYMMDDHHmmSSdd” which is useful in some contexts. This predates the availability of the strftime procedure.

`(unlink path)`

Deletes the specified entry from the directory stucture. **U**

`(untrace fcn1 fcn2 ...)`

Untraces the specified procedures.

`(url-decode str)`

Performs the url decode operation on the str. See people.delphi.com/gjc/chtml.html for example usage.

`(url-encode str)`

Locates characters in the str which should not appear in a url, and returns a new string where they have been converted to the %NN hex representation. Spaces are converted to ”+” signs.

`(utime path modification-time access-time)`

Sets the file modification and access times. **U**

`(verbose arg)`

Sets the verbosity level of SIOD to the specified level or returns the current level if not specified.

Verbose Level	Effect on System
0	No messages.
1	Error messages only.
2	Startup messages, prompts, and evaluation timing.
3	File loading and saving messages.
4 (default)	Garbage collection messages.
5	display of data loaded from files and fetched from databases.

`(wait pid options)`

Waits on a child process by calling the waitpid function, where options may be a list containing (WCONTINUED WNOWAIT WNOHANG WUNTRACED). Returns a list of the process pid and final exit status. The fork-test.scm and http-stress.scm modules provide example usage. **U**

`(while pred-form form1 form2 ...)`

If pred-form evaluates true it will evaluate all the other forms and then loop.

`(writes stream data1 data2 data3 ...)`

Outputs the data arguments to the stream without quoting strings or special characters.

appendix

E

Links and References

In this appendix you will find all kinds of Gimp related links. You will also find some bibliographic links to useful books, or books that we use at Frozenriver.

LINKS

WEB

Resources	Address
The one and only Gimp.org	www.gimp.org
Latest Gimp news	xach.dorknet.com/gimp/news/index.html
Plug-in registry to the latest plug-ins available for Gimp. You can also upload your own plug-ins here	registry.gimp.org
Gimp faq's	www.rru.com/~meo/gimp
GUM	manual.gimp.org
GUM in French	

MAIL

List	address
gimp-developer	info at www.gimp.org/mailing_list.html subscribe send mail to "majordomo@scam.xcf.berkeley.edu" and put "subscribe gimp-developer yourname@your.domain" in the body
gimp-user	info at www.gimp.org/mailing_list.html subscribe send mail to "majordomo@scam.xcf.berkeley.edu"and put "subscribe gimp-user yourname@your.domain" in the body
gimp-announce	info at www.gimp.org/mailing_list.html subscribe send mail to "majordomo@scam.xcf.berkeley.edu" and put "subscribe gimp-announce you name@your.domain" in the body

List	address
japanese gimp mailing list	info at www.gimp.org/mailing_list.html subscribe send mail to ”gimp@hypercore.co.jp”
hungarian gimp mailing list	info at www.gimp.org/mailing_list.html subscribe send mail to ”listproc@kva.hu” with ”SUBSCRIBE GIMP YOURNAME” as the subject where your name is yourname@your.domain

IRC (DEV CHAT) CHANNEL #GIMP

Server	Port	Location
irc.mint.net / irc.gimp.org	6666	Maine, USA
irc.canweb.net	6667	Toronto, Canada
irc.canberra.edu.au	6666	Canberra, Australia
irc.giblets.com	6667	Texas, USA
dazed.nol.net	6667	Texas, USA
irc.eanut.org	6667	Texas, USA
pandora.hrz.uni-bielefeld.de	6667	Germany
irc.chillin.org	6666	Florida, USA
irc.coherent.net	6667	Chicago, USA
irc.olg.com	6667	Maryland, USA

FTP

The main ftp site is naturally ftp.gimp.org but try to use a mirror near you

Location	Address
Africa	ftp://ftp.is.co.za/applications/gimp/
Australia	ftp://gimp.zeta.org.au/
Austria	ftp://gd.tuwien.ac.at/graphics/gimp/
France	ftp://stef.u-picardie.fr/mirror/ftp.gimp.org/
Germany	ftp://infosoc.uni-koeln.de/pub/ftp.gimp.org/

Location	Address
Greece	ftp://sunsite.ics.forth.gr/sunsite/pub/gimp/
Japan	ftp://SunSITE.sut.ac.jp/pub/archives/packages/gimp/
	ftp://ftp.u-aizu.ac.jp/pub/graphics/tools/gimp/
Korea	ftp://ftp.kreonet.re.kr/pub/tools/X11/ftp.gimp.org/
Taiwan	ftp://linux.cis.nctu.edu.tw/pub/packages/X/gimp/
United Kingdom	ftp://ftp.flirble.org/pub/X/gimp/
US	ftp://moloko.insync.net/pub/mirrors/ftp.gimp.org/
	ftp://ftp.cs.umn.edu/pub/gimp/
	ftp://hal.res.wpi.net/mirror/ftp.gimp.org/
	ftp://froody.res.cmu.edu/pub/gimp/
	ftp://ftp.randomc.com/pub/mirrors/gimp/
	ftp://gimp.cs.stevens-tech.edu/mirrors/gimp/
	ftp://gimp.cs.stevens-tech.edu/mirrors/gimp/
	ftp://gimp.chillin.org/pub/gimp.org-mirror/

COMMERCIAL SUPPORT

Company	**Address**
Frozenriver	Frozenriver N.Dragspelsg 12 S-421 43 V.FROLUNDA SWEDEN Tel: +46 (0)31 47 43 56 Email: info@frozenriver.com Web: www.frozenriver.com
WilberWorks	WilberWorks Inc 3121 Kingsley DR Bloomington, IN 47404 USA Tel:+1 812 332-8375 Email: wilber@wilberworks.com

BOOKS

Name	**ISBN**
GUM	
Photoshop Web techniques	1-56205-733-2
Preparing Digital Images for print	0-07-882146-0
Photoshop WOW book	0-201-88370-8

index

Index

Numerics
- 4-color 133
- 8-bit 130

A

- active layer 201
- add alpha channel 206
- add layer mask 203
- add layer mask tips 204
- adding layers 200
- addition mode 191
- additive 130
- Adjust 209
- adjust layers 209
- airbrush 21, 98
  - brush stroke 98
  - pressure 98
- alien map 246
- align
  - horizontal 206
  - vertical 207
- align visible layers 206
- allow window resizing 106, 109
- alpha
  - channel 212
  - channel what is 212
  - decomposing 175
  - holes 178
  - selection edit 213
  - selection store 213
  - to selection 205
  - web image 253
- alpha to selection 205
- AM screening 151
- animation gif 229
- animation optimize 228
- animation playback 228
- animframe 382
  - basic concept 382
  - beyond basics 384
  - concept basic 382
  - controls 386
  - convert 387
  - duplicate 388
  - exchange 387
  - flatten 387
  - gallery 388
  - how to 383
  - layerdel 387
  - make frame 382
  - move path 384
  - navigate frames 382
  - preview 385, 387

    source select 385
    to image 387
    tutorial 388
    undo 387
animframe see also animation filters
antialiasing 81
apply canvas 232
apply layer mask 204
apply lens 302
Authors xxxix
    Karin Kylander xxxix
    Olof S Kylander xxxix
autocrop 178
auto-stretch hsv 169

B

background 55
    color 100
balance 161
bbs 140
behind mode 193
bezier 21, 84
    control points move 84
    sharp corners 85
bilinear 94
black level calibration 143
blackness 116
blend 125
blend tool 21, 92
    bilinear 94
    color options 92
    custom gradient 93
    fill 21
    gradient type 93
    linear 93
    offset 92
    radial 94
    sawtooth 96
    shapeburst 95
    square 94
    symmetric asymmetric 95
    triangular 96
blinds 268
blur 235, 242
    gaussian 242
    iir 242
    motion 243
    rle 242
    variable 244
blurring 99
bmp 60
border 116, 117, 183
brightness 162
brush
    dialog 120
    directory 45
    make 120
    name 120
    opacity 120
    pop up 120
    spacing 120
    stroke 98
brush selection 99, 104
    mode 99
    opacity 99
    spacing 99
bucket fill 92, 103
    fill opacity 92
    fill threshold 92
    mode 92
    pattern fill 92
    sample merged 92
bump map 322
bzip 60

C

calibration 142
    black level 143
    color 145
    gamma 143
    white level 143
carve it 401

cel 60
central reflection 302
Changelog xliii
channel 212
    add alpha 206
    alpha 212
    alpha edit 213
    alpha selection 213
    alpha what is 212
    layers 212
    rgb 212
    save to 183
channel ops 170
checkerboard 348
chrome it 401
clear 103
clone 98
    retouch 98
close 55
cml explorer 346
cms 148
    Caldera 145
    Mentalix 145
cmy 130
cmyk 130, 148
    decomposing 174
code 453
color 139, 158, 197
    balance 161
    brightness-contrast 162
    calibration 145
    calibration cms 145
    calibration Frozenriver 145
    calibration plain 145
    calibration poor man 147
    calibration rgb & cmyk 147
    cmyk decomposing 174
    colorify 247
    compose 172
    contrast auto stretch 169
    curves 164
    curves workflow 164
    decompose 172
    desaturate i.e "grayscale" 168
    dialog 100
    exchange 247
    filter pack 250
    gamut 148
    highlights 161
    hot colors 252
    hsv decomposing 173
    hue 162
    hue hints 163
    invert 159
    levels 166
    levels examples 167
    levels rgb or alpha 167
    midtone 161
    mode 192
    normalize 170
    posterize 159
    rotate 248
    saturation 162
    selection 184
    selection, modes 184
    shadow 161
    threshold 160
color calibration 145
color calibration cms 145
color calibration Frozenriver 145
color calibration plain 145
color calibration poor man 147
color calibration rgb & cmyk 147
color dialog 100
color exchange 247
color gamut 148
color info 90
color management 148
    cms 148
    gamut 148
    profile 148
    rgb & cmyk 148
color map rotation 248
color mode 192

color models 130
  additive 130
  cmy 130
  cmyk 130
  complementary colors 134
  grayscale 134
  hsv 132
  indexed 131
  indexed gif 131
  inverted colors 134
  munsell 133
  natural color system 133
  ncs 133
  Pantone 133
  rgb 130
  spot color 133
  subtractive 130
  Truematch 133
color picker 90
  color info 90
  sample merged 90
color proof 139
color see also color models
color select 83
colorify 247
coloring 125
combine
  depth merge 258
  file 260
  fuse 260
comparing different modes 193
compile 452
  code 453
  configure 459
  flag see flag
  gcc 454
  how to 453
  include files 455
  libraries 455
  link 455
  makefile 457
  multiple source files 456
  programs 452
  variables 458
complementary colors 134
compose 172
convolver 99
  blurring 99
  sharpen 99
configure 452, 459
configure flags 459
conical anamorphose 302
contrast 162
contrast auto stretch 169
Contributions xli
  Gimp contributions xli
  GUM contributions xlii
convert
  grayscale 175
  indexed 175
  rgb 175
convolution matrix 294
coordinate map 323
copy 102
  named 102
  visible 104
correct color 158
creating image objects 20
crop 110
  move 110
  nudge 110
  resize 110
  selection 110
  snap to guides 110
crypt 264
cubism 232
curtain 268
curves 164
  workflow 164
cut 102
  named 102

D

darken mode 191
db browser 71
decompose 172
decompress 453
decrypt 264
deinterlace 288
delete layer 202
desaturate 168
despeckle 288
destripe 289
difference mode 21, 190
diffraction patterns 349
digital signature 264
displace 323
    black 327
    calculations 324
    description 324
    examples 325
    smear 327
    tips & tricks 327
    wrap 327
dissolve mode 189
dithering 176
dots 138
dpi 138, 149
draw 352
drop shadow 403
duplicate 170
duplicate layers 202

E

edge 284
edge detect 284
emboss 269
emboss see also bumpmap
encoding 445
encrypt 264
engrave 270
eps 61, 65, 136
    resolution 65
equalize 158
eraser 97
eye 200
Ezdrive 140

F

fade out 97
family 445
faxG3 60
feather 81, 182
FF 300
fig 352
figures 350
file formats 59, 136
    bmp 60
    bzip 60
    cel 60
    eps 61, 65, 136
    faxG3 60
    fits 60
    fli 60
    gbr 60, 62
    gicon 60, 62
    gif 60, 63
    gzip 60
    header 60
    hrz 60
    jpeg 60, 64
    mpeg 61
    pat 61, 64
    pcx 61
    pdf 61, 65
    pix 61
    png 61, 65
    pnm 61, 65
    postscript 61
    ps 61, 65
    psd 61
    sgi 61
    snp 61
    sunras 61, 66

  tga 61, 66
  tiff 61, 67, 136
  url 62
  xcf 59, 62
  xpm 62, 67
  xwd 62
file transfer 140
filesystem format 141
fill 103
  opacity 92
  threshold 92
film 260
  how to 260
filter all layers 229
filter factory 300
filters
  alien map 246
  all layers 229
  animation optimize 228
  animation playback 228
  apply lens 302
  artistic 232
  blinds 268
  blur 235, 242
  blur gaussian 242
  blur iir 242
  blur motion 243
  blur rle 242
  blur variable 244
  bump map 322
  bump map usage 322
  canvas 232
  central reflection 302
  checkerboard 348
  cml explorer 346
  color exchange 247
  color filter pack 250
  color map rotation 248
  colorify 247
  conical anamorphose 302
  convolution matrix 294
  coordinate map 323
  crypt 264
  cubism 232
  curtain 268
  deinterlace 288
  depth merge 258
  despeckle 288
  destripe 289
  diffraction patterns 349
  digital signature 264
  displace 323
  displace see displace
  edge 284
  edge detect 284
  emboss 269
  engrave 270
  figures 350
  film 260
  filter factory 300
  filter pack 250
  filter pack color 250
  flame 350
  flarefx 306
  fractal trace 329
  fuse 260
  gfig 352
  gfig see also gfig
  gflare 306
  gif animation 229
  glass tile 303
  gradient map 251
  grid 360
  hide file 265
  hot (colors) 252
  ifs compose 360
  ifs compose see also ifs
  illusion 329
  iwrap 270
  laplace 284
  lic 234
  lic how to 234
  light effect 312
  light effect see also map object

  - make seamless 330
  - map object 330
  - mathmap 296
  - max rgb 252
  - maze 374
  - mosaic 233
  - mosaic tip 233
  - motion blur 243
  - nl 290
  - oilify 234
  - paper tile 335
  - pinch 280
  - pixelize 244
  - plasma 374
  - polar coords 273
  - qbist 375
  - quantize 253
  - randomize 342
  - refract 303
  - ripple 275
  - scatter hsv 253
  - semi flatten 253
  - sharpen 291
  - shift 275
  - sinus 376
  - small tile 335
  - smooth palette 254
  - sobel 285
  - solid noise 378
  - sparkle 318
  - spread 343
  - stegano 265
  - stereogram 339
  - super nova 319
  - texture 234
  - tile 336
  - twist 21, 276
  - universal 299
  - user 300
  - value propagate 278
  - van gogh 234
  - van gogh how to 234
  - variable blur 244
  - video 340
  - warp 236
  - waves 279
  - whirl 280
- fits 60
- flag 455
  - cflags 459
  - configure flags 459
  - -I 456
  - --includedir 459
  - -L 455
  - ldflags 459
  - --libdir 459
  - -o 455
- flame 350
- flare
  - flarefx 306
  - gflare 306
- flarefx 306
- flatten 59, 387
- flatten image 205
- flatten semi 253
- fli 60
- flip 113, 125
- float 182
- floating selection
  - tips 219
- floating selection
  - what is 218
- FM screening 151
- font 116, 442
  - copy 443
  - encoding 445
  - family 445
  - file 444
  - foundry 445
  - install 442
  - loading 444
  - management 443
  - origin 116
  - path 442

  - scalable 442
  - slant 445
  - typ1inst 443
  - type 1 443
  - weight 445
  - width 445
- font see also text tool
- fonts how they work 442
- foreground color 100
- foundry 116, 445
- fractal trace 329
- Frozenriver xl
- ftp 140
- fuse 260

G

- gallery 388
- gamma calibration 143
- gaussian blur 242
- gbr 60, 62
- gcc 454
- gfig 352
  - brush 357
  - curves 353
  - directory 46
  - example 359
  - lines 353
  - options 358
  - paint 356
  - preview 352
  - select 358
  - settings 355
  - user interface 352
- gflare 306
  - directory 46
- ghostscript 47, 68
- ghostscript Alladin 47
- gicon 60, 62
- gif 60, 63
  - animation 63, 229
  - animation loop 63
  - comment 63
  - interlaced 63
  - save 63
  - transparent background color 63
- Gimp
  - about 4
  - authors 4
  - features 4
  - future 8
  - history 0.54 5
  - history 0.60 6
  - history 0.99 7
  - history 1.0 8
- gimprc 45
- glass tile 303
- gradient 122, 399
  - directory 45
  - editor 122
  - map 21, 251
- gradient editor 122
  - blend 125
  - coloring 125
  - endpoint color 124
  - flip 125
  - menu 124
  - points 123
  - pov-ray 125
  - replicate 125
  - save 125
  - section 123
  - segments 124
  - segments split 124
- gradient fill see blend tool
- gradient map 21, 251
- gray shades 154
- grayscale 134, 175
  - convert 175
- grayscale levels 166
- grid 360
- grow 183
- guash 47, 55
- guides 78, 105

- snap 105
- toggle 105

Gum
- conventions xlvii
- usage xlv

gunzip 453

GYUM xl

gzip 47, 60, 453

H

halftone 138
- cell 149
- dot 150
- matrix 151

header 60

highlights 161

histogram 177

holes 178

hot 252

how to add layer mask 204

how to compile 453

how to lic 234

how to make a make file 457

how to write script-fu 406

hrz 60

hsv 132, 162
- auto stretch 169
- decomposing 173
- hue 132
- saturation 132
- scatter 253
- value 132

hue 21, 132, 162, 197
- hints 162

hue mode 192

I

ifs 360
- example 361
- options 363
- settings 361
- usage 360

ifs compose see ifs

iir 242

illusion 329

image
- adding layers 200
- auto correction 169, 170
- auto-stretch hsv 169
- background 55
- brightness 162
- channel ops 170
- color 158
- color balance 161
- compose 172
- contrast 162
- contrast auto stretch 169
- convert 175
- correct color 158
- curves 164
- decompose 172
- desaturate 168
- duplicate 170
- equalize 158
- flatten 205
- guash 55
- histogram 177
- hsv 162
- indexed 55
- invert 159
- line art 160
- mail 67
- new 55
- normalize 170
- objects create 20
- open 55
- open file menu 56
- open guash 55
- open postscript 58
- open type 56
- open type automatic 57
- posterize 159
- print 68

- resize 176
- rgb 55
- rotate 179
- saturation 162
- save 58
- save by extension 58
- save flatten 59
- save formats supported 59
- scale 176
- transparent 55

image size 137
image table 153
import layers 209
include files 455
indexed 131, 175
- convert 175
- convert options 175
- custom palette 176
- dithering 176
- options 175
- posterize 159
- quantize 253

install
- .gimp 45
- Alladin ghostscript 47
- binary 48
- bzip 47
- configure 48
- directory brushes 45
- directory gfig 46
- directory gflare 46
- directory gradients 45
- directory palettes 45
- directory pattern 45
- directory plug-ins 45
- directory script-fu 45
- extra data 49
- font 442
- fonts 49
- fonts by hand 444
- gimp-1.0.X.tar.gz 46
- gimprc 45
- GNU ghostscript 47
- gzip 47
- libraries 47
- libgtk 47
- make 48
- personal files 44
- sane 47
- script-fu 396
- shared data 48
- source 46
- wget 47
- xv 47

install see also compile
intelligent scissors 85
interpolation 69, 138
invert 159, 182
inverted colors 134
Iomega 140
iwrap 270

J

jaz 140
jpeg 60, 64
- compression 64

K

Karin Kylander xxxix
Key 10

L

laplace 284
layer
- anchor 204

layers 200
- active 201
- add alpha channel 206
- add mask 203
- add mask tips 204
- adding 200
- adjust 209
- align visible 206

  - alpha to selection 205
  - apply mask 204
  - channel 212
  - copy visible 104
  - delete 202
  - dialog 200
  - duplicate 202
  - eye 200
  - filter 229
  - flatten 205
  - how to add mask 204
  - import 207, 209
  - introduction 200
  - layerdel 387
  - lower 202
  - mask to selection 206
  - merge visible 205
  - move 210
  - naming 202
  - new 201
  - raise 202
  - resize 203
  - rotate 179
  - scale 202
  - symbols explanation 201
  - thumbnail icon 200
- Legals i–iii
  - Printing ii
- levels 166, 167
  - examples 167
- lic 234
- light effect 312
- light effect see also map object
- lighten mode 191
- line art 134, 160
- linear 93
- link libs 455
- lower layers 202
- lpi 138, 149
- lpi table 153

M

- magnifying 109
  - allow window resizing 109
  - new view 109
  - shrink warp 109
- mail 67
- make 121, 452
- make seamless 330
- makefile 457
  - example 458
  - how to 457
- map object 330
- mask 78
- mask to selection 206
- mathmap 296
- max rgb 252
- maze 374
- merge visible layers 205
- midtone 161
- mode 92, 99
  - addition 191
  - addition 195
  - behind 193
  - color 192, 197
  - comparing different modes 193
  - darken 191
  - darken only 196
  - difference 21, 190
  - dissolve 189
  - explanation between color, hue 196
  - explanation between multiply, darken 196
  - explanation between screen, addition, lighten 195
  - few colors screen, addition, lighten 195
  - hue 192, 197
  - lighten 191
  - lighten only 195
  - multiply 21, 189, 196
  - normal 188
  - overlay 21, 190
  - saturation 21, 192

  - screen 189, 195
  - subtraction 191
  - value 193
- modem 140
- modes
  - what is 188
- mosaic 233
- mosaic tip 233
- Motion 243
- move 78, 108, 110
  - float 108
  - floating selection 108
  - hints 109
  - hole image 108
  - layers 210
  - nudge 109
  - object floating selection 219
  - path 384
  - selections 109
- mpeg 61
- multiply mode 21, 189
- munsell 133

N

- name 120
- naming layers 202
- natural color system 133
- new layer 201
- new view 105, 109
- nl filter 290
- normal mode 188
- normalize 170
- nudge 109, 110

O

- offset 170
  - move 170
  - tile 171
  - tip 170
- oil paint 134
- oilify 234
- Olof S Kylander xxxix
- opacity 99, 120
- overlay mode 21, 190

P

- paintbrush 96
  - fade out 97
- palette 91, 121, 178
  - add color 91
  - create from image 91
  - delete 121
  - dialog 121
  - directory 45
  - edit 121
  - from image 122
  - hints 122
  - new 121
  - save 178
- Pantone 133, 140
- paper tile 335
- paste 102
  - into 102
  - named 102
- pat 61, 64
- pattern
  - directory 45
  - make 121
- pattern fill 92
- pcx 61
- pdf 61
- pencil 96
- perspective 112, 403
- pinch 280
- pix 61
- pixelize 244
- plasma 374
- plug-in
  - compile programs 452
- plug-ins 224, 452
  - categories 225
  - compile 452

  configure 452
  directory 45
  filters 224
  gcc 454
  make 452
  see also filters
  what is 224
plug-ins see also filters
png 61, 65
pnm 61, 65
pop up 120
postscript 58
power goo 270
ppi 137, 138, 139
preference 69
  directories 71
  display 69
  display interpolation 69
  environment 70
  interface 70
  suggestions 71
  swap 71
preparing prepress 138
prepress 136
  AM screening 151
  bbs 140
  Caldera 142
  calibration 142
  calibration black level 143
  calibration gamma 143
  calibration white level 143
  color 139
  color calibration 145
  color calibration cms 145
  color calibration Frozenriver 145
  color calibration plain 145
  color calibration poor man 147
  color calibration rgb & cmyk 147
  color management 148
  color proof 139
  dots 138
  dpi 138, 149
  email 141
  eps 136
  file formats 136
  file transfer 140
  filesystem format 141
  FM screening 151
  ftp 140
  halftone 138
  halftone cell 149
  halftone dot 150
  halftone matrix 151
  image size 137
  image table 153
  internet 140
  interpolation 138
  lpi 138, 149
  lpi table 151, 153
  Mentalix 142
  modem 140
  Pantone 140
  ppi 137, 138, 139
  preparing 138
  printer table 153
  printing 136
  removable drives 140
  resolution 137, 149
  resolution grid 150
  sane 141
  scan resolution 139
  scanning 141
  scanning Caldera 142
  scanning Mentalix 142
  scanning resolution 138
  scanning sane 141
  scanning XVscan 142
  screen frequencies 149
  screening matrix 154
  shades of gray 154
  spot color 140
  stochastic screening 151
  tiff 136
  XVscan 142

pressure 98
primary 130
print 68
    ghostscript 68
    settings 68
    supported printers 68
printer table 153
printing 136
profile 148
ps 61, 65
    resolution 65
psd 61

Q

qbist 375
quantize 253
quit 55

R

radial 94
raise layer 202
randomize 342
redo 104
refract 303
removable drives 140
replicate 125
resize 110, 176
resize layer 203
resolution 137, 149
    grid 150
retouch 98
rgb 55, 130, 148, 175
    channels 212
    composing 172
    convert 175
    decomposing 172
    invert 159
    levels 167
    max 252
ripple 275
rle 242
rotate 111, 179
rulers 105

S

sample merged 90, 92, 185
sane 47, 141
saturation 21, 132, 162
    mode 192
saturation mode 21
save 125, 183
    native file format 59
    palette 178
    to channel 183
sawtooth 96
scale 111, 176
scale layer 202
scanning resolution 138, 139
scatter hsv 253
screen mode 189
screen frequency 149
screen shot 72
screening matrix 154
script-fu 396
    carve-it 401
    chrome-it 401
    directory 45
    don’t and do:s 396
    drop shadow 403
    gradient 399
    image dependent 400
    install 396
    kinds of 397
    perspective 403
    stand alone 397
    what is 396
script-fu how to write 406
section 123
select all 182
select by color 184
select by color modes 184
select by color options 185

select non 182
selection 109, 110
  all 182
  alpha 205
  alpha edit 213
  alpha store 213
  antialiasing 81
  bezier 83
  bezier control points 84
  bezier control points move 84
  bezier sharp corners 85
  border 183
  by color 184
  by color options 185
  by color, modes 184
  color 83
  control 76
  difference 77
  ellipse 79
  feather 81, 182
  float 182
  floating move 219
  floating tips 219
  floating what is 218
  free-hand 82
  fuzzy 82
  gfig 358
  grow 183
  guide 78
  intelligent scissors 85
  intersection 77
  invert 182
  mask 78
  mask to 206
  move 78
  non 182
  options 81
  rectangular 79
  sharpen 183
  shrink 183
  toggle 76, 182
  union 76
semi flatten 253
sgi 61
shades of gray 154
shadow 161
shapeburst 95
sharpen 99, 183, 291
shearing 112
shift 275
short cuts 11–16
shrink 183
shrink warp 109
shrink wrap 106
sinus 376
slant 116, 445
small tile 335
smooth palette 254
smoothing 113
snap to guides 110
anchor layer 204
snp 61
sobel 285
solid noise 378
source multiple 456
source unpack 453
spacing 99
sparkle 318
spot color 133, 140
spread 343
square 94
stegano 265
stereogram 339
stochastic screening 151
stroke 104
  brush selection 104
subtraction mode 191
sunras 61, 66
super nova 319
support
  commercial xl
symbols 201
symmetric asymmetric 95
SyQuest 140

T

tar 453
text 116
text tool 116
  blackness 116
  border 116, 117
  c 116
  foundry 116
  italic 116
  m 116
  p 116
  regular 116
  roman 116
  slant 116
  spacing 116
  weight 116
  width 116
texture explorer 375
texture explorer see also cml explorer
textures 234, 346, 349, 350, 376
tga 61, 66
threshold 134, 160
  alpha 178
thumbnail icon 200
tiff 61, 67, 136
  compression 67
  fill order 67
  mac 67
  pc 67
tile 336
  offset 171
tile small 335
tip of the day 71
TODO xliv
toggle 76, 105, 182
  marching ants 182
transform 110
  autocrop 178
  options 113
  perspective 112
  rotate 111
  scale 111
  shearing 112
  smoothing 113
  zealous crop 179
triangular 96
truecolor 130
Truematch 133
twist 21, 276

U

undo 104, 387
universal filter 299
unpack 453
url 62
user filter 300

V

value 132
value mode 193
value propagate 278
van gogh 234
variables 458
video 340

W

warp 236
water colors 72
watercolor 134
waterselect 72
waves 279
weight 116
wget 47
what is modes 188
whirl 280
white level calibration 143
width 116, 445
weight 445
window info 106

X

xcf 59, 62
xpm 62, 67
XVscan 142
xwd 62

Z

zealous crop 179
zip 140
zoom 104, 109
    allow window resizing 106, 109