fractint3

COLLABORATORS

	TITLE : fractint3	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		July 20, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

fractint3

1.1 Doodads, Bells, and Whistles

1.2 Drawing Method

The "passes option" (<X> options screen or "passes=" parameter)
selects one of the single-pass, dual-pass, triple-pass, solid-guessing
(default), boundary tracing, or tesseral modes. This option applies
to most fractal types.

Single-pass mode ("1") draws the screen pixel by pixel.

Dual-pass ("2") generates a half-resolution screen first as a preview
using 2x2-pixel boxes, and then generates the rest of the dots with a
second pass. Dual-pass uses no more time than single-pass.

Triple-pass ("3") generates the coarse first pass of the solidguessing
mode (see "g" below), then switches to either "1" (with low resolution
video modes) or "2" (with higher resolution video modes). The advantage
of '3' vs '2' is that when using high resolution modes, the first pass has
a much lower resolution (about 160x120) and is therefore much quicker than
the first pass of the passes=2 mode. However, with the '2' mode, the first
pass does not represent wasted time. The '3' mode wastes the effort of
generating the coarse first screen.

The single, dual, and triple pass modes all result in identical images.

These modes are for those who desire the highest possible accuracy. Most people will want to use the guessing mode, described next.

Solid-guessing ("g") is the default. It performs from two to four visible passes -
more in higher resolution video modes. Its first visible pass is actually two passes - one pixel per 4x4, 8x8, or 16x16 pixel box is generated, and the guessing logic is
applied to fill in the blocks at the next level (2x2, 4x4, or 8x8). Subsequent passes fill in the display at the next finer resolution, skipping blocks which are surrounded by the same color. Solid-guessing can guess wrong, but it sure guesses quickly!

Boundary Tracing ("b"), which only works accurately with fractal types (such as the Mandelbrot set, but not the Newton type) that do not contain "islands" of colors, finds a color boundary, traces it around the screen, and then "blits" in the color over the enclosed area.

Tesseral ("t") is a sort of "super-solid-guessing" option that successively divides the image into subsections and colors in rectangles that have a boundary of a solid color. It's actually slower than the solid-guessing algorithm, but it looks neat, so we left it in. This mode is also subject to errors when islands of color appear inside the rectangles.

The "fillcolor=" option in the <X> screen or on the command line sets a fixed color to be used by the Boundary Tracing and Tesseral calculations for filling in defined regions. The effect of this is to show off the boundaries of the areas delimited by these two methods.

1.3 Autokey Mode

The autokey feature allows you to set up beautiful self-running demo "loops". You can set up hypnotic sequences to attract people to a booth, to generate sequences for special effects, to teach how Fractal exploring is done, etc.

A sample autokey file (DEMO.KEY) and a batch to run it (DEMO.BAT) are included with Fractint. Type "demo" at the DOS prompt to run it.

Autokey record mode is enabled with the command line parameter "AUTOKEY=RECORD". Keystrokes are saved in an intelligible text format in a file called AUTO.KEY. You can change the file name with the "AUTOKEYNAME=" parameter.

Playback is enabled with the parameter "AUTOKEY=PLAY". Playback can be terminated by pressing the <Esc> key.

After using record mode to capture an autokey file, you'll probably want to touch it up using your editor before playing it back.

Separate lines are not necessary but you'll probably find it easier to understand an autokey file if you put each command on a separate line. Autokey files can contain the following:

  Quoted strings. Fractint reads whatever is between the quotes just as if

you had typed it. For example,
    "t" "ifs"
issues the "t" (type) command and then enters the letters i", "f", and "s" to select the ifs type.

Symbols for function keys used to select a video mode. Examples:
    F3 -- Function key 3
    SF3 --<Shift> and <F3> together

Special keys: ENTER ESC F1 PAGEUP PAGEDOWN HOME END LEFT RIGHT UP DOWN INSERT DELETE TAB

WAIT <nnn.n> -- wait nnn.n seconds before continuing

CALCWAIT -- pause until the current fractal calculation or file save or restore is finished. This command makes demo files more robust since calculation times depend on the speed of the machine running the demo - a "WAIT 10" command may allow enough time to complete a fractal on one machine, but not on another. The record mode does not generate this command - it should be added by hand to the autokey file whenever there is a process that should be allowed to run to completion.

GOTO target -- The autokey file continues to be read from the label "target". The label can be any word that does not duplicate a key word. It must be present somewhere in the autokey file with a colon after it. Example:
    MESSAGE 2 This is executed once
    start:
    MESSAGE 2 This is executed repeatedly
    GOTO start
GOTO is mainly useful for writing continuous loop demonstrations. It can also be useful when debugging an autokey file, to skip sections of it.

; -- A semi-colon indicates that the rest of the line containing it is a comment.

MESSAGE nn <Your message here> -- Places a message on the top of the screen for nn seconds

Making Fractint demos can be tricky. Here are some suggestions which may help:

Start Fractint with "fractint autokeyname=mydemo.key autokey=record". Use a unique name each time you run so that you don’t overwrite prior files.

When in record mode, avoid using the cursor keys to select filenames, fractal types, formula names, etc. Instead, try to type in names. This will ensure that the exact item you want gets chosen during playback even if the list is different then.

Beware of video mode assumptions. It is safest to build a separate demo for different resolution monitors.

When in the record mode, try to type names quickly, then pause. If you pause partway through a name Fractint will break up the string in the .KEY file. E.g. if you paused in the middle of typing fract001, you

```
might get:
    "fract"
    WAIT 2.2
    "001"
No harm done, but messy to clean up. Fractint ignores pauses
less than about 1/2 second.

DO pause when you want the viewer to see what is happening during
playback.

When done recording, clean up your mydemo.key file. Insert a CALCWAIT
after each keystroke which triggers something that takes a variable
amount of time (calculating a fractal, restoring a file, saving a file).

Add comments with ";" to the file so you know what is going on in
future.

It is a good idea to use INSERT before a GOTO which restarts the demo.
The <insert> key resets Fractint as if you exited the program and
restarted it.
```

Warning: an autokey file built for this version of Fractint will probably require some retouching before it works with future releases of Fractint. We have no intention of making sure that the same sequence of keystrokes will have exactly the same effect from one version of Fractint to the next. That would require pretty much freezing Fractint development, and we just love to keep enhancing it!

1.4 Distance Estimator Method

This is Phil Wilson's implementation of an alternate method for the M and J sets, based on work by mathematician John Milnor and described in "The Science of Fractal Images", p. 198. While it can take full advantage of your color palette, one of the best uses is in preparing monochrome images for a printer. Using the 1600x1200x2 disk-video mode and an HP LaserJet, we have produced pictures of quality equivalent to the black and white illustrations of the M-set in "The Beauty of Fractals."

The distance estimator method widens very thin "strands" which are part of the "inside" of the set. Instead of hiding invisibly between pixels, these strands are made one pixel wide.

Though this option is available with any escape time fractal type, the formula used is specific to the mandel and julia types - for most other types it doesn't do a great job.

To turn on the distance estimator method with any escape time fractal type, set the "Distance Estimator" value on the <Y> options screen (or use the "distest=" command line parameter).

Setting the distance estimator option to a negative value -nnn enables edge-tracing mode. The edge of the set is display as color number nnn. This option works best when the "inside" and "outside" color values are also set to some other value(s).

In a 2 color (monochrome) mode, setting to any positive value results in the inside of the set being expanded to include edge points, and the outside points being displayed in the other color.

In color modes, setting to value 1 causes the edge points to be displayed using the inside color and the outside points to be displayed in their usual colors. Setting to a value greater than one causes the outside points to be displayed as contours, colored according to their distance from the inside of the set. Use a higher value for narrower color bands, a lower value for wider ones. 1000 is a good value to start with.

The second distance estimator parameter ("width factor") sets the distance from the inside of the set which is to be considered as part of the inside. This value is expressed as a percentage of a pixel width, the default is 71. Negative values are now allowed and give a fraction of a percent of the pixel width. For example: -71 gives 1/71 % of the pixel width.

You should use 1 or 2 pass mode with the distance estimator method, to avoid missing some of the thin strands made visible by it. For the highest quality, "maxiter" should also be set to a high value, say 1000 or so. You'll probably also want "inside" set to zero, to get a black interior.

Enabling the distance estimator method automatically toggles to floating point mode. When you reset distest back to zero, remember to also turn off floating point mode if you want it off.

Unfortunately, images using the distance estimator method can take many hours to calculate even on a fast machine with a coprocessor, especially if a high "maxiter" value is used. One way of dealing with this is to leave it turned off while you find and frame an image. Then hit <B> to save the current image information in a parameter file (see Parameter Save/Restore Commands). Use an editor to change the parameter file entry, adding "distest=1", "video=something" to select a high-resolution monochrome disk-video mode, "maxiter=1000", and "inside=0". Run the parameter file entry with the <@> command when you won't be needing your machine for a while (over the weekend?)

1.5 Inversion

Many years ago there was a brief craze for "anamorphic art": images painted and viewed with the use of a cylindrical mirror, so that they looked weirdly distorted on the canvas but correct in the distorted reflection. (This byway of art history may be a useful defense when your friends and family give you odd looks for staring at fractal images color-cycling on a CRT.)

The Inversion option performs a related transformation on most of the fractal types. You define the center point and radius of a circle; Fractint maps each point inside the circle to a corresponding point outside, and vice-versa. This is known to mathematicians as inverting (or if you want to get precise, "everting") the plane, and is something they can contemplate without getting a headache. John Milnor (also mentioned in connection with the Distance Estimator Method), made his name in the 1950s with a method for everting a seven-dimensional sphere, so we have a

lot of catching up to do.

For example, if a point inside the circle is 1/3 of the way from the center to the radius, it is mapped to a point along the same radial line, but at a distance of (3 * radius) from the origin. An outside point at 4 times the radius is mapped inside at 1/4 the radius.

The inversion parameters on the <Y> options screen allow entry of the radius and center coordinates of the inversion circle. A default choice of -1 sets the radius at 1/6 the smaller dimension of the image currently on the screen. The default values for Xcenter and Ycenter use the coordinates currently mapped to the center of the screen.

Try this one out with a Newton plot, so its radial "spokes" will give you
something to hang on to. Plot a Newton-method image, then set the inversion radius to 1, with default center coordinates. The center "explodes" to the periphery.

Inverting through a circle not centered on the origin produces bizarre effects that we're not even going to try to describe. Aren't computers wonderful?

1.6 Decomposition

You'll remember that most fractal types are calculated by iterating a simple function of a complex number, producing another complex number, until either the number exceeds some pre-defined "bailout" value, or the iteration limit is reached. The pixel corresponding to the starting point is then colored based on the result of that calculation.

The decomposition option ("decomp=", on the <X> screen) toggles to another coloring protocol. Here the points are colored according to which quadrant of the complex plane (negative real/positive imaginary, positive real/positive imaginary, etc.) the final value is in. If you use 4 as the parameter, points ending up in each quadrant are given their own color; if 2 (binary decomposition), points in alternating quadrants are given 2 alternating colors.

The result is a kind of warped checkerboard coloring, even in areas that would ordinarily be part of a single contour. Remember, for the M-set all points whose final values exceed 2 (by any amount) after, say, 80 iterations are normally the same color; under decomposition, Fractint runs [bailout-value] iterations and then colors according to where the actual final value falls on the complex plane.

When using decomposition, a higher bailout value will give a more accurate plot, at some expense in speed. You might want to set the bailout value (in the parameters prompt following selection of a new fractal type; present for most but not all types) to a higher value than the default. A value of about 50 is a good compromise for M/J sets.

1.7 Logarithmic Palettes and Color Ranges

By default, Fractint maps iterations to colors 1:1. I.e. if the calculation for a fractal "escapes" (exceeds the bailout value) after N iterations, the pixel is colored as color number N. If N is greater than the number of colors available, it wraps around. So, if you are using a 16-color video mode, and you are using the default maximum iteration count of 150, your image will run through the 16-color palette 150/16 = 9.375 times.

When you use Logarithmic palettes, the entire range of iteration values is compressed to map to one span of the color range. This results in spectacularly different images if you are using a high iteration limit near the current iteration maximum of 32000 and are zooming in on an area near a "lakelet".

When using a compressed palette in a 256 color mode, we suggest changing your colors from the usual defaults. The last few colors in the default IBM VGA color map are black. This results in points nearest the "lake" smearing into a single dark band, with little contrast from the blue (by default) lake.

Fractint has a number of types of compressed palette, selected by the "Log Palette" line on the <X> screen, or by the "logmap=" command line parameter:

  logmap=1: for standard logarithmic palette.

  logmap=-1: "old" logarithmic palette. This variant was the only one used before Fractint 14.0. It differs from logmap=1 in that some colors are not used - logmap=1 "spreads" low color numbers which are unused by logmap=-1's pure logarithmic mapping so that all colors are assigned.

  logmap=N (>1): Same as logmap=1, but starting from iteration count N. Pixels with iteration counts less than N are mapped to color 1. This is useful when zooming in an area near the lake where no points in the image have low iteration counts - it makes use of the low colors which would otherwise be unused.

  logmap=-N (<-1): Similar to logmap=N, but uses a square root distribution of the colors instead of a logarithmic one.

  logmap=2 or -2: Auto calculates the logmap value for maximum effect.

Another way to change the 1:1 mapping of iteration counts to colors is to use the "RANGES=" parameter. It has the format:

   RANGES=aa/bb/cc/dd/...

Iteration counts up to and including the first value are mapped to color number 0, up to and including the second value to color number 1, and so on. The values must be in ascending order.

A negative value can be specified for "striping". The negative value specifies a stripe width, the value following it specifies the limit of the striped range. Two alternating colors are used within the striped range.

```
Example:
   RANGES=0/10/30/-5/65/79/32000
This example maps iteration counts to colors as follows:

    color    iterations
    -------------------
      0      unused (formula always iterates at least once)
      1       1 to 10
      2      11 to 30
      3      31 to 35, 41 to 45, 51 to 55, and 61 to 65
      4      36 to 40, 46 to 50, and 56 to 60
      5      66 to 79
      6      80 and greater
Note that the maximum value in a RANGES parameter is 32767.
```

1.8 Biomorphs

Related to Decomposition are the "biomorphs" invented by Clifford Pickover, and discussed by A. K. Dewdney in the July 1989 "Scientific American", page 110. These are so-named because this coloring scheme makes many fractals look like one-celled animals. The idea is simple. The escape-time algorithm terminates an iterating formula when the size of the orbit value exceeds a predetermined bailout value. Normally the pixel corresponding to that orbit is colored according to the iteration when bailout happened. To create biomorphs, this is modified so that if EITHER the real OR the imaginary component is LESS than the bailout, then the pixel is set to the "biomorph" color. The effect is a bit better with higher bailout values: the bailout is automatically set to 100 when this option is in effect. You can try other values with the "bailout=" option.

The biomorph option is turned on via the "biomorph=nnn" command-line option (where "nnn" is the color to use on the affected pixels). When toggling to Julia sets, the default corners are three times bigger than normal to allow seeing the biomorph appendages. Does not work with all types - in particular it fails with any of the mandelsine family. However, if you are stuck with monochrome graphics, try it - works great in two-color modes. Try it with the marksmandel and marksjulia types.

1.9 Continuous Potential

Note: This option can only be used with 256 color modes.

Fractint's images are usually calculated by the "level set" method, producing bands of color corresponding to regions where the calculation gives the same value. When "3D" transformed (see 3D Images), most images other than plasma clouds are like terraced landscapes: most of the surface is either horizontal or vertical.

To get the best results with the "illuminated" 3D fill options 5 and 6, there is an alternative approach that yields continuous changes in colors.

Continuous potential is approximated by calculating

```
   potential =  log(modulus)/2^iterations
```

where "modulus" is the orbit value (magnitude of the complex number) when the modulus bailout was exceeded, at the "iterations" iteration. Clear as mud, right?

Fortunately, you don't have to understand all the details. However, there ARE a few points to understand. First, Fractint's criterion for halting a fractal calculation, the "modulus bailout value", is generally set to 4. Continuous potential is inaccurate at such a low value.

The bad news is that the integer math which makes the "mandel" and "julia" types so fast imposes a hard-wired maximum value of 127. You can still make interesting images from those types, though, so don't avoid them. You will see "ridges" in the "hillsides." Some folks like the effect.

The good news is that the other fractal types, particularly the (generally slower) floating point algorithms, have no such limitation. The even better news is that there is a floating-point algorithm for the "mandel" and "julia" types. To force the use of a floating-point algorithm, use Fractint with the "FLOAT=YES" command-line toggle. Only a few fractal types like plasma clouds, the Barnsley IFS type, and "test" are unaffected by this toggle.

The parameters for continuous potential are:

```
    potential=maxcolor[/slope[/modulus[/16bit]]]
```

These parameters are present on the <Y> options screen.

"Maxcolor" is the color corresponding to zero potential, which plots as the TOP of the mountain. Generally this should be set to one less than the number of colors, i.e. usually 255. Remember that the last few colors of the default IBM VGA palette are BLACK, so you won't see what you are really getting unless you change to a different palette.

"Slope" affects how rapidly the colors change -- the slope of the "mountains" created in 3D. If this is too low, the palette will not cover all the potential values and large areas will be black. If it is too high, the range of colors in the picture will be much less than those available. There is no easy way to predict in advance what this value should be.

"Modulus" is the bailout value used to determine when an orbit has "escaped". Larger values give more accurate and smoother potential. A value of 500 gives excellent results. As noted, this value must be <128 for the integer fractal types (if you select a higher number, they will use 127).

"16bit": If you transform a continuous potential image to 3D, the illumination modes 5 and 6 will work fine, but the colors will look a bit granular. This is because even with 256 colors, the continuous potential is being truncated to integers. The "16bit" option can be used to add an extra 8 bits of goodness to each stored pixel, for a much smoother result when transforming to 3D.

Fractint's visible behavior is unchanged when 16bit is enabled, except

that solid guessing and boundary tracing are not used. But when you save an image generated with 16bit continuous potential:
 o The saved file is a fair bit larger.
 o Fractint names the file with a .POT extension instead of .GIF, if you didn't specify an extension in "savename".
 o The image can be used as input to a subsequent <3> command to get the promised smoother effect.
 o If you happen to view the saved image with a GIF viewer other than Fractint, you'll find that it is twice as wide as it is supposed to be. (Guess where the extra goodness was stored!) Though these files are structurally legal GIF files the double-width business made us think they should perhaps not be called GIF - hence the .POT filename extension.

A 16bit (.POT) file can be converted to an ordinary 8 bit GIF by <R>estoring it, changing "16bit" to "no" on the <Y> options screen, and <S>aving.

You might find with 16bit continuous potential that there's a long delay at the start of an image, and disk activity during calculation. Fractint uses its disk-video cache area to store the extra 8 bits per pixel - if there isn't sufficient memory available, the cache will page to disk.

The following commands can be used to recreate the image that Mark Peterson first prototyped for us, and named "MtMand":

```
TYPE=mandel
CORNERS=-0.19920/-0.11/1.0/1.06707
INSIDE=255
MAXITER=255
POTENTIAL=255/2000/1000/16bit
PASSES=1
FLOAT=yes
```

Note that prior to version 15.0, Fractint:
 o Produced "16 bit TGA potfiles" This format is no longer generated, but you can still (for a release or two) use <R> and <3> with those files.
 o Assumed "inside=maxit" for continuous potential. It now uses the current "inside=" value - to recreate prior results you must be explicit about this parameter.

1.10 Starfields

Once you have generated your favorite fractal image, you can convert it into a fractal starfield with the 'a' transformation (for 'astronomy'? - once again, all of the good letters were gone already). Stars are generated on a pixel-by-pixel basis - the odds that a particular pixel will coalesce into a star are based (partially) on the color index of that pixel.

(The following was supplied by Mark Peterson, the starfield author).

If the screen were entirely black and the 'Star Density per Pixel' were set to 30 then a starfield transformation would create an evenly distributed starfield with an average of one star for every 30 pixels.

If you're on a 320x200 screen then you have 64000 pixels and would end up with about 2100 stars. By introducing the variable of 'Clumpiness' we can create more stars in areas that have higher color values. At 100% Clumpiness a color value of 255 will change the average of finding a star at that location to 50:50. A lower clumpiness values will lower the amount of probability weighting. To create a spiral galaxy draw your favorite spiral fractal (IFS, Julia, or Mandelbrot) and perform a starfield transformation. For general starfields I'd recommend transforming a plasma fractal.

Real starfields have many more dim stars than bright ones because very few stars are close enough to appear bright. To achieve this effect the program will create a bell curve based on the value of ratio of Dim stars to bright stars. After calculating the bell curve the curve is folded in half and the peak used to represent the number of dim stars.

Starfields can only be shown in 256 colors. Fractint will automatically try to load ALTERN.MAP and abort if the map file cannot be found.

1.11 Random Dot Stereograms (RDS)

Random Dot Stereograms (RDS) are a way of encoding stereo images on a flat screen. Fractint can convert any image to a RDS using either the color number in the current palette or the grayscale value as depth. Try these steps. Generate a plasma fractal using the 640x480x256 video mode. When the image on the screen is complete, press <ctrl-s> ("s" for "Stereo"), and press <Enter> at the "RDS Parameters" screen prompt to accept the defaults. (More on the parameters in a moment.) The screen will be converted into a seemingly random collection of colored dots. Relax your eyes, looking through the screen rather than at the screen surface. The image will (hopefully) resolve itself into the hills and valleys of the 3D Plasma fractal.

Because pressing the two-keyed <ctrl-s> gets tiresome after a while, we have made <k> key a synonym for <ctrl-s> for convenience. Don't get too attached to <k> though; we reserve the right to reuse it for another purpose later.

The RDS feature has five and sometimes six parameters. Pressing <ctrl-s> always takes you to the parameter screen.

The first parameter allows you to control the depth effect. A larger value (positive or negative) exaggerates the sense of depth. If you make the depth negative, the high and low areas of the image are reversed. If your RDS image is streaky try either a lower depth factor or a higher resolution.

The second parameter indicates the overall width in inches of the image on your monitor or printout. The default value of 10 inches is roughly the width of an image on a standard 14" to 16" monitor. This value does not normally need to be changed for viewing images on standard monitors. However, if your monitor or image hardcopy is much wider or narrower than 10 inches (25 cm), and you have trouble seeing the image, enter the image width in inches. The issue here is that if the widest separation of left and right pixels is greater than the physical separation of your eyes, you will not be able to fuse the images. Conversely, a too-small separation may cause your eyes to

hyper-converge (fuse the wrong pixels together). A larger width value reduces the width between left and right pixels. You can use the calibration feature to help set the width parameter - see below. Once you have found a good width setting, you can place the value in your SSTOOLS.INI file with the command monitorwidth=<nnn>.

The third parameter allows you to control the method use to extract depth information from the original image. If your answer "no" at the "Use Grayscale value for Depth" prompt, then the color number of each pixel will be used. This value is independent of active color palette. If you answer "yes" and the prompt, then the depth values are keyed to the brightness of the color, which will change if you change palettes.

The fourth parameter allows you to set the position of vertical stereo calibration bars to the middle or the top of the image, or have the bars initially turned off. Use this feature to help you adjust your eye's convergence to see the image. You will see two vertical bars on the screen. You can turn off and on these bars with the <Enter> or <Space> keys after generating the RDS image. If you save an RDS image by pressing <s>, if the bars are turned on at the time, they become a permanent part of the image.

As you relax your eyes and look past the screen, these bars will appear as four bars. When you adjust your eyes so that the two middle bars merge into one bar, the 3D image should appear. The bars are set for the average depth in the area near the bars. They should always be closer together than the physical separation of your eyes, but not much less than about 1.5 inches. About 1.75 inches is ideal for many images. The depth and screen width controls affect the width of the bars.

At the RDS Parameters screen, you can select bars at the middle of the screen or the top. If you select "none", the bars will initially be off, but immediately after generation of the image you can still turn on the bars with <Enter> or <Space> before you press any other keys. If the initial setting of the calibration bars is "none", then if the bars are turned on later they will appear in the middle. Hint: if you cycle the colors and find you can't see the calibration bar, press <Enter> or <Space> twice, and the bars will turn to a more visible color.

The fifth parameter asks if you want to use an image map GIF file instead of using random dots. An image map can give your RDS image a more interesting background texture than the random dots. If you answer "yes" at the Use image map? prompt, Fractint will present you with a file selection list of GIF images. Fractint will then go ahead and transform your original image to RDS using the selected image map to provide the "random" dots.

After you have selected an image map file, the next time you reach the RDS Parameters screen you will see an additional prompt asking if you want to use the same image map file again. Answering "yes" avoids the file selection menu.

The best images to use as image maps are detailed textures with no solid spots. The default type=circle fractal works well, as do the barnsley fractals if you zoom in a little way. If the image map is smaller than your RDS image, the image map will repeated to fill the space. If the image map is larger, just the upper left corner of the image map will be used.

The original image you are using for your stereogram is saved, so if you want

to modify the stereogram parameters and try again, just press <ctrl-s> (or <k>) to get the parameter screen, changes the parameters, and press <Enter>. The original image is restored and an RDS transform with the revised parameters is performed. If you press <s> when viewing an RDS image, after the RDS image is saved, the original is restored.

Try the RDS feature with continuous potential Mandelbrots as well as plasma fractals.

For a summary of keystrokes in RDS mode, see RDS Commands

1.12 Palette Maps

If you have a VGA, MCGA, Super-VGA, 8514/A, XGA, TARGA, or TARGA+ video adapter, you can save and restore color palettes for use with any image. To load a palette onto an existing image, use the <L> command in color-cycling or palette-editing mode. To save a palette, use the <S> command in those modes. To change the default palette for an entire run, use the command line "map=" parameter.

The default filetype for color-map files is ".MAP".

These color-maps are ASCII text files set up as a series of RGB triplet values (one triplet per line, encoded as the red, green, and blue [RGB] components of the color).

Note that .MAP file color values are in GIF format - values go from 0 (low) to 255 (high), so for a VGA adapter they get divided by 4 before being stuffed into the VGA's Video-DAC registers (so '6' and '7' end up referring to the same color value).

Fractint is distributed with some sample .MAP files:

- ALTERN.MAP the famous "Peterson-Vigneau Pseudo-Grey Scale"
- BLUES.MAP for rainy days, by Daniel Egnor
- CHROMA.MAP general purpose, chromatic
- DEFAULT.MAP the VGA start-up values
- FIRESTRM.MAP general purpose, muted fire colors
- GAMMA1.MAP and GAMMA2.MAP Lee Crocker's response to ALTERN.MAP
- GLASSES1.MAP used with 3d glasses modes
- GLASSES2.MAP used with 3d glasses modes
- GOODEGA.MAP for EGA users
- GREEN.MAP shaded green
- GREY.MAP another grey variant
- GRID.MAP for stereo surface grid images
- HEADACHE.MAP major stripes, by D. Egnor (try cycling and hitting <2>)
- LANDSCAP.MAP Guruka Singh Khalsa's favorite map for plasma "landscapes"
- NEON.MAP a flashy map, by Daniel Egnor
- PAINTJET.MAP high resolution mode PaintJet colors
- ROYAL.MAP the royal purple, by Daniel Egnor
- TOPO.MAP Monte Davis's contribution to full color terrain
- VOLCANO.MAP an explosion of lava, by Daniel Egnor

1.13 Bailout Test

The bailout test is used to determine if we should stop iterating before the maximum iteration count is reached. This test compares the value determined by the test to the "bailout" value set via the <Z> screen. The default bailout test compares the magnitude or modulus of a complex variable to some bailout value:

bailout test = |z| = sqrt(x^2 + y^2) >= 2

As a computational speedup, we square both sides of this equation and the bailout test used by Fractint is:

bailout test = |z|^2 = x^2 + y^2 >= 4

Using a "bailout" other than 4 allows us to change when the bailout will occur.

The following bailout tests have been implemented on the <Z> screen:

mod:	x^2 + y^2 >= bailout
real:	x^2 >= bailout
imag:	y^2 >= bailout
or:	x^2 >= bailout or y^2 >= bailout
and:	x^2 >= bailout and y^2 >= bailout

The bailout test feature has not been implemented for all applicable fractal types. This is due to the speedups used for these types. Some of these bailout tests show the limitations of the integer math routines by clipping the spiked ends off of the protrusions.

1.14 3D Images

1.15 3d overview

Fractint can restore images in "3D". Important: we use quotation marks because it does not CREATE images of 3D fractal objects (there are such, but we're not there yet.) Instead, it restores .GIF images as a 3D PROJECTION or STEREO IMAGE PAIR. The iteration values you've come to know and love, the ones that determine pixel colors, are translated into "height" so that your saved screen becomes a landscape viewed in perspective. You can even wrap the landscape onto a sphere for realistic-looking planets and moons that never existed outside your PC!

We suggest starting with a saved plasma-cloud screen. Hit <3> in main command mode to begin the process. Next, select the file to be transformed, and the video mode. (Usually you want the same video mode the file was generated in; other choices may or may not work.)

After hitting <3>, you'll be bombarded with a long series of options. Not to worry: all
of them have defaults chosen to yield an acceptable starting image, so the first time out just pump your way through with the <Enter> key. When you enter a different value for any option, that becomes the default value the next time you hit <3>, so you can change one option at a time until you get what you want. Generally <ESC> will take you back to the previous screen.

Once you're familiar with the effects of the 3D option values you have a variety of options on how to specify them. You can specify them all on the command line (there ARE a lot of them so they may not all fit within the DOS command line limits), with an SSTOOLS.INI file, or with a parameter file.

Here's an example for you power FRACTINTers, the command

```
FRACTINT MYFILE SAVENAME=MY3D 3D=YES BATCH=YES
```

would make Fractint load MYFILE.GIF, re-plot it as a 3D landscape (taking all of the defaults), save the result as MY3D.GIF, and exit to DOS. By the time you've come back with that cup of coffee, you'll have a new world to view, if not conquer.

Note that the image created by 3D transformation is treated as if it were a plasma cloud - We have NO idea how to retain the ability to zoom and pan around a 3D image that has been twisted, stretched, perspective-ized, and water-leveled. Actually, we do, but it involves the kind of hardware that Industrial Light & Magic, Pixar et al. use for feature films. So if you'd like to send us a check equivalent to George Lucas' net from the "Star Wars" series...

1.16 3D Mode Selection

After hitting <3> and getting past the filename prompt and video mode selection, you're presented with a "3d Mode Selection" screen. If you wish to change the default for any of the following parameters, use the cursor keys to move through the menu. When you're satisfied press <Enter>.

Preview Mode:
Preview mode provides a rapid look at your transformed image using by skipping a lot of rows and filling the image in. Good for quickly discovering the best parameters. Let's face it, the Fractint authors most famous for "blazingly fast" code *DIDN'T* write the 3D routines! [Pieter: "But they *are* picking away it and making some progress in each release."]

Show Box:
If you have selected Preview Mode you have another option to worry about. This is the option to show the image box in scaled and rotated coordinates x, y, and z. The box only appears in rectangular transformations and shows how the final image will be oriented. If you select light source in the next screen, it will also show you the light source vector so you can tell where the light is coming from in relation to your image. Sorry no head or tail on the vector yet.

Coarseness:
This sets how many divisions the image will be divided into in the y direction, if you select preview mode, ray tracing output, or grid fill in the "Select Fill Type" screen.

Spherical Projection:
The next question asks if you want a sphere projection. This will take your image and map it onto a plane if you answer "no" or a sphere if you answer "yes" as described above. Try it and you'll see what we mean. See Spherical Projection.

Stereo:

Stereo sound in Fractint? Well, not yet. Fractint now allows you to create 3D images for use with red/blue glasses like 3D comics you may have seen, or images like Captain EO.

Option 0 is normal old 3D you can look at with just your eyes.

Options 1 and 2 require the special red/blue-green glasses. They are meant to be viewed right on the screen or on a color print off of the screen. The image can be made to hover entirely or partially in front of the screen. Great fun! These two options give a gray scale image when viewed.

Option 1 gives 64 shades of gray but with half the spatial resolution you have selected. It works by writing the red and blue images on adjacent pixels, which is why it eats half your resolution. In general, we recommend you use this only with resolutions above 640x350. Use this mode for continuous potential landscapes where you *NEED* all those shades.

Option "2" gives you full spatial resolution but with only 16 shades of gray. If the red and blue images overlap, the colors are mixed. Good for wire-frame images (we call them surface grids), lorenz3d and 3D IFS. Works fine in 16 color modes.

Option 3 is for creating stereo pair images for view later with more specialized equipment. It allows full color images to be presented in glorious stereo. The left image presented on the screen first. You may

photograph it or save it. Then the second image is presented, you may do the same as the first image. You can then take the two images and convert them to a stereo image pair as outlined by Bruce Goren (see below).

Also see Stereo 3D Viewing.

Ray Tracing Output:

Fractint can create files of its 3d transformations which are compatible with many ray tracing programs. Currently four are supported directly: DKB (now obsolete), VIVID, MTV, and RAYSHADE. In addition a "RAW" output is supported which can be relatively easily transformed to be usable by many other products.
One other option is supported: ACROSPIN. This is not a ray tracer, but the same Fractint options apply - see Acrospin.

Option values:

0 disables the creation of ray tracing output
1 DKB format (obsolete-see below)
2 VIVID format
3 generic format (must be massaged externally)
4 MTV format
5 RAYSHADE format
6 ACROSPIN format

Users of POV-Ray can use the DKB output and convert to POV-Ray with the DKB2POV utility that comes with POV-Ray. A better (faster) approach is to create a RAW output file and convert to POV-Ray with RAW2POV. A still better approach is to use POV-Ray's height field feature to directly read the fractal .GIF or .POT file and do the 3D transformation inside POV-Ray.

All ray tracing files consist of triangles which follow the surface created by Fractint during the 3d transform. Triangles which lie below the "water line" are not created in order to avoid causing unnecessary work for the poor ray tracers which are already overworked.
A simple plane can be substituted by the user at the waterline if needed.

The size (and therefore the number) of triangles created is determined by the "coarse" parameter setting. While generating the ray tracing file, you will view the image from above and watch it partitioned into triangles.

The color of each triangle is the average of the color of its verticies in the original image, unless BRIEF is selected.

If BRIEF is selected, a default color is assigned at the begining of the file and is used for all triangles.

Also see Interfacing with Ray Tracing Programs.

Brief output:

This is a ray tracing sub-option.
When it is set to yes, Fractint creates a considerably smaller and somewhat faster file. In this mode, all triangles

```
   use the default color specified at the begining of the file.
   This color should be edited to supply the color of your choice.

Targa Output:

   If you want any of the 3d transforms you select to be saved as a
   Targa-24 file or overlayed onto one, select yes for this option.
   The overlay option in the final screen determines whether you will
   create a new file or overlay an existing one.

MAP File name:

   Imediately after selecting the previous options, you will be given the
   chance to select an alternate color MAP file. The default is to
   use the current MAP. If you want another MAP used, then enter your
   selection at this point.

Output File Name:

   This is a ray tracing sub-option, used to specify the name of the
   file to be written.  The default name is FRACT001.RAY.  The name is
   incremented by one each time a file is written.
   If you have not set "overwrite=yes" then
   the file name will also be automatically incremented to avoid
   over-writing previous files.

When you are satisfied with your selections press enter to go
to the next parameter screen.
```

1.17 Select Fill Type Screen

```
This option exists because in the course of the 3D projection, portions
of the original image may be stretched to fit the new surface. Points of
an image that formerly were right next to each other, now may have a space
between them. This option generally determines what to do with the space
between the mapped dots. It is not used if you have selected a value for
RAY other than 0.

For an illustration, pick the second option "just draw the points", which
just maps points to corresponding points. Generally this will leave empty
space between many of the points. Therefore you can choose various
algorithms that "fill in" the space between the points in various ways.

Later, try the first option "make a surface grid." This option will make a
grid of the surface which is as many divisions in the original "y"
direction as was set in "coarse" in the first screen. It is very fast, and
can give you a good idea what the final relationship of parts of your
picture will look like.

Later, try the second option "connect the dots (wire frame)", then "surface
fills" - "colors interpolated" and "colors not interpolated", the general
favorites of the authors. Solid fill, while it reveals the pseudo-geology
under your pseudo-landscape, inevitably takes longer.

Later, try the light source fill types. These two algorithms allow you to
```

position the "sun" over your "landscape." Each pixel is colored according to the angle the surface makes with an imaginary light source. You will be asked to enter the three coordinates of the vector pointing toward the light in a following parameter screen - see Light Source Parameters.

"Light source before transformation" uses the illumination direction without transforming it. The light source is fixed relative to your computer screen. If you generate a sequence of images with progressive rotation, the effect is as if you and the light source are fixed and the object is rotating. Therefore as the object rotates features of the object move in and out of the light. This fill option was incorrect prior to version 16.1, and has been changed.

"Light source after transformation" applies the same transformation to both the light direction and the object. Since both the light direction and the object are transformed, if you generate a sequence of images with the rotation progressively changed, the effect is as if the image and the light source are fixed in relation to each other and you orbit around the image. The illumination of features on the object is constant, but you see the object from different angles. This fill option was correct in earlier Fractint versions and has not been changed.

For ease of discussion we will refer to the following fill types by these numbers:

    1 - surface grid
    2 - (default) - no fill at all - just draw the dots
    3 - wire frame - joins points with lines
    4 - surface fill - (colors interpolated)
    5 - surface fill - (interpolation turned off)
    6 - solid fill - draws lines from the "ground" up to the point
    7 - surface fill with light model - calculated before 3D transforms
    8 - surface fill with light model - calculated after 3D transforms

Types 4, 7, and 8 interpolate colors when filling, making a very smooth fill if the palette is continuous. This may not be desirable if the palette is not continuous. Type 5 is the same as type 4 with interpolation turned off. You might want to use fill type 5, for example, to project a .GIF photograph onto a sphere. With type 4, you might see the filled-in points, since chances are the palette is not continuous; type 5 fills those same points in with the colors of adjacent pixels. However, for most fractal images, fill type 4 works better.

This screen is not available if you have selected a ray tracing option.

1.18 Stereo 3D Viewing

The "Funny Glasses" (stereo 3D) parameter screen is presented only if you select a non-zero stereo option in the prior 3D parameters. (See 3D Mode Selection.)
We suggest you definitely use defaults at first on this screen.

When you look at an image with both eyes, each eye sees the image in slightly different perspective because they see it from different places.

The first selection you must make is ocular separation, the distance the between the viewers eyes. This is measured as a % of screen and is an

important factor in setting the position of the final stereo image in
front of or behind the CRT Screen.

The second selection is convergence, also as a % of screen. This tends to
move the image forward and back to set where it floats. More positive
values move the image towards the viewer. The value of this parameter
needs to be set in conjunction with the setting of ocular separation and
the perspective distance. It directly adjusts the overall separation of
the two stereo images. Beginning anaglyphers love to create images
floating mystically in front of the screen, but grizzled old 3D veterans
look upon such antics with disdain, and believe the image should be safely
inside the monitor where it belongs!

Left and Right Red and Blue image crop (% of screen also) help keep the
visible part of the right image the same as the visible part of the left
by cropping them. If there is too much in the field of either eye that the
other doesn't see, the stereo effect can be ruined.

Red and Blue brightness factor. The generally available red/blue-green
glasses, made for viewing on ink on paper and not the light from a CRT,
let in more red light in the blue-green lens than we would like. This
leaves a ghost of the red image on the blue-green image (definitely not
desired in stereo images). We have countered this by adjusting the
intensity of the red and blue values on the CRT. In general you should not
have to adjust this.

The final entry is Map file name (present only if stereo=1 or stereo=2
was selected).
If you have a special map file you want
to use for Stereo 3D this is the place to enter its name. Generally
glasses1.map is for type 1 (alternating pixels), and glasses2.map is for
type 2 (superimposed pixels). Grid.map is great for wire-frame images
using 16 color modes.

This screen is not available if you have selected a ray tracing option.

1.19 3D Fractal Parameters

The parameters on this screen are a subset of the zillions of options
available for Fractint's 3D image transformations.
This screen's parameters are those which also affect 3D fractal types like
lorenz3d and kamtorus3d.
Since they are documented elsewhere, we won't repeat ourselves:

For a description of rotation, perspective, and shift parameters, please
see Rectangular Coordinate Transformation. Ignore the paragraphs about
"scaling" and "water level" - those parts apply only to 3D Transforms.

For a description of the stereo option, please see the "stereo" subheading
in 3D Mode Selection.

1.20 Rectangular Coordinate Transformation

The first entries are rotation values around the X, Y, and Z axes. Think of your starting image as a flat map: the X value tilts the bottom of your monitor towards you by X degrees, the Y value pulls the left side of the monitor towards you, and the Z value spins it counter-clockwise. Note that these are NOT independent rotations: the image is rotated first along the X-axis, then along the Y-axis, and finally along the Z-axis. Those are YOUR axes, not those of your (by now hopelessly skewed) monitor. All rotations actually occur through the center of the original image. Rotation parameters are not used when a ray tracing option has been selected.

Then there are three scaling factors in percent. Initially, leave the X and Y axes alone and play with Z, now the vertical axis, which translates into surface "roughness." High values of Z make spiky, on-beyond-Alpine mountains and improbably deep valleys; low values make gentle, rolling terrain. Negative roughness is legal: if you're doing an M-set image and want Mandelbrot Lake to be below the ground, instead of eerily floating above, try a roughness of about -30%.

Next we need a water level -- really a minimum-color value that performs the function "if (color < waterlevel) color = waterlevel". So it plots all colors "below" the one you choose at the level of that color, with the effect of filling in "valleys" and converting them to "lakes."

Now we enter a perspective distance, which you can think of as the "distance" from your eye to the image. A zero value (the default) means no perspective calculations, which allows use of a faster algorithm. Perspective distance is not available if you have selected a ray tracing option.

For non-zero values, picture a box with the original X-Y plane of your flat fractal on the bottom, and your 3D fractal inside. A perspective value of 100% places your eye right at the edge of the box and yields fairly severe distortion, like a close view through a wide-angle lens. 200% puts your eye as far from the front of the box as the back is behind. 300% puts your eye twice as far from the front of the box as the back is, etc. Try about 150% for reasonable results. Much larger values put you far away for even less distortion, while values smaller than 100% put you "inside" the box. Try larger values first, and work your way in.

Next, you are prompted for two types of X and Y shifts (now back in the plane of your screen) that let you move the final image around if you'd like to re-center it. The first set, x and y shift with perspective, move the image and the effect changes the perspective you see. The second set, "x and y adjust without perspective", move the image but do not change perspective. They are used just for positioning the final image on the screen. Shifting of any type is not available if you have selected a ray tracing option.

3D Color Parameters are also requested on the same input screen.

If you are doing a Spherical Projection, special parameters for it are requested at the start of this screen.

1.21 3d color parameters

You are asked for a range of "transparent" colors, if any. This
option is most useful when using the 3D Overlay Mode. Enter the
color range (minimum and maximum value) for which you do not want to
overwrite whatever may already be on the screen. The default is no
transparency (overwrite everything).

Now, for the final option. This one will smooth the transition between
colors by randomizing them and reduce the banding that occurs with some
maps. Select the value of randomize to between 0 (for no effect) and 7 (to
randomize your colors almost beyond use). 3 is a good starting point.

That's all for this screen. Press enter for these parameters and the
next and final screen will appear (honestly!).

1.22 Light Source Parameters

This one deals with all the aspects of light source and Targa files.

You must choose the direction of the light from the light source. This will
be scaled in the x, y, and z directions the same as the image. For
example, 1,1,3 positions the light to come from the lower right front of
the screen in relation to the untransformed image. It is important to
remember that these coordinates are scaled the same as your image. Thus,
"1,1,1" positions the light to come from a direction of equal distances to
the right, below and in front of each pixel on the original image.
However, if the x,y,z scale is set to 90,90,30 the result will be from
equal distances to the right and below each pixel but from only 1/3 the
distance in front of the screen i.e.. it will be low in the sky, say,
afternoon or morning.

Then you are asked for a smoothing factor. Unless you used
Continuous Potential
illumination when using light source fills may appear "sparkly", like a
sandy beach in bright sun. A smoothing factor of 2 or 3 will allow you to
see the large-scale shapes better.

Smoothing is primarily useful when doing light source fill types with plasma
clouds. If your fractal is not a plasma cloud and has features with
sharply defined boundaries (e.g. Mandelbrot Lake), smoothing may cause
the colors to run. This is a feature, not a bug. (A copyrighted response
of [your favorite commercial software company here], used by permission.)

The ambient option sets the minimum light value a surface has if it has no
direct lighting at all. All light values are scaled from this value to
white. This effectively adjusts the depth of the shadows and sets the
overall contrast of the image.

If you selected the full color option, you have a few more choices.
The next is the haze factor. Set this to make distant objects more hazy.
Close up objects will have little effect, distant objects will have most.
0 disables the function. 100 is the maximum effect, the farthest objects
will be lost in the mist. Currently, this does not really use distance

from the viewer, we cheat and use the y value of the original image. So the effect really only works if the y-rotation (set earlier) is between +/- 30.

Next, you can choose the name under which to save your Targa file. If you have a RAM disk handy, you might want to create the file on it, for speed. So include its full path name in this option. If you have not set "overwrite=yes" then the file name will be incremented to avoid over-writing previous files. If you are going to overlay an existing Targa file, enter its name here.

Next, you may select the background color for the Targa file. The default background on the Targa file is sky blue. Enter the Red, Green, and Blue component for the background color you wish.

Finally, absolutely the last option (this time we mean it): you can now choose to overlay an existing Targa-24, type 2, non mapped, top-to-bottom file, such as created by Fractint or PVRay. The Targa file specified above will be overlayed with new info just as a GIF is overlayed on screen. Note: it is not necessary to use the "O" overlay command to overlay Targa files. The Targa_Overlay option must be set to yes, however.

You'll probably want to adjust the final colors for monochrome fill types using light source via color cycling.
Try one of the more continuous
palettes (<F8> through <F10>), or load the GRAY palette with the
<A>lternate-map command.

Now, lie down for a while in a quiet room with a damp washcloth on your forehead. Feeling better? Good -- because it's time to go back almost to the top of the 3D options and just say yes to:

1.23 spherical projection

Picture a globe lying on its side, "north" pole to the right. (It's our planet, and we'll position it the way we like.) You will be mapping the X and Y axes of the starting image to latitude and longitude on the globe, so that what was a horizontal row of pixels follows a line of longitude. The defaults exactly cover the hemisphere facing you, from longitude 180 degrees (top) to 0 degrees (bottom) and latitude -90 (left) to latitude 90 (right). By changing them you can map the image to a piece of the hemisphere or wrap it clear around the globe.

The next entry is for a radius factor that controls the over-all size of the globe. All the rest of the entries are the same as in the landscape projection. You may want less surface roughness for a plausible look, unless you prefer small worlds with big topography, a la "The Little Prince."

WARNING: When the "construction" process begins at the edge of the globe (default) or behind it, it's plotting points that will be hidden by subsequent points as the process sweeps around the sphere toward you. Our nifty hidden-point algorithms "know" this, and the first few dozen lines may be invisible unless a high mountain happens to poke over the horizon.

If you start a spherical projection and the screen stays black, wait for a while (a longer while for higher resolution or fill type 6) to see if points start to appear. Would we lie to you? If you're still waiting hours later, first check that the power's still on, then consider a faster system.

1.24 3D Overlay Mode

While the <3> command (see 3D Images)
creates its image on a blank screen, the <#> (or <shift-3> on some keyboards) command
draws a second image over an existing displayed image. This image can be any restored image from a <R> command or the result of a just executed <3> command. So you can do a landscape, then press <#> and choose spherical projection to re-plot that image or another as a moon in the sky above the landscape. <#> can be repeated as many times as you like.

It's worth noting that not all that many years ago, one of us watched Benoit Mandelbrot and fractal-graphics wizard Dick Voss creating just such a moon-over-landscape image at IBM's research center in Yorktown Heights, NY. The system was a large and impressive mainframe with floating-point facilities bigger than the average minicomputer, running LBLGRAPH -- what Mandelbrot calls "an independent-minded and often very ill-mannered heap of graphics programs that originated in work by Alex Hurwitz and Jack Wright of IBM Los Angeles."

We'd like to salute LBLGRAPH, its successors, and their creators, because it was their graphic output (like "Planetrise over Labelgraph Hill," plate C9 in Mandelbrot's "Fractal Geometry of Nature") that helped turn fractal geometry from a mathematical curiosity into a phenomenon. We'd also like to point out that it wasn't as fast, flexible or pretty as Fractint on a 386/16 PC with S-VGA graphics. Now, a lot of the difference has to do with the incredible progress of micro-processor power since then, so a lot of the credit should go to Intel rather than to our highly tuned code. OK, twist our arms -- it IS awfully good code.

1.25 special note for cga or hercules users

If you are one of those unfortunates with a CGA or Hercules 2-color monochrome graphics, it is now possible for you to make 3D projection images.

Try the following unfortunately circuitous approach. Invoke Fractint, making sure you have set askvideo=yes. Use a disk-video mode to create a 256 color fractal. You might want to edit the fractint.cfg file to make a disk-video mode with the same pixel dimensions as your normal video. Using the "3" command, enter the file name of the saved 256 color file, then select your 2 or 4
color mode, and answer the other 3D prompts. You will then see a 3D projection of the fractal. Another example of Stone Soup responsiveness to our fan mail!

1.26 making terrains

If you enjoy using Fractint for making landscapes, we have several new features for you to work with. When doing 3d transformations banding tends to occur because all pixels of a given height end up the same color. Now, colors can be randomized to make the transitions between different colors at different altitudes smoother. Use the new "RANDOMIZE= " variable to accomplish this. If your light source images all look like lunar landscapes since they are all monochrome and have very dark shadows, we now allow you to set the ambient light for adjusting the contrast of the final image. Use the "Ambient= " variable. In addition to being able to create scenes with light sources in monochrome, you can now do it in full color as well. Setting fullcolor=1 will generate a Targa-24 file with a full color image which will be a combination of the original colors of the source image (or map file if you select map=something) and the amount of light which reflects off a given point on the surface. Since there can be 256 different colors in the original image and 256 levels of light, you can now generate an image with *lots* of colors. To convert it to a GIF if you can't view Targa files directly, you can use PICLAB (see Other Programs), and the following commands:

```
SET PALETTE 256
SET CREZ 8
TLOAD yourfile.tga
MAKEPAL
MAP
GSAVE yourfile.gif
EXIT
```

Using the full color option allows you to also set a haze factor with the "haze= " variable to make more distant objects more hazy.

As a default, full color files also have the background set to sky blue. Warning, the files which are created with the full color option are very large, 3 bytes per pixel. So be sure to use a disk with enough space. The file is created using Fractint's disk-video caching, but is always created on real disk (expanded or extended memory is not used.) Try the following settings of the new variables in sequence to get a feel for the effect of each one:

```
;use this with any filltype
map=topo
randomize=3; adjusting this smooths color transitions

;now add this using filltype 5 or 6
ambient=20; adjusting this changes the contrast
filltype=6
smoothing=2; makes the light not quite as granular as the terrain

;now add the following, and this is where it gets slow
fullcolor=1; use PICLAB to reduce resulting lightfile to a GIF

;and finally this
haze=20; sets the amount of haze for distant objects
```

When full color is being used, the image you see on the screen will represent the amount of light being reflected, not the colors in the final image. Don't be disturbed if the colors look weird, they are an artifact

of the process being used. The image being created in the lightfile won't look like the screen.

However, if you are worried, hit ESC several times and when Fractint gets to the end of the current line it will abort. Your partial image will be there as LIGHT001.TGA or with whatever file name you selected with the lightname option. Convert it as described above and adjust any parameters you are not happy with. Its a little awkward, but we haven't figured out a better way yet.

1.27 making 3d slides

Bruce Goren, CIS's resident stereoscopic maven, contributed these tips on what to do with your 3D images (Bruce inspired and prodded us so much we automated much of what follows, allowing both this and actual on screen stereo viewing, but we included it here for reference and a brief tutorial.)

"I use a Targa 32 video card and TOPAS graphic software, moving the viewport or imaginary camera left and right to create two separate views of the stationary object in x,y,z, space. The distance between the two views, known as the inter-ocular distance, toe-in or convergence angle, is critical. It makes the difference between good 3-D and headache-generating bad 3-D.

"For a 3D fractal landscape, I created and photographed the left and right eye views as if flying by in an imaginary airplane and mounted the film chips for stereo viewing. To make my image, first I generated a plasma cloud based on a color map I calculated to resemble a geological survey map (available on CIS as TARGA.MAP). In the 3D reconstruction, I used a perspective value of 150 and shifted the camera -15 and +15 on the X-axis for the left and right views. All other values were left to the defaults.

"The images are captured on a Matrix 3000 film recorder -- basically a box with a high-resolution (1400 lines) black and white TV and a 35mm camera (Konica FS-1) looking at the TV screen through a filter wheel. The Matrix 3000 can be calibrated for 8 different film types, but so far I have only used Kodak Ektachrome 64 daylight for slides and a few print films. I glass mount the film chips myself.

"Each frame is exposed three times, once through each of the red, blue, and green filters to create a color image from computer video without the scan-lines which normally result from photographing television screens. The aspect ratio of the resulting images led me to mount the chips using the 7-sprocket Busch-European Emde masks. The best source of Stereo mounting and viewing supplies I know of is an outfit called Reel 3-D Enterprises, Inc. at P.O. Box 2368, Culver City, CA 90231, tel. 213-837-2368. "My platform is an IBM PC/AT crystal-swapped up to 9 MHz. The math co-processor runs on a separate 8-MHz accessory sub-board. The system currently has 6.5 MB of RAM."

1.28 Interfacing with Ray Tracing Programs

(Also see "Ray Tracing Output", "Brief", and "Output File Name"
in 3D Mode Selection.)

Fractint allows you to save your 3d transforms in files which may be
fed to a ray tracer (or to "Acrospin").
However, they are not ready to be traced by
themselves. For one thing, no light source is included. They are
actually meant to be included within other ray tracing files.

Since the intent is to produce an object which may be included in a larger
ray tracing scene, it is expected that all rotations, shifts, and final
scaling will be done by the ray tracer. Thus, in creating the images, no
facilities for rotations or shifting is provided. Scaling is provided to
achieve the correct aspect ratio.

WARNING! The files created using the RAY option can be huge. Setting
COARSE to 40 will result in over 2000 triangles. Each triangle can utilize
from 50 to 200 bytes each to describe, so your ray tracing files can
rapidly approach or exceed 1Meg. Make sure you have enough disk space before
you start.

Each file starts with a comment identifying the version of Fractint
by which it was created. The file ends with a comment giving the number
of triangles in the file.

The files consist of long strips of adjacent triangles. Triangles are
clockwise or counter clockwise depending on the target ray tracer.
Currently, MTV and Rayshade are the only ones which use counter clockwise
triangles. The size of the triangles is set by the COARSE setting in the
main 3d menu. Color information about each individual triangle is included
for all files unless in the brief mode.

To keep the poor ray tracer from working too hard, if WATERLINE is set
to a non zero value, no triangle which lies entirely at or below the
current setting of WATERLINE is written to the ray tracing file. These
may be replaced by a simple plane in the syntax of the ray tracer you are
using.

Fractint’s coordinate system has the origin of the x-y plane at the upper
left hand corner of the screen, with positive x to the right and positive
y down. The ray tracing files have the origin of the x-y plane moved to
the center of the screen with positive x to the right and positive y up.
Increasing values of the color index are out of the screen and in the +z
direction. The color index 0 will be found in the xy plane at z=-1.

When x- y- and zscale are set to 100, the surface created by the
triangles will fall within a box of +/- 1.0 in all 3 directions. Changing
scale will change the size and/or aspect ratio of the enclosed object.

We will only describe the structure of the RAW format here. If you want
to understand any of the ray tracing file formats besides RAW, please
see your favorite ray tracer docs.

The RAW format simply consists of a series of clockwise triangles. If
BRIEF=yes, Each line is a vertex with coordinates x, y, and z. Each
triangle is separated by a couple of CR’s from the next. If BRIEF=no, the

first line in each triangle description if the r,g,b value of the triangle.

Setting BRIEF=yes produces shorter files with the color of each triangle removed - all triangles will be the same color. These files are otherwise identical to normal files but will run faster than the non BRIEF files. Also, with BRIEF=yes, you may be able to get files with more triangles to run than with BRIEF=no.

The DKB format is now obsolete. POV-Ray users should use the RAW output and convert to POV-Ray using the POV Group's RAW2POV utility. POV-Ray users can also do all 3D transformations within POV-Ray using height fields.