GNU Image Manipulation Program
Technology White Paper

Part I
Introduction

The **G**NU[1] **I**mage **M**anipulation **P**rogram (GIMP) is a powerful tool for the preparation and manipulation of digital images. The GIMP provides the user with a wide variety of image manipulation, painting, processing, and rendering tools. The key to the GIMP's power lies in its flexible core and easily extensible design. The GIMP's open design and extensible architecture make for a very powerful end product which can continue to be extended to meet the needs of the photo compositor, image retoucher, web graphics designer, or digital illustrator for a very long time.

The GIMP's extensible plug-in architecture allows for image manipulation procedures and other functionality to be easily added without requiring any change to the application core. A plug-in can provide functionality as simple as rotating an image, or as complicated as rendering iterated function system fractals. There are nearly 150 plug-ins available in version 1.0, and more are sure to follow.

The plug-in architecture also allows the GIMP to support a wide variety of file formats. File operations are implemented by special *file plug-ins*, allowing additional file formats to be added without modification to the core. File formats supported in version 1.0 include the popular GIF and JPEG standards, as well as PNG, TIFF, XPM, SGI, PCX, and Windows BMP.

An innovative tile management system allows the GIMP to be used to edit images much larger than can be stored in system memory; the user's available disk space is the only real limit to the size of the image a user can edit.

[1] Please see `http://www.gnu.org` for information on the GNU Project.

While the GIMP's primary emphasis is on image manipulation, it also offers a complete set of painting tools for use in image creation. Version 1.0 offers pencil, paintbrush, eraser, airbrush, and cloning tools, as well as a variable-strength convolver. All of these tools (except the eraser) can be operated in any of the GIMP's 15 painting modes. A powerful gradient generator, with a very versatile custom gradient editor, makes colored blends easy.

A full battery of image manipulators are available, including rotation, scaling, translation, color, brightness, and contrast adjustment, and gamma correction. In addition, a great many other useful transformations are available as plug-ins, and the ease of extensibility here means that new capabilities are being added all the time.

Finally, the most impressive feature of the GIMP is that it is available under the terms of the GNU General Public License (GPL).[2] The entire source code is freely available and distributed. This openness has fostered a very active development community and large user base, out of which a superior product has arisen.

Part II
Core architecture

1 Images

The basic operating element of any digital image editor is the *image*. In the GIMP, images are constructed out of *layers*, which are stacked on top of on another through a process called *composition* to produce a *projection*, which is what is displayed to the user. In addition to having any number of layers, a GIMP image may have one or more user-defined *channels*, as well as a *selection mask*. Channels and selection masks are discussed later. Together, these three objects (layers, channels, and selection masks) along with *layer masks* (also to be discussed later) are known as *drawables*, because the drawing tools work on all of them and they are the only internal structures into which the program can paint to.[3]

Images in the GIMP are typed, and there are presently three types of image:

[2] Please see `http://www.gnu.org/copyleft/gpl.html` for the full text of the GPL.

[3] There is presently no way to draw directly onto a selection mask using the user interface. However, the user can create an invisible channel, paint there, and ask the program to turn the channel into a selection mask.

Type	Data channels	Contents of channels
RGB	3	red, green, blue
RGB w/ alpha	4	red, green, blue, alpha
Grayscale	1	intensity
Grayscale w/ alpha	2	intensity, alpha
Indexed	1	color index
Indexed w/ alpha	2	color index, alpha

Table 1: Drawable Types

RGB, grayscale, and indexed. The type of image determines the representation of the pixels in the image. In an RGB image, each pixel of the composited image is represented by a 24-bit RGB tuple; all 16 million possible colors are potentially available in the composited image. Grayscale images are monochromatic, and each pixel is a single 8-bit gray value, yielding 256 shades of gray. Indexed images represent each pixel as an index into a color table, each entry of which is a 24-bit RGB tuple. The type of all layers within an image must be compatible with the image type.

2 Drawables

A *drawable* is a linear array of pixel data; however, the contents of a drawable need not necessarily be used solely for rendering as pixel data (for example, selection masks are drawables internal to the program, and they are not directly "paintable" by the user).

Each drawable contains from one to four data channels (not to be confused with the channels spoken of elsewhere in this document), depending on the type of the drawable. Each data channel is eight bits deep.

There are six types of drawables (see table **??**).

2.1 Layers

Each *layer* is a drawable. Layers of any drawable type are possible, but the type of a layer must be compatible with the type of the image of which it is a part. A layer type is compatible with an image type if the two are the same, or if the layer type is the same as the image type with an added alpha channel (for example, a

grayscale image cannot contain an RGB layer). Every layer is part of exactly one image.

The user may selectively make any layer of an invisible; invisible layers do not contribute to the composition process.

2.1.1 Layer masks

Optionally, any layer with an alpha channel may have an added *layer mask*. The layer mask is a separate channel which is multiplied into that layer's drawable alpha channel whenever that layer contributes to a projection. The user can elect to temporarily disable this effect of a layer mask for editing purposes, or cause the layer mask to be substituted for the main drawable; these effects are mainly for convenience in editing. The user can also merge the layer mask back into the layer's main drawable, or discard it.

2.2 Channels

The term "channels" actually refers to three different things in the GIMP: layer masks, selection masks, and custom channels. In all three cases, a channel consists of an array of 8-bit values; the interpretation of these values varies depending on the type of channel.

In addition to these channel types, each image also has either one or three "virtual" channels (one for grayscale and indexed images, three for RGB). These virtual channels (sometimes called "color channels") are merely a convenience to the user and are not drawables by themselves. These virtual channels can be made visible or invisible, and for images where there is more than one virtual channel, each can be made visible or invisible independently of the others. Normally all the virtual channels are active, and all painting operations modify all three the channels. When a virtual channel is invisible, both layer composition and layer painting operations ignore that channel. This is useful in two situations: first, when the user wants to see only a subset of the color channels (RGB); and second, when the user wants to paint on the image but only modify a subset of the channels.

3 Tiles

4 Pixel Regions

5 The Paint Core

5.1 Brushes

All brushes in gimp are implemented as greyscale bitmap masks. They are 8-bit greyscale and provide 256 levels of grey.

The format for the Brush file is the .gbr format (Gimp brush). Gimp includes a file plugin to load and save this format, so brushes can be created and edited using GIMP itself.

These brushes can be used by any of the paint tools (paintbrush, airbrush, clone, etc).

Brushes are completely customizable and are not limited to simple pre-made shapes. Any greyscale image can conceivibly used as a brush.

6 Selections

7 Tools

8 xcf format

8.1 Gradient tool

The gradient tool lets the user easily create color gradients. It supports several gradient types: linear, bilinear, radial, square, conical, and the special “shapeburst” mode, which makes the gradient follow the shape of the active selection.

Normally, the blend tool creates color gradients between the current foreground and background colors. It supports four blending modes: linear interpolation between the RGB components, linear interpolation between the HSV versions of the colors (for rainbow effects), blending from the foreground color to transparent, and the very powerful custom gradient feature.

Gradients are calculated with respect to a direction vector. The user can specify whether the gradient is to be rendered normally (only once along the direction

vector), or repeated along the direction vector using a sawtooth or triangular wave pattern.

A unique feature of the GIMP's gradient tool is its support for user-defined, custom color gradients. The GIMP sports a fully-featured color gradient editor that lets the user create color gradients with an arbitrary number of color transitions.

A custom gradient is represented internally as a list of contiguous, non overlapping segments that define a partition of the range $[0, 1]$. Each segment has the following properties:

- Left and right colors that blend smoothly inside the segment.
- An off-center midpoint, which can be used to bias the color blend to the left or to the right.
- A blending function, which can be linear, curved (with an exponent), sinuosidal, or spherical (increasing and decreasing).
- A coloring type, which can be RGB interpolation, clockwise or couterclockwise HSV interpolation.

The user can drag the segments' endpoints left and right. Segments can be inserted and deleted at any time. The gradient editor provides several useful functions for manipulating segments (split, flip, replicate) that make creation of custom gradients easy and convenient. The color segments support full transparency information, making for even more flexible gradients.

To avoid sampling artifacts (the "jaggies"), the gradient rendering engine supports adaptive supersampling with customizable threshold and recursion depth parameters. With adaptive supersampling even the most complex custom gradients will be rendered smoothly without artifacting.

*** *FIXME: drawing or screenshot?* ***

Part III
Plug-ins and the PDB

9 Procedures

The GIMP core consists of 215[4] procedures which operate on images in a great variety of ways.

The procedures register themselves in the Procedural Database (or PDB). All plugins have access to all the procedures in the PDB, and most of the scripting extentions provide full access to the PDB. The PDB is basically an API for programming high level internal GIMP functions and procedures.

*** *FIXME: General overview* ***

[4] In version 0.99.16. This number tends to go up with time.