Image processing with Gimp, part 5

PLUGGED IN

SIMON BUDIG

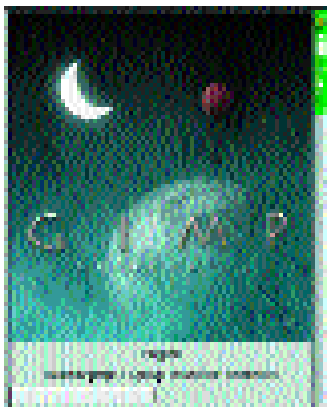


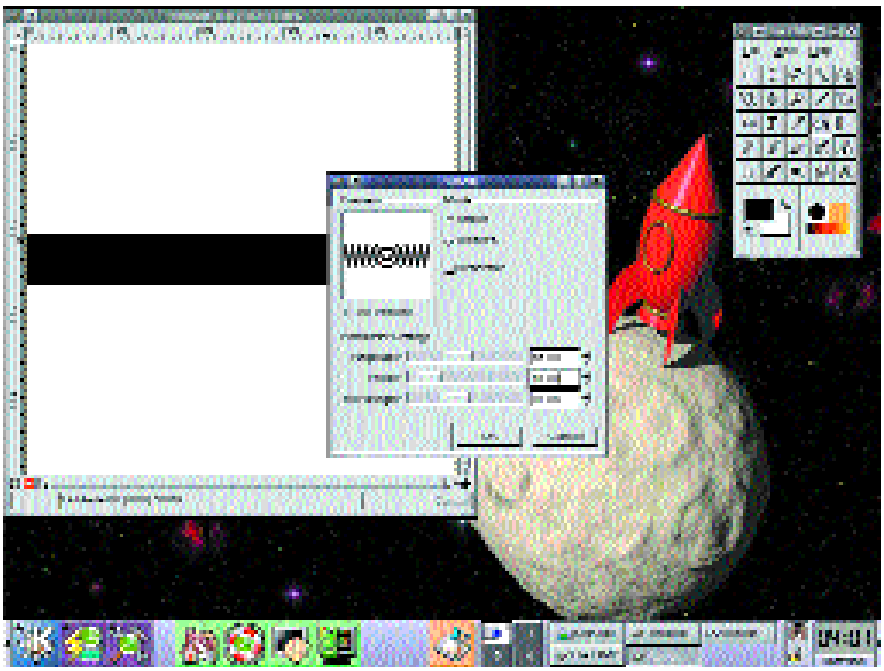
Figure 1: This is where plugins are checked

One large area which we have barely touched on in this series is plugins. These are separate little programs that take over special tasks and are included in the Gimp menu.

When starting Gimp for the first time, the dialog from Figure 1 is visible for a relatively long time. This is when the search for new plugins takes place. These are asked where they should be included in the menu. The search results are stored in the file *~/.gimp-1.2/pluginrc* – hence subsequent starts will be noticeably quicker.

We had the idea initially of arranging all submenus of *<Image>/Filters* and *<Image>/Script-Fu* on the screen and making an impressive screenshot, but then we changed our minds – there are just too many. So for this part of the Workshop we're faced with something of a dilemma: Which plugins should we present?

Figure 2: First make a few waves...



You may already have been wondering how wecreated the abstract pattern from the last part. Basically, we used two *Distorting* plugins for this.

Open a new image (for example 500x500 pixel) and select, in the centre of the image, a rectangular area (in our example from (0, 220) to (500, 280)). Fill in this area with black colour. With *Select/None*, deselect this again. Now start the plugin *Filter/Distorts/Waves*. With this plugin the image becomes distorted as if one had thrown a stone into the (liquid) surface of the image. Using the lower slide control you can affect the waves – in the preview you will then see what awaits you. In our example we've taken the values 45, 40 and 20 for the amplitude, phase and wavelength respectively (Figure 2).

In the next step this image is distorted spirally. The easiest way to do this is with *Filter/Distort/Whirl and Pinch* (Figure 3). Make sure that white is set as the background colour, otherwise strangely coloured areas will appear at the edge (where, as it were, something is being screwed out of nothingness). We've rotated the image to the max here (360 degrees). If you like, you can also use the *Pinch Amount* slide control to contract or expand the centre of the image.

So now we have something slightly similar to what was used in the last part as an example. But the edge does not look especially great. It looks a bit frayed. Interestingly, a soft focus filter can help to create a neater edge.

We came across the main soft focus filters last time: */Filter/Soft focus/Gaussian blur*. This plugin comes in two variants, although their results are

only marginally different. Normally it makes very little difference whether one uses *IIR* or *RLE*. The soft focus tool can be used for various purposes. In the last part we saw how to use it for light and shadow effects. But it can also be used to create neater anti-aliasing. Yes, that may sound daft, but it works.

Draw the image blurred with a radius of about 10. This will necessarily mean losing details, but in our special case, this is what we want. The little indentations are meant to disappear. Incidentally, it is standard practice for the radii in horizontal and vertical directions to be linked to each other, which means they always have the same value. If this is not what you want, you must click on the little button with the chain link.

Here comes the trick: Using the *Values* tool already introduced in the second part, (*<Image>/Image/Colours/Levels*) you can now create a clean, sharp edge. We have achieved good results with the source values 115, 1.0 and 145 (Figure 5).

Now we can simply copy the steps from the last part. Duplicate the levels, let a soft focus run over the lower levels... er, hang on. We forgot the transparency. In fact, we really wanted to have the black motif against a transparent background.

No problem whatsoever – there are more plugins. In this case we are helped out by *Filter/Colors/Color to alpha*. Agreeably enough, white is already set as the colour here, which is to be converted into transparency. After a click on *OK* we have the image in the form we need.

One more comment on the last plugin: *Colour to alpha* tries to make every pixel as transparent as possible, on the assumption that the colour set will be placed in a level underneath. This is sometimes a good method for making certain colours transparent, but the effect is usually stronger than desired. Depending on the task, though, it's worth a try.

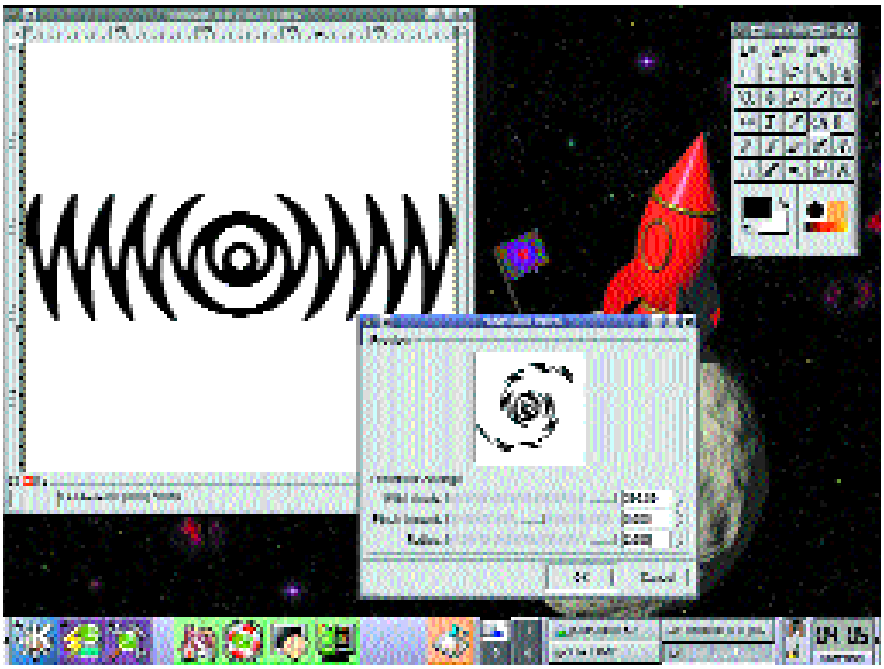


Figure 3:
...and then stir

Three-dimensional

Not so long ago in Web design, buttons which appeared somewhat three-dimensional and encouraged you to click on them were common. Even if the current trend is towards flat design, it is still good to know how to create such effects.

Gimp offers the fairly flexible bumpmap plugin for this type of effect, which you will find under *Filter/Map/Bump Map*.

Graphics tablets

I will not go into the various types of graphics tablets at this point - for this, you should refer to the article available at http://www.gtk.org/~otaylor/ximput/ .
Gimp obtains information via so-called Xinput devices (which also includes graphics tablets) through the GTK+ library. This in turn receives its information direct from the X-Server. So in order to be able to use Gimp with graphics tablets, the X-Server must be configured such that the graphics tablets are recognised as Xinput devices. Typically, this happens via /etc/X11/XF86Config, for which details can be found in the article mentioned above.
To support GTK+ one doesn't have to do very much, since normally all the current packages support Xinput devices as a matter of course. Formerly it might have been necessary to compile GTK+ yourself, since the support for XInput had to be explicitly selected (the ./configure switch is called —with-xinput).
To be able to make use of the expanded options within Gimp, you must now select, under File/Dialogs/Input devices the Xinput devices, by setting each one (apart from SWITCH under XFree 3.3.x) to the mode Screen. After a click on Save and Close you should now be able to paint in the image window with the paintbrush tool pressure-sensitively.
A quick word about troubleshooting, in case it doesn't work. The command xinput list lists all the input devices. If the X-Server has been correctly configured, one or more XextensionDevices should pop up here. If so and if No Input Devices appears within Gimp when the above dialog is selected, GTK+ has not been compiled correctly.

The author

The fact that so much can be written about bumpmap surprised even Simon Budig himself. In particular, when writing this article he understood for the first time how the Waterlevel parameter functions...

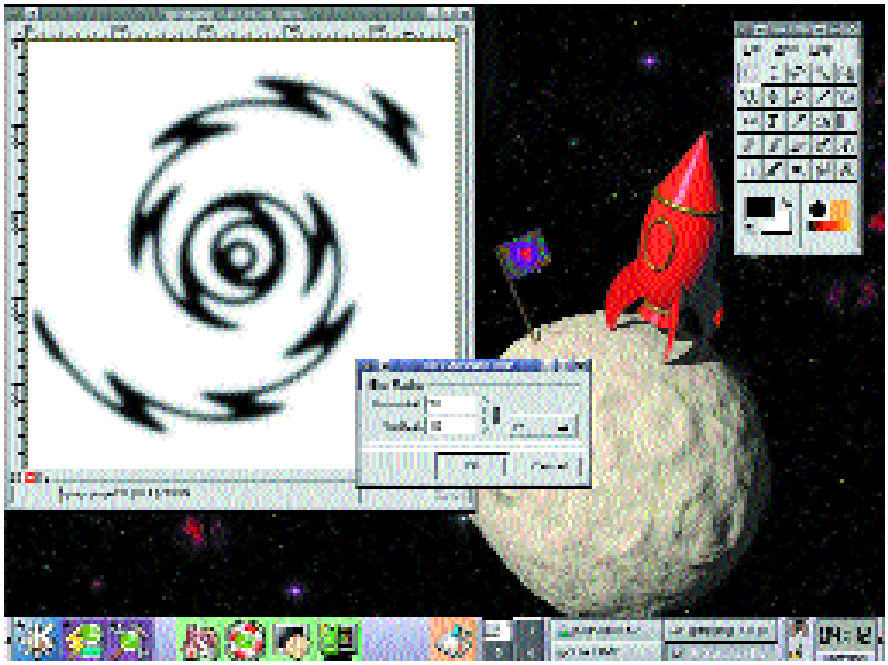
The dialog can be seen in Figure 6.

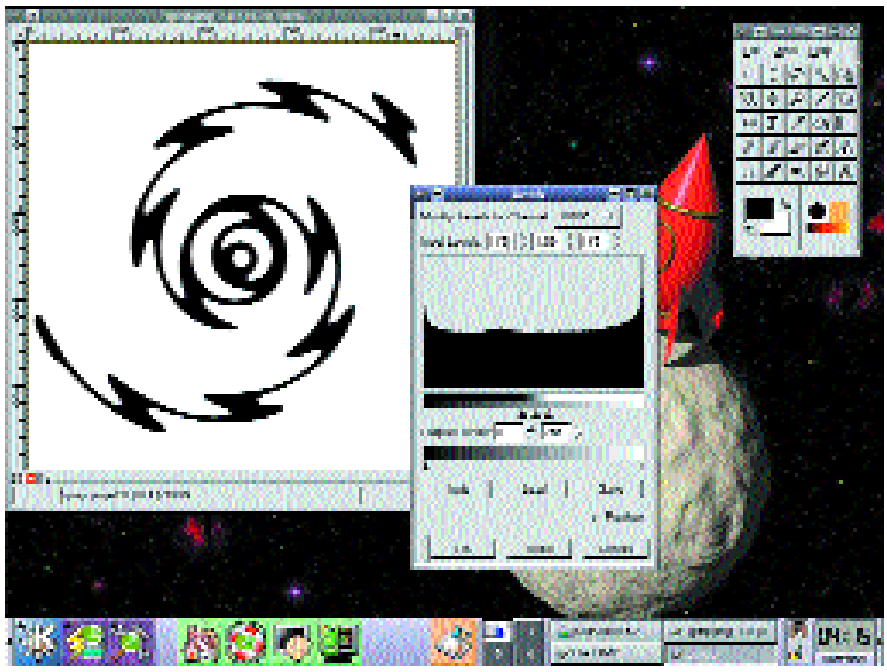
Before we leap in, a quick bit of theory. The bumpmap plugin is based on the idea that a mountain range, seen from above, still displays something of its structure, since the sun illuminates the various slopes differently. The mountain range is typically depicted within Gimp with an image in shades of grey. The different heights by various shads of grey. The valleys are black and the peaks are white.

Let's try it out. Create a new image with black background and write a white text inside. With *Filters/Blur/Gaussian blur* you can ensure that our mountain range with raised letters at the edges gets somewhat softer slopes. Now make a second image of the same size, but this time with a white background. Now start, by right-clicking in the white image *Filter/Map/Bumpmap*.

[top]
Figure 4: Soft focus for sharp edges

[below]
Figure 5: The levels dialog makes it sharp again





First of all, select within the *Parameter settings* under *Bumpmap* the image which shows our mountain range. In the preview at the top left you should see a three-dimensional effect, although you may need to scroll up or down a bit (either using the scrollbar or by clicking in the preview), until you find the interesting area.

Using *Azimuth* and *Elevation* you can define the direction from which our mountain range is to be lit. *Azimuth* defines from which side the light source shines onto the mountain range. If you pull on the slider control, you will see how the light spots move around the mountain range. *Elevation* defines from what height above the horizon the light source shines onto the mountain range. Play around with it a little to get a feel for the parameters. After a click on *OK* the effect will be included in the image.

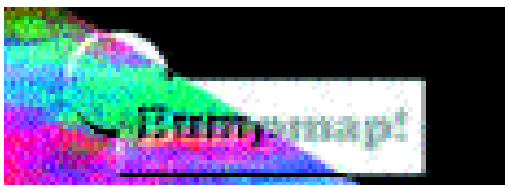
The *Depth* parameter controls the maximum height of the mountain range. The greater this is, the steeper the slopes, so the illumination gains a stronger contrast.

Now it gets lively

So far we have only found out about the basic variants of the bumpmap effect. Obviously the plugin cannot be used only on white images – because then only grey shaded results would ever come out. In Figure 7 we have applied the bumpmap plugin to a coloured image. In the right upper area you can see the grey-shaded map, at lower left the effect it has on a coloured image. In this case we have created the coloured pattern with *Filter/Render/Clouds/Plasma*.

Let's take a look at the parameters in the dialog at top right (Figure 6). The *Map Type* states how the grey shades are converted into height levels. In Figure 8 you can see the three basic functions: linear, spherical and sinusoidal. A short aid to interpretation: If you have an even gradation of colour on one edge from black to white, the various functions convert this into a straight, quadrant-shaped or sinusoidal slope. In the image you can see how the various forms affect a small pyramid. *Linear* produces a pointed cone, *spherical* a sort of drop of mercury and *sinusoidal* a gentle hill, which tapers out gently at the peak and at the bottom.

The bumpmap algorithm has the extremely undesirable side effect that the image becomes noticeably darker. If the rays of light strike it at an angle, the area is not completely illuminated. In order to reduce this effect, you can use the option



Top right the bumpmap, lower left the effect created

Compensate for darkening. The option *Invert bumpmap* interprets the grey shade image in exactly the opposite way — white for valleys, black for the height features. In principle one could achieve the same effect by rotating the light source to the opposite side (± 180°) – this button is a shortcut.

If the bumpmap is smaller than the image to which it is applied, it can be repeated with the option *Tile bumpmap* and thus create an even surface structure. The parameters *X/Y Offset* displace the bumpmap with respect to the image. With *Ambient* you can control the diffuse lighting. The higher this value, the brighter the side facing away from the light source appears.

To clarify the parameter *Waterlevel*, we will have to go back a bit to present one neat trick. The bumpmap plugin can also evaluate transparency information as height information. This means that if a white spot goes transparent, a hill appears after applying the effect. But if a black spot goes transparent, a hollow appears. How strongly a hill stands out or how deeply a hollow is indented is defined by the waterlevel.

Take a look at Figure 9. On the left you can see how spots in various shades of grey become transparent against a green background. Next to this you can see the images which are created when you apply this bumpmap with the water heights 0, 64, 128, 192 and 256 to a white image. Bear in mind that with a water height of 0 only hills, and with a water height of 256, only hollows will be produced (the light comes from top left).

Contours

The three different functions for affecting the shape of the contour are all very nice, but sometimes you need more control. The bumpmap plugin alone cannot do it, but it is very easy to foist various contours onto the plugin. To do this, you can use the*Image/Colours/Curves* dialog.

Start again with a blurred (radius approx. 10 pixels) white text against a black background. Now open the curve dialog and model a contour, similar to the one you can see in Figure 10.

Now create a new image of the same size. We have created a plasma effect again, in order to make something colourful. Then apply the bumpmap plugin with the signature. The result should look something like Figure 11.

Hmm, we have to admit it – from the foray through the menu we promised last time, all we've covered is a couple of distortion plugins and an intensive discussion of the bumpmap plugin. We promise, that next time we will be dealing with more colourful matters. And we will restrain ourselves when it comes to Gimpressionist – which has about ten times as many parameters as bumpmap – and keep strictly to the main points.

Gimp's plugins are flexible -– but with a bit of creativity and combining one can achieve considerably more. Have a go and see what happens when you combine different plugins with each other. If you stumble across any interesting effects, don't hesitate to email a brief description (sbudig@linux-user.de). ■

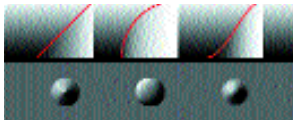


Figure 8: Image functions forming the basis of the bumpmap plugin

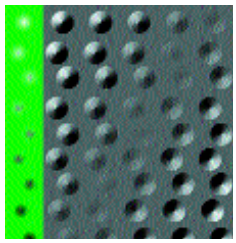


Figure 9: The effect of the *Waterlevel* parameter

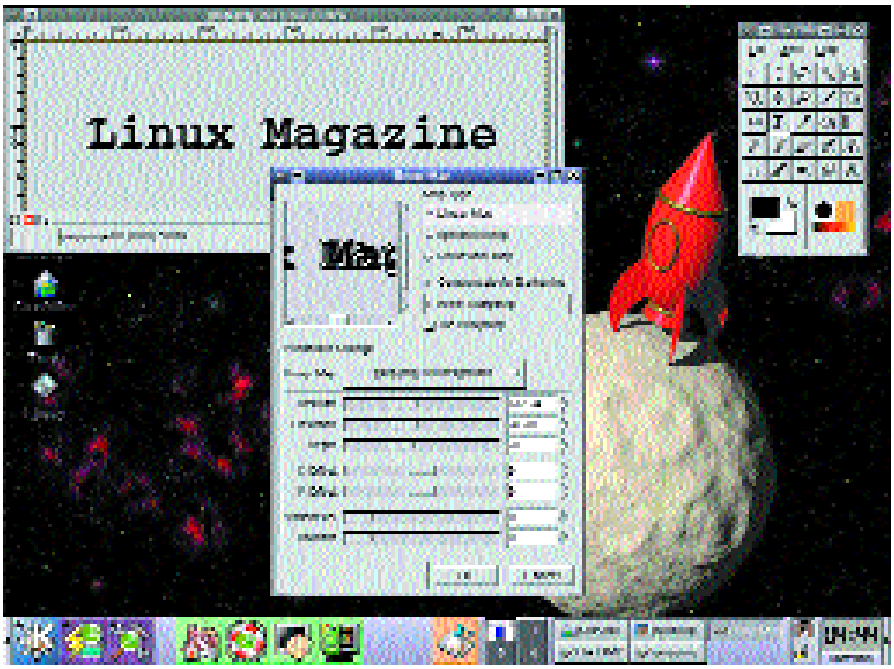


Figure 6: Lots of parameters for the bumpmap plugin

Figure 10: Using the curve tool to create a contour for bumpmap

Figure 11: The result: A text with a fancy contour