Fractal Zplot Application Help

Table of Contents

1 Main Index

Fractal Zplot Help Index

Commands

1.1 Title Bar

Title Bar

The title bar is located along the top of a window. It contains the name of the application and drawing.

To move the window, drag the title bar. Note: You can also move dialog boxes by dragging their title bars.

A title bar may contain the following elements:

- Application Control-menu button
- Drawing Control-menu button
- Maximize button
- Minimize button
- Name of the application
- Name of the drawing
- Restore button

1.2 Scroll bars

Scroll bars

Displayed at the right and bottom edges of the drawing window. The scroll boxes inside the scroll bars indicate your vertical and horizontal location in the drawing. You can use the mouse to scroll to other parts of the drawing.

1.3 Size

Size command (System menu)

Use this command to display a four-headed arrow so you can size the active window with the arrow keys.

After the pointer changes to the four-headed arrow:

1. Press one of the DIRECTION keys (left, right, up, or down arrow key) to move the pointer to the border you want to move.
2. Press a DIRECTION key to move the border.
3. Press ENTER when the window is the size you want.

Note: This command is unavailable if you maximize the window.

Shortcut

Mouse: Drag the size bars at the corners or edges of the window.

1.4 Move

Move command (Control menu)

Use this command to display a four-headed arrow so you can move the active window or dialog box with the arrow keys.

Note: This command is unavailable if you maximize the window.

Shortcut

Keys: CTRL+F7

1.5 Minimize Command

Minimize command (application Control menu)

Use this command to reduce the Fractal Zplot window to an icon.

Shortcut

Mouse: Click the minimize icon  on the title bar.

Keys: ALT+F9

1.6 Maximize Command

Maximize command (System menu)

Use this command to enlarge the active window to fill the available space.

Shortcut

Mouse: Click the maximize icon on the title bar; or double-click the title bar.
Keys: CTRL+F10 enlarges a drawing window.

1.7 Next Window

Next Window command (drawing Control menu)

Use this command to switch to the next open drawing window. Fractal Zplot determines which window is next according to the order in which you opened the windows.

Shortcut

Keys: CTRL+F6

1.8 Previous Window

Previous Window command (drawing Control menu)

Use this command to switch to the previous open drawing window. Fractal Zplot determines which window is previous according to the order in which you opened the windows.

Shortcut

Keys: SHIFT+CTRL+F6

1.9 Close

Close command (Control menus)

Use this command to close the active window or dialog box.

Double-clicking a Control-menu box is the same as choosing the Close command.

Shortcuts

Keys: CTRL+F4 closes a drawing window

ALT+F4 closes the application

1.10 Restore

Restore command (Control menu)

Use this command to return the active window to its size and position before you chose the Maximize or Minimize command.

1.11 Switch to

Switch to command (application Control menu)

Use this command to display a list of all open applications. Use this "Task List" to switch to or close an application on the list.

Shortcut

Keys: CTRL+ESC

Dialog Box Options

When you choose the Switch To command, you will be presented with a dialog box with the following options:

Task List

Select the application you want to switch to or close.

Switch To

Makes the selected application active.

End Task

Closes the selected application.

Cancel

Closes the Task List box.

Cascade

Arranges open applications so they overlap and you can see each title bar. This option does not affect applications reduced to icons.

Tile

Arranges open applications into windows that do not overlap. This option does not affect applications reduced to icons.

Arrange Icons

Arranges the icons of all minimized applications across the bottom of the screen.

2 Fractal Zplot Remote

Fractal Zplot Remote

The remote provides access to many of the most-used commands in Fractal Zplot. Info about each button can be obtained by using the '?' located near the close box in the top right-hand corner.

2.1 New button

New button

Use this button to create a new drawing window in Fractal Zplot.

2.2 Undo button

Undo button

Use this command to undo the last action. An image can be continued after an undo, if continue was enabled before the last action, but not after an image is resized. Color-cycling is disabled after using Undo.

2.3 Size button

Size button

This allows you to set the drawing area for a picture, independent of the Windows screen size. It also shows which size is currently in use. The aspect for the drawing is based on the ratio of X (horizontal width) to Y (vertical height.) The custom setting allows for any size/aspect that system memory will permit. Videos are limited to the standard 4/3 vga aspect or 1/1. The minimum size for an image is 40X30. Note: if the image is less than 100 width, the aspect must be 4/3 for solid-guessing to work properly.

2.4 Color button

Color button

Use the palette editor to modify the palette(s) in use.

2.5 Batch button

Batch button

Here you set parameters for batching and saving random-generated images to disk.

2.6 Fvr button

FVR button

The window opened varies with the fractal type selected, and contains all the major variables that Fractal Zplot now scales between key frames of an avi stream.

2.7 Draw button

Draw button

Use this button to draw or redraw the image for the current fractal variables. Clicking inside the draw window with the left-mouse button stops all plotting. Use the Cont button on the toolbar to

restart plotting from the current column. When the lsystem option is selected, a custom dialog is opened to enter lsystem draw options. Note: the draw window that a plot is initialized in (by clicking on an Okay or Apply button or Draw) must have the focus to begin drawing. With a lot of parameter or formula windows open for different draw windows, it is easy to mistake one window for another. So if the plot doesn't start immediately in the window that has the focus, check to see that that window is actually the one you wanted to start a new plot in. If it isn't, click on the title bar for the window that the plot is activated in. The plot should then commence.

2.8 Abort button

Abort button

Use this command to stop drawing. Clicking inside a window's drawing area or close box (or the program close box) will also stop the drawing. Note: once a plot has started Fractal Zplot continues to draw the image for that window regardless of which drawing window has the input focus, until done or aborted. You can open and close other drawing windows without affecting the current drawing, but only one drawing is active at any time.

2.9 Scan button

Scan button

This generates a Julia set or quaternion Julia set from a formula's Mandelbrot 'P' space. Random points in a formula's current Mandelbrot space are scanned for an interesting Julia set. Rendering options are maintained in the current fractal. Equivalent to the 'F' hot keys. Not used with lsystem, landscape or orbital fractals.

2.10 Rend button

Rend button

For quaternion, lsystems, landscape, orbital and 3D height fields, the current coloring filter and lighting variables are applied. This allows you to see what the surface texture looks like before the fractal is finished drawing. Note: to randomize the coloring filter, select Random Render from the Demo menu.

2.11 View button

View button

Displays the entire plot, expanding or shrinking the image to fit in a maximized window without title bar, scroll bars or menu bar. At all other times, part of the picture is hidden by the inclusion of the title bar, toolbar, scroll bars and menu bar. To exit full-screen mode, press any key or click the left-mouse button.

2.12 Help button

Help button

Use this button to open the help index for Fractal Zplot.

2.13 Pilot button

Opens the Pilot window to adjust key parameters, rotate, zoom and redraw the figure interactively. The current image is reduced to one quarter normal for faster redraw. Each click on a Pilot button increments or decrements a parameter. The Speed slider controls the rate at which the buttons operate (default is 10.)

Press the space bar or Click on Ok to open a new window and draw the altered image full-size. Press Esc or click on Cancel to exit this mode without opening a new window. Note: when using this option while an AVI stream is open, a new window isn't opened, but the altered figure is drawn in the current draw window, the changed parameters replacing the previous ones.

2.14 Formula button

Formula button

Use this button to change formulas or type.

2.15 Light button

Light button

Edit lightpoint and viewpoint variables

2.16 Random Render button

Random Render button

The rendering options for the current fractal are randomized. Does not affect formula or range variables. For quaternion and 3D height fields, a random coloring filter is applied.

2.17 Channel Guide

Channel Guide

The sixteen channels accessed via the FZ remote:

J1: Random Julia -- basic Julia sets, using one formula and many different rendering options

J2: Random Julia 2 -- includes more advanced composite-type fractals, using two formulas

ES: Random Escher-like fractals -- Julia fractal set using Escher mapping

S/B: Random Bubbles and Stalks -- based on Paul Carlson's bubble and stalk methods

Q1: Random Quaternion/Hypernion -- traditional 3D quaternion and hypercomplex quaternion fractals

Q2: Random Quaternion/Hypernion 2 -- extended search through non-traditional formulas

C1: Random Cubic Mandelbrot fractals

C2: Random Cubic Mandelbrot2 -- relaxed formulas/parameters

CJ: Random Cubic Julia fractals

O1: Random Octonion fractals

O2: Random Octonion2 fractals -- extended dimensional search

QT: Random Quat-trap fractals

HF: Random Julia height fields -- 3D rendering of Julia sets using ray-tracing and continuous potential smoothing

LD: Random Landscapes -- 3D rendering using ray-traced mid-point displacement algorithm

LS: Random Lsystems -- 3D turtle graphics with ray-traced highlights

OR: Random Orbital fractals -- 3D rendering of strange attractors

2.18 Channel J1

Random Julia (Channel J1 button)

A random Julia fractal is generated. Many of the rendering options of Fractal Zplot are selected on a random basis, and the Mandelbrot space for one of the 180 built-in formulas is scanned for an interesting Julia set. The palette used is also randomized. Note: In most case the Julia search is a short one, but sometimes the "seek" mode can seem to get stuck when the criteria for an interesting Julia set fails to match the formula used. In the latter case, either click the left mouse button and restart the search process, or pressing a palette key will sometimes override the current search and restart it (if HSV filtering has been selected.) Tip: some things remain to be done after the Julia set is drawn. The algorithm doesn't check the type of split palette that is used, so a mismatch in the "Divide by" color option may need correcting, e.g. Divide-by-four may be selected for a divide-by-eight palette. Feel free to experiment with all the parameters, reframe the image, change palettes etc. This routine provides a fast intro to many options in Fractal Zplot that the user may be unfamiliar with: no knowledge of fractal science/math required!

2.19 Channel J2

Random Julia (Channel J2 button)
A random Julia fractal is generated. Many of the rendering options of Fractal Zplot are selected on a random basis, and the Mandelbrot space for one of the 180 built-in formulas is scanned for an interesting Julia set. The palette used is also randomized. Note: In most case the Julia search is a short one, but sometimes the "seek" mode can seem to get stuck when the criteria for an interesting Julia set fails to match the formula used. In the latter case, either click the left mouse button and restart the search process, or pressing a palette key will sometimes override the current search and restart it (if HSV filtering has been selected.)

This is like the Random Julia command, except that more options are randomized, including spin and switch, and the Formula Type can be composite or Escher, so the search/draw time may be somewhat longer, and the results not as certain. But the images can be quite weird!

Tip: some things remain to be done after the Julia set is drawn. The algorithm doesn't check the type of split palette that is used, so a mismatch in the "Divide by" color option may need correcting, e.g. Divide-by-four may be selected for a divide-by-eight palette. Feel free to experiment with all the parameters, reframe the image, change palettes etc. This routine provides a fast intro to many options in Fractal Zplot that the user may be unfamiliar with: no knowledge of fractal science/math required!

2.20 Channel ES

Random Escher (Channel QT button)
A random Julia fractal is generated using Escher-like tilings (Type 10 in the Formula window.) The Rise variable in the Parameters window sets the degree of tiling (1-235). Note: Sometimes it helps to reduce iterations if the tilings do not appear to be distributed throughout the circular plane. 20-25 iterations is sufficient for many Escher plots.

2.21 Channel SB

Random Stalks and Bubbles (Channel S/B button)
A random Julia fractal is generated using one of Paul Carlson's orbit traps or bubble method.

2.22 Channel Q1

Random Quaternion (Channel Q1 button)
A random quaternion/ hypernion fractal is generated. Like Random Julia, a set of formulas appropriate for quaternions is scanned to find an interesting Julia set, and then the parameters are adjusted to produce a quaternion or hypernion image. The ranges are reset, Hj is set to 2.0, and the lighting is set for optimum viewing.

Note: for some images an hj value of 2.0 may result in a partially clipped image. Sometimes it helps to increase this value to 2.5 or 3.0, but too high a value may interfere with Solid guessing.

2.23 Channel Q2

Random Quaternion2 (Channel Q2 button)
A random quaternion/hypernion fractal is generated. Like Random Julia, a set of formulas appropriate for quaternions is scanned to find an interesting Julia set, and then the parameters are adjusted to produce a quaternion or hypernion image. The ranges are reset, Hj is set to 2.0, and the lighting is set for optimum viewing.

This option uses an extended set of formulas, not all of which may produce useable images all the time. The images can be quite different from the traditional quaternion.

Note: for some images an hj value of 2.0 may result in a partially clipped image. Sometimes it helps to increase this value to 2.5 or 3.0, but too high a value may interfere with Solid guessing.

2.24 Channel C1

Random Cubic Mandelbrot (Channel C1 button)
A random cubic Mandelbrot fractal is generated. Like the inverse of Random Julia, the essential cubic parameters are randomly adjusted to point into the four-dimensional formula G0, and then the Mandelbrot set is scanned to find an interesting area to zoom into. The ranges are reset, and the lighting is set for optimum viewing.

2.25 Channel C2

Random Cubic Mandelbrot2 (Channel C2 button)
A random cubic Mandelbrot fractal is generated. Like the inverse of Random Julia, the essential cubic parameters are randomly adjusted to point into the four-dimensional formula G0, and then the Mandelbrot set is scanned to find an interesting area to zoom into. The ranges are reset, and the lighting is set for optimum viewing.

This option uses the cubic formulas G0 and G1, with relaxed parameters to create cubic-Mandelbrot like fractals that may extend to six or more dimensions.

2.26 Channel CJ

Random Cubic Julia (Channel CJ button)
A random cubic Julia fractal (the Julia analog of a cubic Mandelbrot fractal) is generated. The essential cubic parameters are randomly adjusted to point into a four-dimensional formula. Like Random Julia, a set of formulas (G0 and G1) appropriate for cubic Julias is scanned to find an interesting Julia set, and then the parameters are adjusted to produce the cubic image. The ranges are reset, Hj is set to 2.0, and the lighting is set for optimum viewing. Note: This is a quasi-Julia

approximation that doesn't follow traditional cubic Mandelbrot theory. The "quaternions" produced by this method do exhibit characteristics of cubic Mandelbrots, but here I am more interested in esthetics than mathematical conformity.

2.27 Channel O1

Random Octonion (Channel O1 button)
A random octonion Julia fractal is generated. The essential octonion parameters are randomly adjusted to point into an eight-dimensional formula H0-H9, and then the octonion Mandelbrot set is scanned to find an interesting area to zoom into. The ranges are reset, and the lighting is set for optimum viewing (if lighting is checked in the Random Batch window.)

2.28 Channel O2

Random Octonion2 (Channel O2 button)
A random octonion Julia fractal is generated. The essential octonion parameters are randomly adjusted to point into an eight-dimensional formula H0-H9, and then the octonion Mandelbrot set is scanned to find an interesting area to zoom into. The ranges are reset, and the lighting is set for optimum viewing.

This option uses the octonion formulas H0-H9, with random dimensional switching (one of OE-OK for Oi) to create octonion fractals that may extend to eight dimensions.

2.29 Channel QT

Random Quat-Trap (Channel QT button)
A random quat-trap fractal is generated using one of Paul Carlson's orbit trap methods and a quaternion image. A separate image is created for both quaternion and orbit-trap formulas, in figures 1 and 2, then the figures are merged and drawn as one image. See Orbit-Trap options for further details on quat-trap pictures.

2.30 Channel HF

Random 3D Julia Field (Channel HF button)
A 3D height field fractal is generated for a random Julia set. Like Random Julia, a set of formulas appropriate for 3D height fields is scanned to find an interesting Julia set, and then the parameters are adjusted to produce a height field image. The ranges are reset and the lighting is set for optimum viewing.

2.31 Channel LD

Random Landscape (Channel LD button)
A random 3D landscape (midpoint displacement fractal) is generated.

2.32 Channel LS

Random Lystem (Channel LS button)
A random lsystem fractal is generated. An .ls file is selected at random from the startup directory, the palette randomized, and then a random amount of mutation is applied. The lighting is set for optimal viewing, and the figure rotated a random amount on the x y and z-axis.

2.33 Channel OR

Random Orbital (Channel OR button)
A 3D orbital fractal is generated by randomizing parameters for one of nine strange attractors. Based on Sprott's method of using random affine transformations to find interesting strange attractors.

2.34 Save button

Save button

Use this button to save and name the active drawing. Fractal Zplot displays the Save As dialog box so you can name your drawing.

To save a drawing with its existing name and directory, use the File/Save command.

2.35 Load button

Load button

Use this button to open an existing data/image file in a new window. You can open multiple image files at once. Use the Window menu to switch among the multiple open images.

2.36 Bmp radio button

BMP button

Use this button to select the BMP format when loading and saving fractals. This is the default Windows bitmap format, readable by most Windows programs that use image files. This is also the fastest method of loading and saving fractals, but requires more disk space, since no compression is used. Windows keeps track of the last six BMP files saved or loaded (displayed in the Files menu.)

2.37 Png radio button

PNG radio button

Use this button to select the PNG format when loading and saving fractals. This format uses medium compression without loss of image quality.

2.38 Jpg radio button

JPG radio button

Use this button to select the JPEG format when loading and saving fractals. This format uses moderate compression but with some loss of image quality. Preferable for posting to the net, since most browsers can display jpeg files.

2.39 ||||| button

||||| button

Through a series of windows, this allows you to name and open an avi animation stream and choose a compression method. After choosing the frame rate (1-60) and using the file requester to name the file, you are given a choice of compression methods. You can also choose no compression for optimum view quality. (All compression methods degrade the original images, some more than others.) The first key frame in the stream is then drawn and written to the file.

Note: after the stream is opened, the size of the fractal that can be drawn is fixed at the size of the frame. No changes can be made to the size until the stream is closed.

2.40 > button

> button

With this option, frames are written to a stream based on the difference between the current key frame and the previous key frame. The first key frame is written when you open a stream. The next key frame is created each time you use this option. In between you can zoom or change Fvr variables as much as necessary. The stream is only written to when this option is used. The last key frame is automatically saved after the 'tween' series is written. The number of frames may range from 1-1500 frames between keys. With a frame number of 1 only the key frames are written. This allows animation to be created that incorporate all scalable variables in Fractal Zplot.

Use the Cancel button to exit this dialog without initializing a new series of frames.

Check the Log Scaling box if you want the frames to be written with logarithmic space between frames, else linear space is used. Useful when zooming, where frames would otherwise be packed together at the end of the frame series.

2.41 [] button

[ ] button

Closes any open avi stream file. You need to do this before viewing the file or creating a new avi file. The stream is also closed when you exit Fractal Zplot.

2.42 V button

V button

Opens an avi file for viewing. You can preview any multimedia file by clicking on its file name. A multimedia box will appear to the right of the file list. Click on okay to open the main view window.

There are buttons to Play a file forwards or Backwards, or forward automatically with Auto rewind/ repeat. Click on Slow to slow down a video. Each click on Slow halves the viewing speed. A click on Stop freezes viewing and restores the view speed to normal playback.

Use the Open button to view a different avi file. Use the Save button to save the file in a different compression format. You must use a different name to save the file than the name that was used to open it. Click on the left-mouse button or any key to abort a save operation.

Note: the view avi requester can be used to preview any multimedia file, including midi files.

3 File menu

File menu commands

The File menu offers the following commands:

New	Creates a new drawing.
Open	Opens an existing drawing.
Close	Closes an opened drawing.
Save	Saves an opened drawing using the same file name.
Save As	Saves an opened drawing to a specified file name.
Load Parameters	Load parameters from an existing drawing.
Load Par File	Load a parameters definition file.
Load Palettes [PL]	Load palettes file.
Open [JPEG]	Load jpeg.
Open [PNG]	Load png.
Save Parameters	Save parameters for an opened drawing to a specified file name.
Save Par File	Save a parameters definition file.
Save Palettes [PL]	Save palettes to file.
Save As [JPEG]	Save in jpeg format.
Save As [PNG]	Save in png format.

Import

Load LSystem [LS]	Load a LParser compatible file.
Load Palette [PQZ]	Load QuaSZ palette file.
Load Palette [MAP]	Load a Fractint map file.
Load Texture	Load QuaSZ texture file [QTX]

Export

Save Palette [PQZ]	Save current palette.
Save Texture	Save texture file [QTX].

Save QuaSZ Parameters	Save quaternion variables to QuaSZ x64 data file [QS6].
Save LSystem [LS]	Save a LParser compatible file.
Save as OBJ	Save polygonized quaternion as Wavefront object.
Simplify Mesh	Simplify mesh.
Save as POV	Save polygonized quaternion as a pov triangle object.
Smooth Triangles	Convert triangle mesh to smooth_triangle mesh.
Save as WRL	Save polygonized quaternion as virtual reality file.
Save as DXF	Save polygonized quaternion as AutoCad dxf file.
Set Max Vertices	Set maximum number of vertices allocated for Q polygon.
Save Lsystem to OBJ	Save Lsystem to Wavefront Object file.
Save Lsystem to STL	Save Lsystem to STL solid file.
Save Lsystem to POV	Save Lsystem as POV triangle object.
Save Lsystem to DXF	Save Lsystem to AutoCad dxf file.
Save Orbital to OBJ	Save orbital fractal to Wavefront Object file.
Save Orbital to DXF	Save orbital fractal to AutoCad dxf file.
Save Height Field to OBJ	Save height field to Wavefront Object file.
Save Height Field to POV	Save height field fractal as POV triangle object.
Save Height Field to DXF	Save height field fractal to AutoCad dxf file.
Mesh Setup	Edit/View parameters for exporting meshes.
Exit	Exits Fractal Zplot.

3.1 File New command

New command (File menu)

Use this command to create a new drawing window in Fractal Zplot. The image and data for the opening picture are used to create the new window.

You can open an existing data/image file with the Open command.

Shortcuts

Keys: CTRL+N

3.2 File Open command

Open command (File menu)

Use this command to open an existing data/image file in a new window. You can open multiple image files at once. Use the Window menu to switch among the multiple open images. See Window 1, 2, ... command.

You can create new images with the New command.

Shortcuts

Toolbar: 
Keys: CTRL+O

3.2.1 File Open dialog box

File Open dialog box

The following options allow you to specify which file to open:

File Name
Type or select the filename you want to open. This box lists files with the extension you select in the List Files of Type box.

List Files of Type
Select the type of file you want to open.

Drives
Select the drive in which Fractal Zplot stores the file that you want to open.

Directories
Select the directory in which Fractal Zplot stores the file that you want to open.

Network...
Choose this button to connect to a network location, assigning it a new drive letter.

3.3 File Close command

Close command (File menu)

Use this command to close the window containing the active image. If you close a window without saving, you lose all changes made since the last time you saved it.

You can also close a drawing by using the Close icon on the drawing window, as shown below:

3.4 File Save command

Save command (File menu)

Use this command to save the active drawing to its current name and directory. When you save a drawing for the first time, Fractal Zplot displays the Save As dialog box so you can name your drawing. If you want to change the name and directory of an existing drawing before you save it, choose the Save As command.

Shortcuts

Toolbar: 

Keys: CTRL+S

3.5 File Save As command

Save As command (File menu)

Use this command to save and name the active drawing. Fractal Zplot displays the Save As dialog box so you can name your drawing.

To save a drawing with its existing name and directory, use the Save command.

3.5.1 File Save as dialog box

File Save As dialog box

The following options allow you to specify the name and location of the file you're about to save:

File Name
Type a new filename to save a drawing with a different name. Fractal Zplot adds the extension .zp6.

Drives
Select the drive in which you want to store the drawing.

Directories
Select the directory in which you want to store the drawing.

Network...
Choose this button to connect to a network location, assigning it a new drive letter.

3.6 File Load Parameters command

Load Parameters command (File menu)

Use this command to load a data file [.zp6]. The data file contains all variables to recreate an image created previously with Fractal Zplot.

3.7 Load Par File

Load Par File

Opens a window that allows the user to select and convert Fractint/Fractal Zplot parameter files to Fractal Zplot format. A list of par titles is displayed for each .par file loaded. The user then selects a title and clicks on Convert to translate it to the current image data. This works for most Fractint v18 and v19 escape-time fractals up to 19.3 (and is downward compatible with most Fractal Zplot fractals.) Fractint 19.6 par files with internal formula definitions (and/or the if/then/else structure) are supported. Excluded are the frothy basin and complex phoenix types. Formula types may now be loaded, although some options such as rand and LastSqr are not supported. (Some built-in

functions like p0 use the LastSqr speedup.) 3-D fractals and other types, such as Orbit Fractals and Bifurcation are not translated. Some of Fractint's 2D options are implemented differently in Fractal Zplot, so the translation may not be exact.

Some Fractint functions are not supported through the f1-f4 boxes, but when they are encountered in a par file, Fractal Zplot writes the name of the unsupported function in the fun#2 box.

The Fractint palette specified in the colors option (if any) is loaded into F12. The skew parameter in the center-mag coordinate mapping mode is not supported.

Select Load Palette to load a par palette *only* without changing the current function data.

Related Topics:
Edit Palette describes the Fractal Zplot palette editor.

3.8 File Load Palettes command

Load Palettes command (File menu)

Use this command to load a palette file [.pl]. The palette file contains 21 palettes created previously with Fractal Zplot (or another version of the program.)

3.9 File Open [JPG] command

Open [JPEG] command (File menu)

Use this command to load parameters and a bitmap file that were saved in jpeg format. There is an option in the file-type box to load only the bitmap too. This replaces the Open command for those who need a smaller sized bitmap file. Note: the last files list doesn't keep track of images loaded in JPEG format. This option uses the jpeg library written by the Independent JPEG Group.

3.10 File Open [PNG] command

Open [PNG] command (File menu)

Use this command to load parameters and a bitmap file that was saved in png format. There is an option in the file-type box to load only the bitmap too. This replaces the Open command for those who need a smaller sized bitmap file. Note: the last files list doesn't keep track of images loaded in PNG format.

3.11 File Save Parameters command

Save Parameters command (File menu)

Use this command to save all data elements for the current image in a data file [.zp6].

3.12 Save Par File

Save Par File

This option allows you to save Fractal Zplot parameters and color info in a text format similar to Fractint's .par format. You can build par libraries of your favorite fractals to share with other users of Fractal Zplot. The par definitions can be easily posted on the net and reloaded in Fractal Zplot through the Load Par command.

Specify the par file name with the Filename button. This can be a generic file name, such as "complex.par", for a library of par titles, or a specific file name based on the fractal's .zp6. The later file name is the default. Use the Par Title box to specify the fractal's name or description. Add a comment to the par definition with the Comment box. When you click on Okay, if a file with the Par Filename doesn't exist, it will be created and the par definition added to it. If the file exists, Fractal Zplot will scan the file for a par title the same as the one being added. If it doesn't exist, the par definition will be added to the end of the par file. If it exists, you have the option to replace the old definition with the new one. If you choose not to replace the old definition, the existing par file remains unchanged and no further action is taken.

For 3D plots, built with multiple images, you need to save a par definition for each image (figure 1-4) used.

Compatibility issues: Since Fractal Zplot has many options not found in Fractint, no attempt was made to make the Fractal Zplot par format compatible with Fractint's. The object was to make Fractal Zplot picture parameters more accessible to Fractal Zplot users. As formula files add another unnecessary layer to the parameter puzzle, Fractal Zplot's par format saves any user-defined formulas in the par definition rather than creating a separate formula file. The color info is converted into a condensed ASCII format for the par definition that also differs from Fractint's color-coding.

This command is not available for the orbital fractal type.

3.13 File Save Palettes command

Save Palettes command (File menu)

Use this command to save all palettes for the current session in a palette file [.pl].

3.14 File Save As [JPG] command

Save As [JPEG] command (File menu)

Use this command to save the parameters and active bitmap in jpeg format. There is an option in the file-type box to save only the bitmap too. This replaces the Save and Save As command for those who need a smaller sized bitmap file. Note: the last files list doesn't keep track of images saved in JPEG format. This option uses the jpeg library written by the Independent JPEG Group.

3.15 File Save As [PNG] command

Save As [PNG] command (File menu)

Use this command to save the parameters and active bitmap in png format. There is an option in the file-type box to save only the bitmap too. This replaces the Save and Save As command for those who need a smaller sized bitmap file. Note: the last files list doesn't keep track of images saved in PNG format.

3.16 File Load Lsystem command

Load Lsystem [LS] command (File menu)

Use this command to load an LParser-compatible .ls file. The level of recursion, basic angle, starting line width, axiom and rules are extracted from the file and converted to Fractal Zplot format.

3.17 File Load Palette [PQZ] command

Load Palette [PQZ] command (File menu)

Use this command to load a QuaSZ-style palette file [.pqz]. The palette file contains a single palette that replaces the current palette.

3.18 File Load Palette [MAP] command

Load Palette [MAP] command (File menu)

Use this command to load a Fractint-type map file. The palette in the map file replaces the currently selected palette.

3.19 File Load Texture command

Load Texture command (File menu)

Use this command to load variables that make up the texture and noise parameters. This also loads the palette, coloring filter, orbit trap and coloring options in a texture file [qtx].

3.20 File Save Palette command

Save Palette [PQZ] command (File menu)

Use this command to save the current palette to a QuaSZ-style palette file [.pqz].

3.21 File Save Texture command

Save Texture command (File menu)

Use this command to save the variables that make up the texture and noise parameters for the current figure. This also saves the palette, coloring filter, orbit trap and coloring options in the texture file [qtx].

3.22 File Save QuaSZ command

Save QuaSZ parameters command (File menu)

Use this command to save the quaternion variables that make up the current figure in a QuaSZ x64 compatible data file [QS6]. You can load this file into QuaSZ x64 via the File/Load Parameters command.

3.23 File Save Lsystem command

Save Lsystem [LS] command (File menu)

Use this command to save the current lsystem in an LParser-compatible .ls file. This is useful for exporting an lsystem to Crocus or another program that supports the LParser format.

3.24 File Save Q Polygon[OBJ] command

Save Q Polygon [OBJ] command (File menu)

Use this command to save a quaternion as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a quaternion formula, and then writes the triangles to a Wavefront object file. Precision is set with the Steps variable in the Quaternion window, where precision=10/Steps.

After entering the name of the object in the file requester, you click on a starting point where the exporter can begin mapping the 3-D fractal. The starting point must lie on a solid (non-empty) part of the 3-D fractal.

Note: some formulas produce unsymmetrical object files with this routine, where one side of the q polygon isn't resolved completely. Usually one side is markedly smoother in this case.

3.25 File Simplify mesh command

Export -> Simplify Mesh command (File menu)

When this flag is set (default on) the object meshes are simplified using Garland's mesh-simplification

algorithm before outputting to a Wavefront obj or POV mesh file, resulting in a much smaller export file. Set the number of facets in the target mesh file with the Mesh Setup command. You can set the resolution of the object as high as necessary (with the Params/Steps variable) to produce a finely detailed quaternion, but the output file remains about the same. Use the smoothing feature in Bryce to smooth the resulting object mesh.

3.26 File Save Q Polygon[POV] command

Export Q Polygon [POV] command (File menu)

Use this command to save a quaternion as a true 3-D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a quaternion formula, and then outputs the triangles to a pov file. The pov file is written as a simple scene, the triangles part of a “union” object, with camera and lighting elements compatible with POV 3.5. This can be used as a starting point for more complex compositions. Precision is set with the Steps variable in the Quaternion window, where precision=10/Steps.

After entering the name of the object in the file requester, you click on a starting point where the exporter can begin mapping the 3-D fractal. The starting point must lie on a solid (non-empty) part of the 3-D fractal.

Note: some formulas produce asymmetrical object files with this routine, where one side of the q polygon isn't resolved completely. Usually one side is markedly smoother in this case.

3.27 File Smooth command

Export -> Smooth command (File menu)

When this flag is set (default on) the object facets are converted to smooth_triangles before outputting to a POV mesh file. Surface normals are calculated for all triangles that share common vertices.

3.28 File Save Q Polygon[WRL] command

Save Q Polygon [WRL] command (File menu)

Use this command to save a quaternion as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a quaternion formula, and then writes the triangles to a virtual reality file. Precision is set with the Steps variable in the Quaternion window, where precision=10/Steps.

After entering the name of the object in the file requester, you click on a starting point where the exporter can begin mapping the 3-D fractal. The starting point must lie on a solid (non-empty) part of the 3-D fractal.

Note: some formulas produce unsymmetrical object files with this routine, where one side of the q polygon isn't resolved completely. Usually one side is markedly smoother in this case.

3.29 File Save Q Polygon[DXF] command

Export Q Polygon [DXF] command (File menu)

Use this command to save a quaternion as a true 3-D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a quaternion formula, and then outputs the triangles to a dxf file. The memory requirements for this routine are 20MB or more for a typical Julia set quaternion rendered at 320X240. The output file can be very large too, up to 40MB or more, at the highest precision. The higher the precision, the smoother the finished object becomes. Precision is set with the Steps variable in the Quaternion window, where precision=10/Steps.

After entering the name of the object in the file requester, you click on a starting point where the exporter can begin mapping the 3-D fractal. The starting point must lie on a solid (non-empty) part of the 3-D fractal.

Note: some formulas produce asymmetrical object files with this routine, where one side of the q polygon isn't resolved completely. Usually one side is markedly smoother in this case.

3.30 File Set Max Vertices command

Set Max Indices (File menu)

Use this command to set the maximum number of indices that are allocated by the polygonizing routine. Default is 5,000,000 indices. Use less to limit the amount of memory used while polygonizing. Use more if necessary for higher resolution. Note: unless you have an application that can use very large object files, there's a limit to how much resolution is obtainable with the polygonizing routine. Bryce6 has problems with object files produced by FZ that are much larger than 2 MB.

3.31 File Save Lsystem to Obj command

Save Lsystem to Obj command (File menu)

Use this command to save an lsystem as a true 3D object. This is useful to export an lsystem for use in Bryce or another program that supports the Wavefront format. Precision is defined by the number of Steps in the Draw LS window (default 1). Hint: You can reduce the complexity of an lsystem (and its obj file size) by decreasing the Max. Lines variable.

3.32 File Save Lsystem to Stl command

Save Lsystem to Stl command (File menu)

Use this command to save an lsystem as a stereolithographic solid object. Precision is defined by the number of Steps in the Draw LS window (default 1). Hint: You can reduce the complexity of an lsystem (and its stl file size) by decreasing the Max. Lines variable.

3.33 File Save Lsystem to POV command

Save Lsystem to POV command (File menu)

Use this command to save a lsystem as a POV triangle object. Precision is defined by the number of Steps in the Draw LS window (default 1). Hint: You can reduce the complexity of an lsystem (and its pov file size) by decreasing the Max. Lines variable.

3.34 File Save Lsystem to Dxf command

Save Lsystem to Dxf command (File menu)

Use this command to save an lsystem as a true 3D object. This is useful to export an lsystem for use in Bryce or another program that supports the AutoCad dxf format. Precision is defined by the number of Steps in the Draw LS window (default 1). Hint: You can reduce the complexity of an lsystem (and its dxf file size) by decreasing the Max. Lines variable.

3.35 File Save Height Field to POV command

Save Height Field to POV command (File menu)

Use this command to save a height field as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a 3D Julia formula, and then writes the triangles to a POV ray-tracing file. The output file can be very large too, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object. Precision is set with the Steps variable in the Parameters window, where precision=10/Steps. A value of 200-300 works best.

Note: there is limited support for zooming into height fields (by switching from a zoomed 2-D fractal to the 3-D Height field type, a closer view of the height field surface can be seen.). The polygonization routine works best for “complete” objects. However if you want to experiment with partial views, the Y-offset value controls the depth of the scan. Increasing this value decreases the height of the partial object generated. The object is clipped at the x ranges, but not the y ranges. The Steps value is limited to about 230, due to memory allocation restraints. For Obj files this isn't such a problem because the object can be smoothed later after exporting. Pov files of this resolution are best without too much Phong highlights, as these tend to exaggerate the triangular nature of the height field object.

3.36 File Save Height Field to dxf command

Save Height Field to Dxf command (File menu)

Use this command to save a height field as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a 3D Julia formula, and then writes the triangles to an AutoCad dxf file. The output file can be very large too, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object. Precision is set with the Steps

variable in the Parameters window, where precision=10/Steps. A value of 200-300 works best.

Note: there is limited support for zooming into height fields (by switching from a zoomed 2-D fractal to the 3-D Height field type, a closer view of the height field surface can be seen.). The polygonization routine works best for "complete" objects. However if you want to experiment with partial views, the Y-offset value controls the depth of the scan. Increasing this value decreases the height of the partial object generated. The object is clipped at the x ranges, but not the y ranges. The Steps value is limited to about 230, due to memory allocation restraints.

3.37 File Save Height Field to Obj command

Save Height Field to Obj command (File menu)

Use this command to save a height field as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a 3D Julia formula, and then writes the triangles to a Wavefront object file. The output file can be very large too, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object. Precision is set with the Steps variable in the Parameters window, where precision=10/Steps. A value of 200-300 works best.

Note: there is limited support for zooming into height fields (by switching from a zoomed 2-D fractal to the 3-D Height field type, a closer view of the height field surface can be seen.). The polygonization routine works best for "complete" objects. However if you want to experiment with partial views, the Y-offset value controls the depth of the scan. Increasing this value decreases the height of the partial object generated. The object is clipped at the x ranges, but not the y ranges. The Steps value is limited to about 230, due to memory allocation restraints.

3.38 File Orbital to Obj command

Save Orbital fractal to Obj command (File menu)

Use this command to save an orbital fractal as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize an orbital formula, and then writes the triangles to a Wavefront object file. Precision is set with the Steps variable in the orbital parameters window, where precision=10/Steps.

Notes: This works slightly differently from the other polygonizing routines in that you have to draw the fractal first (or anytime the fractal data file is reloaded) before the command is enabled. This routine doesn't work with many orbital fractals, where the bands are narrow or there is empty space in the center of the fractal. The polygonizing routine needs a solid area to start with. If the routine finishes very quickly, it probably didn't find a starting point. This usually results in a blank obj file or one less than 10kb. The steps required to produce a useable obj model may be substantially more than that of quaternions or height fields. It's best to start with 200-400 and increase gradually until the output file size is in the megabytes. It helps to increase iterations as much as possible to improve object smoothness. Hint: by turning off the Auto Redraw option in the Image menu you can adjust the Steps variable without redrawing the orbital each time you polygonize it. So you only draw the orbital once at the maximum resolution needed.

3.39 File Orbital to Dxf command

Save Orbital fractal to Dxf command (File menu)

Use this command to save an orbital fractal as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize an orbital formula, and then writes the triangles to an AutoCad dxf file. Precision is set with the Steps variable in the orbital parameters window, where precision=10/Steps.

Notes: This works slightly differently from the other polygonizing routines in that you have to draw the fractal first (or anytime the fractal data file is reloaded) before the command is enabled. This routine doesn't work with many orbital fractals, where the bands are narrow or there is empty space in the center of the fractal. The polygonizing routine needs a solid area to start with. If the routine finishes very quickly, it probably didn't find a starting point. This usually results in a blank obj file or one less than 10kb. The steps required to produce a useable obj model may be substantially more than that of quaternions or height fields. It's best to start with 200-400 and increase gradually until the output file size is in the megabytes. It helps to increase iterations as much as possible to improve object smoothness. Hint: by turning off the Auto Redraw option in the Image menu you can adjust the Steps variable without redrawing the orbital each time you polygonize it. So you only draw the orbital once at the maximum resolution needed.

3.40 File Orbital to POV command

Save Orbital fractal to POV command (File menu)

Use this command to save an orbital fractal as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize an orbital formula, and then writes the triangles to a POV mesh file. Precision is set with the Steps variable in the orbital parameters window, where precision=10/Steps.

Notes: This works slightly differently from the other polygonizing routines in that you have to draw the fractal first (or anytime the fractal data file is reloaded) before the command is enabled. This routine doesn't work with many orbital fractals, where the bands are narrow or there is empty space in the center of the fractal. The polygonizing routine needs a solid area to start with. If the routine finishes very quickly, it probably didn't find a starting point. This usually results in a blank obj file or one less than 10kb. The steps required to produce a useable obj model may be substantially more than that of quaternions or height fields. It's best to start with 200-400 and increase gradually until the output file size is in the megabytes. It helps to increase iterations as much as possible to improve object smoothness. Hint: by turning off the Auto Redraw option in the Image menu you can adjust the Steps variable without redrawing the orbital each time you polygonize it. So you only draw the orbital once at the maximum resolution needed.

3.41 Mesh Setup command

Mesh Setup command (File menu)

Here you edit or view the parameters for simplifying meshes when outputting in Wavefront [obj] or

POV format.

Max export faces determines the size of the export mesh, if the original mesh is larger than the target size. Triangle faces are "collapsed" until the target face size is reached.

Max input faces and max input vertices determine how much memory is set aside as buffers for processing the meshes. Increase or decrease from the default values as the size of the mesh warrants, or as system memory permits.

The "min width" variable controls how narrow individual segments in a lsystem are allowed to shrink at the highest level of recursion. Increase this value if the cylinders in the object file become too narrow to render correctly in Bryce. Decrease if the cylinders look oversized.

The weld factor controls how close adjacent triangle vertices of a mesh may be before they are merged into one vertex. This effectively flattens adjacent triangles or "collapses" them and reduces mesh size. Use a small enough weld factor that produces an evenly simplified mesh without destroying the integrity of the smallest elements of the mesh. Use the smoothing routine in Bryce to restore the mesh to optimum curvature. POV triangle meshes are further processed into "smooth_triangles" before exporting into the final mesh file.

3.42 File 1, 2, 3, 4, 5, 6 command

1, 2, 3, 4, 5, 6 command (File menu)

Use the numbers and filenames listed at the bottom of the File menu to open the last six drawings you closed. Choose the number that corresponds with the drawing you want to open.

3.43 File Exit command

Exit command (File menu)

Use this command to end your Fractal Zplot session. You can also use the Close command on the application Control menu. Note: if you choose to exit while plotting, the program does not terminate, but stops the plotting so the program can be safely exited.

Shortcuts

Mouse: Double-click the application's Control menu button.

Keys: ALT+F4

4 Edit menu

Edit menu commands

The Edit menu offers the following commands:

Undo	Undo last edit, action or zoom.
Copy	Copy the active view and put it on the Clipboard.
Clip	Define area of view and copy to clipboard.
Paste	Insert Clipboard contents.
Copy Data	Copy fractal data to buffer.
Paste Data	Copy data from copy buffer.
Formulas/Type	Edit formula/type data.
Fractal Variables	Edit fractal variables.
Cubic Values	Edit cubic parameters.
Octonion Values	Edit octonion constants.
Quat-Trap	Edit quat-trap parameters.
Size	Sets the image size.
Ray-Tracing Variables	Edit lighting and viewpoint variables.
Palette Editor	Edit palette.
RGB Thresholds	Set threshold values for color interpolation.
Text	Edit and add text to drawing.
Preferences	Startup preferences and defaults.

4.1 Edit Undo command

Undo command (Edit menu)

Use this command to undo the last action. An image can be continued after an undo, if continue was enabled before the last action. Color-cycling is disabled after using Undo, though.
Undo is disabled for bitmaps whose width is not a multiple of 8.

Shortcut

Keys: CTRL+Z

4.2 Edit Copy command

Copy command (Edit menu)

Use this command to copy the active view to the clipboard. The entire view is copied to the clipboard.

Shortcut

Keys: CTRL+C

4.3 Edit Clip command

Clip command (Edit menu)

Use this command to copy a part of the active view to the clipboard. A zoom box is used to select the part to be copied. Click outside the view frame or press escape to exit this option.

Shortcut
Keys: CTRL+L

4.4 Edit Paste command

Paste command (Edit menu)

Use this command to paste from the clipboard. The clipboard must contain a bitmap. If the bitmap is larger than the view, it is clipped. The zoom cursor is used to set the left/top corner in the view where the bitmap will be pasted. Click outside the view frame or press escape to exit this option.

Shortcut
Keys: CTRL+V

4.5 Edit Copy Data command

Copy Data command (Edit menu)

Use this command to copy the fractal data for the active view to the file "c:\zcopy.zp6". The current palette for the view is also copied.

Shortcut
Keys: CTRL+F

4.6 Edit Paste Data command

Paste Data command (Edit menu)

Use this command to paste the data in the file "c:\zcopy.zp6" to the active view. The palette stored in the file is copied to palette 10(F11).

Shortcut
Keys: CTRL+R

4.7 Formula Window

Formula/Type Window
Fun #1 and Fun #2 are combo controls for entering/selecting up to two formulas either from their

drop-down lists or in the custom form of Z=AZ+BZ+c. Z is the complex variable or function, 'c' is the complex constant, and A and B are optional real constants. There are additionally a Type control, an Arg control and an Arg Limit control that determine how the above formulas are processed. Note: when the fractal type is quaternion, custom formulas are only processed through the Fun #1 or the Formula box. The Fun #2 combo only uses the built-in formulas in its drop-down list.

The Type control accepts a value of 0 to 11. For a value of 0, the first formula is always used and the second formula is ignored. For a value of 1, the second formula is processed and the first formula is ignored.

Type 1 is of use only if you are switching between two functions and don't want to reenter them each time you plot the other one.

For a value of 2, the first formula is processed if the real component of Z is greater than 0, else the second formula is used.

For values of 3, the first formula is processed and its output becomes the input of the second formula, which is then processed.

Type 4 takes the average of Fun#1 and Fun#2.

Type 5 alternates between the two functions while iterating.

Type 6 takes the quotient of both functions.

Type 7 uses the lowest iterative results of both Fun#1 and Fun#2

Type 8 uses the highest iterative results of both Fun#1 and Fun#2

Type 9 uses the Formula box to enter up to 1000 characters per formula.
Text can be pasted from the clipboard to the formula box by using the keystrokes shift-insert. Text may be moved from box to box by using shift-delete to move it first to the clipboard.

Type 10 produces Escher-like tilings for Julia sets, as described in The Science of Fractal Images. This uses both fun#1 and fun#2 in combination. Fun#1 sets the stage for the target set fun#2. To reproduce the images in The Science of Fractal Images, use z=z*z for fun#1 and p0 for fun#2. Fun#1 may be any formula other than z=z*z, but may produce unsymmetrical tilings. Fun#2 can be any built-in formula. This type is mainly applicable to Julia-type sets, but may be used with limited effect with the MandelbrotP-type set. Use the Rise box in the Parameters window to set the degree of tiling desired. The higher the Rise number (1-235), the greater the density of the tiling. Note: for quaternion "Escher" fractals, the Rise number has no effect, and Fun#1 is processed without hypercomplex extensions. This results in a 3D tiling effect on the x/y axis that can be multiplied by increasing the number of terms in Fun #1, as in z=z*z*z.

Type 11 uses Fun#1 to produce "genetic" style fractals. Here you enter a character string or word

that can contain any of 25 letters 'a' through 'y'. Each letter or "gene" references a built-in formula, as defined in the Genes window. For every iteration the program cycles through the character string and uses whatever formula is referenced by the characters. Notes: If Max. Iter is shorter than the gene string then only part of the string is used. For quaternions, which usually have a very short iteration cycle, changing Max. Iter can have a dramatic effect on the fractal output.

The f1-f4 combo boxes are used to designate the 'fn' user-definable part of a generalized function. This can be one of the 41 function types listed (sin(w),cos(w) etc.) A few of the options are formulas themselves (L3 (w) the Legendre function, the gamma function and G(w)the Gaussian function), so quite complex (though mathematically unintelligible) formulas are possible.

The S control is used to enter the variable 's' used in many of the built-in functions.
The Si control is used to enter the variable 'si' the imaginary component for s in some of the built-in functions.

The Converge control is used to enter the convergence limit for Renormalization or Convergence plots. This is a measure for how close z needs to approach an attractor to be mapped to that attractor color. A small value .0001-.000001 is normally used. Smaller values of the convergence variable produce a more accurate plot, while increasing the computation time somewhat. On some formulas it may be necessary to reduce this variable to its smallest value (.00000001) to eliminate some artifacts (spots) caused by non-convergence (at a higher limit value.)

For the random displacement routines, the S box is used to control the fractal dimension (valid inputs are 0-1.0). A lower number gives the fractal a higher dimension, and thus more ragged and choppy appearance. In 3D mode the magnify box works as for other 3D plots, except that a value of .4-1.0 is adequate for most plots. For 2D plots or backgrounds, the magnification factor is used as a color scalar. A value less than 1.0 reduces the number of colors by the same factor. This is necessary to create more realistic clouds. The color palette for 2D plots is rotated by the cutoff number. This allows some independent color control given the limitations of a palette-based system. The arg box is used to control shaping and fullness of the finished plot. The plot may include random additions at every point or just the end points. The shape may be scaled linearly or non-linearly. The following values for the arg box control random additions and shape:

0 -- random additions at end points only; linear shaping.
1 -- random additions at every point; linear shaping.
2 -- random additions at end points only; non-linear shaping.
3 -- random additions at every point; non-linear shaping.

About formula syntax: This applies if you elect to enter your own formula into one of the function boxes and use the parser to generate the plot. The use of parenthesis is necessary around complex exponents or variables. E.g.: '(z-i)^(1.5e)'. For a complete list of variables, operators and functions recognized by the parser, see Parser Information.

Up to 500 user-named-complex variables and constants may be included in a formula. A variable must begin with a letter and may contain numbers and letters only. A variable may be up to 9 characters long. A constant may be up to 20 digits long, including the decimal point. Fractal Zplot

uses syntax similar to Fractint's formula style with an initialization section, followed by the main formula, and an optional bailout routine. Comments may be entered on the same lime with a preceding ';'. Some variables such as 'pixel' and p1 are named after Fractint's predefined variables. These are provided to allow Fractal Zplot users to more easily convert Fractint formula types to Fractal Zplot use. However, Fractal Zplot doesn't prompt you to enter values into p1 (the cr and ci boxes) or p2 (the s and si boxes) or p3 (the limit and converge boxes). Since p1 is used in the iteration process as 'c', p1 cannot be used as a variable independent of c. A ':' terminates the initialization section. Multiple phrases may be entered in the main formula or initialization sections on the same line by using the terminator ',' between phrases. Use ctrl-enter to terminate a line in the formula box. An optional bailout routine may be entered as a phrase at the end of the formula. If the bailout phrase equals a value other than TRUE during iteration, the iteration loop is exited. There are other flags such as Convergence and Biomorph, if set, that can force exit also.

The arg control is limited to 25 characters.

The Title text box is used with the hot key 'T' to annotate a picture with text. Use the Edit/Text command to change font, text color or format text into multiple lines. Text in this box is not saved in a picture's data file, but once entered the same text can be used over and over for different pictures. Useful for adding copyright/author info to batches of pictures. Since the same title text may be used many times, it is shared among views and saved in the file "prefs.txt" in Fractal Zplot's startup directory.

Click on the Apply button to use the formulas currently displayed in the window. Use the Okay button to apply the formulas and close the window, or use Cancel to exit the window without making any changes or cancel current changes. If changes have been applied before clicking on Cancel, then the formulas will revert to the state when the window was last opened. Note: commands exterior to this window (as in the Demo menu) may cause the window to be closed and reopened with new values. In this case Cancel only returns the formula values to the "new" values.

The Reset button returns all boxes and slider values to their original values when the window was opened.

The three buttons named Rand fun#1, Rand fun#2 and Rand f1-f4 are used to pick formulas/ functions at random. Clicking on Rand fun#1, a formula is chosen (from the 200+ built-in formulas) for fun #1. Clicking on Rand fun#2, a formula is chosen for fun #2. Clicking on Rand f1-f4, functions are chosen for f1-f4.

Select FraSZle Formulas to switch to the FraSZle formula set for more compatibility with QuaSZ. You can then export any quaternion done with Fractal Zplot as a QuaSZ parameter file [QSZ] and import it into QuaSZ for more advanced surface coloring, etc.

4.8 Fractal Variables

Fractal Variables (Edit Menu)

The window opened varies with the fractal type selected, and contains all the major variables that Fractal Zplot now scales between key frames of an avi stream.

4.8.1 Parameters Window

Parameters Window

There are edit controls for entering the complex constant (real and imaginary parts), and the min/max ranges for the real and imaginary window coordinates. Fractal Zplot uses three-corner plotting for easier rotating, so boxes are provided for Top Left, Top Right and Bottom Right real/imaginary coordinates. These reflect the current range values that may have been derived from zooming with the Zoom option. Slider-type controls affect the number of iterations (1-9999), the z-limit (1-9999). Cj, ck, and hj are for entering hypercomplex parameters. Cx, cy, cz and cw are used as complex C increments for Julia-Tower types. The Size slider controls the overall size of the picture. The Size slider sets the horizontal resolution, while the vertical resolution is then scaled according to the full-screen VGA ratio, 4 to 3(1:1 if that aspect is selected through the Auto menu.) The Sector slider controls which of 4 sectors the picture will be drawn in, if the Size is less than or equal to (the full-screen horizontal resolution)/2. Otherwise the picture is centered according to the full-screen dimensions. This allows you to show zooms of a particular function by using different sectors, or show the affect of different plotting options. Each sector is erased individually. Note: if you try to continue a plot in a different sector than you started with, the plot will continue in the original sector. The Thumbnail button next to the Size slider is used to set a thumbnail size quickly. The thumbnail size toggles between 1/4 and 1/8 of the horizontal screen resolution, e.g. 200X150 or 100X75 for an 800X600 screen.

The more iterations used, the longer it takes to plot a function, but more detail will be present. 10-20 is sufficient for most biomorphs, while more iterations will be required for Mandelbrot and Julia sets, depending on the detail required.

Start color and end color boxes are provided to limit the palette colors that the current image uses. Use start color of 0 and end color 235 to use the full palette. The color-cycling keys (arrows and enter/backspace) work for only those colors that the start/end colors designate.

Zooming is not available while plotting height fields in 3D.

The screen or sector is always cleared before beginning a new 2D plot, but not with 3D plots or 3D backgrounds. Use 'C' to clear the screen for a new 3D plot. Use 'B' to clear a 3D background.

The Reset button returns all boxes and slider values to their original values when the window was opened.

For 3D plots there are four additional edit boxes, the Magnification Factor, Y-Offset, Slope/Cutoff and Rise. The Magnification Factor controls the amount of Z-Axis depth for each point plotted. This can range from 0 to 10000.0. Use less for smaller plots. The amount of magnification used depends on the smoothing factor, size of plot and iterations. Excess magnification can result in a plot being clipped off the screen. Note: for a magnification of 0.0, the curve is totally flat and the lower boundary is clipped to the line being drawn, instead of a level starting from the boundary of the plot sector. In order to create full-screen three-dimensional plots, a larger triangular plot is clipped to the sector's square limits. The Size slider must be set to at least double the sectors 2D horizontal

resolution for a seamless 3D rendering. The 3D plot is drawn left to right, so that the lower edge of the plot can be set with the Y-Offset box. Potential is mapped with continuous color changes from the bottom of the plot to the starting line. The Y-offset should be adjusted so that the starting line is nearly invisible, with as smallest a section shown as possible, to minimize odd color changes at the start of the plot. For some views, it is impossible to avoid a clipped-polygon effect if the potential are high at the start of the function's min/max zones. You need to zoom out somewhat, to retain the 2D image area in full-screen 3D. Also, a horizon is established on all 3D plots at approximately 1/3 down from the top of the plot. This works by clipping all points above the horizon, but retaining the height of all points at or below the horizon.

The Y-Offset controls centering of the plot up and down. The plot is automatically centered left to right. Use a small positive or negative offset to start (under 0.5 or zero), then go from there. A positive y-offset moves the plot down on the screen. This value will change with the size of the plot or with increases in magnification. Also, less Y-Offset is generally required on Low plots than High plots. The slope factor is used to determine how steep points near the upper limit of escape times appear on screen. Start with a value from 1 to 2 with Log Plots. A lower limit of 1.0 is imposed for the smoothing factor. Continuous potential plots usually use a slope value of 2.0. A higher magnification factor is necessary to bring out detail on close-ups.

Since the magnification needed for extreme close-ups increases exponentially with the iterations used, the magnification variable also represents an exponentially increasing factor: 'magnify^(log (iterations))'.

For 3D (height field) plots, the Rise box is used to specify a height for the set potential. Normally the set is mapped at zero or the highest potential (# of colors-1.) You can change this to create Mandelbrot lakes or sheer cliffs. Using a Rise value of less than 0 automatically maps the set potential at the highest potential. The rise value also sets the color index for the Mandelbrot lake with ray-traced plots.

For 2D plots, the Cutoff box acts as a palette multiplier or divider, depending on whether the value entered is less than or greater than 1.0. The palette color is divided by the Cutoff to speed up or slow down color changes. Cutoff values are limited to a low minimum of .001. For level curves and the add and multiply color-scaling options, use a negative cutoff value to maintain a smooth palette. This ensures that the multiplier is used before the (floating-point) palette values are converted to (integer) palette indexes.

For random displacement plots, the Rise value determines the minimum height plotted. Heights below the Rise value are treated as 0 altitude. For 2D plots, pixels at minimum height are not plotted.

The Rotate box is used to rotate the picture by any degree. This rotates all three points that define the Z-plane. The rotational angle is reset to zero after each use.

Related Topic:
Quaternion describes the Quaternion generator's data-collection window.

4.8.2 Quaternion Window

Quaternion Window

This is the data-collection window for Fractal Zplot's Quaternion generator. Minx, maxx, miny and maxy are the spatial variables for framing the quaternion object. These are usually updated automatically when you use the zoom box. Min Z and Max Z define the three-dimensional space that is used to map the quaternion image. Normally Min Z is the negative of Max Z, but Min Z can be adjusted in the positive direction to shear off the front of the quaternion object. This has the effect of exposing the insides of a quaternion. Const1-4 are cr, ci, cj and ck respectively. Maxiter is the same as iterations in the Parameter's window. Three rotate variables determine the 3D angle of rotation. Step and Fine are pitch adjustments that bear on the quality of the plot at an expense of lengthier calculations.

The Bailout variable can range from .000001 to 65000. For most escape-time formulas that have an attractive point at infinity, the bailout is set to a value 4 or greater. A larger value tends to make the figure smoother and fuller with some loss of detail. When the variable is set to a value less than one, then a bailout routine geared to convergent formulas such as built-in formulas that use Newton's method. The bailout uses Alessandro Rosa's "cutoff rate" routine to limit the basins of attraction (to viewable 3D areas) for formulas that use Newton's or Halley's method, etc. This method also works for escape-time formulas, though slower and requiring more iterations to bring out detail.

When you click on Okay, the quaternion generator looks at the Fun#1 gadget in the New Formula window. If this contains a preset variable that function is iterated for its escape time, then the results are ray-traced in any 3D object that may be created. If the Type is set to 9, any custom formula (Fractint-style: as in 'z=z^3+c#') in the Formula box is used. Not all functions may produce a usable 3D object with this method, but it's interesting to experiment with. The Quaternion option works for both Mandelbrot and Julia sets, depending on which type is selected.

The Slice variables may be altered to display different 3-D planes of the quad set.

The Plane variables B (Back), F (Front), and P (Position) allow you to flatten part of the quaternion figure. For normal plotting, these variables default to 0. To have any effect on the image, the Back variable must be different in value from the Front variable. The difference between back and front variables determines where the image is flattened. These variables are limited to +/-9.99, with normal values being in the range –z to +z. The position variable sets the color of the flattened plane.

4.8.3 Axiom Window

Axiom Window

Enter the basic angle, 0.5-360. This angle will be used for any turning command that doesn't use an additional argument. Enter an axiom, up to 255 characters in the axiom box. Enter the initial width of the line, (>=1.) Angle and width variables can be non-integer. A line with width greater than one

causes the turtle to draw a tubular line in 3D space. See the 's' rule (Demo/Lsystem info) in the extended rule set for info on changing the line shape. Click on Apply to apply the current axiom to the image. Click on Okay to set the new axiom into memory and close the window. Click on Sample or Random Sample to use one of the pre-existing lsystems in the startup directory. Click on Rules to edit the rules associated with the current axiom.

4.8.4 Rules Window

LSystem Rules

The top row contains the command line buttons, and the input boxes below are used to enter production rules and labels.

The input boxes:

'Rule' -- enter up to 255 characters for each production rule.

'Label' -- the label for the production rule. Can be any ASCII character. If the character has no keyboard equivalent, you can enter it as its decimal number (0-255.) This may or may not produce a visible character when drawn in the rules list. Using shift-delete and shift-insert for these characters enables them to be copied and pasted into any rule.

The command line buttons:

'Add' -- adds whatever production rule has been entered into the string gadgets, if the label does not duplicate a label already in use. Up to 255 rules can be entered. The window below the string gadgets displays the current production rules in use. Use the slider device to scroll through the rules, if more than nineteen rules are used.

'#' & 'Recall' are used in conjunction to edit a previously added rule. Enter the number of the rule, as displayed below, into the # box and click on Recall. The rule is entered into the input boxes. (Alternately, you may click on a displayed rule with the cursor and left-mouse button, and it will be recalled.) This enables 'Change' and 'Delete' to change or delete the rule. An extra step here prevents changing or deleting the wrong rule by mistake. Hint: you can also change the rule and then Add it in the same operation, if the label is changed. Use 'Cancel' to change another rule, if you don't really want to change or delete the rule you Recalled. (# box is disabled during a Recall operation.)

'Cancel' -- cancels the recall operation.

'Insert' -- inserts a rule before the numbered rule as entered in the # box. You can insert a rule after recalling and changing it, too. You can't insert a rule with a label already in use. You must add at least one rule before you can insert a rule.

'Okay' -- terminates the edit-rule session and closes the edit rule window.

4.8.5 Orbit Function

Orbit Function

Function is the edit control for entering the two-letter code for one of 17 built-in orbital functions. A quick reference to the built-in formulas is found here.

N is the number of iterations used to produce the fractal. The demo menu defaults to 1000000 for exploratory purposes. Increase to its maximum value (999999999) for a smooth finished fractal. Change the angle of rotation with the three rotational variables. Options for the Flame-type function (R6) are provided as a set of check boxes.

Click on Apply to start a new plot, or Okay to apply and close the window.

After the fractal is drawn you may want to try to export it as a model for other 3D graphics programs. The Steps variable controls the resolution of the export model. The Export Orbital command (File menu) only works with some orbital fractals, mainly those that have a broad surface area at the center.

The Reset button returns all edit boxes to their original values when the window was last opened.

4.8.5.1 Orbits Quick Reference

Orbits Quick Reference

p0: -- Sprott strange attractor #1
p1: -- Sprott strange attractor #2
p2: -- Sprott strange attractor #3
p3: -- Sprott strange attractor #4
p4: -- Sprott strange attractor #5
p5: -- Latoocarfian
p6: -- Latoocarfian Mutation Alpha
p7: -- Latoocarfian Mutation Beta
p8: -- Latoocarfian Mutation Ceta
p9: -- 3D Attractor (C. Pickover)
r0: -- Lorenz 3D Attractor (E. Lorenz)
r1: -- Million-Point Sculptures (C. Pickover)
r2: -- Quadratic by Julian C. Sprott
r3: -- Modified Quadratic by Julian C. Sprott
r4: -- Chaotic Iterated Map
r5: -- Chaotic Iterated Map #2
r6: -- Flame #1(variation of Scott Drave's Flame method)

4.9 Cubic Values Window

Cubic Values Window

To support the cubic Mandelbrot formulas (g0 thru g9), variables have been added to adjust z-origin and magnify the z-plane while calculating pixel depth. Normally you can keep origin set at (0,0,0) and z-mag at 1.0. But these can be useful to tweak in small increments when drawing Mandelbrot quaternions. Then a small change in z-origin can rotate details on a close-up slightly, so you can adjust the view accordingly. Each shift in z-origin will require re-zooming to get back to the area of interest. The z-mag variable makes the z-plane non-symmetrical; pushing up some details while other details recede. So it too is useful to change viewpoint. The affect on Julia quaternions is less noticeable for z-origin shifts. You can usually re-zoom to produce the same Julia image.
Cubic Mandelbrots quaternions use the S and Si variables to set the initial value and range of the 3rd dimension. Starting from an initial value of 'S', Si is normally set to 2*abs (S), when the Type is Mandelbrot. Si can also be set to center the 3rd dimension on something besides 0.0. Si has no affect in 2-D hypercomplex mode. (In 2-D mode, the S variable acts as one of the two fixed dimensions, along with the 4th Center variable.)

The 4th Center points to the center of the fourth dimension (with quaternions/hypernions the 4th Dimension variable is used.)

In addition, the Arg value has the following affect on cubic Mandelbrots:
0 -- compute M+, using Z for z space
1 -- compute M+, using greater of S or Z for z space
2 -- compute M-, using greater of S or Z for z space
3 -- compute M+ and M-, use lesser of M+/M- for pixel depth
4 -- compute M+ and M-, use greater of two for pixel depth
5 -- compute M+ and M-, use difference of two for pixel depth
6 -- compute M+ and M-, use sum of two for pixel depth
7 -- compute M+ and M-, use vector magnitude of two for pixel depth
8 -- compute M+ and M-, use intersection of two (CCL)
9 -- compute M+ and M-, use M+ or difference of two if M+>M-

Args 3-9 affect only cubic Mandelbrots.

For args 0-9, a second argument determines the plane that is used for the fourth dimension:
b -- b-imag (default)
B -- b-real
a -- a-imag
A -- a-real

A third argument also works for args 0-9:
j -- alternate cubic Mandelbrot mapping

Therefore an arg value of '3B' uses the union of M+ and M- and b-real as the fourth dimension.

4.10 Octonion Values Window

Octonion Values Window

Octonions (built-in functions H0-H9) have a form of xr+xi+xj+xk+xE+xI+xJ+xK. Additional options are entered via the Arg box in the Edit/Formula window. Use the Randomize button to set a random value for octonion planes E-K and complex constants cj-cK.

4.11 Edit Quat-trap

Quat-Trap (Edit menu)

With this window you can edit and display the two separate functions that make up the quat-trap fractal. You can change the orbit-trap width, complex c or type, then display the orbit-trap with the Draw Trap button. You can change the quaternion's function or complex c, then display the quaternion with the Draw Quat button. Use Apply or Okay to display the composite fractal. See Orbit-Trap options for further details on quat-trap pictures.

4.12 Size

Size (Edit menu)

This allows you to set the drawing area for a picture, independent of the Windows screen size. It also shows which size is currently in use. The aspect for the drawing is based on the ratio of X (horizontal width) to Y (vertical height.) The custom setting allows for any size/aspect that system memory will permit. Videos are limited to the standard 4/3 vga aspect or 1/1. Midi output is limited to images with the standard 4/3 aspect. The minimum size for an image is 40X30. Note: if the image is less than 100 width, the aspect must be 4/3 for solid-guessing to work properly.

4.13 Ray-Tracing Window

Ray-Tracing Window

The Light Point variables (lightx thru lightz) determine the direction of the light source used in the ray-tracing algorithm. The ViewPoint represents the angle at which the object is ray-traced, which can affect Phong highlights greatly. This has no effect on the camera view.

The Lighting variables shininess, highlight, gamma and ambient are used to adjust ambient light and highlights. The ranges for these variables appear beside their label. Decreasing the shininess value increases light reflected by the object and the apparent sheen on the object's surface. The ambient value controls the amount of ambient light that illuminates the object. The highlight value increases or decreases the specular (Phong) highlighting, while the gamma value increases or decreases the intensity of the light source's illumination. Once a plot is started, the lighting variables and light point can be changed without redrawing the object.

Click the Apply button to redisplay a plot after changing the lighting variables or light point. Click

the Okay button to close the Ray-Tracing Window, applying new settings, if the variables were modified. Click on Cancel to revert to the state that existed when the ray-tracing window was opened. Click on Defaults to set the lighting and viewpoint variables to the built-in defaults for these variables.

4.14 Edit Palette

Edit Palette

Use the palette editor to modify the palette(s) in use.

There are copy and spread options to smooth or customize the existing palettes in Fractal Zplot. You can then save all the palettes in a .pl file, or by saving the entire function and bitmap (v1.08+ saves all the palettes in the data file.)

Colors are shown in 8 groups of 32 colors. This makes it easy to create divide-by-8, divide-by-4 and divide-by-2 palettes with 256 colors (Use a start color of 0 and an end color of 255.) With Fractal Zplot, a palette is actually 65280 colors, with each succeeding color (except the last) followed by 255 colors that are evenly spread from one color to the next.

Use the RGB-slider controls to edit any color in the palette. Select Copy to copy any color to another spot in the palette. Select Spread to define a smooth spread of colors from the current spot to another spot in the palette. Copy and Spread take effect immediately when you select another spot with the mouse button. You can cancel the operation with the Cancel button. In Fractal Zplot, colors do not cycle smoothly when you adjust the RGB/HSV sliders. This would be too slow with true color. The Map button is used to map color changes to an image after you are done adjusting the sliders. In the HSV mode, color spreads are based on HSV values instead of RGB values, which in some cases results in brighter color spreads.

Right-click on any point on the main window and the palette color for that pixel will be displayed in the palette editor. You can use any of the color-cycling keys (after clicking on the main window) to see the effects of the cycling in the palette editor window. Note: color cycling and color-selection-from-pixel only works when the image has been drawn in the current session. If you load a pre-existing image file, you must redraw it to cycle colors, etc. Anti-aliasing, 3D height fields, undoing an action and hsv filtering also disable color cycling.

Use Reset to reset the colors of the palette in use, to where it was before it was cycled or modified. Note: if you change palettes with one of the function keys, any modifications to a previous palette are unaffected by the Reset button.

Use Reverse to reverse the order of the colors in the palette. This affects only those colors in the start-color to end-color range. Useful for reversing divide-by-eight palettes, etc., for orbit-trap pictures that require a reversed palette.

Use Neg to create a palette that is the complement of the current palette.

Use Smooth to create a random palette with smooth color spreads. Use Scramble to create a palette with random color indexes.

Use SRG to switch the red and green components of all palette colors.
Use SRB to switch the red and blue components of all palette colors. SRB and SRG are disabled in HSV mode. You can use these buttons to form eight different palettes by repeatedly switching red, green and blue components.

Use the Random palette button to randomize the current palette. The Randomize variables, rmin, rmax, bmin, bmax, gmin, and gmax act as limits that are applied after the palette after initial randomizing, to make the palette conform to the desired spectrum of colors.

Note: unless you click on Reset before exiting the editor, changes are permanent to the palette edited, no matter which way you close the editor (Okay button or close box.)

4.14.1 Reverse button

Reverse button
Use Reverse to reverse the order of the colors in the palette. This affects only those colors in the start-color to end-color range. Useful for reversing divide-by-eight palettes, etc., for orbit-trap pictures that require a reversed palette.

4.14.2 Neg Button

Neg button
Use Neg to create a palette that is the complement of the current palette.

4.14.3 Map Button

Map button
In Fractal Zplot, colors do not cycle smoothly when you adjust the RGB/HSV sliders. This would be too slow with true color. The Map button is used to map color changes to an image after you are done adjusting the sliders.

4.14.4 H/R Button

H/R button
Change from HSV to RGB mode or back. In the HSV mode, color spreads are based on HSV values instead of RGB values, which in some cases results in brighter color spreads.

4.14.5 Spread Button

Spread button
Select Spread to define a smooth spread of colors from the current spot to another spot in the palette.

4.14.6 Copy Button

Copy button
Select Copy to copy any color to another spot in the palette.

4.14.7 SRG Button

SRG button
Use SRG to switch the red and green components of all palette colors. RGB mode only.

4.14.8 SRB Button

SRB button
Use SRG to switch the red and blue components of all palette colors. RGB mode only.

4.14.9 Okay Button

Okay button
Click on Okay to exit the palette editor, applying unmapped color changes to picture (if color-cycling is enabled.)

4.14.10 Reset Button

Reset button
Use Reset to reset the colors of the palette in use, to where it was before it was cycled or modified. Note: if you change palettes with one of the function keys, any modifications to a previous palette are unaffected by the Reset button.

4.14.11 Cancel Button

Cancel button
You can cancel a copy or spread operation with the Cancel button.

4.14.12 Red Slider

Red slider
Use the RGB/HSV-slider controls to edit any color in the palette.

4.14.12.1 Red edit box

Red edit box
Shows red/hue value of selected color index.

4.14.13 Green Slider

Green slider
Use the RGB/HSV-slider controls to edit any color in the palette.

4.14.13.1 Green edit box

Green edit box
Shows green/saturation value of selected color index.

4.14.14 Blue Slider

Blue slider
Use the RGB/HSV-slider controls to edit any color in the palette.

4.14.14.1 Blue edit box

Blue edit box
Shows blue/value magnitude of selected color index.

4.14.15 Smooth Button

Smooth palette button

Use to create a random palette with smooth color spreads.

4.14.16 Scramble

Scramble

Use to create a palette with random color indexes.

4.15 RGB Thresholds

RGB Thresholds (Edit menu)

When Fractal Zplot plots pixels using a simulated palette, the colors are normally scaled (interpolated) to fit between one of 236 color indexes. These 236 colors can be edited by the palette editor. Sometimes you don't want an intermediate color to be plotted, if adjacent color indexes differ a lot.

There are two tests that can be made to determine the degree of color difference between adjacent color indexes. The first is RGB, where the difference in individual components is tested. The thresholds are set so that color indexes that differ in values exceeding the individual RGB thresholds will bypass the interpolation process. The values can be set from 0 to 255, with 255 the default. At 255, all pixels are interpolated except the first and last colors. At 0, no pixels are interpolated.

Test 2 measures the difference in the sum of the RGB values of adjacent colors. If the sum is greater than a preset limit (0-765), no interpolation is done.

These two tests can be combined with the Both option.

4.16 Edit Text command

Text (Edit menu)

Allows you to edit text and font and apply it to a drawing. Select the font button to set the font style, size and color. In the text window click on Okay to add a line of text to the current image. (You can add multiple lines of text too, up to 80 characters.) The cursor will change to a crosshair. Position the cursor where you want the text to start and left-click the mouse. Note: font and title text are saved in the file "prefs.txt" in Fractal Zplot's startup directory. Title text can also be edited (as a single line only) in the Edit/Formula window.

4.17 Preferences

Preferences (Edit menu)

Each time you use the Reset command, Fractal Zplot restores data variables to built-in defaults. The Set Defaults button allows you to change some of the data variable defaults to whatever the current settings are. Some of the customizable variables include step, fine, formula, viewpoint, lighting, rotational angles, Phong and x/y space. (Iterations, Type, Constants, and a few other variables are excluded to maintain compatibility with the 'G' command.) The new Reset defaults are saved in the file "prefs.txt" when you close the program (if the Defaults check box is selected.) The check boxes in the group "Save on Program Close" allow you to change the default startup mode of a few Auto options, such as Auto Redraw, and the Random Setup variables. By keeping the boxes selected, Fractal Zplot saves the last changes you make to these options. If you want to go back to the initial settings (the way Fractal Zplot was packaged originally) you can click on the Reset Defaults button. This restores the data, Auto variables and random setup defaults.

Use the Reset Dialog Positions button to reset all non-modal dialog positions to x/y positions that will fit within a 640X480 screen. Sometimes when you switch screen resolutions to a lower resolution there might be dialogs that are off the screen and thus are inaccessible when reopened. This can happen too if the Registry key for the dialog position becomes corrupt. (The program keeps track of all non-modal dialogs' last positions in the Software portion of the Registry.) Close all open dialog windows you want to reset before using this command, or open a New draw window before using the command.

Use the Default Directories tab to change the default directories for saved and loaded items in Fractal Zplot. Click on the "…" button next to each default directory box, and use the folder requester to pick a different directory, or create a new directory with the Make New Folder button. The default directories are saved at program close in the Registry and reloaded when you next open Fractal Zplot.

5 Image menu

Image menu commands

The Image menu offers the following commands:

Draw	Draw the picture.
Draw Composite	Draw composite from figures 1-4.
Plot To File	Plot large bitmap images directly to png file.
Plot Files In Directory	Disk render .zp6 files in working directory.
Auto Redraw	Redraw image on command.
Auto Clear	Clear drawing area before new plot.
Auto Sound Alert	Enable or turn off sound alerts.
Auto Remote	Open remote automatically at startup.
Auto Time	Show time used to plot image.
Merge Sum	Merge current pixel color with previous color summing colors.
Merge And	Merge current pixel color with previous color anding colors.
Merge Or	Merge current pixel color with previous color oring colors.
Merge High	Merge current pixel color with previous color by choosing highest rgb.
Merge Low	Merge current pixel color with previous color by choosing lowest rgb.
Merge Back	Merge current pixel color with previous color by excluding background color.
Merge Diff	Merge current pixel color with previous color by using difference of colors.

Abort	Abort drawing.
Continue	Continue drawing.
Zoom	Zoom into rectangle.
New View on Zoom	New view on zoom.
Clone	Clone current view.
Scan	Scan Mandelbrot border for quaternion Julia set.
Dive	Peel off outer layer of quaternion.
Full Screen	View image full-screen.
Pilot	Use Pilot to rotate and alter key cubic variables.
Reset->	Reset coordinates, current figure or all figures
Figure 1	Switch to figure one.
Figure 2	Switch to figure two.
Figure 3	Switch to figure three.
Figure 4	Switch to figure four.
Composite	Select figures to merge.
Count Colors	Count colors used in picture.

5.1 Image Draw command

Draw command (Image menu)

Use this command to draw or redraw the image for the current fractal variables. Clicking inside the draw window with the left-mouse button stops all plotting. Use the Continue command to restart plotting from the current column. When the lsystem option is selected, a custom window is opened to enter lsystem draw options.

5.1.1 Image Draw Lsystem command

Draw (Lsystem) command (Image menu)

The lsystem draw window has edit boxes to set projection angles, roll pitch and yaw. The angles have a range of -360 to 360 degrees. The initial turtle heading is 0,1,0 -- which moves the turtle up in the direction of the y-axis when the projection angles are all 0. The view may be improved when some of the projection angles are non-zero, or to rotate an object done with LParser to an appropriate viewing angle.

The skewing angles affect the line styles when the line width is greater than 1. These can warp the lines(tubular or poly) to a non-uniform width.

The light point is similar to that used when Fractal Zplot draws 3D quaternions.

The Lighting variables shininess, highlight, gamma and ambient are used to adjust ambient light and Phong highlights. The ranges for these variables appear beside their label. Decreasing the shininess value increases light reflected by the quaternion and the apparent sheen on the quaternion's surface. The ambient value controls the amount of ambient light that illuminates the quaternion. The highlight value increases or decreases the specular (Phong) highlighting, while the gamma correction increases or decreases the intensity of the light source's illumination.

Note: once a plot has been drawn, the lighting variables and light point can be changed without redrawing the lsystem. Click the light bulb button to redisplay a picture after changing the lighting variables or light point.

You enter the level of recursion into the level box. The maximum level of recursion is limited to 50.

Use the scale slider to set the size of the drawing, .05 to 1.0. Each lsystem is pre-scaled prior to drawing, to approximate the size of the draw window. To speed up the pre-scaling, the width of the line drawn isn't used to map image limits. So it's possible an object may extend off the window's edges, when a line width is greater than 1 and the scale is 1. In this case, use a scale less than 1 to "shrink fit" the drawing.

Click on Draw to recompute and draw the current axiom.

Select Wire Frame to draw a wire frame model of the current axiom/rules. This is useful to speed up drawing while prototyping an lsystem.

Enter additional Steps (default 1), to increase the resolution of solid rendering. Each additional step increases rendering time proportionally, but can increase the smoothness of certain area fills, as in polygon rendering of large objects.

Enter a non-zero mutation value to mutate the current lsystem. This follows the same mutation rules as LParser. Small values (less than 5) can produce extremely lengthy mutations. Use Fixed Mutation to limit the size of the mutated lsystem to the original size. In this case the no rules are appended or inserted. Only angle and movement commands are switched.

Select the extended set box to enable the extended rule set. Most input files made for LParser will work with the extended set enabled, but the extended rules may not work if the command is overridden by a production rule.

5.2 Image Draw Composite command

Draw Composite command (Image menu)

Use this command to draw or redraw an image defined in the Composite command as a merging of figures 1-4. Clicking inside the draw window with the left-mouse button stops all plotting. Continue is disabled for this command.

5.3 Plot to file

Plot to File

Allows you to plot a large bitmap directly to a .png file without the added system requirements of keeping the whole bitmap in memory. The Target group sets the bitmap resolution (width 800 to 14400. Drawing aspect is that of the current image.) Click on Okay to set the target file name and start a new plot to file. Note: the 3200X2400 bitmap size is suitable for 8 1/2X11 printouts at 320-720 dpi. The larger bitmap sizes are suitable for poster-size printouts. This option is not available with the merging options, or with anti-aliasing. Also, solid-guessing is disabled when using this option.

5.4 Plot Files in Directory

Plot Files in Directory

Allows you to plot a set of large bitmaps directly to a .png files without the added system requirements of keeping any of the images in memory. The Target group sets the bitmap resolution (width 800 to 14400. Drawing aspect is that of the current image.) All data files (.zp6) in the working directory are enlarged to this resolution. Click on Okay to start. Note: the 3200X2400 bitmap size is suitable for 8 1/2X11 printouts at 320-720 dpi. The larger bitmap sizes are suitable for poster-size printouts. Merging, anti-aliasing and solid-guessing are disabled when using this option.

5.5 Image Redraw command

Auto Redraw command (Image menu)

With this command disabled (on by default), redraw does not occur except when the Draw command is executed, or Continue. Most of the time you want to see the results of changing a parameter or mapping option, so redraw occurs automatically with parameter or mapping changes.

Sometimes you want to change more than one parameter before redrawing the image, or you may want to *continue* a mid-point displacement plot (after loading the file, this is the only way to recreate the plot.) So you need to turn this option off then.

5.6 Image Auto Clear command

Auto Clear command (Image menu)

With this command enabled (on by default), the drawing area is cleared before starting a new plot. You can turn off this option when you want to see the effect of minor changes to parameters, as they affect the plot pixel by pixel, or when setting up a multiple-layered fractal, as in a 3D landscape. You can use the shift-c command (hot keys) to clear the drawing area at any time.

5.7 Image Auto Alert command

Auto Sound Alert command (Image menu)

With this command enabled (on by default), the user is notified by a sound clip when a drawing is completed or user-canceled. By disabling this command the completion exclamation is suppressed and also any alert that contains a message box. Note: some sound clips are automatically generated by Windows, or there is no text alert for a given error condition. In these cases the sound alert is unaffected by the Auto Alert command.

5.8 Image Auto Remote command

Auto Remote command (Image menu)

With this command enabled (on by default), the remote is opened immediately at program startup. Handy if you find the remote useful and don't want to click on the toolbar button each time the program starts up.

5.9 Image Auto Time command

Auto Time command (Image menu)

With this command enabled (on by default), the time that an image takes to plot is displayed when the plot is complete. Fractal Zplot saves the condition of this option at session's end, so if you disable it and close the program, the option will be disabled when you restart Fractal Zplot.

5.10 Image Merge Sum command

Merge Sum command (Image menu)

With this command enabled (off by default), current pixel color is not overwritten when a new image is drawn. Instead the colors are merged using a summing algorithm. The auto-clear option must be disabled and solid-guessing off to choose this option. Useful to merge two or more separate fractal images/types with the initial image(s) "bleeding" through.

5.11 Image Merge And command

Merge And command (Image menu)

With this command enabled (off by default), current pixel color is not overwritten when a new image is drawn. Instead the colors are merged using an anding algorithm. The auto-clear option must be disabled and solid-guessing off to choose this option. Useful to merge two or more separate fractal images/types with the initial image(s) "bleeding" through.

5.12 Image Merge Or command

Merge Or command (Image menu)

With this command enabled (off by default), current pixel color is not overwritten when a new image is drawn. Instead the colors are merged using an oring algorithm. The auto-clear option must be disabled and solid-guessing off to choose this option. Useful to merge two or more separate fractal images/types with the initial image(s) "bleeding" through.

5.13 Image Merge High command

Merge High command (Image menu)

With this command enabled (off by default), current pixel color is not overwritten when a new image is drawn. Instead the colors are merged using the highest rgb values of both images. The auto-clear option must be disabled and solid-guessing off to choose this option. Useful to merge two or more separate fractal images/types with the initial image(s) "bleeding" through.

5.14 Image Merge Low command

Merge Low command (Image menu)

With this command enabled (off by default), current pixel color is not overwritten when a new image is drawn. Instead the colors are merged using the lowest rgb values of both images. The auto-clear option must be disabled and solid-guessing off to choose this option. Useful to merge two or more separate fractal images/types with the initial image(s) "bleeding" through.

5.15 Image Merge Back command

Merge Back command (Image menu)

With this command enabled (off by default), current pixel color is not overwritten when a new image is drawn. Instead the colors are merged using the rgb components of the new image if the new color index is not zero; else the old rgb values are retained. The auto-clear option must be disabled and solid-guessing off to choose this option. Useful to merge two or more separate fractal images/types with the initial image(s) "bleeding" through.

5.16 Image Merge Diff command

Merge Diff command (Image menu)

With this command enabled (off by default), current pixel color is not overwritten when a new image is drawn. Instead the colors are merged using the difference of the rgb values of both images. The auto-clear option must be disabled and solid-guessing off to choose this option. Useful to merge two or more separate fractal images/types with the initial image(s) "bleeding" through.

5.17 Image Abort command

Abort command (Image menu)

Use this command to stop drawing. Clicking inside a window's drawing area or close box (or the program close box) will also stop the drawing. Note: once a plot has started Fractal Zplot continues to draw the image for that window regardless of which drawing window has the input focus, until done or aborted. You can open and close other drawing windows without affecting the current drawing, but only one drawing is active at any time.

5.18 Continue Draw

Continue Draw (Image menu)

Continues a plot that was aborted early. The plot is restarted at the beginning of the last row drawn. Continue is disabled when an Image/Merge option is selected.

5.19 Zoom

Zoom (Image menu)

Turns on zoom mode, so that detail of the current plot may be magnified. Alternatively, just click inside any drawing window, move the mouse, and the zoom box will appear. Using the mouse, move the zoom box over the portion of the plot you wish to magnify. Hold the left mouse button to shrink the box or the right button to enlarge it. Use the left and right arrow keys to rotate the box counter-clockwise or clockwise. Use the up and down arrow keys to squash or expand the box, changing the aspect of the image. You start a zoom by pressing the space bar. You abort a zoom by clicking outside the main window or in the title bar, or by pressing the escape key. The program will begin a new plot at the new coordinates. You may zoom in by defining a box inside the current drawing area. You zoom out by drawing a box outside the current drawing area. The outer zoom limits are between -1000 and 1000. The precision is that of double precision (64 bits)
Note: Zooming in a three-dimensional plot is not supported, except for quaternion plots, nor is zooming on a random (midpoint displacement) fractal. Rotating a quaternion while zooming is possible, but inexact. Depending on the rotational angles set in the quaternion window, you may end up rotating on the z-axis, which may rapidly result in overshooting the area of zoom. So avoid large rotations combined with high levels of zoom. It is better to rotate at low zoom levels, and then zoom without rotating, for close-ups. If you change screen resolutions, you must redraw the bitmap image for a function before you can accurately zoom on it.

Zooming with orbital-type fractals turns on magnify mode, so that details of the current screen may be examined. The selected area is magnified to fill the current sector or screen. The x and y ranges do not change though, so the magnified area may not show up all at once, since other areas of the function may have to be recalculated first. The magnify mode can only be used on one area of a screen at a time. Magnifying is then disabled until a new map is generated though the New button in the Edit/Orbit Function window. You can increase the resolution of a magnified map, though, via the iterations slider. Use Okay to redraw the map at increased resolution. Rotating is not supported with orbital fractals.

5.20 Image New View on Zoom command

New view on zoom (Image menu)

With this option enabled, a new window is opened with each zoom, instead of the zoom box area replacing the original image. Ignored in avi mode.

5.21 Image Clone

Clone (Image menu)

A new draw window is opened that contains the same fractal data as the window it was opened from. This is useful for comparing minor changes in texturing options, etc. Similar to using the copy/ paste data commands except that all figures are copied to the new view.

5.22 Scan

Scan (Image menu)

Equivalent to the Shift+G hot key. Enabled when the Type is Mandelbrot. A quaternion Julia set is generated in sector 2, using an iteration count of 10 and other parameters are changed temporarily to suit quaternion plots. (Z is set to 2.0, the rotational variables are reset and the light source is set to the default, if random lighting is enabled.) Quaternion math is used where possible if the Type is set Quaternion, else hypercomplex math is used for the Hypernion type, etc. except that quaternion math is used to generate cubic Julia sets. Once you find an interesting quaternion set using "G", like the J command another window is opened that sets the fractal parameters to those in the exploratory qjulia window. The parameters in the exploratory window revert to their original Mandelbrot settings.

5.23 Dive

Dive (Image menu)

Select Dive to go beneath the surface of a quaternion. Some quaternions have a smooth border that doesn't show the turbulence below the surface. Using the Dive option strips off the border layer to reveal what's underneath.

5.24 Full Screen

Full Screen (Image menu)
Displays the entire plot, expanding or shrinking the image to fit in a maximized window without title bar, scroll bars or menu bar. At all other times, part of the picture is hidden by the inclusion of the title bar, toolbar, scroll bars and menu bar. To exit full-screen mode, press any key or click the left-mouse button.

5.25 Pilot

Pilot (Image menu)
Opens the Pilot window to adjust key parameters, rotate, zoom and redraw the figure interactively. The current image is reduced to one quarter normal for faster redraw. Each click on a Pilot button increments or decrements a parameter. The Speed slider controls the rate at which the buttons operate (default is 10.)

Press the space bar or Click on Ok to open a new window and draw the altered image full-size. Press Esc or click on Cancel to exit this mode without opening a new window. Note: when using this option while an AVI stream is open, a new window isn't opened, but the altered figure is drawn in the current draw window, the changed parameters replacing the previous ones.

5.26 Reset

Reset
Reset the current figure or all figures to an empty Mandelbrot. All functions in the New Formula data are blanked. All options on the Flags menu are reset to their default settings. The Print Function Data ignores any reset figures.

The Ranges Only command resets only the real Z and imaginary Z ranges in the Parameters window (to +/-2.0 and +/-1.5.) No other menus or variables are affected. This is useful in conjunction with the "P" command to generate and view Julia sets. After setting the complex-C variable via shift-P (Caps Lock off), you need to reset the Z ranges to see the entire Julia set after zooming into a Mandelbrot set. The Reset All option resets all figures.

5.27 Figure #1

Figure #1
Switch to Function #1. Current settings are saved under the previous image.

5.28 Figure #2

Figure #2
Switch to Function #2. Current settings are saved under the previous image.

5.29 Figure #3

Figure #3
Switch to Function #3. Current settings are saved under the previous image.

5.30 Figure #4

Figure #4
Switch to Function #4. Current settings are saved under the previous image.

5.31 Image Composite command

Composite command (Image menu)

Opens the Composite Figure window, where you can define a set of figures to merge into one image. All the merging options in the Merge Color menu are supported, plus "ALL" which is usually used for the first figure to be drawn. The "ALL" option transfers all rgb information for a figure to the drawing area, without checking the rgb state of the pixel. You can define up to four figures (layers), as part of the composite, but each figure should contain an image (if used in the composite.)

5.32 Count Colors

Count Colors

The colors are counted in the picture. A box is displayed showing how many unique colors are used out of how many possible colors in the palette. Since red, green and blue components of a an individual color can range only from 0-255, there is a real limit on how many colors can be used in a palette and still maintain smooth color spreads (even with true color.) For pictures using a split palette, like orbit-trap pictures, it is very important to keep the split-palette sections continuous. For most purposes this limits a picture to about 5000 colors with a divide-by-eight palette. Put into practice, the number of colors used out of the palette will usually be less than half, depending on the color-scaling method and other coloring factors.

The method to count the colors varies depending on whether color-cycling is enabled or not. With color-cycling enabled, the colors used in the picture are matched to the palette before counting. This is much faster than when color-cycling is disabled (as when anti-aliasing or undo is used), where all the colors in the picture have to be referenced back to the picture's pixels instead. Note: if color-cycling is not enabled, as when a picture drawn with Fractal Zplot is loaded, the latter method of counting pixel colors is used whether or not the Use Palette flag is set. The time it takes to count colors varies with the number of unique colors in the picture, so this process can be very slow at times... To see how the count is progressing, you can press a key or the right mouse button, and a dialog will show how many different colors have been counted. To stop a count, click the left mouse button.

6 Type menu

Type menu commands

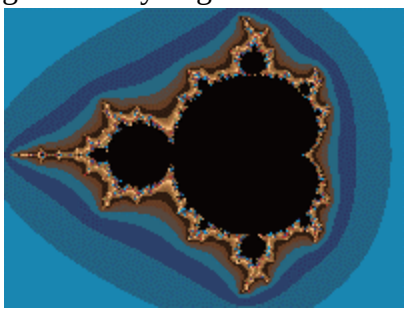
The Type menu offers the following commands:

Mandelbrot	Mandelbrot set.
MandelbrotP	Mandelbrot set (orbit starts at pixel.)
Julia	Julia set.
Julia Tower	Julia tower of varying C.
3D Height Field	Set fractal type to 3D height field.
3D Background	Use Julia or Mandelbrot as background for 3D type.
3D Landscape	Set fractal type to 3-D landscape (random displacement)
Quaternion	Set fractal type to quaternion.
Hypernion	Set fractal type to hypercomplex quaternion.
Cubic Mandelbrot	Set fractal type to cubic Mandelbrot.
Complexified Quaternion	Set fractal type to complexified quaternion.
2D Hypercomplex Map	Set fractal type to two-dimensional hypercomplex mapping.
2D Quaternion Map	Set fractal type to two-dimensional quaternion mapping.
2D Complexified Map	Set fractal type to two-dimensional complexified quaternion mapping.
2D Cubic Map	Set fractal type to two-dimensional cubic Mandelbrot mapping.
Random Displacement	Set fractal type to 2-D random displacement
Orbits	Set fractal type to orbital fractal.
Lsystem	Set fractal type to lsystem.
Zero Init	Sets initial 'z' to zero for Mandelbrot sets.

6.1 Mandelbrot

Mandelbrot

Mandelbrots base their mapping on varying inputs of complex C, which corresponds to the min/max values set in the Parameters window. With Mandelbrot0, Cr and Ci represent the initial value of Z before the first iteration. This is normally zero, but can be changed to produce non-symmetrical Mandelbrots, or Mandelbrots based on formulas whose initial value of Z must be non-zero to generate anything.



Mandelbrot set

6.2 MandelbrotP

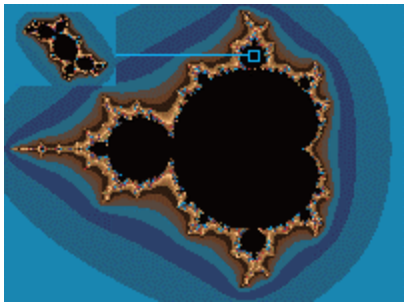
MandelbrotP

Mandelbrots base their mapping on varying inputs of complex C, which corresponds to the min/max values set in the Parameters window. With MandelbrotP, the initial value of Z is set to the value of the pixel being iterated. This produces interesting effects with some Mandelbrot formulas that normally start their orbits at zero.

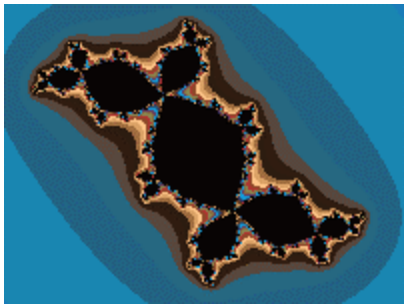
6.3 Julia

Julia

Julia sets normally have a fixed complex C, with varying inputs of Z, which corresponds to the min/max values set in the Parameters window. This option, without the Bound flag set, generates the so-called 'filled-in' Julia set, which includes non-escaping points as well as the Julia set.



Julia from Mandelbrot

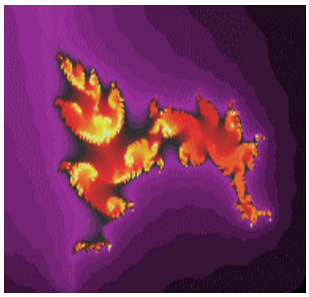


Julia set

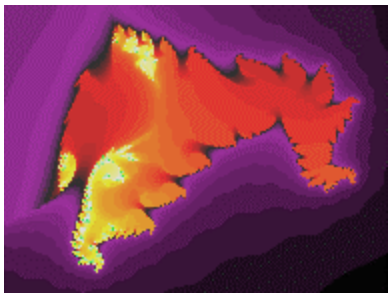
6.4 Julia Tower

Julia Tower

Julia sets normally have a fixed complex C, with varying inputs of Z, which corresponds to the min/max values set in the Parameters window. The Tower option, without the Bound flag set, generates a "stacked" Julia set. This Julia set has its constant incremented by a value of Cx, where Cx represents four variables (cx, cy, cz and cw) for incrementing each part of the complex C (cr, ci, cj, ck), specified in the Parameters window. The constant is scaled from its initial value c to the value of c+Cx. Each row of the set has a different slice of the constant. Some functions that rely heavily on both cr and ci for their shape show a marked "meltdown" as a tower. Note: zooming with this flag set changes both complex C and Cx to maintain the correct scaling factors.



Original Julia set



Julia Tower

6.5 Type 3D Height Field command

3D Height Field (Type menu)

Use this command to set the fractal type to a 3D height field. Variables that affect this fractal type are defined in the Edit/Parameters window.

6.6 Type 3D background command

3D Background (Type menu)

Use this command to use the current 2D parameters for plotting a backdrop to a 3D height field. The 3D height field must have been previously drawn and a 3D image map must reside in memory. (The image map is usually stored with the 3D height field, if the height field is drawn as figure one and the image saved with figure one as the current image. You don't want to draw another figure as figure one, as that would clear the 3D image map.) The background will be mapped into the drawing area that has not been drawn into by the height field. You can erase a background by using the shift-B command (see hot keys.)

6.7 3D Landscape

3D Landscape

Uses the midpoint displacement routine from **The Science of Fractal Images** to create random hills and valleys. Can be combined with a 3-D Background and quaternion "moon" (separate figures) to complete the picture.

6.8 Type Quaternion command

Quaternion (Type menu)

Use this command to set the fractal type to a 3D quaternion. Variables that affect this fractal type are defined in the Edit/Quaternion window.

6.9 Type Hypernion command

Hypernion (Type menu)

Use this command to set the fractal type to a hypercomplex 3D quaternion. Variables that affect this fractal type are defined in the Edit/Quaternion window. A hypernion uses hypercomplex math to shape the 3D object, which usually results in a squared-off shape, rather than the rounded shape of the typical quaternion.

6.10 Type Cubic command

Cubic (Type menu)

Use this command to set the fractal type to a cubic Mandelbrot. Variables that affect this fractal type are defined in the Edit/Quaternion window. Built-in formulas that support this type are G0-G9. Fun#1 must be set to one of these formulas to enable this type.

Cubic Mandelbrots quaternions use the S and Si variables to set the initial value and range of the 3rd dimension. Starting from an initial value of 'S', Si is normally set to 2*abs(S), when the Type is Mandelbrot. Si can also be set to center the 3rd dimension on something besides 0.0. Si has no affect in 2D hypercomplex mode. (In 2D mode, the S variable acts as one of the two fixed dimensions, along with the 4th Center variable.)

The 4th Center points to the center of the fourth dimension (with quaternions/hypernions the 4th Dimension variable is used.)

In addition, the Arg value (in the Formula window) has the following affect on cubic Mandelbrots:
0 -- compute M+, using Z for z space
1 -- compute M+, using greater of S or Z for z space
2 -- compute M-, using greater of S or Z for z space
3 -- compute M+ and M-, use lesser of M+/M- for pixel depth
4 -- compute M+ and M-, use greater of two for pixel depth
5 -- compute M+ and M-, use difference of two for pixel depth
6 -- compute M+ and M-, use sum of two for pixel depth
7 -- compute M+ and M-, use vector magnitude of two for pixel depth
8 -- compute M+ and M-, use intersection of two (CCL)
9 -- compute M+ and M-, use M+ or difference of two if M+>M-
Args 3-9 affect only cubic Mandelbrots.

For args 0-9, a second argument determines the plane that is used for the fourth dimension:
b -- b-imag (default)
B -- b-real
a -- a-imag
A -- a-real

A third argument also works for args 0-9:
j -- alternate cubic Mandelbrot mapping

Therefore an arg value of '3B' uses the union of M+ and M- and b-real as the fourth dimension.

6.11 Type Complexified Quaternion command

Complexified Quaternion (Type menu)

Use this command to set the fractal type to a 3D complexified quaternion. Variables that affect this fractal type are defined in the Edit/Quaternion window. A compquat uses another variation of quad math to shape the 3D object, which usually results in a more chaotic shape than the rounded lines of the typical quaternion. Not all formulas support the Complexified Quaternion type. In that case, the formula will default to hypercomplex algebra when this type is selected.

6.12 Type 2D Quaternion Map command

2D Quaternion Map (Type menu)

This command sets the fractal type to a 2D quaternion mapping.

6.13 Type 2D Hypercomplex Map command

2D Hypercomplex Map (Type menu)

This command sets the fractal type to a 2D hypercomplex mapping.

6.14 Type 2D Cubic Map command

2D Cubic Map (Type menu)

This command sets the fractal type to a 2D cubic Mandelbrot mapping. Only enabled with the built-in cubic formulas (G0-G0.)

6.15 Type 2D Complexified Map command

2D Complexified Map (Type menu)

This command sets the fractal type to a 2D complexified quaternion mapping.

6.16 Random Displacement

Random Displacement

2D version of the midpoint displacement routine from **The Science of Fractal Images**. Used to simulate cloudy skies, etc. Can be combined with a 3-D Landscape and quaternion "moon" (separate figures) to complete the picture.

6.17 Type Orbits command

Orbits (Type menu)

Use this command to set the fractal type to an orbital-type fractal instead of the default Mandelbrot/ Julia contour-type fractals. Orbital fractals are often used as a faster method to produce Julia sets, etc, though results are normally not as connected as contour plots. Variables for the orbital fractals are edited inside the Edit/Orbit Function window. Most options for the contour plots don't apply to orbital plots. Clicking inside the draw window with the left-mouse button stops all plotting. Use the Continue command to restart plotting from the current iteration.

6.18 Type Lsystem command

Lsystem (Type menu)

Use this command to set the fractal type to the lsystem-type fractal instead of the default Mandelbrot/Julia contour-type fractals. The Edit/Lsystem Axiom and Edit/Lsystem Rules commands are used to enter the lsystem. Fractal Zplot's lsystem supports 3D drawing, and all plots use a z-buffer for hidden-line removal and ray-tracing. For an introduction to lsystems and complete list of rules recognized by the lsystem parser, see Lsystem Info. Most options for the contour plots don't apply to lsystem plots. Clicking inside the draw window with the left-mouse button stops all plotting. No Continue option is available for lsystems, though a redraw option is offered to cut down on pre-scaling time.

6.19 Type Zero Init command

Zero Init (Type menu)

Use this command to set initial z to zero, excluding the complex constant, before iterating each pixel. Used with Mandelbrot types only. Normally z is set to complex c before iterating Mandelbrot sets. (The complex c may be non-zero to "warp" the orbit.)

7 Map menu

Map menu commands

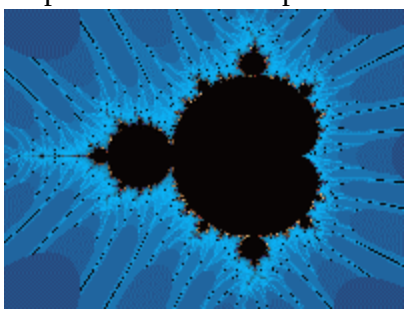
The Map menu offers the following commands:

Z-Real	Mapping based on real part of z only.
Z-Imag	Mapping based on imaginary part of z only.
Abs(Z-Real)	Mapping based on absolute value of real part of z.
Abs(Z-Imag)	Mapping based on absolute value of imaginary part of z.
Z-Real + Z-Imag	Mapping based on sum of parts of z.
Abs(Z-Real)+Abs(Z-Imag)	Mapping based on absolute value of parts of z.
>Abs(Z-Real) or Abs(Z-Imag)	Mapping based on highest absolute value of parts of z.
<Abs(Z-Real) or Abs(Z-Imag	Mapping based on lowest absolute value of parts of z.
Abs(Z)	Mapping based on absolute value of z.

7.1 Z-Real

Z-Real

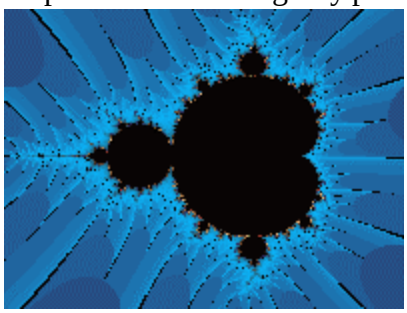
Map based on the real part of the complex number Z; used to map exponential Julia sets, etc.



7.2 Z-Imag

Z-Imag

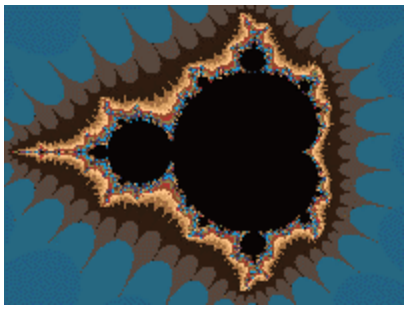
Map based on the imaginary part of the complex number Z.



7.3 Abs(Z-Real)

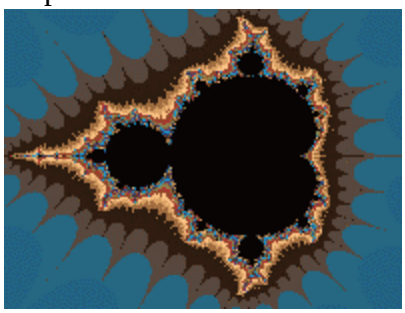
Abs(Z-Real)

Map based on the absolute value of the real part of the complex number Z; used to map exponential Julia sets, etc.



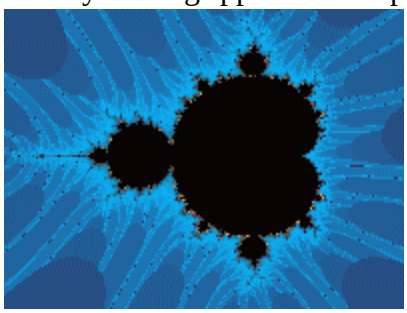
7.4 Abs(Z-Imag)

Abs(Z-Imag)
Map based on the absolute value of the imaginary part of the complex number Z.



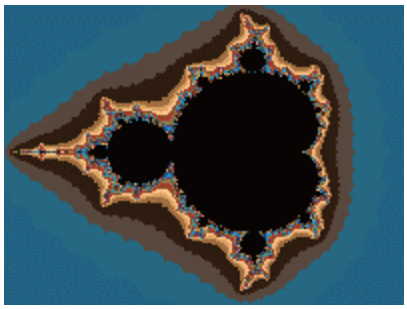
7.5 Z-Real+Z-Imag

Z-Real + Z-Imag
Map based on the sum of the real part and the imaginary part of the complex number Z. Changes the way banding appears in complex mappings.



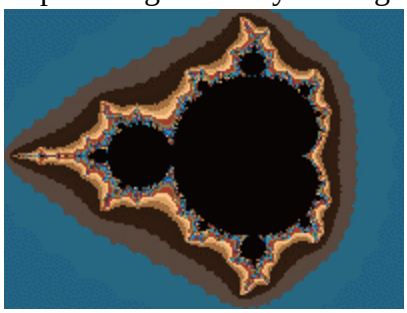
7.6 Abs(Z-Real)+Abs(Z-Imag)

Abs(Z-Real) + Abs(Z-Imag)
Map based on the absolute value of the real part plus the absolute value of the imaginary part of the complex number Z. Changes the way banding appears in complex mappings.



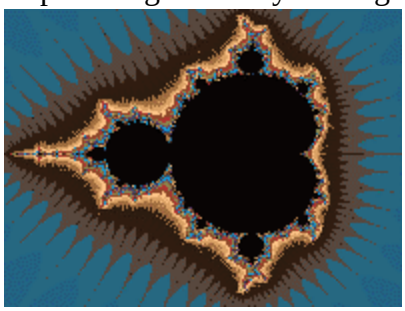
7.7 >Abs(Z-Real) or Abs(Z-Imag)

>Abs(Z-Real) or Abs(Z-Imag)
Map based on the greater of the absolute value of the real part or the imaginary part of the complex number Z. Works like a logical 'or', where either part of z must exceed zlimit to break the iteration loop. Changes the way banding appears in complex mappings.



7.8 <Abs(Z-Real) or Abs(Z-Imag)

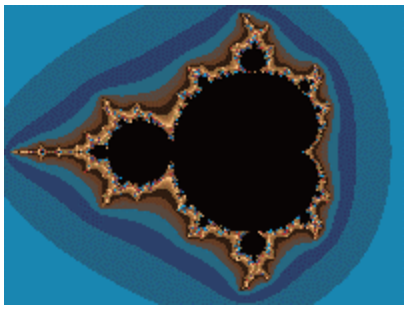
<Abs(Z-Real) or Abs(Z-Imag)
Map based on the lesser of the absolute value of the real part or the imaginary part of the complex number Z. Works like a logical 'and', where both parts of z must exceed zlimit to break the iteration loop. Changes the way banding appears in complex mappings.



7.9 Abs(Z)

Abs(Z)
Map based on the absolute value of the complex number Z (traditionally calculated by taking the square root of the sum of the squares of the real and imaginary parts of Z, but Fractal Zplot uses only the 'sum'(modulus of z) for break-point tests.) The standard method of mapping Julia and

Mandelbrot sets.



8 Break menu

Break menu commands

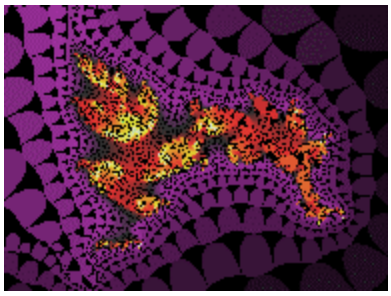
The Break menu offers the following commands:

Biomorph	Escape on any part of z.
Bioconvergence	Relates Biomorph method to convergence.
Biomorph Off	Reset biomorph flag.
Orbit Traps	Set orbit trapping method.
Newton->	Define Newton method or clear Newton flag.
Renormalization	Renormalization.
Convergence	Color converging points with level set colors.
Period Check	Periodic checking(speed option).
Default Function	Use default function type.

8.1 Biomorph

Biomorph

Biomorphs test the real Z and imaginary Z values after breaking the iteration loop. If the absolute value of either is less than the preset zlimit, the point is mapped as part of the set. This method produces biological-like structures in the complex plane. Normally the biomorph tendrils are colored in the set color (the color reserved for non-divergent or inner points.) With the Set Only flag on, the tendrils are colored according to the color-scaling option used (other external points are colored in the background color.) A window is opened each time this option is selected to set a color for the area that falls within the biomorph trap. This can be 0-235.



8.2 Bioconvergence

Bioconvergence

This option relates the Biomorph method to convergence for convergent-type fractals (Newton, Renormalization and Convergence.) A pseudo-biomorphic algorithm is applied to converging points.

8.3 Biomorph Off

Biomorph Off

Turns off the biomorph flag, including bioconvergence too. Alternatively you can enter -1 in the Biomorph window to turn off the bio-flag.

8.4 Orbit traps

Orbit Traps

This includes methods that trap the orbit of a point if it comes in range of a pre-specified area or areas.

The Epsilon-Cross method colors points only if the absolute value of Z-real or Z-imaginary is less than or equal to Epsilon (a small value.) Other points are mapped at the time they blow up (exceed the zlimit.) This produces hair-like structures that branch wildly from the complex set boundaries. For the Epsilon-Inside option, the epsilon method is applied only to points included in the set. For the Epsilon-Outside option, the epsilon method is applied only to points outside the set.

The Globe method uses a circular area around the origin to map a point's orbits. This produces sphere-like structures.

The Ring method uses an area formed by two circles around the origin to map a point's orbits. This produces ring-like structures.

The Four-Circles method (Paul Carlson) uses four circular areas to map a point's orbit. This produces sphere-like structures.

The Square method uses an area formed by two squares around the origin to map a point's orbits. This produces ring-like structures with right angles.

The Petal method (Paul Carlson) also uses four trap areas to form flower-like patterns.

The Formula option allows you to enter your own formula for an orbit trap in the Formula box in the Edit Formula window. This works for built-in formulas and fractal types except type 9 (formula) and 5(random.) An example of how to specify an orbit trap is the following formula for the ring method: a=x#*x#,b=y#*y#,x#=a+b-.25,x#=abs(x#)
where x# is the real part of z and y# is the imaginary part of z at the nth iteration. X# is then compared to the epsilon and epsilon2 values. If x# is less than epsilon and greater than or equal to epsilon2 then x# is subtracted from epsilon and the resulting value is used for coloring purposes (a level curve must be chosen as an option.) This is also the loop-breaking condition.

Epsilon2 is used to create windows into the stalks. The default value is 0.0, which produces solid stalks.

The Parametric Formula option allows you to enter your own parametric formula for an orbit trap in the Formula box in the Edit Formula window. This works for all built-in formulas and fractal types except type 9(formula) and 5(random.) An example of how to specify this type of orbit trap is the following parametric formula: z=0.3*(cos(pixel)^3+i*sin(pixel)^3).
Here, pixel is used to specify the polar angle of z(instead of its usual c or z-plane value), where pixel = atan(imag(z)/real(z)). Note that z is used for setting up the distance variable instead of x# in this case.

The Display Even Only option is used to un-clutter some epsilon plots by coloring points that escape on even iterations only. Odd points are plotted in the background color.

A window is opened to enter a value for Epsilon and Epsilon2, which are used to define the size of the trap areas (.001-2.0 and 0.0-epsilon.) The exclude box is used to exclude the first # iterations (0-99) from orbit trapping.

To produce the maximum 3-D effects (as Phil Pickard and Paul Carlson do) with these options, Level Curve #4 must be set, and the Cutoff value (in the Parameters window) should equal the negation of the epsilon value (-epsilon.) You'll need to set up a special palette with a number of color ranges that matches the split-palette number if set. Built-in examples d3-d6 illustrate how to set up 3d-like fractals.

To automate the process of producing Paul Carlson's 3-D like fractals, a check box has been added to this window for 'Carlson extensions'. This sets the Background and Set Only flags as well as the Level Curve #4 and the cutoff value, and sets the Map to <Abs(Z-Real) or Abs(Z-Imag). An exclude value of 2 is also used. The Map and exclude values are extra parameters that Paul uses in some of his formulas, and may be omitted in other cases. This is easily done by first enabling the Carlson extensions by checking the box and clicking on Okay, then opening the window again and un-checking the box and changing the appropriate variables/flags.

Use Pic as Trap Color -- this option allows you to use a separate bitmap or picture to color the areas hit by the orbit trap. Enabled when a bitmap has been copied to the clipboard (but not with the Formula traps), each pixel inside the trap zone is replaced with its corresponding color in the clipboard image. This produces various mirror-like effects with different orbit traps. Notes: When you first enable this option, whatever is in the clipboard gets copied to a buffer file for use as the picture trap. To change the picture in the buffer, you need to disable this option, and then re-enable it. The clipboard image isn't saved with the data file, so you need to remember which bitmap file is used for the "pic trap", to redo a fractal like this later (you can sometimes leave a comment in the Fun#2 edit box in the Edit/Formula window for this purpose.) For larger file sizes it helps to use higher-resolution clipboard images to reduce graininess in the target image. It also helps to move details around in the clipboard image, as the orbit traps tend to make non-symmetrical use of the pic-trap image.

Use Quaternion as Trap Color -- this option allows you to use a quaternion image to achieve true 3D effects in an orbit-trap. Start by defining an orbit trap image in figure 2. Define a quaternion image in figure 1. Then set the flag for "Use Quaternion as Trap Color" and draw an initial "quat-trap" picture. The orbit-trap data in figure 2 will be transferred to figure 1 and used in conjunction with the quaternion image to produce a hybrid fractal type. The resulting quat-trap retains the borders of the orbit-trap, and incorporates the 3D highlights and depth of the quaternion used. You can zoom into the image like any other 2D fractal. It also works to rotate the original

orbit-trap before applying the quaternion. Note: after the first quat-trap image is drawn, you can modify the orbit-trap values in figure one that have been transferred from figure 2. When you redraw the image, new orbit-trap values are transferred to figure 2. So if you decide to turn off this option to work on each image separately, both figures are up to date with the last quat-trap image drawn.

8.4.1 Orbit trap values

Orbit Trap Values

Enter a value for Epsilon and Epsilon2, which are used to define the size of the orbit trap areas (.001-2.0 and 0.0-epsilon.) The exclude box is used to exclude the first # iterations (0-99) from orbit trapping.

Click on Apply to apply changes without closing the window. Click on Okay to close the window and apply changes, if any. Click on Cancel to exit the window without changing parameters.

Epsilon2 is used to create windows into the stalks. The default value is 0.0, which produces solid stalks. Epsilon2 has no effect on the Petal method.

Check Carlson extensions to apply additional options that are used in Paul Carlson's 3D-like orbit-trap methods.

8.5 Newton Set

Newton Set

The Newton flag is used to map the zeros of a particular function after the Newton transformation has been applied to the function. The program doesn't make the transformation (z-(f(z)/f'(z)), where ' stands for d/dx), but it does allow you to map up to 6 attractors. Each time the Newton flag is set, a window is opened to allow you to enter up to 6 attractors (or repellers) of the function.

This flag is mutually exclusive with the Boundary Scan, Convergence and Renormalization flags, and automatically excludes all points that don't converge to one of the attractors set, within the preset number of iterations. The points that converge are colored with one of up-to-6 possible color spreads (the built-in functions may allow more colors) evenly spaced in the current palette, according to the root they converge to and the time it takes to converge. The non-converging points are mapped with the set color or their level set color (with a level flag set) after the maximum number of iterations. Non-converging points show up typically as round areas or spots.

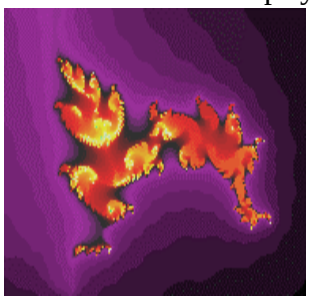
Generally, a limit of 50 iterations gives optimum results. The Newton transformation is normally used with Julia sets, as the attractors (solutions of the formula) can be calculated beforehand. It's also possible to explore the Mandelbrot set applied to Newton's method, but only with some of the built-in formulas mentioned above. It this case, the solutions of the formula for every point on the screen have to be calculated separately, which the program does in a dedicated routine.

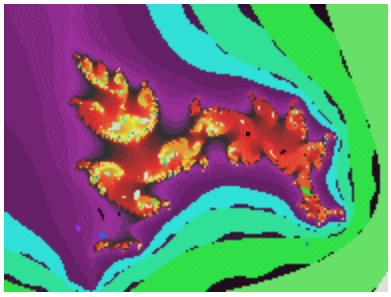
8.6 Newton Off

Newton Off
Turns off the Newton flag, otherwise this option is disabled.

8.7 Renormalize

Renormalization
The Renormalization flag uses a hierarchical lattice transformation to map magnetic phases, with either the Julia set or Mandelbrot set as the iterated function. (Consult The Beauty of Fractals by Pietgen and Richter for appropriate formulas to use.) Basically, the default-mapping algorithm checks orbits for convergence to 1 or infinity, and scales these points in different colors. This flag is mutually exclusive with the Newton, Convergence and Boundary Scan flags. The default method of display actually only checks if z passes through 1. This is similar to the epsilon-cross mapping method. For the original renormalization formulas, there is a strong orbital attraction to 1. For other functions, this mapping produces unusual effects (with obscure mathematical foundations.) For the built-in functions, the convergence tests (type 2 or 3 and 6-8 display types) are the same as the ones used with the Newton flag. So either method produces similar results with the same formula. The differences are worth playing with, though.


Original picture


With renormalization

8.8 Convergence

Convergence
With the Convergence flag set, the program does a convergence/periodic check on all points. This is similar to the convergence checks done with Newton and Renormalization, but also the orbits of each point are saved to determine if the orbit repeats. When an orbit repeats, the iteration loop is

broken and the point colored according to its break time. Depending on the iteration limit, the last 200 points of each orbit are tracked for this check. This flag is mutually exclusive with the Newton, Renormalization, and Boundary Scan flags. May use Newton display methods 3 and 6-8(alternate convergence tests), by setting the arg gadget to these values.

8.9 Period Check

Period Check

With the Period Check flag set, the program does a convergence/periodic check on all points. This is similar to the convergence checks done with the Convergence option, except that only the orbit is escaped from when it repeats. The color of the pixel at escape time is the set color, when no other inside-coloring methods are used. With boundary scan on, the pixel is set to the background color. With continuous potential or level-mapped curves (color menu options), the pixel color will be altered later. This option is useful for speeding-up plots with high iterations, and a large attractive (non-escaping) area. There is a chance of interaction with some inside-mapping options, such as Level Curves, so use with caution.

This flag is mutually exclusive with the Newton and Renormalization flags. May use Newton display methods 3 and 6-8(alternate convergence tests), by setting the arg gadget to these values.

8.10 Default Function

Default Function

When this option is enabled (off by default), convergent functions are iterated according to their original type. Fractal Zplot allows treating a renormalization curve as a Newton curve, or vice versa, but the governing flag must be set through the flags menu. The Default Function option allows a built-in function to work as a Newton or renormalization curve without those flags being set. Newton functions work only as Newtons and likewise for renormalization formulas. Function types that use two built-in functions can distinguish between convergent and non-convergent formulas and use the suitable escape or convergent checking for each formula.

9 Render menu

Render menu commands

The Render menu offers the following commands:

Boundary Scan	Boundary-scanning method.
Level Curve->	Set level curve or reset level curve flag.
Decomposition->	Binary or continuous decomposition.
Decomposition Off	Reset decomposition flag.
Switch->	Switch z components or z for c.
Spin	Increment C by scaled factor of cx at every iteration.
Filter	Choose an optional tail-end filter.
HSV Filters	Define HSV filters.

Coloring Filter	Define coloring filter.
Surface Filter	Define surface filter.
Anti-Alias	Use anti-aliasing, with 1X4 or 1X2 super-sampling.
Link Coloring To Pixel	Set coloring to match absolute coordinates of image.
Atan Coloring	Use Atan algorithm for coloring.
Bof60 Coloring	Use Bof60 algorithm for coloring.
Potential Coloring	Color by magnitude of z.
Add Noise	Add noise to coloring.
Factors	Edit noise factors.
Reset Noise Seed	Re-seed random noise generator.
Texture Scale	Set scaling factor for texture.

9.1 Boundary-Scan

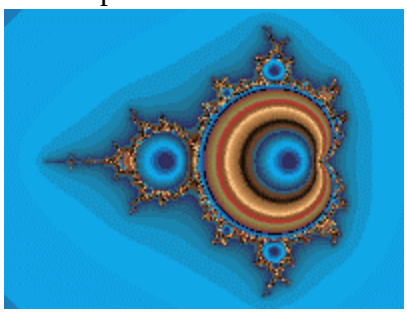
Boundary Scan

This option generates complex sets using a boundary-scanning routine described by C. Pickover. This flag is mutually exclusive with the Convergence, Newton and Renormalization flags.

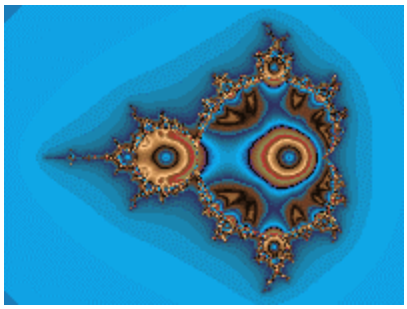
9.2 Level Curve

Level Curve

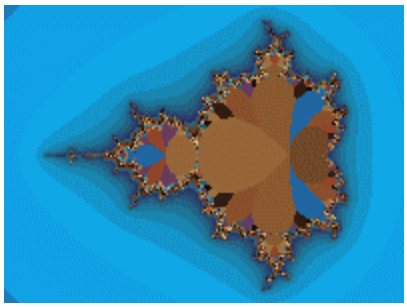
Level-curves map the set points based on how small the value of Z gets. This allows the inside of the complex set to be color-scaled. Log Map #1 produces colored bands on the inside of the complex set. Points are mapped according to what the value of z is at final iteration. Small Log #2 and Linear Map #5 produce circular patterns inside the complex set. Points are mapped according to the smallest value z gets during iteration. Indexed Log #3 and Indexed Linear #6 are mapped according to the time it takes z to reach its smallest value. Level curves 2,3(and 5,6) are described more fully in The Beauty of Fractals. Linear Map #4 is mapped like Log Map #1 (with the mapped value of the function at its final iteration applied to the color palette) and produces 3D-like effects with the Epsilon-Cross method. The Log methods use a log palette, while method #4-6 use linear palettes. This option can override (or may be overridden by) many of the options in the Color-Scaling menu. Decomposition doesn't use Level Curve shading, unless you select the Use Level Curve option.



Log Map #1



Small Log #2



Indexed Log #3

Bubble #7 uses Paul Carlson's contour-mapping method to produce 3D-like bubble pictures. The method is very sensitive to which formula is used, working best with the basic Mandelbrot set z^2+c and the like. Color-mapping should be set to Use Level Curve. This is a trial and error method that uses two other variables to produce the final effect, magnify and cutoff, as entered in the Parameters window. Magnify is used to screen unwanted background contours in the plot, while cutoff is used to fill out the color palette. Magnify should be a low value, usually less than .1, to eliminate the contours that usually appear in escape-type Mandelbrot/Julia sets. If it is too large the bubbles will be too crowded, while too small a value will cause the bubbles to disappear. Cutoff needs to be a small negative value, usually equal to the magnify value times the number of color splits. E.G., for a magnify value of .1 you should use a cutoff value of -.8 for a divide-by-eight palette. An incorrect cutoff value will cause the colors to overlap in the bubbles. For split palette pictures, the colors are divided according to their level index, as in Indexed Linear #6. The color ranges should be graded from light to dark to highlight the bubble centers.

9.3 Decomposition

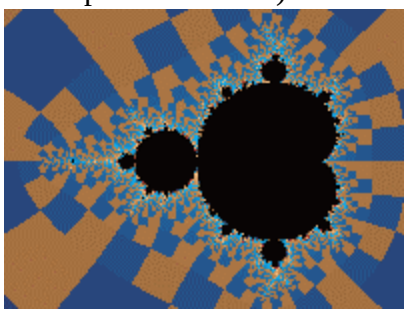
Decomposition

When a Decomposition flag is set, you have the option of performing either a binary or continuous decomposition. Toggle the External/Internal option for either an external or internal decomposition. The Angle-Only option excludes the Coloring-scaling options from consideration when plotting points derived from decomposition. It also excludes escape times in connection with the Angle-Iteration option on the Color-Scaling menu. An external decomposition decomposes points that are outside the complex set. An internal decomposition decomposes the complex set. For Mandelbrot/Julia curves, z-arg is broken into two parts for a binary decomposition. For Newton/Renormalization curves, the binary decomposition is also related to the number of solutions a formula

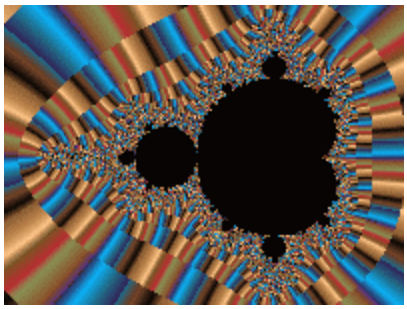
has, if it supports mapping option 1. Continuous decomposition breaks z-arg into n parts, where n=angles (2-256), as set in the Continuous Decomposition window.

Note: With the graded-palette option checked, the decomposition option is extended for extra smoothness in Fractal Zplot. The number of angles is internally multiplied by 236 to track the decomposition angle more closely.

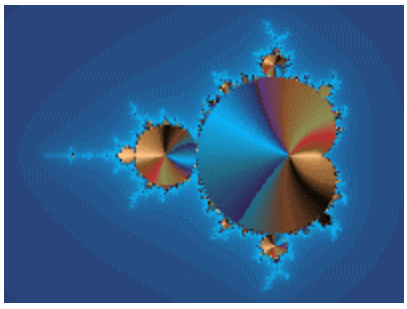
(Consult The Beauty of Fractals by Peitgen & Richter for a mathematical explanation of decomposition.) When Biomorph or Epsilon is decomposed, the tendrils or hairs are decomposed as external points. Use the Set Only flag to emphasize the tendrils and hairs when external decomposition is used.)



Binary Decomposition



Continuous External



Continuous Internal

9.4 Decomposition Off

Decomposition Off

Turns off all decomposition flags and resets the Internal/Eternal option to Eternal.

9.5 Switch

Switch

When a Switch flag is set, you have the option of switching the real and imaginary parts of Z, or switching Z for C. The real part of Z is exchanged with the imaginary part of Z after each iteration. Using this technique with the Mandelbrot set produces a tricorn-like plot. When Z is switched for C, normally you get Mandelbrots from Julia sets and vice versa. This option also offers an inverse to the Julia Tower, with glimpses into the elusive Mandel Tower, a sort of gateway to infinity.

9.6 Spin

Spin

When the Spin flag is set, the complex constant is incremented by a scaled factor of cx(cx*c/ iterations) at every step of iteration. Unavailable for Julia Tower types.

9.7 Filter

Filter

Based on Stephen C. Ferguson's filter algorithms in his program Iterations, this option allows you to choose one of 29 tail-end filters to apply to any 2D plot. The name of the filter corresponds roughly to its effect on the basic Mandelbrot-squared set. The effect will vary with the formula and fractal type chosen. This overrides the Background option on the Color-Scaling menu. Useful to add detail to orbit-trap pictures, as well as perk up any otherwise ordinary picture. Filters 27-29 are generalized filters that use fn4 and fn3 (in the edit/formula window) for expanded scope. The "cross" filters use the epsilon variable from the orbit-trap window. (To set epsilon's value without applying an orbit-trap, first change the value of epsilon to the value desired, and set the orbit-trap flag. Then turn off the orbit-trap flag. Epsilon value remains unchanged when the orbit-trap flag is turned off.)

The Magnify variable is used to intensify or de-intensify the effect of the filter. This value can range from 1-500 nominally. The Add Offset box is checked when you want the filter to add an offset to the color value normally plotted. The Exclude Background box works like the Add Offset box, except that background pixels are unfiltered. With the Replace All box checked, the filter totally replaces the normal color value, which can lead to very different color-rendering. With the Background Only box checked, only pixels which would normally be colored with the background color (index 0) are filtered.

9.8 HSV Filters

HSV Filters

This option allows you to choose up to 3 hsv filters to apply to any 2D plot. These effectively extend the range of the palette used, or reassign colors independent of the limits of the palette. The filters may be assigned to modify hue, saturation or value component of the pixel's normal color. The filters may be Iteration (time-based), Z-Potential (last z), Exit Angle (polar-based), Level Index (time/magnitude based), Coloring Filter (formula/magnitude based), Z-Potential (smallest z), or Orbit Trap. Formula-type orbit traps default to Epsilon Cross for this option.

The Magnify variable is used to intensify or de-intensify the effect of the filter. Begin with 0.5 and increase or decrease as necessary.

The Start slider defines the origin of the filter so that it acts as a decrement or increment for various filter values. For the Iteration filter the values correspond to 0-Max Iterations. Start values range from 0-100 percent of the nominal filter range. For the Iteration filter, any Start value greater than 0 would make all Iterations less than the Start value decrements. The Exit Angle normally ranges from -PI to PI, so a Start value of 50 would make all angles less than 0 decrements. The Z-Potential filter has a range of 0-zlimit^2, for values that do not escape the iteration loop. For escape values, a modulus is used to bring the filter within hsv parameters. For the Level Index filter and Coloring filter, the Start slider represents a decrement/increment of -1 to 1.

The Add Offset box is checked when you want the filter to add an offset to the color value normally plotted. With the Replace All box checked, the filter uses only the offset for the color value, which can lead to very different color rendering. This differs somewhat from how the Replace All box in the Filter's windows works. Here the offset is still loosely tied to the pixel's normal color. With the Background Only flag checked, only pixels which would normally be colored with the background color (index 0) are filtered.

Notes: The Level Index filter was designed specifically for modifying a pixel's hue, though may produce interesting results applied to a pixel's saturation or value component. Hsv filtering is not available with solid-guessing. Use the Coloring Filter window to define a coloring formula. Sample coloring formula: sin(x*x)+cos(y*y).

9.9 Coloring Filter

Coloring Filter

Here you define an hsv filter based on a real function. A generalization of Earl Hinrichs' sine-wave coloring method, the function can be any formula, up to 80 characters, that uses the z components x and y. X and y are the real and imaginary parts of the last z value in the iteration loop. Sample function: sin(x+y)+cos(x*x). The Magnify slider is used to control the intensity of the filter (in conjunction with the Magnify variable in the HSV filter window.) Use the Preview button to see what the filter looks like with x and y ranges of -2pi to 2pi.

Note: the coloring filter can also be used with quaternions, ray-traced orbital fractals and 3D height fields (when the Ray Trace option is selected) to map palette indexes or spreads to the x/y ranges of these fractal types. The variable 'z' is equated to the zbuffer value in this case, so may also be included in the formula. Quaternions and ray-traced fractals normally use palette index one (the second index, zero being reserved for the background color) for their predominant color, with pixel intensities/colors affected by the lighting variables. When the coloring filter formula is defined, up to 235 colors can be used (the full palette) to create mixed textures.

The trig and exponential functions translated include sine (sin), arc sine (asn), cosine (cos), arc cosine(acs), tangent (tan), hyperbolic tangent (th), hyperbolic sine (sh), hyperbolic cosine (ch), log

(log), natural log (ln), power (pow), arc tangent (atn), absolute value (abs), exponential (exp) and square root (sqr.)

The math functions are *(multiply),-(subtract),/(divide), and +(add).

The constants are PI and E (ln (1)), plus any floating-point number up to 9 digits (including the decimal point).

The power function (x to the y power) is entered in standard notation: x^y', with optional parenthesis necessary around complex exponents or variables.

Note: Range limits exist for arguments to these functions: exp, arc sine, hyperbolic sine, arc cosine, hyperbolic cosine, arc tangent, and hyperbolic tangent (+/-100.0 for the exponential , +/-200.0 for hyperbolic functions, +/-1.0 for the arc functions), the log functions (must be >0) and the power function (x must be integral and non-zero when y<0, and 0^0 is undefined). Square root is undefined for x<0. No filtering is done when these limits are exceeded.

Syntax for an acceptable formula is AS([XY])+bs([xy])...
.up to 80 characters per formula. Algebraic notation is supported to a limited degree. E.G. you can enter a variable as 2x^2, instead of 2*x*x.
A and B are optional constants.

S is an optional trig function (1 to three letters: 1 will work for sine, cosine and tangent, but use the above abbreviations for the other functions. X and Y are the standard variables. The '+' could be any of the math functions.

The parser interprets up to 10 levels of parenthesis. Use parenthesis to separate complex expressions. Use parenthesis to embed trig functions within other trig functions, etc.

9.10 Surface Filter

Surface Filter
Here you define a quaternion surface filter based on a real function. Like a coloring filter, except that the formula is used to warp the quaternion's shape. The Magnify variable is used to control the intensity of the filter. Use the Preview button to see what the filter looks like with x and y ranges of -2pi to 2pi.
Use the Random Filter button to generate a random surface filter. The best surface filters will use the z value and one or both of the other variables (x or y.)

9.11 Anti-Alias

Anti-Alias
Applies a 2 to 1 or 4 to 1 averaging filter to every pixel plotted, to reduce jaggies and other high-frequency noise. This increases the processing time 4 to 8 times, so is mainly a final rendering method, not for general development use. Not available for most 3D-type fractals (for quaternions,

only single-pass mode is supported -- or without ray-tracing, solid-guessing can be used), plot-to-file, midpoint-displacement, 3D backgrounds or with the tesseral solid-guessing method. Note: because of the lengthy time required for applying the anti-aliasing filter, and because anti-aliasing calculates different smoothing colors each time the palette is changed, all color-cycling and palette-switching hot keys are disabled with the anti-alias flag set.

9.12 Link Coloring To Pixel

Link Coloring To Pixel
Set coloring to match absolute coordinates of (quaternion) image. This uses extra buffers to track a figure's texture, so that when you rotate it, the texture moves with the figure. Must be enabled to use Atan Coloring, Bof60 Coloring, Potential Coloring, and the Noise functions. Due the extra memory required for this command, you won't be able to open an image much larger than 1600X1200, unless you have more than 128MB of system memory.

9.13 Atan Coloring

Atan Coloring
Uses an Atan algorithm by David Makin to color a quaternion image. Link Coloring To Pixel must be set to enable this option.

9.14 Bof60 Coloring

Bof60 Coloring
A variation of the Bof60 algorithm found in the classic Pietgen/Ricter text, The Beauty of Fractals, adapted by David Makin to color a quaternion image. Link Coloring To Pixel must be set to enable this option.

9.15 Potential Coloring

Potential Coloring
The magnitude of z (at the quaternion border) is used to color the (quaternion) image. Link Coloring To Pixel must be set to enable this option.

9.16 Add Noise

Add Noise
Add noise to (quaternion) image texture. A variation of Perlin's noise algorithm is used to add natural randomness to an image's coloring. Link Coloring To Pixel must be set to enable this option.

9.17 Factors

Factors
Edit noise factors. The blend variable determines how much noise is added to an image. The higher the blend, the more pronounced the noise appears. This also tends to darken an image, which can

be compensated for by decreasing Gamma. The Grain variable determines the frequency of the noise. The higher the grain, the noisier the image appears. You can adjust how the noise maps to an image by changing the scale factors. Higher scale factors make the image noisier on the respective axis (x, y and z.)

The Surface Warp variable allows you to apply the same noise to a (quaternion) figure's shape also, like a surface filter. Small values are best for creating realistic surface variations, like stone and wood grain.

9.18 Reset Noise Seed

Reset Noise Seed

The random noise generator is re-seeded. Use this to create variations on the noise texture.

9.19 Texture Scale

Texture Scale

Opens a window to edit texture scale factors. The higher the scale factors, the more repetitive the texture becomes. You can adjust the factors to make the texture asymmetrical on the x, y or z-axis. Scale A is used to adjust the texture scale for the atan and Bof60 coloring options.

10 Pixel menu

Pixel menu commands

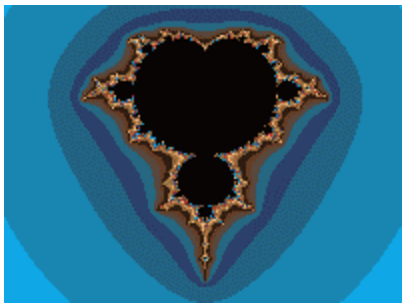
The Pixel menu offers the following commands:

Phoenix	Phoenix Curve.
Invert	Invert image around circle.
Invert Off	Reset inversion flag.
Symmetry->	Horizontal, vertical or XY symmetry.
Fast->	Speed up five basic types of fractals.
Solid-Guessing	Solid-guessing plotting mode.
Tesseral	Tesseral solid-guessing plotting mode.
Segment	Plot image as part of segmented array.
Torus	Use torus method.
Torus Off	Reset torus flag.
Use Stencil	Use border on picture.

10.1 Phoenix

Phoenix

The Phoenix flag rotates the planes, so that the imaginary plane is mapped horizontally and the real plane is mapped vertically.



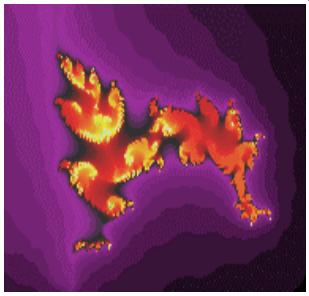
This option is normally used for mapping Phoenix curves (Shigehiro Ushiki), which are Julia-related curves based on the formula $f(z+1)=z^2+p+qz$. 'p' and 'q' are constants, and the 'z' term of 'qz' is actually the value of z^n-1, or the previous value of z before the current iteration. 'zn' is reserved by Fractal Zplot to represent this value, while the complex constant set in the Parameters window becomes 'p' and 'q'. The real part of the complex constant is 'p' and the imaginary part of the constant is 'q' (when the Phoenix option is chosen).

If the Phoenix flag is used with the Mandelbrot option, 'j' and 'k' should be used as the constants, since the complex constants p and q are already used as the starting value of 'z0'.

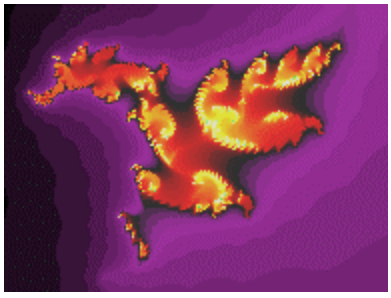
10.2 Invert

Invert

The Invert flag inverts the plane around a circle. A window is opened that allows the user to specify the circle's radius and center coordinates. Select Auto Coords to let Fractal Zplot calculate the center coordinates and circle radius. Using Auto Coords, the new radius and center coordinates are calculated when the picture is next drawn. You can zoom on an inverted picture as long as radius and center coordinates remain the same. Use the Perspective box to alter the X/Y symmetry of the inversion. A smaller Perspective value (less than 1.0) stretches the inversion in the vertical direction.



Original picture



Inverted

10.3 Invert Off

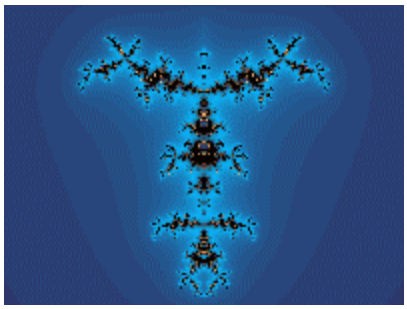
Invert Off

Turns off the inversion flag. Alternatively you can set the inversion radius to 0.0 to turn off inversion.

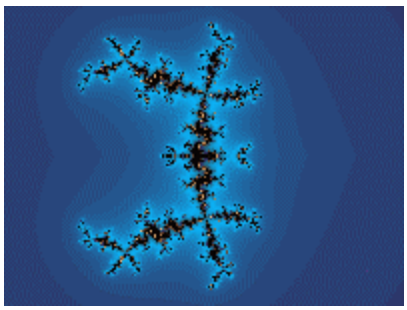
10.4 Symmetry

Symmetry

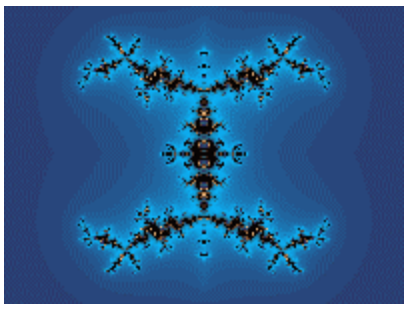
This produces a mirror image from left to right (vertical) or top to bottom (horizontal) or both (xy). You can zoom with symmetry, but the results will be uncertain if the zoom box is off-center on the window or if rotation is used. Symmetry has no effect when used with the Tesseral option, 3D, or random fractals.



Vertical symmetry



Horizontal symmetry



XY symmetry

10.5 Fast

Fast

This option alters flags and variables to speed up drawing time for 5 basic fractal types.

Fast Mandelbrot/Julia sets up the flags and filters to do non-convergent type plots. This is fully optimized for the non-hypercomplex Mandelbrot set (p0; z^2+c.) New code has been included in v1.16+ to speed up the basic Mandelbrot/Julia set even more. This is based on an assembly language routine by Damien M. Jones. It's over twice as fast when used with pictures that don't need filtering or exponential smoothing.

Fast Orbit-Trap sets up the flags and filters to do a Carlson-style orbit-trap picture.

Fast Newton sets up the flags and filters to do a basic Newton's method picture.

Fast Biomorph sets up the flags and filters to do a Pickover-style biomorph picture.

Fast Quaternion sets up the flags and filters to optimize quaternion drawing (q^2+c only.) Note: Any mode uses the new z-buffered qjulia mode, which can speed up drawing up to 3X. This option uses the qjulia set q^2+c directly in its iteration loop, which is faster for generating quaternions than any of the user-selectable formulas in Fractal Zplot.

Fast modes are not available with Formula Types other than 0 or 9(for Mandelbrot/Julia sets only.), or a user-defined bailout or initialization.
The speedup averages 1/3X faster on general types, but can be up to 10X on the Mandelbrot option.

10.6 Solid Guessing

Solid Guessing

In the solid-guessing plotting mode, the program guesses at colors that lie inside rectangular areas of the plot. It first computes all the perimeter pixels of a rectangle, and checks if all the pixels have the same color. If so, all the pixels inside the rectangle are colored the same and no further calculations are done on that rectangle. Otherwise the rectangle is broken into four parts and the above procedure is repeated for each part. If any of the perimeter pixels are different at this point, all the remaining pixels in the smaller rectangle are computed. The screen is updated in groups of 16 lines.

This method can be much faster than the default single-pass mode that Fractal Zplot starts in. This is especially true for plots that have large areas of a single color. For very intricate plots that have little open space, the solid-guessing mode can still be 15-20% faster.

The solid-guessing mode can fail for some plots that have small areas of one color that are islands inside the rectangular areas under test.

Solid-guessing is not available for lsystems, orbital fractals, 3D height fields or midpoint

displacement.

10.7 Tesseral

Tesseral

Tesseral is a variation of the solid-guessing plotting mode, where the program guesses at colors that lie inside rectangular areas of the plot. The main difference is in the size of the rectangle that tesseral uses to start with and how the screen is updated. Tesseral starts with the whole screen and divides that into quarters, eighths etc until it reaches a solid block or a minimum size to fill in pixel by pixel. This is a recursive routine, so the whole screen is updated every 100 recursions, or when done, rather than by lines.

This method can be much faster than the default single-pass mode that Fractal Zplot starts in. It runs about the same speed as the solid-guessing mode. The main advantage Tesseral has over solid-guessing is when there are very large areas of one color in the plot (such as a Mandelbrot island) that take a long time to compute, as when the iterations variable is set to a large number.

Like solid-guessing, Tesseral can fail for some plots that have small areas of one color that are islands inside the rectangular areas under test.

When selected, tesseral opens a window to set an optional fill color. The areas filled in by the tesseral routine are filled using this color, if zero or greater is entered. Tesseral is turned off by selecting Solid Guessing or entering a negative number less than -1 for the fill color.

Tesseral is not available for 3D, 3D background, midpoint displacement plots.

10.8 Segment

Segment...

This option allows you to break a large bitmap (larger than would fit into memory) into tiles that can be reassembled later with a program such as Richard Paasen's Image Arithmetic. The Segments box defines how many tiles the image is broken into (1-225.) The number of tiles horizontally must equal the number of tiles vertically; for example, a 4X4 tiling would consist of 16 tiles total. Segments are numbered 0-(no. of segments - 1), with 0 being the upper-left corner tile and following left to right in rows to the last tile at the lower-right corner.

10.9 Torus

Torus

Pixels are mapped around a torus, and then expanded to fit the drawing area. A generalized form of Earl Hinrichs' torus method, variables are provided for center x and center y to define the c and z radii and may both equal 0.0. Results will vary with the formula used, but resembles the warping effect found in hypercomplex images. Two versions of this method are provided: the Pixel method which uses pixel values to map the torus to the fractal space, and the Two-Pi method which uses an initial rectangle 2 pi by 2 pi to map the torus to a fractal image. With the Two-Pi method, when you

zoom the rectangle's size and starting points are changed to match the zooming area. The rectangle's coordinates are saved with the fractal. If you turn off the torus flag after zooming and then reinitialize the torus flag, the rectangle reverts to a 2X2 area, so the image will change accordingly. Rotating is not supported for the Two-Pi method, but does work in a limited way with the Pixel method.

10.10 Torus Off

Torus Off

Turns off the torus flag. Alternatively you can enter a negative value to turn off this flag.

10.11 Use Stencil

Use Stencil

A border is created around the plot as it is drawn. This can be a circular or oval border. The border uses the background color. Not available with 3D plots, quaternion plots, midpoint displacement plots.

11 Color menu

Color menu commands

The Color menu offers the following commands:

Cycle	Cycle colors.
Color-Scaling menu	
Palette menu	
Divide By One Palette	No split palette.
Divide By Two Palette	Split palette into two sections.
Divide By Four Palette	Split palette into four sections.
Divide By Eight Palette	Split palette into eight sections.

11.1 Color Cycle command

Cycle command (Color menu)

Use this command to cycle colors when not plotting. Works with any coloring mode, but not with hsv filtering or anti-aliasing. Undoing an action disables the cycle command until the image is redrawn.

11.2 Color-Scaling menu

Color-Scaling menu commands

The Color-Scaling menu (in Color menu) offers the following commands:

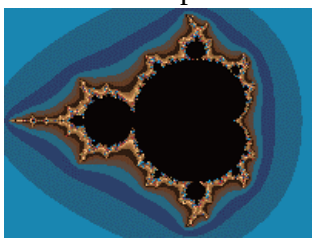
Escape->	Escape-time color scaling.
Level	Color scaling based on log(z).
Continuous Potential	Color scaling based on continuous potential.
Use Level	Use level curve option for all coloring.
Background	Set external points to background color.
Set Only	Plot points in complex set only.
Graded Palette->	Use non-repeating vs modulus palette.
Use Palette 1 ...	Use palette 1 for background filter.
Log High	Logarithmic mapping of level potential.
Log Low	Logarithmic mapping of reversed potential.
Continuous Potential High	Linear mapping of level potential.
Continuous Potential Low	Linear mapping of reversed level potential.
Ray Trace	Ray trace 3D plot.
Sea Level	Set sea level at ground zero.

11.2.1 Escape

Escape

Five options are included that color a point based on its escape time (when it blows up.)

The Iteration option uses only the point's escape time.



Escape-time coloring.

The Iteration+ option uses the sum of a point's escape time and the value chosen (which can be picked from a menu that mirrors the Map menu.) A window is opened to set a q factor (1-200), which scales the sum value.

The Iteration* option uses the product of a point's escape time and the value chosen. .) A window is opened to set a q factor (1-200), which scales the product value.

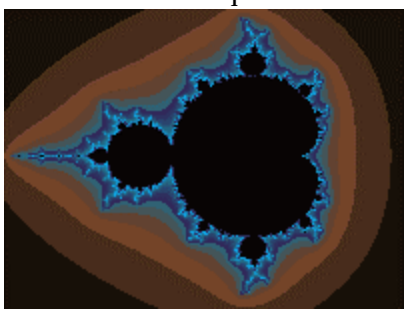
The Angle option use the absolute value of a point's exit angle (theta.) This is the atan method in Fractint.

The Angle-Iteration option uses the angle formed by the difference between a point's last two exit values and subtracts the point's escape time. Using the Angle-Only option on the Decomposition menu, escape times are not subtracted from the difference angle. This is Paul Carlson's atan method.

11.2.2 Level

Level

A point is colored based on its logarithmic escape. A window is opened to set a q factor, which controls the smoothness of picture color. A higher q factor results in grainier pictures and excess detail. Too low a q factor results in loss of colors and detail.

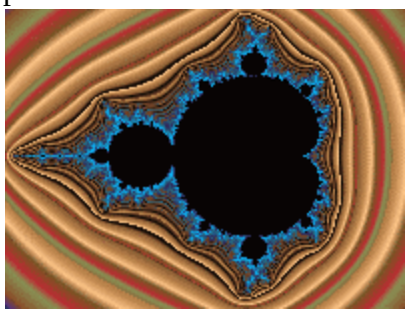


Level coloring.

11.2.3 Continuous Potential

Continuous Potential

A point is colored based on its continuous potential (when it blows up.) A window is opened to set a q factor, which controls the smoothness of picture color. A higher q factor results in grainier pictures and excess detail. Too low a q factor results in loss of colors and detail.



Continuous-Potential coloring.

11.2.4 Use Level Curve

Use Level Curve

All points are colored according to the choice selected from the Level-Curve Flag option. Defaults to Linear Map #4 if no Level Curve is checked. This works with Decomposition and other methods that would normally not use Level-Curve shading.

11.2.5 Set Only

Set Only

The Set Only flag plots all external points in the background color.

11.2.6 Background

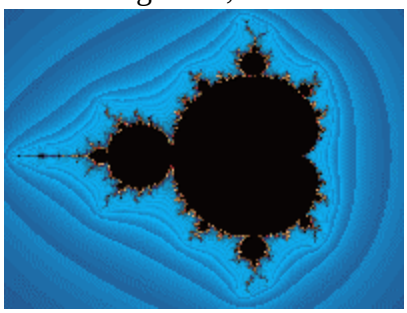
Background

An external point is colored with the background color. This works like the Set Only flag, except with decomposition plots and Biomorph/Epsilon plots. Normally, when a point is decomposed, its escape time or level color is added to its arg (exit angle) to determine its final coloring. With Background color-scaling, only a point's arg determines its color.
With Biomorph/Epsilon plots, all external points are colored with the background color and all Biomorph/Epsilon points are colored with the set color.

11.2.7 Graded Palette

Graded Palette

All points are colored with a non-repeating graded palette versus the default repeating(modulus) color scaling. Has no effect on the Background or Use Level Curve options, or when you use the cutoff value in the Parameters window as a color multiplier. When used with the Escape/Iteration coloring mode and a negative cutoff value, iterations are interpolated to reduce banding in escape-time pictures. Three options are provided for smoothing. The Interpolated version works for most escape-time formulas except convergent types (Newton and renormalization.) The Mandelbrot version is based on Linas Vepstas' log log algorithm, and is designed mainly for formulas that use z^2 as their main focus, such as z^2+c. The Exponential smoothing method is based on Ron Barnett's algorithm, and works for both escape time and convergent-type fractals.



Level-graded coloring.

11.2.8 Use Palette

Use Palette 1 for Background Filter

This allows you to use palette 1 with a background filter, instead of the default palette, to add highlights to a picture. When Background filter is checked in the Filter window and a filter is selected, FZ uses palette 1 for any color indexes modified by the filter. This option is not available with solid-guessing.

11.2.9 Log High

Log High

Maps hill or mountain-type plots with a logarithmic slope based on iterations (escape time of Z.) Logarithmic slopes smooth out fast changes in point potentials. User specifies the level of the set point with the Rise box (0 to [# colors-1]).

11.2.10 Log Low

Log Low
Maps valley or lake-type plots with a logarithmic slope based on iterations (escape time of Z.) Logarithmic slopes smooth out fast changes in point potentials. User specifies the level of the set point with the Rise box (0 to [# colors-1]).

11.2.11 Continuous Potential High

Continuous Potential High
Maps hill or mountain-type plots with a logarithmic slope based on potential (the absolute value of Z when the point escapes.) This option usually produces smoother contours with the Mandelbrot set than the Log options. User specifies the level of the set point with the Rise box (0--[# colors-1]).

11.2.12 Continuous Potential Low

Continuous Potential Low
Maps valley or lake-type plots with a logarithmic slope based on potential (the absolute value of Z when the point escapes.) This option usually produces smoother contours with the Mandelbrot set than the Log options. User specifies the level of the set point with the Rise box (0--[# colors-1]).

11.2.13 Ray Trace

Ray Trace
Add a light source, with Phong highlights, to 3d plots. Color palettes should be continuous (dark to light to dark) to take best advantage of this option. The light source parameters may be altered in the Edit Quaternion window. Starting with version 1.22, this option is the default for all 3D fractal types. New: if you prefer to generate 3D plots, such as quaternions, without ray-tracing, deselect this flag before drawing the plot. The image draw will be the pre-image, using the existing palette, as is, without ray-tracing (Phong and shading.)

11.2.14 Use Sea level

Use Sea Level
Imposes a water level at zero potential for 3D plots. All points below zero potential are mapped at zero to create lakes or oceans around 'land' areas.

11.3 Palette menu

Palette menu commands

The Palette menu (in Color menu) offers the following commands:

Palette #1-21 Use one of 21 palettes.

11.3.1 Palette 1-21 command

Palette command (Palette menu)

Switch to palette #. Used with palette-coloring mode.

11.4 Color Divpal1 command

Divide by One Palette (Color menu)

Palette is not split before applying to pixel.

11.5 Color Divpal2 command

Divide by Two Palette (Color menu)

Palette is split into two parts before applying to pixel.

11.6 Color Divpal4 command

Divide by Four Palette (Color menu)

Palette is split into four parts before applying to pixel.

11.7 Color Divpal8 command

Divide by Eight Palette (Color menu)

Palette is split into eight parts before applying to pixel.

12 View menu

View menu commands

The View menu offers the following commands:

Toolbar	Shows or hides the toolbar.
Status Bar	Shows or hides the status bar.

12.1 View Toolbar command

Toolbar command (View menu)

Use this command to display and hide the Toolbar, which includes buttons for some of the most common commands in Fractal Zplot, such as File Open. A check mark appears next to the menu item when the Toolbar is displayed.

See Toolbar for help on using the toolbar.

12.1.1 toolbar

Toolbar

The toolbar is displayed across the top of the application window, below the menu bar. The toolbar provides quick mouse access to many tools used in Fractal Zplot,

To hide or display the Toolbar, choose Toolbar from the View menu (ALT, V, T).

Click	To
	Open the remote which contains shortcut buttons for many common tasks and options in Fractal Zplot
	Open an existing drawing. Fractal Zplot displays the Open dialog box, in which you can locate and open the desired file.
	Save the active drawing or template with a new name. Fractal Zplot displays the Save As dialog box.
	Draw Mandelbrot set
	Draw Julia set
	Zoom into rectangle.
	Set image size.
	Edit palette.
z^2+c	Edit formula/type data.
	Edit fractal parameters.
	Draw image from current parameters.
	Continue drawing.
	Reset coordinates.
	Show picture full-screen.
	Display info about Fractal Zplot.
H	Display Fractal Zplot's help index.

12.2 View Status Bar Command

Status Bar command (View menu)

Use this command to display and hide the Status Bar, which describes the action to be executed by the selected menu item or depressed toolbar button, and keyboard latch state. A check mark appears next to the menu item when the Status Bar is displayed.

See Status Bar for help on using the status bar.

12.2.1 status bar

Status Bar

The status bar is displayed at the bottom of the Fractal Zplot window. To display or hide the status bar, use the Status Bar command in the View menu.

The left area of the status bar describes actions of menu items as you use the arrow keys to navigate through menus. This area similarly shows messages that describe the actions of toolbar buttons as you depress them, before releasing them. If after viewing the description of the toolbar button command you wish not to execute the command, then release the mouse button while the pointer is off the toolbar button.

The right areas of the status bar indicate which of the following keys are latched down:

Indicator	Description
CAP	The Caps Lock key is latched down.
NUM	The Num Lock key is latched down.
SCRL	The Scroll Lock key is latched down.

13 Window menu

Window menu commands

The Window menu offers the following commands, which enable you to arrange multiple images in the application window:

Cascade	Arranges windows in an overlapped fashion.
Tile	Arranges windows in non-overlapped tiles.
Arrange Icons	Arranges icons of closed windows.
Size Desktop	Size drawing area to window frame.
Window 1, 2, ...	Goes to specified window.

13.1 Cascade

Cascade command (Window menu)

Use this command to arrange multiple opened windows in an overlapped fashion.

13.2 Tile

Tile command (Window menu)

Use this command to arrange multiple opened windows in a non-overlapped fashion.

13.3 Arrange Icons

Window Arrange Icons Command

Use this command to arrange the icons for minimized windows at the bottom of the main window. If there is an open drawing window at the bottom of the main window, then some or all of the icons may not be visible because they will be underneath this drawing window.

13.4 Size DeskTop

Window Size DeskTop Command

Use this command to size the active drawing window to its frame size. Use after Tile command to reduce white space around a drawing that is smaller than screen size.

13.5 1, 2, ...

1, 2, ... command (Window menu)

Fractal Zplot displays a list of currently open drawing windows at the bottom of the Window menu. A check mark appears in front of the drawing name of the active window. Choose a drawing from this list to make its window active.

14 A/V menu

A/V menu commands

The A/V menu offers the following commands:

Open AVI Stream	Open AVI file for writing and draw initial frame.
Write Frames	Write frames to AVI file.
AVI Variables	Edit AVI variables.
Close AVI Stream	Close an existing AVI stream.
View AVI	View an AVI animation file.
Start Midi	Enable midi sampling and set sample size.
End Midi	Disable midi sampling.
Save Midi	Save midi data to mdf and mid files.
Load Midi	Load midi variables from mdf file.
AVI Composite	Generate composite video.
AVI Object	Output video frames as obj file.

AVI WRL Output video frames as wrl files.

14.1 Open Avi Stream

Open Avi Stream…
Through a series of windows, this allows you to name and open an avi animation stream and choose a compression method. After using the file requester to name the file, you are given a choice of compression methods. The compression methods include Intel Indeo Video®, Microsoft Video 1 and Cinepak Codec by Radius. (All compression methods degrade the original images, some more than others.) The first key frame in the stream is then drawn and written to the file.

Notes: after the stream is opened, the size of the fractal that can be drawn is fixed at the size of the frame. No changes can be made to the size until the stream is closed. New: If you open a video stream after setting up a batch mode (Demo menu), then the frames will be written as a series of bmp, obj or wrl files, depending on whether AVI Object or AVI WRL is also checked.

14.2 Write Frames

Write Frames…
With this option, frames are written to a stream based on the difference between the current key frame and the previous key frame. The first key frame is written when you open a stream. The next key frame is created each time you use this option. In between you can zoom or change Avi variables as much as necessary. The stream is only written to when this option is used. The last key frame is automatically saved after the 'tween' series is written. The number of frames may range from 1-1500 frames between keys. With a frame number of 1 only the key frames are written. This allows animation to be created that incorporates all scalable variables in Fractal Zplot.

Use the Cancel button to exit this dialog without initializing a new series of frames.

Check the Log Scaling box if you want the frames to be written with logarithmic space between frames, else linear space is used. Useful when zooming, where frames would otherwise be packed together at the end of the frame series.

Notes: key frames are saved in parameter files (.zp6), with filenames of "bvf_image#_title.zp6", where '#' is the number of the keyframe and 'title' is the name of the working fractal file.

14.3 Avi Variables

Avi Variables…
This window contains all the major variables that Fractal Zplot now scales between key frames of an avi stream. They are identical to some of the variables found in the Parameters, Formula and Epsilon windows, plus a few other windows. If you decide to change a variable not included in this window, no scaling occurs. A few exceptions are the newlimit, limit, invert, torus, quaternion variables and the palette indexes (if you change palettes between key frames.) These are scaled also. If you change the basic function type, formula or color-scaling method, morphing isn't supported for these sorts of changes.

Note 1: when a formula is changed between key frames, the formula in the last key frame is used for tweening purposes. This may or may not produce useable images.

Note 2: when a frame number of 1 is used in the Write Frames window, all variables in FZ that can be scaled are useable for animation. In this case the animation is composed of single key frames and optional tween frames.

14.4 Close Avi Stream

Close Avi Stream
Closes any open avi stream file. You need to do this before viewing the file or creating a new avi file. The stream is also closed when you exit Fractal Zplot.

14.5 View Avi

View Avi…
Opens an avi file for viewing. You can preview any multimedia file by clicking on its file name. A multimedia box will appear to the right of the file list. Click on okay to open the main view window.

There are buttons to Play a file forwards or Backwards, or forward automatically with Auto rewind/ repeat. Click on Slow to slow down a video. Each click on Slow halves the viewing speed. A click on Stop freezes viewing and restores the view speed to normal playback.

Use the Open button to view a different avi file. Use the Save button to save the file in a different compression format. You must use a different name to save the file than the name that was used to open it. Click on the left-mouse button or any key to abort a save operation.

Note: the view avi requester can be used to preview any multimedia file, including midi files.

14.6 Start Midi..

Start Midi…
Start collecting sample data for the current fractal. This allows you to translate fractal data into a form that can be used later to create fractal music. You can specify the sample width for sampling (40-100.) This controls how many samples will be gathered and thus how large the midi file will be. The size of the current fractal must be larger than the sample size. A width of 40 creates a midi file length of about 3 minutes run time (at 24 clicks per division.) Notes: samples are collected when a fractal is drawn. Start Midi is not available when Solid-Guessing is selected, with 3D fractal types or orbital fractals. Disabled also when AVI is enabled, or anti-aliasing is checked.

14.7 Stop Midi..

Stop Midi…
Stop collecting sample data for the current fractal. You select this command to remove midi sampling from the iteration loop. Warning: you abandon any sample data collected when using this

command.

Note: it's necessary to use this command sometimes when you need to use solid-guessing or anti-aliasing which are not available after starting midi sampling.

14.8 Save Midi

Save Midi…
Saves the sample data to a midi text file(c:\miditext.txt) then converts the text file to midi binary format. The format for the text file follows Piet van Oostrum's specifications for his text-to-midi converter, T2MF, which is used for this conversion. An mdf file is also created that saves the current midi parameters in this window and the sample size. The sample data is not saved. You need to recreate the sample data each time you reopen Fractal Zplot. But once the data is created, it can be modified again and again during the same working session.

After specifying a filename through the filename button and window, you can change any other parameter in the window to customize midi output. The divisions' slider controls click per quarter note (1-100, default: 24), or the overall tempo of the music. You select 1-16 channels, and then apply a patch (instrument voice) to each channel. The pitch and volume can be adjusted separately for each channel (0-20, default: 5.) Since instruments tend to be pitch sensitive, min and max pitch controls are provided to tailor frequency response for each channel. Initial note (pitch), volume and clicks (time between notes) are set by choosing a filter that translates sample data for each parameter. The filter can be based on average color for a sample, last level or smallest level of a sample, the average exit angle or escape time for a sample, or a constant (default: 6*volume setting for each channel.)

14.9 Load Midi..

Load Midi…
Load midi parameters from an mdf (midi-data) file. This allows you to recreate/edit a midi file using larger samples/different voices, etc.

14.10 Avi Composite

AVI Composite
When this flag is set, Fractal Zplot generates composite frames for a video according to the settings in the Image/Composite window. Each frame may then consist of a merging of up to 4 figures (1-4). You must set this flag and the composite options before beginning a video. After an avi stream has been opened, you can then use variations of any figure in the composite to produce tweens while using the Write Frames option. As usual, you vary data in the figure(s) before writing frames.

14.11 AVI Object

AVI Object
When this flag is set, Fractal Zplot generates single frames in obj format instead of opening a video stream. The 3D object files can be exported into a program such as Bryce, for post-processing into

videos. This works in conjunction with the Demo/Batch mode command, to set the target disk directory and file name, so is only enabled when Batch mode is set.

14.12 AVI WRL

AVI WRL

When this flag is set, Fractal Zplot generates single frames in wrl format instead of opening a video stream. The 3D object files can be exported into a program such as Bryce, for post-processing into videos. This works in conjunction with the Demo/Batch mode command, to set the target disk directory and file name, so is only enabled when Batch mode is set.

15 Demo menu

Demo menu commands

The Demo menu offers the following commands, which illustrate various features of Fractal Zplot:

Random Julia	Generate random Julia fractal.
Random Julia2	Generate random Julia fractal(includes composites).
Random Escher	Generate random Julia fractal using Escher mapping.
Random Newton	Generate random Julia fractal using Newton or Halley's method.
Random Stalks and Bubbles	Generate random orbit-trap or bubble fractal.
Random Quaternion	Generate random quaternion/hypernion fractal.
Random Quaternion2	Generate random quaternion/hypernion fractal(extended formula search).
Random Newton/Halley 3D	Generate random quaternion/hypernion fractal using "cutoff rate" bailout
Random Cubic Mandelbrot	Generate random cubic Mandelbrot fractal
Random Cubic Mandelbrot2	Generate random cubic Mandelbrot(relaxed formulas/parameters)
Random Cubic Julia	Generate random cubic Julia fractal.
Random Octonion	Generate octonion/hyper-octonion fractal.
Random Octonion2	Generate random octonion/hyper-octonion fractal (extended dimensional search).
Random Quat-Trap	Generate random quat-trap fractal.
Random 3D Julia Field	Generate random 3d Julia field.
Random Landscape	Generate random 3D landscape.
Random Lsystem	Generate random lsystem fractal.
Random Orbital	Generate random orbital fractal.
Random Render	Select a random rendering.
Batch Mode	Repeat random fractal and save to file.

15.1 Random Julia

Random Julia (Demo menu)
A random Julia fractal is generated. Many of the built-in options of Fractal Zplot are selected on a random basis, and the Mandelbrot space for one of the hundred built-in formulas is scanned for an interesting Julia set. The palette used is also randomized. Note: In most case the Julia search is a short one, but sometimes the "seek" mode can seem to get stuck when the criteria for an interesting Julia set fails to match the formula used. In the latter case, either click the left mouse button and restart the search process, or pressing a palette key will sometimes override the current search and restart it (if HSV filtering has been selected.) Tip: some things remain to be done after the Julia set is drawn. The algorithm doesn't check the type of split palette that is used, so a mismatch in the "Divide by" color option may need correcting, e.g. Divide-by-four may be selected for a divide-by-eight palette. Feel free to experiment with all the parameters, reframe the image, change palettes etc. This routine provides a fast intro to many options in Fractal Zplot that the user may be unfamiliar with: no knowledge of fractal science/math required! See the hot keys section also for a description of the 'F' command.

15.2 Random Julia2

Random Julia (Demo menu)
A random Julia fractal is generated. Many of the built-in options of Fractal Zplot are selected on a random basis, and the Mandelbrot space for one of the hundred built-in formulas is scanned for an interesting Julia set. The palette used is also randomized. Note: In most case the Julia search is a short one, but sometimes the "seek" mode can seem to get stuck when the criteria for an interesting Julia set fails to match the formula used. In the latter case, either click the left mouse button and restart the search process, or pressing a palette key will sometimes override the current search and restart it (if HSV filtering has been selected.)

This is like the Random Julia command, except that more options are randomized, including spin and switch, and the Formula Type can be composite or Escher, so the search/draw time may be somewhat longer, and the results not as certain. But the images can be quite weird!

Tip: some things remain to be done after the Julia set is drawn. The algorithm doesn't check the type of split palette that is used, so a mismatch in the "Divide by" color option may need correcting, e.g. Divide-by-four may be selected for a divide-by-eight palette. Feel free to experiment with all the parameters, reframe the image, change palettes etc. This routine provides a fast intro to many options in Fractal Zplot that the user may be unfamiliar with: no knowledge of fractal science/math required! See the hot keys section also for a description of the 'F' command.

15.3 Random Escher

Random Escher (Demo menu)
A random Julia fractal is generated using Escher-like tilings (Type 10 in the Formula window.) The Rise variable in the Parameters window sets the degree of tiling (1-235). Note: Sometimes it helps to reduce iterations if the tilings do not appear to be distributed throughout the circular plane. 20-25

iterations is sufficient for many Escher plots.

15.4 Random Newton

Random Newton/Halley (Demo menu)
A random Julia fractal is generated using Newton or Halley's method.

15.5 Random Stalks and Bubbles

Random Stalks and Bubbles (Demo menu)
A random Julia fractal is generated using one of Paul Carlson's orbit traps or bubble method, or a custom orbit-trap formula for Newton's method renderings.

15.6 Random Quaternion

Random Quaternion (Demo menu)
A random quaternion/hypernion fractal is generated. Like Random Julia, a set of formulas appropriate for quaternions is scanned to find an interesting Julia set, and then the parameters are adjusted to produce a quaternion or hypernion image. The ranges are reset, Hj is set to 2.0, and the lighting is set for optimum viewing.

Note: for some images an hj value of 2.0 may result in a partially clipped image. Sometimes it helps to increase this value to 2.5 or 3.0, but too high a value may interfere with Solid guessing.

See the hot keys section also for a description of the 'G' command.

15.7 Random Quaternion2

Random Quaternion2 (Demo menu)
A random quaternion/hypernion fractal is generated. Like Random Julia, a set of formulas appropriate for quaternions is scanned to find an interesting Julia set, and then the parameters are adjusted to produce a quaternion or hypernion image. The ranges are reset, Hj is set to 2.0, and the lighting is set for optimum viewing.

This option uses an extended set of formulas, not all of which may produce useable images all the time. The images can be quite different from the traditional quaternion.

Note: for some images an hj value of 2.0 may result in a partially clipped image. Sometimes it helps to increase this value to 2.5 or 3.0, but too high a value may interfere with Solid guessing.

See the hot keys section also for a description of the 'G' command.

15.8 Random Newton/Halley 3D

A random 3D fractal is generating using Alessandro Rosa's "cutoff rate" method for bailout. This bailout is optimized for formulas that use Newton's or Halley's method, but can be used with escape-time formulas too. The Bailout variable is set to .000001 if it is initially greater than or equal to 1.0. See the description of the Bailout variable in the Quaternion window for further information on the "cutoff rate" algorithm.

15.9 Random Cubic Mandelbrot

Random Cubic Mandelbrot (Demo menu)
A random cubic Mandelbrot fractal is generated. Like the inverse of Random Julia, the essential cubic parameters are randomly adjusted to point into the four-dimensional formula G0, and then the Mandelbrot set is scanned to find an interesting area to zoom into. The ranges are reset, and the lighting is set for optimum viewing.

15.10 Random Cubic Mandelbrot2

Random Cubic Mandelbrot2 (Demo menu)
A random cubic Mandelbrot fractal is generated. Like the inverse of Random Julia, the essential cubic parameters are randomly adjusted to point into the four-dimensional formula G0, and then the Mandelbrot set is scanned to find an interesting area to zoom into. The ranges are reset, and the lighting is set for optimum viewing.

This option uses the cubic formulas G0 and G1, with relaxed parameters to create cubic-Mandelbrot like fractals that may extend to six or more dimensions.

15.11 Random Cubic Julia

Random Cubic Julia (Demo menu)
A random cubic Julia fractal (the Julia analog of a cubic Mandelbrot fractal) is generated. The essential cubic parameters are randomly adjusted to point into a four-dimensional formula. Like Random Julia, a set of formulas (G0 and G1) appropriate for cubic Julias is scanned to find an interesting Julia set, and then the parameters are adjusted to produce the cubic image. The ranges are reset, Hj is set to 2.0, and the lighting is set for optimum viewing. Note: This is a quasi-Julia approximation that doesn't follow traditional cubic Mandelbrot theory. The "quaternions" produced by this method do exhibit characteristics of cubic Mandelbrots, but here I am more interested in esthetics than mathematical conformity.

15.12 Random Octonion

Random Octonion (Demo menu)
A random octonion Julia fractal is generated. The essential octonion parameters are randomly adjusted to point into an eight-dimensional formula H0-H9, and then the octonion Mandelbrot set is scanned to find an interesting area to zoom into. The ranges are reset, and the lighting is set for

optimum viewing.

15.13 Random Octonion2

Random Octonion2 (Demo menu)
A random octonion Julia fractal is generated. The essential octonion parameters are randomly adjusted to point into an eight-dimensional formula H0-H9, and then the octonion Mandelbrot set is scanned to find an interesting area to zoom into. The ranges are reset, and the lighting is set for optimum viewing.

This option uses the octonion formulas H0-H9, with random dimensional switching (one of OE-OK for Oi) to create octonion fractals that may extend to eight dimensions.

15.14 Random Quat-Trap

Random Quat-Trap (Demo menu)
A random quat-trap fractal is generated using one of Paul Carlson's orbit trap methods and a quaternion image. A separate image is created for both quaternion and orbit-trap formulas, in figures 1 and 2, then the figures are merged and drawn as one image. See Orbit-Trap options for further details on quat-trap pictures.

15.15 Random 3D Julia Field

Random 3D Julia Field (Demo menu)
A 3D height field fractal is generated for a random Julia set. Like Random Julia, a set of formulas appropriate for 3D height fields is scanned to find an interesting Julia set, and then the parameters are adjusted to produce a height field image. The ranges are reset and the lighting is set for optimum viewing.

15.16 Random Landscape

Random Landscape (Demo menu)
A random 3D landscape (midpoint displacement fractal -- Formula Type 4) is generated.

15.17 Random Lsystem

Random LSystem (Demo menu)
A random lsystem fractal is generated. An .ls file is selected at random from the startup directory, the palette randomized, and then a random amount of mutation is applied. The lighting is set for optimal viewing, and the figure rotated a random amount on the x y and z-axis.

15.18 Random Orbital

Random Orbital (Demo menu)
A random 3D orbital fractal is generated. The fractal formula is chosen from the 17 built-in orbital formulas. The pixel iteration (n variable) is set to 1 million iterations.

15.19 Random Render

Random Render (Demo menu)
The rendering options for the current fractal are randomized. Does not affect formula or range variables. For quaternion and 3D height fields, a random coloring filter is applied.

15.20 Batch Mode

Batch mode (Demo menu)
Here you set parameters for batching and saving random-generated images to disk. When the Repetitions value is non-zero, up to 1000 random images can be generated and saved to disk. Use a unique Filename to prevent batch files from overwriting existing image files. The Scan Limit directs the program on how many scans it makes through each formula before it skips to a new formula (if an interesting Julia fractal hasn't been found.) For lsystems, the scan limit is a limit on how many mutations on the rules may be done before drawing. There are check boxes that affect only random quaternions or lsystems also.

New: If you select this option before opening a video stream, then instead of an AVI stream, the program initializes a set of bmp image files. Each 'frame' is written to the directory specified in the Demo/Batch mode window. If AVI Object or AVI WRL is checked before opening the stream, then the frames are written as a series of 3D object files. Each frame is numbered with a postfix to the Batch-mode name from 0000 to 9999, e.g. 'quat0001.bmp'.

There are radio boxes that allow you to customize how random variables and menu options are processed to create new 3-D fractals:

Formula -- (default on) check to randomize built-in formula used
Lighting -- (default off) check to set default lighting
Symmetry -- (default off) check to randomize symmetry used
Rotation -- (default on) check to randomize camera angles
Coloring -- (default off) check to reset coloring parameters
Iteration -- (default off) check to randomize iterations
Z-Space -- (default on) check to set default z-space
Constants -- (default on) check to randomize the complex constants cj-cK
Flame -- (default off) check to randomize options for the flame orbital fractal
Genetic -- (default off) check to randomize genetic word strings and 'genes' in Formula window
Torus -- (default off) check to randomize torus option

Orbit Trap -- (default off) check to randomize orbit traps
Filters -- (default off) check to randomize filters
Biomorph -- (default off) check to randomize biomorph option
Converge -- (default off) check to randomize convergence options (Newton, Renormalization or Converge)
Levelcurve -- (default off) check to randomize level curves
Decomposition -- (default off) check to randomize decomposition option
Escape -- (default off) check to randomize escape coloring methods
S/Arglimit -- (default off) check to randomize 's' variable and arglimit variable
Invert -- (default off) check to randomize invert option

For quaternions you have the option of selecting only quaternion types, only hypernion types (hypercomplex) or a mixture of quaternion/hypernion types. For lsystems, you can set the mutation mode to Fixed, to allow a limited number of mutation factors. This results in an lsystem that remains the same number of lines as the original (un-mutated) lsystem. The base lsystem can be chosen at random, or only the current lsystem is mutated over and over (Use the Edit/Axiom window to set the current lsystem.) Each random lsystem starts with the base lsystem, to keep some consistency in the mutated image. You can also control the size of the lsystem (small mutations can create very large lsystems, which can bog down the batch mode) by setting the Max Lines variable to a smaller value (default is 25000.) A good value to use is 5000. Up to 5000 lines of the lsystem will be drawn before moving on to the next lsystem in the batch. Note: the max lines variable isn't saved with the image's data file, so the image can be recreated with any suitable lines limit, when not in batch mode. Many lsystems don't exceed 5000 lines.

16 Help menu

Help menu commands

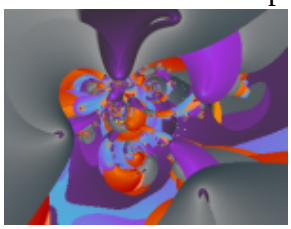
The Help menu offers the following commands, which provide you assistance with this application:

Getting Started	Tutorial for new users of Fractal Zplot.
Index	Offers you an index to topics on which you can get help.
Hot Keys	Quick reference to Fractal Zplot's hot keys.
Parser Info	Quick reference to Fractal Zplot's parser variables and functions.
Built-in Formulas	Quick reference to Fractal Zplot's built-in formulas.
Bibliography	Sources for fractal information and complex numbers.
Lsystem info	Quick reference to Fractal Zplot's lsystem (description and syntax).
About Fractal Zplot	Displays the version number and author info for this application.

16.1 Getting Started

Getting Started

Welcome to Fractal Zplot!



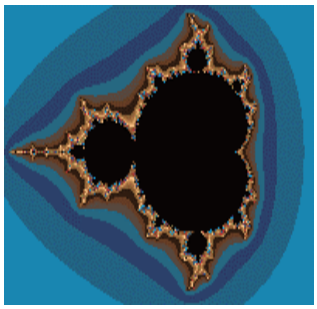
This is a short tutorial that will cover basic commands and background material necessary for a new user to create an initial picture with Fractal Zplot. For help on any menu command, press F1 while the command is highlighted. For help on the Edit Formula or Parameters window, click on the Help button inside that window.

The two methods of fractal composition that most fractal programs use are iterative and orbital. With the iterative method, which Fractal Zplot uses, a formula is successively iterated until some criteria are met then the point is plotted. The process of iteration goes like this: starting from some initial value, a formula is evaluated, and then the result is used to reevaluate the formula. The formula is usually complex (based on complex variables of the form x+yi, where i=sqrt(-1)), the most commonly used formula being z^2+c. For two-dimensional fractals, the screen is mapped into ranges of the variables z or c. When the c variable is changed during iteration, the map produced is called the Mandelbrot set, after Benoit Mandelbrot. The z variable may be initialized at zero or some other value. When the z variable is changed during iteration, the c variable remaining fixed, a Julia set is formed. The Mandelbrot set is frequently considered a map of all Julia sets. If you choose a point for c anywhere on the border of the Mandelbrot set, a Julia set can be generated that looks very similar to the area it was taken from. The formula that came to be known as the Mandelbrot set, z^2+c, is only one of many complex formulas that can be used. Fractal Zplot provides 100 such formulas, with an example picture for each.

Since most fractaliers are initially interested in the Mandelbrot set, we will begin there. We're first going to generate a two-dimensional version of the Mandelbrot set, then redraw it as a 3D plot and sky.

First, use the menu command Image/Reset/All to clear the screen to a two-dimensional plot of the Mandelbrot set. If you click on Okay or Apply a two-dimensional plot of the Mandelbrot set will be redrawn. First click on the thumbnail button to reduce the image size to one quarter of the drawing window. Alternately, you can click to the left of the carat in the Size slider and the size will decrease by 4 with every click. Clicking on an end arrow changes the size by one. This works for all the sliders too. Next, delete the value in the Cutoff/Slope box by clicking in it and using the delete or backspace buttons. Enter 0.00039 in the Cutoff box. For 2D plots the cutoff value acts as a color multiplier or divider on the entire palette. This speeds up or slows down changes in color. With a value of .00039 the palette is cycled through ten times the normal rate, so stripes of color are quite noticeable on a Mandelbrot plot with escape-time coloring.

Now click on Apply. Fractal Zplot's main window will be erased and a small plot of the Mandelbrot set will be drawn in the center of the drawing window.



Go back to the Parameters window now and move the Sector to a value of 2. Press Okay. The right upper quarter of the window will be erased and the same Mandelbrot set will be drawn there. You can experiment with sector values 1-4 also. Each sector value puts the plot in a different quarter of the window. This works as long as the Size value is one-half of the full-screen horizontal size or less. A size above that forces Fractal Zplot to plot a 2D picture in the center of the screen and change the sector to 0. You want to use sectors and smaller plot sizes to save time and when zooming out or framing. You may have noticed that only the sector that is being drawn into is erased before drawing. This allows you to draw a different plot in all four sectors. You can prevent accidental erasures of the screen entirely by deselecting the auto-clear option in the Image menu. This must be done also when creating composite 3D drawings.

Change the size to anything above the full-size screen resolution, and then click on Okay. Fractal Zplot limits a 2D plot size to full-screen resolution so a full-screen rendition of the Mandelbrot set will be drawn. Clicking the mouse button inside the plot zone or pressing a key (other than the hot keys) stops the plot. You can continue the plot later by using the Continue command off the Image menu, or the Continue button in the Parameters window. Some commands disable the continue feature, as when changing to Solid-Guessing or Tesseral as a plot mode. These use a different plotting mode, incompatible with the default single-pass mode in Fractal Zplot.

If the Auto-Alert option is enabled (on by default), Fractal Zplot will give an audible sound when the plot is completed. The sound may be customized via the Sounds editor in the Windows 95 control panel. The plot-complete alert uses the Exclamation sound.

You'll probably be doing a lot of zooming and framing on your plots later, so we'll cover that briefly here. After the full-screen version of the Mandelbrot set is finished (or as much of it is finished that you want to zoom in on), select the Zoom In/Out command off the Image menu, or just point and click the left-mouse button over any area of the drawing. A box a quarter the size of the window will appear that you can move around with the mouse. Hold the left-mouse button down to shrink the box, or the right-mouse button down to expand the box. Move the box over the area you are zooming in on, size the box if necessary and when it includes the details you want, press the space bar. The plot will be redrawn at zoom scale. To zoom out, you need to draw a smaller size plot, then zoom using a box larger than the plot drawn. You can also rotate the zoom box by using the left and right arrow keys. This effectively rotates the area being zoomed into in the reverse direction as the zoom box is rotated. When rotating the zoom box, think of the final image as being a horizontal version of the image in the zoom box.

Drawing a 3D plot: load the title_old picture by using the Load button on the Remote, and double

click on title_old.zp6. Click on the autoclear option in the Image menu to turn that option off. Click on the title bar of the draw window and use the Clear Screen command, shift-C to erase the screen. Now use the Remote's Draw button to redraw the 3D part of the title picture. The picture starts drawing at the bottom left corner of the main window. Initially nothing will be present at the bottom left corner of the window, as the steepness of Mandelbrot mountain is built up off-screen. Fractal Zplot uses a 3D projection method similar to Fractint's height fields. Fractal Zplot has the capability to produce seamless 3D pictures by cropping the plot at the corners and adding a horizon at the top of the plot.

Next, change the image to Image/Figure 2. Use the Draw command to redraw the chaos moon. (It is drawn invisibly then displayed in whole.) This is actually another quaternion figure with added noise for surface texture.

The last part of the title picture involves redrawing the random sky background. Change the image to Image/Figure 3. Click on the Continue icon on the toolbar to redraw the sky. Fractal Zplot allows you to redraw random backgrounds by saving the random seed that produced them. You can erase the 3D background separately from the 3D height field by using the shift-B command. This allows you to change the background after saving a composite 3D plot, without having to redraw the height field too. When you use one of the hot keys like shift-B be sure that the draw window has the focus. The title bar should not be "grayed".

The second part of this tutorial involves creating Julia sets based on points inside a Mandelbrot set. There are three ways to generate Julia sets using Fractal Zplot:

The manual way involves entering the complex constant values for a known Julia set into the complex c boxes in the Parameters window.

The semi-automatic way uses the "P" command, or keyboard hotkey. Reset the Figure (Using Image/Reset Figure or Reset All) and draw a Mandelbrot set using MandelbrotP or Mandelbrot0 on the Type menu. Zoom into it until you find an area that would make an interesting Julia set. Press shift-P (Caps Lock off), and the cursor will change to a crosshatch. Position the cross hatch over the area of the Mandelbrot set you want to use as a starting basis for the Julia set. Click the left mouse button, and the pixel's coordinates will be entered into the complex c boxes as above. Use the Reset Ranges Only command on the Image menu to reset the Z ranges to full-scale. Change the type to Julia and redraw the picture. You should see a Julia set that closely resembles the area you picked from the Mandelbrot set.

The third way to locate and generate Julia sets uses the hotkey "J". With this command, a small copy of the Julia set is immediately drawn in the second sector of the window each time you click on an area of the Mandelbrot set. When you find an interesting Julia set, click on the title bar or outside the window to exit this mode. A new window will be opened and automatically set the parameters to those necessary to recreate the "found" Julia set. (The parameters in the original window are unchanged.)

If you just want to find interesting Julia sets, click on the Remote's Scan button and the program will do the searching for you and draw a random Julia set.

Special note: As you explore the many options included in Fractal Zplot you'll find that many of the variable windows are non-modal, so they can stay open while the fractal is being plotted. This allows you to change some coloring and lighting variables without redrawing the fractal, or repeatedly experiment with other aspects of the fractal-design process. All of the non-modal windows have an Apply button for applying changes directly without closing the window, or an Okay button for applying changes and closing the window. To close the window without making any further changes, click on the window's close button. The Cancel button, if present, allows you to revert to when the window was last opened. Some commands external to the window may cause it to close and reopen if variables were changed externally. In this case Cancel "goes back" to after the window was reopened.

Fractal Zplot allows you to Undo the last command in most cases. However this is mostly a failsafe command, as it disables color-cycling and requires you to redraw the fractal to change colors or lighting variables.

This completes the Getting Started tutorial. Be sure to read the hot keys and built-in formulas sections for additional info. The Bibliography lists additional reference material for a better understanding of the fractal types and functions contained in Fractal Zplot.

16.2 Index

Index command (Help menu)

Use this command to display the opening screen of Help. From the opening screen, you can jump to step-by-step instructions for using Fractal Zplot and various types of reference information.

Once you open Help, you can click the Contents button whenever you want to return to the opening screen.

16.3 Hot Keys

Hot keys

Ctrl+F1-Ctrl+F9,Ctrl+F11, Ctrl+0-Ctrl+9 --- change to one of 21 color palettes -- useable during plotting.
Ctrl+F12 holds the palette of the most recently loaded function.

Tab --- Replaces the currently-selected palette with the palette in F11.
Useful when you want to make a palette file(.pl) from the palettes in a lot of individual bitmap files. Use the copy data and paste data commands to move the palette from another drawing window into F11. Select the palette(Ctrl+F1-Ctrl+F9,Ctrl+F12,Ctrl+0-Ctrl+9) you want to move Ctrl+F11 into, then press Tab.
Hint: you can copy data from and to the same window, to move the currently selected palette into Ctrl+F11.

up arrow --- forward cycle colors one step, including set color -- useable during plotting.
down arrow --- back cycle colors one step, including set color -- useable during plotting.

Shift-B -- erases the 3D background, leaving the 3D plot intact.

Shift-C -- clear the screen to the current background color.

Shift-P --- grab point from Mandelbrot set(real and imaginary parts) and put values in complex constant. Cursor changes to a cross-hatch, which you position over the area of the Mandelbrot set of interest. Then click the left-mouse button to transfer the pixel's coordinates to the c constant. Click outside window or in window frame to exit routine without "grabbing" a point.

Shift-J -- like "P", except that a Julia set is drawn immediately in sector 2 at size 100 and with iterations of 100. This is a fast exploratory routine for finding interesting Julia sets that can also be used where the Mandelbrot set is discontinuous as in the Phoenix formula. Unlike the "P" command, "grabbing,"(and drawing) continues until you click on the window's frame. Note: when you exit the J command, once you find an interesting Julia set; another window is opened with the Julia type set. The parameters in the original window revert to their original Mandelbrot settings.

Shift-G -- like "J", except that a quaternion set is generated in sector 2, using an iteration count of 10 and other parameters are changed temporarily to suit quaternion plots. (Hj is set to 2.0, the rotational variables are reset and the light source is set to -1, 1 and -3.) Quaternion math is used where possible. Once you find an interesting quaternion set using "G", like the J command another window is opened that sets the fractal parameters to those in the exploratory qjulia window (fast qjulia mode.) The parameters in the exploratory window revert to their original Mandelbrot settings.

Shift-W -- like "G" except this modifies the cj and ck elements of the complex constant to locate a quaternion Julia set.

Shift-S -- a combination of the "G" and "W" commands. The "W" command is implemented with the right mouse button.

Shift L: -- Like 'G' except this modifies the ci and cj elements of the complex constant to locate a quaternion Julia set. This complements the loxodromic formula (L0) that requires a non-zero cj for best results. Cr should be close to zero to maintain symmetry in the loxodromic figure.

Shift-U -- like "J" and "G" except that you choose the target type and beginning parameters. This is useful for exploring cubic Mandelbrot sets, insofar as you can scan the complex c planes for interesting variations. Also works for other multi-dimensional Mandelbrot formulas that are dependent on the c planes. Hint: turn off the Image/Auto-Redraw option after drawing the mapping picture. You can then set the target set (Mandelbrot, quaternion, etc.) before using this key sequence, without erasing the mapping picture.

Shift-D -- use the mouse to examine pixel depth in a drawing. By clicking with the left-mouse button on any area in the current fractal, the pixel's depth and bailout values are displayed in the status bar,

along with the pixel's coordinates. Useful to locate Mandelbrot islands while zooming on an image. Click on the title bar or press Esc to exit this command.

Shift-F -- generate a Julia set from a formula's MandelbrotP space. Random points in a formula's current Mandelbrot space are scanned for an interesting Julia.

Shift-Z -- zoom in/out coordinates. Like the menu command except does not immediately redraw the picture. This allows you to zoom into another screen sector without erasing the previous picture.

Shift-T -- annotate a picture with text. Cursor changes to a crosshatch, which you position over the area where you want the text to start. Then click the left-mouse button to transfer any text (from the Edit/Text window) to the picture. Can be used with Undo. Use the Edit/Text command to change font, text color or format text into multiple lines. This is useful for adding copyright/author info to a finished picture.

16.4 Parser

Parser Information

Functions (capital letters are optional, and parenthesis are necessary around complex expressions)

The following information takes the form "standard function" ---"form used by Fractal Zplot to represent standard function".

sine z --- sin(z) or SIN(Z) ; where Z can be any complex expression
hyperbolic sine z --- sinh(z) or SINH(Z)
arcsine z --- asin(z) or ASIN(Z)
cosine z --- cos(z) or COS(Z)
hyperbolic cosine z --- cosh(z) or COSH(Z)
arccosine z --- acos(z) or ACOS(Z)
tangent z --- tan(z) or TAN(Z)
hyperbolic tangent z --- tanh(z) or TANH(Z)
arctangent z --- atan(z_) or ATAN(Z)
cotangent z --- cotan(z) or COTAN(Z)
arccotangent z --- acotan(z) or ACOTAN(Z)
e^z --- exp(z) or EXP(z) -- the exponential function
natural log of z --- log(z) or LOG(Z)
absolute value of z --- abs(z) or ABS(Z)
square root of z --- sqrt(z) or SQRT(Z)
z squared --- sqr(z) or SQR(Z)
real part of z --- real(z) or REAL(Z)
imaginary part of z --- imag(z) or IMAG(Z)
modulus of z --- mod(z) or MOD(Z) or |z| -- (x*x + y*y)
conjugate of z -- conj(z) or CONJ(z) -- (x-yi)
flip(z) --- flip(z) or FLIP(Z) -- exchange real and imaginary parts of z (y+xi)

polar angle of z -- theta(z)

if/then/endif – if(argument), then (phrase) endif -- if argument is true then do phrase else skip phrase ('then' tag is optional, but a comma should follow argument or put 'if(argument)' on separate line)

if/then/else/endif - if(argument), then (phrase) else (phrase) endif -- if argument is true then do phrase else skip phrase and do alternate phrase('then' tag is optional, but a comma should follow argument or put 'if(argument)' on separate line)

Note: if/then/endif and if/then/else/endif loops can be nested only when endifs follow each other at the end of the loops. For example: if(argument) if(argument) then (phrase) endif endif.

Math operators

+ --- addition
- --- subtraction
* --- multiplication
/ --- division
^ --- power function
< --- less than
<= --- less than or equal to
> --- greater than
>= --- greater than or equal to
!= --- not equal to
== --- equal to
|| --- logical or (if arg1 is TRUE(1) or arg2 is TRUE)
&& --- logical and (if arg1 is TRUE and arg2 is TRUE)

Constants and variables

complex constant --- c# or C#, read/write.
complex conjugate --- cc# or CC#, read-only.
convergence limit --- cl# or CL# -- the constant entered in the Converge gadget, read-only.
cr -- the constant entered in the cr box in the Parameters window(use j# for parser)
ci -- the constant entered in the ci box in the Parameters window(use k# for parser)
e --- e or E -- 1e^1 -- 2.71828, read/write.
i --- i or I -- square root of -1,read/write.
iteration --- iter# -- iteration loop counter
j --- j# or J# -- real part of the complex constant, read-only.
k --- k# or K# -- coefficient of the imaginary part of the complex constant, read-only.
Note: j and k are the actual values of the complex constant terms as they are used in the iteration process, so will vary when the Mandelbrot option is used.
m --- m# or M# or pixel --a complex variable mapped to the pixel location as defined by the z coordinates entered in the Parameters window, read/write.
maxit -- the maximum number of iterations, as set in the Parameters window, read only
p --- p# or P# -- real constant used in phoenix maps; uses the real part of the complex constant

when the Phoenix option is chosen, read-only.
p1 – the complex constant entered in the cr and ci gadgets, read-only.
pi --- pi or PI -- 3.14159, read/write.
q --- q# or Q# -- real constant used in phoenix maps; uses the imaginary part of the complex constant when the Phoenix option is chosen, read-only
x --- x# or X# -- real part of Z, read/write.

y --- y# or Y# -- coefficient of the imaginary part of Z, read/write.

z --- z or Z -- function value at any stage of the iteration process, read/write.
zn# or ZN# -- the value of z at the previous stage of iteration, read-only.

16.5 Built-in Formulas

Built-in Formulas (enter the following prefix into the Function #1 or Function #2 edit boxes)

a0 -- spiral network -- C. Pickover.
a1 -- z-(1/z+c) -- try with renormalization applied. Sequel to t9.
a2 -- fn(z)-(fn(z)+c) -- generalized form of a1.
a3 -- alternate Newton/Halley map of sinz-c. see t3 variant.
a4 -- user-defined complex set: fn(z)+fn(z)+c.
a5 -- hypercomplex Newton/Halley map of h^3+c.
a6 -- Hypercomplex Newton/Halley map of fn(h)+c.
a7 -- user-defined complex set: fn(z)+fn(c).
a8 -- fn(z)+zn+c -- from Fractal Creations.
a9 -- q^3+c -- cubic Quaternion set.

b0 -- alternate Newton/Halley map of exp(z)-c. as for t3 variant.
b1 -- alternate Newton/Halley map of log(z)-c.
b2 -- Newton/Halley map of phoenix curve.
b3 -- cfn(z)+zn -- user-defined complex formula.
b4 -- fn(z)+kzn+j -- generalized phoenix curve formula.
b5 -- fn(z) a preformula for use with type 3 composite fractals. uses limit gadget to select function.
b6 -- Newton/Halley map of fn(z)+fn(z)+c.
b7 -- Newton/Halley map of cfn(z).
b8 -- fn(z)*fn(z)+c.
b9 -- Newton/Halley map of foggy coastline #1.

c0 -- Newton/Halley map of foggy coastline #2.
c1 -- Newton/Halley map of fn(fn(z))+c.
c2 -- cfn'(z), where fn'(z)=first derivative of user-defined function.
c3 -- fn(z)+fn'(z)+c.
c4 -- fn'(z)+fn(c).
c5 -- fn(fn'(z))).
c6 -- first order gamma function: (z/e)^z*sqr(2*pi*z)+c.

c7 -- Newton/Halley map of fifth degree Legendre polynomial: 1/8(63z^5-70z^3+15z; display method 2 default.
c8 -- (z^2+e^(-z))/(z+1): second-order convergence formula for finding root of ze^z-1=0.
c9 -- Newton/Halley map of fn(z)*fn(z)+c.

d0 -- z^s/limit+c: anti-derivative of z^n; s may be complex using the si variable as its imaginary component.
d1 -- Sterling expansion of gamma function: (z/e)^z*sqr(2*pi/z)+c.
d2 -- Newton map of fn'(z)-fn(z)+c: generalized first degree Laguerre polynomial. Newton map only. (Note 13).
d3 -- fn(1/(fn(z)+c)).
d4 -- z^2-c; where zreal=abs(zreal)(Paul Carlson's "alien" Julia set).
d5 -- z^2; where zreal=abs(zreal)-cr, zimag=zimag-ci (Paul Carlson Julia set).
d6 -- cfn(z)+c.
d7 -- Newton's method applied to (x^3+y^2-cr=0 and y^3-x^2+ci=0). Newton map only. from Sylvie Gallet and Fract19.par.
d8 -- Newton's method applied to fn1(x)+fn2(y)-cr=0 and fn3-fn4+ci=0 (Note 15) Newton map only.
d9 -- Bill13 from Bill Rossi via the Internet.

e0 -- solves Newton/Halley transformation of (z+j)(z+k)(z^2+1) for either Julia or Mandelbrot set.
e1 -- solves Newton/Halley transformation of (z+j)(z^2+z+k) for Mandelbrot and Julia set.
e2 -- solves Newton/Halley transformation of (z-1)(z^2+z+c) for either Julia or Mandelbrot set.
e3 -- solves Newton/Halley transformation of (z+j)(z+k)(z+1) for either Julia or Mandelbrot set.
e4 -- Chaos Game Julia IFS (M. Barnsley).
e5 -- snowflake Julia IFS (as described in Fractals Everywhere by M. Barnsley).
e6 -- solves Newton/Halley transformation of logz-c.
e7 -- solves Newton/Halley transformation of exp(z)-c.
e8 -- solves Newton/Halley transformation of (z-c)(z+1)(z-1) for Mandelbrot or Julia set.
e9 -- solves Newton/Halley transformation of (z-c)(z+c)(z^2+c^2) --- z^4-c^4.

f0 -- generalized form of Earl Hinrichs' sophomore sine function(ssin) -- limit*fn(z)+s+si, where fn(z) =(fn1(x),fn2(y)).
f1 -- c^2/(1-cz^2) -- variant of p4.
f2 -- Gallet-4-01, from Sylvie Gallet's extensive Internet collection (Note 16).
f3 -- Gallet-4-02, from Sylvie Gallet.
f4 -- Gallet-6-01, from Sylvie Gallet.
f5 -- Gallet-6-02, from Sylvie Gallet.
f6 -- Gallet-6-03, from Sylvie Gallet.
f7 -- Gallet-6-04, from Sylvie Gallet.
f8 -- Gallet-6-05, from Sylvie Gallet.
f9 -- Gallet-7-01, from Sylvie Gallet.

g0 -- z^3-3c^2z+s -- cubic Mandelbrot (Note 17)
g1 -- z^3-3sz+c -- alternate cubic Mandelbrot

g2 -- z^3-3c^2z^2+s -- cubic Mandelbrot variant
g3 -- z^3-3sz^2+c -- alternate cubic Mandelbrot variant
g4 -- z^3-3c^2/z+s -- cubic Mandelbrot variant
g5 -- z^3-3s/z+c -- alternate cubic Mandelbrot variant
g6 -- z^3-3c^3*z+s -- cubic Mandelbrot variant
g7 -- z^3-3s/z^2+c -- alternate cubic Mandelbrot variant
g8 -- z^3-3c^2+z+s -- cubic Mandelbrot variant
g9 -- z^3-3s+z+c -- alternate cubic Mandelbrot variant

h0 -- (O*C^-1)(C*O)+c -- octonion set (Note 4)
h1 -- cO*CC(1-C*O) -- octonion set
h2 -- O*C(O-CC*O)+c -- octonion set
h3 -- cO^2+O^2*CC-c -- octonion set
h4 -- O^2*CC+O^2*C+c -- octonion set
h5 -- O^3*CC+c -- octonion set
h6 -- O^3*CC+O^3*C+c -- octonion set
h7 -- O^4*CC+c -- octonion set
h8 -- cO^3*CC+c -- octonion set
h9 -- O^2*CC+O^3*C+c -- octonion set

i0 -- 2*z*c#*cos(pi/z) -- Godwin Vickers
i1 -- 2*z*c#*sin(pi/z) -- Godwin Vickers
i2 -- 2*z*c#*tan(pi/z) -- Godwin Vickers
i3 -- 1/z^3+sin(z)*c^2
i4 -- cosh(z)/c*z+c^2
i5 -- exp(z)/z^3-c
i6 -- cosh(z)*z^2-c^2
i7 -- 1/z^2-cz-c
i8 -- tan(z)-czc
i9 -- (tan(z)-1/z^3)/c^2

j0 -- cosh(z)*cos(z)+1/c
j1 -- z^4/(z^3-c^2)
j2 -- 1/z^2-tan(z)+c
j3 -- tan(z)/z^3+c
j4 -- 1/z^3-cz+1/c
j5 -- z^3+tan(z)*c
j6 -- 1/z^3-tan(z)-c
j7 -- tan(z)-sin(z)+c
j8 -- 1/z^2-tan(z)+1/c
j9 -- z^4+sin(z)/c

k0 -- Mandelbrot set(sine variation)
k1 -- z^1.5+c -- Godwin Vickers
k2 -- (z^2-z^(2-s))/s+c -- Escher set by Roger Bagula

k3 -- z^2+z/(|z|+c) -- Roger Bagula
k4 -- quantum set -- S.M. Ulam
k5 -- prey predator #1 -- Roger Bagula
k6 -- prey predator #2 -- Roger Bagula
k7 -- Klein group #1 -- Roger Bagula
k8 -- Klein group #2 -- Roger Bagula
k9 -- Klein group #3 -- Roger Bagula

L0 -- Loxodromic by Thomas Kromer (fixed type)
L1 -- squared loxodrome
L2 -- Gedatou by Thomas Kroner (fixed type)
L3 -- Ventri by Thomas Kroner (fixed type)
L4 -- squared gedatou
L5 -- fn(z)-cfn(z) (Note 5)
L6 -- fn(z)+fn(z)+c"
L7 -- cfn(z)+c"
L8 -- fn(z)+cfn(z)+1"
L9 -- fn(z)+c

m0 -- fn(fn(z))+c
m1 -- fn(z)+fn(c)
m2 -- fn(z)+zn+c
m3 -- cfn(z)+zn
m4 -- fn(z)+kzn+j -- generalized phoenix curve
m5 -- fn(z)*fn(z)+c
m6 -- fn(1/(fn(z)+c))
m7 -- (1/fn(z))^2+c
m8 -- (1/fn(z))^3+c
m9 -- fn(z)/(1-fn(z))+c

n0 -- Sinus by Thomas Kromer (fixed type)
n1 -- Sinus #2 by Thomas Kromer (fixed type) (Note 18)
n2 -- Rings of Fire by Thomas Kromer (fixed type) (Note 18)
n3 -- Teres by Thomas Kromer (fixed type)
n4 -- z^2+y+c
n5 -- z^2+y[n+1]+c
n6 -- z^2+zi+c
n7 -- z^2+zi[n+1]+c
n8 -- zr^2+3zi+c
n9 -- zr^2+4zi+c

o0; z^2+timewave[n]+c -- Note 7
o1; Loxodromic #2 by Thomas Kromer
o2; Loxodromic #3 by Thomas Kromer
o3; Sinus #4 by Thomas Kromer

o4; Sinus #5 by Thomas Kromer
o5; Marmor by Thomas Kromer
o6; Armor by Thomas Kromer
o7; Three-Parameter Julia set by Kenneth Gustavsson
o8; Three-Parameter Julia set #2 by Kenneth Gustavsson
o9; Three-Parameter Cubic Julia set -- Kenneth Gustavsson

p0 -- z^2+c --- the standard Mandelbrot or Julia set.
p1 -- cz(1-z) --- the self-squared dragon set.
p2 -- c(z-1/z) --- alternate Mandelbrot or Julia set.
p3 -- cz^2-1 --- alternate Mandelbrot or Julia set.
p4 -- c^2/(c+z^2) --- alternate Mandelbrot or Julia set.
p5 -- z^3+c --- cubic Mandelbrot or Julia set.
p6 -- ((z^2+c-1)/(2z+c-2))^2 -- renormalization formula #1 for x-plane or q-plane pictures (Note 9).
p7 -- z^2+j+kzn --- Phoenix curve (Ushiki). Uses Fractint extensions for degree(>2 or <-3.).
p8 -- Julia/Mandelbrot set (modified from M. Barnsley) (Note 2).
p9 -- fn(z)-cfn(z) -- generalized frothy basin (J. Alexander.) (Note 7).

q0 -- (2z^3+c)/(3z^2)
q1 -- (2z^3+c)/(3z^2+1)
q2 -- Newton's method applied to exp(z)-c
q3 -- Newton method applied to sin^2z-c
q4 -- Newton's method applied to sin3z-c
q5 -- Newton's metod applied to sinz+cosz-c
q6 -- Newton's method applied to exp(z)sinz-c
q7 -- Newton's method applioed to exp(sinz)-c
q8 -- Newton's method applied to fn(z)+c
q9 -- Newton's method applied to fn(z)+fn(z)+c

r0 -- Newton/Halley map of z^3+conj(z)c -- exploratory function based on modified frothy basin.
r1 -- z^z+z^s+c --- Biomorphs, etc.; s may be complex using the si variable as its imaginary component.
r2 -- z^s-z+c --- Biomorphs, etc.; s may be complex using the si variable as its imaginary component.
r3 -- fn(z)+exp(z)+c -- Biomorphs, etc.
r4 -- solves Newton/Halley transformation of (z^2-c)(z+1). (Notes 3,4,5,11).
r5 -- cfn(z) -- transcendental Julia curve, etc.
r6 -- cexp(z) -- exponential Julia curve, etc. with additional plane checking when real value of Z exceeds 50. If cos(imag-Z) >= 0, point is considered part of Julia set.
r7 -- fn(z)+cfn(z)+1 -- generalized form of t9.
r8 -- foggy coastline #1 Mandelbrot IFS (M. Barnsley) (Note 14).
r9 -- foggy coastline #2 Mandelbrot IFS (M. Barnsley) (Note 14).

s0 -- solves Newton/Halley transformation of sinz-c.

s1 -- sexpz+c -- transcendental Mandelbrot or Julia set.
s2 -- c(1+z^2)^2/(z^2-1) -- alternate Mandelbrot/Julia set.
s3 -- solves Newton/Halley transform of tan(z)-c .
s4 -- IFS (x=sy+j,y=-sx+k (x>0); else x=sy-j,y=-sx-k (modified from M. Barnsley) (An alternate version of this formula is executed when the limit value is non-integral.)
s5 -- solves Newton/Halley transform of z^s-1 (Julia set only).
s6 -- composite function cz-c/z & z^2+c (C. Pickover).
s7 -- transcendental function fn(z)+c .
s8 -- ((z^3+3(c-1)z+(c-1)(c-2))/(3z^2+3(c-2)z+c^2-3c+3))^2 -- renormalization formula #2 for x-plane or q-plane pictures .
s9 -- Newton/Halley map of z(z^limit-1) (Julia set only).

t0 -- Newton/Halley map of z(z^limit-c) (Julia or Mandelbrot set ; display method 2 default; limit>=1.0 ; use 1.0 for initial z with Mandelbrot0 type, or use MandelbrotP type.).
t1 -- Newton/Halley map of Chebyshev function cos(n*arccos x).
t2 -- Newton/Halley map of Hermite polynomial: 16x^4-48x^2+12.
t3 -- alternate Newton/Halley map of tanz-c (Julia or Mandelbrot set.) the twist is in the second derivative of the Halley type.
t4 -- Newton/Halley map of z^limit-c (Julia or Mandelbrot set ; display method 2 default; limit>=1.0 ; use 1.0 for initial z with Mandelbrot0 type, or use MandelbrotP type.).
t5 -- fn(fn(z))+c -- user-defined complex set. When the first function is z^2 and the second function is conj(z), this becomes the z-conjugate set, zz^2+c, the tricorn set. (Note 12).
t6 -- Volterra-Lotka equations discretized by modified Huen method (from The Beauty of Fractals).
t7 -- c^z+c -- tetration of z.
t8 -- q^2+c -- Quaternion set (from Computer, Pattern, Chaos and Beauty) (Note 8).
t9 -- z+cz+1 --try with Newton's method applied. A buggy algorithm found this one.

u0 -- Mandelbulb (sine-based non-trig version) -- Note 8
u1 -- Mandelbulb (cosine-based non-trig version)
u2 -- Mandelbulb (sine-based trig version)
u3 -- Mandelbulb (cosine-based trig version)

QuaSZ formulas
a0 -- spiral network -- C. Pickover
a1 -- z+z^3/c+c
a2 -- tan(z)-(z^2+c)
a3 -- z^2+arcsinz-c
a4 -- cos(z)+csc(z)+c
a5 -- cos(z^3)+c
a6 -- z^7+c
a7 -- sin(z)+c^3
a8 -- z^3+zn+c
a9 -- Quaternion set -- q^3+c^3

b0 -- 1/z^2+exp(z)-c

b1 -- 1/z^3+logz-c
b2 -- z^3+j+kzn
b3 -- cz^2+zn+c
b4 -- sin(z)+kzn+j
b5 -- vers(z)+c
b6 -- vers(z)+covers(z)+c
b7 -- c*vers(z)+c
b8 -- vers(z)*covers(z)+c
b9 -- (foggy coastline #1)^2+c -- Barnsley

c0 -- (foggy coastline #2)^2+c -- Barnsley
c1 -- sin(vers)+c
c2 -- csec(z^2)+c
c3 -- z^3+sech(z^2)+c
c4 -- c*z^(c-1)+z^2
c5 -- cos(-1/w^2)+c
c6 -- (z/e)^z*sqr(2*pi*z)+c
c7 -- 1/8(63z^5-70z^3+15z)+c
c8 -- (z^2+e^(-z))/(z+1)+c
c9 -- z^2*exp(z)+c

d0 -- (z^3)/c+c
d1 -- z^3*sqr(2*pi/z)+c
d2 -- z^2-csc(h)cot(h)+c
d3 -- (1/(sin(z)+c))^3
d4 -- z^2-c ; where x=abs(real(z)), y=imag(z) -- Paul Carlson
d5 -- z^2 ; where x=abs(real(z))-cr, y=imag(z)-ci -- Paul Carlson
d6 -- c*cos(z)+c
d7 -- z*cos(z)+c
d8 -- z*2+z/sin(z)+c
d9 -- z^2+z^pi+c

e0 -- (z+j)(z+k)(z^2+1)+c
e1 -- (z+j)(z^2+z+k)+c
e2 -- (z-1)(z^2+z+c)
e3 -- (z+j)(z+k)(z+1)+c
e4 -- chaos formula -- Barnsley
e5 -- snowflake IFS -- Barnsley
e6 -- cos(z)+c
e7 -- z^3+exp(z)+c
e8 -- (z-c)(z+1)(z-1)+c
e9 -- z^4-c^4

f0 -- z^2+z^3+sin(c)+cos(c)+c
f1 -- z^2*(1-cz^2)+c

f2 -- 1/(z*z/c)+c
f3 -- 1/(z*z)-z+c
f4 -- (1+c)/(z*z)-z
f5 -- (1+c)/(z*z/c)+c
f6 -- (1/c)/(z*z/c)+c
f7 -- 1/((z*z)/c)+(c/(1/z-c))
f8 -- 1/(z*z*z/c)+c
f9 -- 1/(z*z*z)-z+c

g0 -- z^3-3c^2z+s -- cubic Mandelbrot (Note 3)
g1 -- z^3-3sz+c -- alternate cubic Mandelbrot
g2 -- z^3-3c^2z^2+s -- cubic Mandelbrot variant
g3 -- z^3-3sz^2+c -- alternate cubic Mandelbrot variant
g4 -- z^3-3c^2/z+s -- cubic Mandelbrot variant
g5 -- z^3-3s/z+c -- alternate cubic Mandelbrot variant
g6 -- z^3-3c^3*z+s -- cubic Mandelbrot variant
g7 -- z^3-3s/z^2+c -- alternate cubic Mandelbrot variant
g8 -- z^3-3c^2+z+s -- cubic Mandelbrot variant
g9 -- z^3-3s+z+c -- alternate cubic Mandelbrot variant

h0 -- (O*C^-1)(C*O)+c -- octonion set (Note 4)
h1 -- cO*CC(1-C*O) -- octonion set
h2 -- O*C(O-CC*O)+c -- octonion set
h3 -- cO^2+O^2*CC-c -- octonion set
h4 -- O^2*CC+O^2*C+c -- octonion set
h5 -- O^3*CC+c -- octonion set
h6 -- O^3*CC+O^3*C+c -- octonion set
h7 -- O^4*CC+c -- octonion set
h8 -- cO^3*CC+c -- octonion set
h9 -- O^2*CC+O^3*C+c -- octonion set

i0 -- 2*z*c#*cos(pi/z) -- Godwin Vickers
i1 -- 2*z*c#*sin(pi/z) -- Godwin Vickers
i2 -- 2*z*c#*tan(pi/z) -- Godwin Vickers
i3 -- 1/z^3+sin(z)*c^2
i4 -- cosh(z)/c*z+c^2
i5 -- exp(z)/z^3-c
i6 -- cosh(z)*z^2-c^2
i7 -- 1/z^2-cz-c
i8 -- tan(z)-czc
i9 -- (tan(z)-1/z^3)/c^2

j0 -- cosh(z)*cos(z)+1/c
j1 -- z^4/(z^3-c^2)
j2 -- 1/z^2-tan(z)+c

j3 -- tan(z)/z^3+c
j4 -- 1/z^3-cz+1/c
j5 -- z^3+tan(z)*c
j6 -- 1/z^3-tan(z)-c
j7 -- tan(z)-sin(z)+c
j8 -- 1/z^2-tan(z)+1/c
j9 -- z^4+sin(z)/c

k0 -- Mandelbrot set(sine variation)
k1 -- z^1.5+c -- Godwin Vickers
k2 -- (z^2-z^(2-s))/s+c -- Escher set by Roger Bagula
k3 -- z^2+z/(|z|+c) -- Roger Bagula
k4 -- quantum set -- S.M. Ulam
k5 -- prey predator #1 -- Roger Bagula
k6 -- prey predator #2 -- Roger Bagula
k7 -- Klein group #1 -- Roger Bagula
k8 -- Klein group #2 -- Roger Bagula
k9 -- Klein group #3 -- Roger Bagula

L0 -- Loxodromic by Thomas Kromer (fixed type)
L1 -- squared loxodrome
L2 -- Gedatou by Thomas Kroner (fixed type)
L3 -- Ventri by Thomas Kroner (fixed type)
L4 -- squared gedatou
L5 -- fn(z)-cfn(z) (Note 5)
L6 -- fn(z)+fn(z)+c"
L7 -- cfn(z)+c"
L8 -- fn(z)+cfn(z)+1
L9 -- fn(z)+c

m0 -- fn(fn(z))+c
m1 -- fn(z)+fn(c)
m2 -- fn(z)+zn+c
m3 -- cfn(z)+zn
m4 -- fn(z)+kzn+j -- generalized phoenix curve
m5 -- fn(z)*fn(z)+c
m6 -- fn(1/(fn(z)+c))
m7 -- (1/fn(z))^2+c
m8 -- (1/fn(z))^3+c
m9 -- fn(z)/(1-fn(z))+c

n0 -- Sinus by Thomas Kromer (fixed type)
n1 -- Sinus #2 by Thomas Kromer (fixed type) (Note 6)
n2 -- Rings of Fire by Thomas Kromer (fixed type) (Note 6)
n3 -- Teres by Thomas Kromer (fixed type)

n4 -- z^2+y+c
n5 -- z^2+y[n+1]+c
n6 -- z^2+zi+c
n7 -- z^2+zi[n+1]+c
n8 -- zr^2+3zi+c
n9 -- zr^2+4zi+c

o0 -- z^2+timewave[n]+c -- Note 7
o1 -- Loxodromic #2 by Thomas Kromer
o2 -- Loxodromic #3 by Thomas Kromer
o3 -- Sinus #4 by Thomas Kromer
o4 -- Sinus #5 by Thomas Kromer
o5 -- Marmor by Thomas Kromer
o6 -- Armor by Thomas Kromer
o7 -- Three-Parameter Julia set by Kenneth Gustavsson
o8 -- Three-Parameter Julia set #2 by Kenneth Gustavsson
o9 -- Three-Parameter Cubic Julia set -- Kenneth Gustavsson

p0 -- z^2+c (Note 2)
p1 -- cz(1-z)
p2 -- z(z-1/z)+c
p3 -- cz^2-c
p4 -- z^2+cz^2+c
p5 -- z^3+c
p6 -- ((z^2)*(2z+c))^2+c
p7 -- z^2+j+kzn
p8 -- (x^2-y^2-j,2xy-k) when x>0 ; else (x^2-y^2-c+jx,2xy+kx-k) -- Barnsley (Note 1)
p9 -- z^2-cz^3+c

q0 -- (2z^3+c)/(3z^2)
q1 -- (2z^3+c)/(3z^2+1)
q2 -- Newton's method applied to exp(z)-c
q3 -- Newton method applied to sin^2z-c
q4 -- Newton's method applied to sin3z-c
q5 -- Newton's metod applied to sinz+cosz-c
q6 -- Newton's method applied to exp(z)sinz-c
q7 -- Newton's method applioed to exp(sinz)-c
q8 -- Newton's method applied to fn(z)+c
q9 -- Newton's method applied to fn(z)+fn(z)+c

r0 -- z^3+conj(z)c+c
r1 -- z^z+z^3+c
r2 -- z^3-z+c
r3 -- z^2+exp(z)+c
r4 -- (z^2-c)(z+1)

r5 -- cz^3+c
r6 -- z^2+cexp(z)+c
r7 -- sin(z)+cz^2+c
r8 -- (z-1)/c when x>=0; else (z+1)/cc -- Barnsley
r9 -- (z-1)/c when kx-jy>=0; else (z+1)/c -- Barnsley

s0 -- sin(z)-c2
s1 -- z^4+exp(z)+c
s2 -- (c(z^2+1)^2)/(z^2-1)
s3 -- z^2+tan(z)+c
s4 -- strange attractor IFS (s=1.4142) -- Barnsley
s5 -- z^5-c^4
s6 -- composite function cz-c/z && z^2+c
s7 -- 1/z^2+c
s8 -- (z^3+3(c-1)z+(c-1)(c-2)
s9 -- z(z^5+c^2)

t0 -- z(z^6+c^2)
t1 -- z^7-z^5+z^3-z+c
t2 -- z^4-z^2+c
t3 -- z^3+tanz+c
t4 -- z^2+z^c+c
t5 -- z^3+c/z+c
t6 -- discretizes Volterra-Lotka equations via modified Heun algorithm
t7 -- z^3+c^z+c
t8 -- Quaternion set -- q^2+c^2
t9 -- z^2+cz^2+c

u0 -- Mandelbulb (sine-based non-trig version) -- Note 8
u1 -- Mandelbulb (cosine-based non-trig version)
u2 -- Mandelbulb (sine-based trig version)
u3 -- Mandelbulb (cosine-based trig version)

Note 1: all pertinent menu flags must be set for built-in functions to work as described.

Note 2: For further info on Michael Barnsley's formulas, see his "Fractals Everywhere".

Note 3: Halley map requires the Newton flag to be set. This is another numerical approximation method for finding complex roots. For all Newton/Halley functions, the Newton map is the default. The Halley option is specified through the Arg Gadget, the second character being set to 'h', after the display method(1-9). E.g. '1hr' would designate a relaxed Halley map with display method 1.

Note 4: Halley and Newton maps can use one of nine display methods:
#1 (the default mode, except for functions using sinz, expz, logz and tanz, or fn(z), which default to method 2, and don't use methods 1,4 or 5): ---colors represent the root(the zero) which a point converges to.

#2 (if the Arg Gadget is set to 2, or for functions of sinz, tanz, logz, expz, and fn(z)): ---colors represent the number of iterations a point takes to converge.
#3 (if the Arg Gadget is set to 3) -- colors represent the number of iterations a point takes to converge according to an alternate formula described by C. Pickover in Computers, Pattern, Chaos and Beauty.
#4 (if the Arg Gadget is set to 4) -- a merging of methods 1 and 3. After the point converges according to the alternate formula #3, its roots are colored according to #1.
#5 (if the Arg Gadget is set to 5) -- a variation of method 1, with double-convergence checking inside the loop.
#6, #7 and #8-- alternate convergent formulas.
#9 -- a variation of method 3, with double-convergence checking.

Note 5: An additional third argument that affects the convergence speed of Newton/Halley maps may be one of the following six methods:
'r': relaxed Newton method uses the formula z = z-sf(z)/f'(z).
'm': modified Newton transformation uses the formula: z-(f(z)/(f'(z)+si)). Note: Si here references the s variable * i, not the complex variable s+si.
'd': relaxed modified Newton method uses the formula: z-(sf(z)/f'(z)+si)).
'p': premodified Newton transform uses the formula: sz-(f(z)/f'(z)).
'c': complex Newton transform uses the formula : z-(f(z)/(f'(z)+c)), where c is the complex constant.
'n': Nova variation by Paul Derbyshire, z-(f(z)/(f'(z))+c.

The s constant is entered via the S gadget.

Note 7: the term 'fn(w)' represents any one of 47 user-defined functions chosen through the f1-f4 gadgets:
0: sin(w). 1: sinh(w). 2: cos(w). 3: cosh(w).
4: tan(w). 5: tanh(w). 6: exp(w). 7: ln(w).
8: w^c 9: w^z. 10: 1/w. 11: w^2.
12: w^3. 13: abs(w). 14: sqrt(w). 15: w.
16: conj(w). 17: csc(w). 18: csch(w). 19: sec(w).
20: sech(w). 21: cot(w). 22: coth(w). 23: cw.
24: 1. 25: arsin(w). 26: arcsinh(w).
27: arccos(w). 28: arccosh(w). 29: arctan(w).
30: arctanh(w). 31: arccot(w). 32: arccoth(w).
33: vers(w). 34: covers(w). 35: L_3(w): 3rd degree Laguerre polynomial. 36: gamma (w): first order gamma function.
37: G(w): Gaussian probability function -- (1/sqr(2pi))*e^(.5w^2).
38: c^(s+si). 39: zero. 40: w^(s+si). 41: |(wx)|+|(wy)|*i(abs).
42: wy+wx*i(flip). 43: conj(cos(w))--cosxx. 44: theta(w) -- polar angle(w).
45: real(w). 46: imag(w).
When only fun#1 or fun#2 is used and a single user-defined function is involved, the function is taken from f1. When two user-defined functions appear in a function, the f2 gadget supplies the second function type, except as noted below. For Newton/Halley maps involving z^z, the first derivative is

defined as z^z*(1+ln(z)). An alternate derivative formula (z*z^(z-1)) is used when a non-integral value is entered as an arg limit (e.g.: 0.1). (This produces interesting effects, though mathematically inaccurate.) For plots that use both fun#1 and fun#2(type 2 or 3, etc), fun#1 takes its functions from f1 and f2 and fun#2 takes its functions from f3 and f4.

Note 8: The quaternion and hypercomplex functions use the complex c gadgets to input cr, ci, cj and ck. These may be zero when generating a Mandelbrot-like set of these functions. Julia sets may then be mapped by grabbing points (cr,ci) from interesting areas near this set. Cj and ck must be entered manually for Julia sets. The hj and hk gadgets are used to input the z and w coefficients of the j and k planes. Use small amounts to start for these variables (0-1.0.) Values of 0 for hj, hk, cj and ck result in a two-dimensional slice that matches the standard (non-hypercomplex) type. Higher values of z and w (as well as cj and ck) produce more pronounced asymmetry in the complex mapping.

Note 9: Renormalization functions use the Arg Gadget for plotting options (1-4,6-8) as follows:

0 or 1: default renormal, with anti-ferromagnetic points mapped only for Julia sets. Paramagnetic points (those converging to 1) and ferromagnetic points (those escaping to infinity) are mapped for both Mandelbrot and Julia sets.

2: anti-ferromagnetic points are mapped for the Mandelbrot set. This is actually a level-set mapping for points that do not escape to infinity or converge to 1.

3: uses an alternate convergence formula for paramagnetic and anti-ferromagnetic points.

4: a combination of methods 0 and 3, with characteristics of both methods appearing in plot.

6-8: alternate convergence methods, same as those used with Newton/Halley maps

An optional argument for renormalization 'n' follows the convergence method. This is an alternate bailout method for ferromagnetic points.

Note 10: Most of the built-in functions (except for real Newtons and the Gallet formulas) have hypercomplex extensions when values of cj, ck, hj or hk are non-zero.

Note 11: Hypercomplex Newton/Halley maps use only type 2 and type 3 convergence tests.

Note 12: The default version of hypercomplex conjugate is defined as conjugate(h)=hr-hi-hj+hk. A variant of the hypercomplex conjugate uses an arg limit with a non-integral value (e.g.: 2.1.) This makes all imaginary components of h negative, such that conjugate(h)=hr-hi-hj-hk.

Note 13: The formula for a first degree Laguerre polynomial is e^t(d/dt(t/e^t))=d/dt(t)-t.

Note 14: With a non-integral value entered in the limit gadget, an alternate (Fractint) version of this formula is executed.

Note 15: For real Newtons, the function selected from the f1-f4 boxes is a real function. For a real conjugate, the negation of the real term is used.

Note 16: formulas by Sylvie Gallet have been modified to allow both Mandelbrot and Julia sets to be drawn from them. Except for Gallet-6-02, the bailout is set with the built-in variable zlimit in the

Parameters window. Gallet-6-02 uses the complex constant p3 (limit and converge) for bailout.

Note 17: For the traditional cubic Mandelbrot (example in The Science of Fractal Images), hj, hk, cj and ck (hypercomplex components of z and c) should be set to zero when used with the quaternion type.

Cubic Mandelbrots quaternions use the S and Si variables (in the New Formula window) to set the initial value and range of the 3rd dimension. Starting from an initial value of 'S', Si is normally set to 2*abs (S), when the Type is MandelbrotP or Julia Tower. Si can also be set to center the 3rd dimension on something besides 0.0. Si has no affect on Mandelbrot0 or Julia types, height fields or in 2D mode. (In 2D mode, the S variable acts as one of the two fixed dimensions, along with the Limit variable.) Note: the previous version of FZ (1.19b) used Si as an increment to S, instead of the range of the 3rd dimension. Assuming Steps was 200 in the Quaternion window, you need to multiply Si by 200 to retain compatibility with cubic pictures done with version 1.19b. The current version allows you to change the Steps variable (for smoother pictures) without having to change Si also.

The Limit variable (NF window) points to the fourth dimension (in the Quaternion window the 4th Dim. variable points to hk.)

In addition, the Arg value (entered in NF window) has the following affect on cubic Mandelbrots:
0 -- compute M+, using hj for z space
1 -- compute M+, using greater of S or hj for z space
2 -- compute M-, using greater of S or hj for z space
3 -- compute M+ and M-, use lesser of M+/M- for pixel depth
4 -- compute M+ and M-, use greater of two for pixel depth
5 -- compute M+ and M-, use difference of two for pixel depth
6 -- compute M+ and M-, use sum of two for pixel depth
7 -- compute M+ and M-, use vector magnitude of two for pixel depth
8 -- compute M+ and M-, use intersection of two (CCL)
9 -- compute M+ and M-, use M+ or difference of two if M+>M-
Args 3-9 affect only quaternion-type cubic Mandelbrots, while args 0-2 can be used in 2D mode, or with height fields.

For args 0-9, a second argument determines the plane that is used for the fourth dimension:
b -- b-imag(default)
B -- b-real
a -- a-imag
A -- a-real

A third argument also works for args 0-9:
j -- alternate cubic Mandelbrot mapping(for compatibility with alpha versions of Fractal Zplot 1.19)

Therefore an arg value of '3B' uses the union of M+ and M- and b-real as the fourth dimension.
Note 18: For the loxodromic functions, Sinus #2 and Rings of Fire, the Arg Limit variable (in the Edit Formula/Type window) is used as an additional ingredient. Try values -1.5 to 1.5 for Sinus #2

and 1.5 or PI/2 for Rings of Fire.
Note 19: Octonions have a form of xr+xi+xj+xk+xE+xI+xJ+xK. For these formulas C is the octonion constant (1,1,1,1,1,1,1,1) and CC is the octonion conjugate (1,-1,-1,-1,-1,-1,-1,-1). Additional options are entered via the Arg box in the Quaternion editor window. To rotate the extra four-octonion dimensions (E-K) use the following syntax:

0I -- rotate OE-OK to OI-OE
0J -- rotate OE-OK to OJ-OI
0K -- rotate OE-OK to OK-OJ

To normalize the C and CC constants:

n -- normalize C and CC (default is un-normalized)

Alternate octonion initialization:

c -- set OE-OK to .01 at beginning of each iteration

With octonions, you have your choice of two different algebraic systems (depending on whether the Type is set to Quaternion or Hypernion.) Hyper-octonions use alternate definitions of the basic octonion multiplication tables. This is similar to the difference between hypercomplex quaternions (hypernions) and quaternions. The algebra for octonion and hyper-octonions differ in how they conform to (or fail in) the associative and commutative laws.

Note 20: The timewave array is derived from the formula of Matthew Watkins and Peter Meyer's translation to 'c'. In the Julia set version the array acts as a continuously varying complex constant of the form, "tc1=timearray[(2^iteration)%384]/25, tci= timearray[(2^iteration)%384+1]/25, etc. In the Mandelbrot version, the array acts as an increment to zreal and zimag, with initialization being in the same form as the Julia set. The divisor "25" was chosen strictly for esthetics, as this value enhances the openness of the timewave fractal.

Note 21: These formulas were designed by Paul Nylander and David White, in their search for the true "3D Mandelbrot." The non-trig versions are limited to exponents -2 to -8 and 2 to 8, inclusive. Use the trig-based formulas for exponents beyond -8 and 8 or non-integral exponents such as 2.5. The non-trig versions are approximately 3-4 time faster. Enter the exponent in the 's' box. Using negative exponents with the Mandelbulb formulas produces inverted fractals, with a large hole in the center of the 3D figure. They are very sensitive to bailout magnitude. Use a low value such as 4 for best results. Since most of the detail in inverted 3D fractals is inside the "shell" the Dive function is also useful to reveal hidden detail. The Mandelbulb formulas use only one type of 3D algebra based on trigonometric functions, so the other Types such as quaternion are N/A. To produce the 3D version of Mandelbulb, select Quaternion, Hypernion or Complexified from the Type menu.

Note 22: These formulas are based on the Mandelbulb formulas using negative exponents. However, the coding was modified to produce a variation that closely resembles the Tricorn formula: z̲^2+c, where the imaginary part of z is multiplied by -1. The Tricorn formula is also known as the three-corner-hat formula... The exponent is limited to the integral range 2-8 in the non-trig formulas. The trig-based formulas support positive non-integral exponents only. Enter the exponent in the 's' box.

16.6 Lsystem info

Lsystem info

Fractal Zplot contains an extended version of the 0L-system (string rewriting) described in The Science of Fractal Images (edited by Pietgen and Saupe.) The basic algorithm has been expanded and modified extensively, and now supports the 3-D syntax of Laurens Lapre's LParser. In addition an expanded set of rules that allow looping and 3D formula mapping (lissajous curves) have been added. To enable the program to run at higher depth levels without extra megabytes of memory, an alternate method of string interpretation has been implemented, using recursion. String size is kept to a minimum, while the axiom is interpreted character by character.
For those who are not familiar with L-systems, a discussion of their basic features and how they differ from other turtle graphics is a good starting point. Lsystems have as their base an axiom, which is a string of character commands, or a single character. The character can be a command in itself, such as 'F' for forward, or it may be a production rule, which is a string of commands also. The axiom and production rules are created and edited with separate editors. When the axiom is 'run', the axiom is scanned for commands and/or rules. Recursion is used to substitute where necessary a production rule string for a single character, to the depth specified at runtime. In 0L-systems, quite long character strings can be built this way and then executed. Once the string is built, it is then stripped of non-commands and executed without displaying the commands on the screen. The size of the drawing object is thus established and it is then scaled and drawn on the screen. The first significant difference from other turtle graphics systems is now apparent: self-scaling. The object (s) will never overrun the screen boundaries. Objects can be displayed to the limits of the window, regardless of window size. Since object size is always relative to the window size, a scaling factor is used to create similar objects of different size. Other turtle systems would vary the length of an object's sides.

L-systems have a stack available to keep track of turtle position and variables whenever it is necessary to return to a particular point in the drawing. This is useful for producing tree structures and other branching patterns. Fractal Zplot puts on its stack the turtle position, headings, line style, and color information.

The third significant difference between L-systems and other turtle graphics interpreters is in its handling of recursion. In L-systems, recursion is a built-in feature. You only need to specify the level of recursion, and the
lsystem does its string interpretation to that level. Fractal Zplot uses recursion to call its own interpretation routines, and thus eliminates the need for lengthy strings to interpret. The price for the above built-in features is speed. Lsystems can never be as fast drawing a dragon curve, for instance, as a dedicated dragon-drawing program. On the other hand, fractal generators have trouble matching the speed of L-systems where branching patterns are involved. There is no need to recurse backwards to reach a branching node in L-systems: just pop the stack, and you're there. Fractal Zplot is capable of rendering polygons in three dimensions, but cannot match the speed of a dedicated 3-D modeler. Hidden-line removal is done via a z-buffer, with simple ray-tracing to enhance the view. L-systems offer the user a chance to experiment with a very different approach to graphics/fractal production.

Syntax for Fractal Zplot's Lsystem (default and extended)

The default syntax follows Laurens Lapre's LParser implementation. The initial heading is (0,1,0), which will move the turtle up in the direction of the y-axis, when no rotation is applied to the axis (or a rotation of 0,0,0). All turning commands are three-dimensional. The following is extracted from Laurens Lapre's LParser package.

Turtle Orientation commands

+ turn left around up vector
+(x) turn x left around up vector
- turn right around up vector
-(x) turn x right around up vector
& pitch down around left vector
&(x) pitch x down around left vector
^ pitch up around left vector
^(x) pitch x up around left vector
< roll left (counter clockwise) around forward vector
<(x) roll x left around forward vector
> roll right (clockwise) around forward vector
>(x) roll x right around forward vector

Special Orientation commands

| turn 180 deg around up vector
% roll 180 deg around forward vector
$ roll until horizontal
~ turn/pitch/roll in a random direction
~(x) " in a random direction with a maximum of x degrees
t correction for gravity with 0.2
t(x) correction for gravity with x

Movement commands when {} active

F move forward and draw full length record vertex
F(x) move x forward and draw record vertex
Z move forward and draw half length record vertex
Z(x) move x forward and draw record vertex
f move forward with full length record vertex
f(x) move x forward record vertex
z move forward with half length record vertex
z(x) move x forward record vertex

```
g     move forward with full length       don't record vertex
g(x)  move x forward                      don't record vertex
.     don't move                          record vertex

-------------------------------------------------------------------------
Structure commands
-------------------------------------------------------------------------
[     push current state
]     pop current state
{     start polygon shape
}     end polygon shape

-------------------------------------------------------------------------
Inc/Dec commands
-------------------------------------------------------------------------
"     increment length with 1.1
'     decrement length with 0.9
"(x)  multiply length with x also '(x)
;     increment angle with 1.1
:     decrement angle with 0.9
:(x)  multiply angle with x also ;(x)
?     increment thickness with 1.4
!     decrement thickness with 0.7
?(x)  multiply thickness with x also !(x)

-------------------------------------------------------------------------
Additional commands
-------------------------------------------------------------------------
c     increment color index
c(x)  set color index to x
-------------------------------------------------------------------------
```

Extended command syntax:

r(n) -- repeat the execution of the following commands by the number specified by the following character or characters. The default is 1, or one repetition. For example: 'r(5)' would repeat a loop 5 times.

rv -- repeat the execution of the following commands until the current heading is reached again. Used to create turtle graphics subroutines that stop when the the total turning is a multiple of 360 degrees.

rp -- repeat the execution of the following commands until the current position(x,y,z) is reached again. Used for mapping lissajous curves.

@ -- end of repeat loop. Repeat loops can be nested to a level of 10. You should begin and end a

repeat loop inside of the same production rule or axiom, to avoid potential problems with the interpretation of that loop. Since recursion is used to interpret a production rule character by character, the '@' can only send the interpreter back to the beginning of that loop. Trying to get back to an axiom's repeat loop using a '@' from a production rule will trigger a syntax error.

's(n)' -- number of sides in line. 0 sides is default for a tubular line. s(3) sets a wedge-shaped line, while s(4) sets a cubic or square line shape. Increasing the number of sides makes the line more rounded, but increases draw time. The sides command only works when the line width is greater than 1. (A narrower line can be rounded with fewer sides, to speed up drawing.)

3D curve(lissajous) commands:

F(Ln) -- draw segment of 3D curve, centered around current position.
f(Ln),g(Ln) -- move as segment of 3D curve, no drawing.
Z(Ln) -- draw half segment or n-length segment of 3D curve.
z(Ln) -- move as half-segment or n-length segment of 3D curve, no drawing.

+(Ln) -- rotates 3D curve's x and z (heading) vectors by n degrees.
-(Ln) -- rotates 3D curve's x and z (heading) vectors by -n degrees.
&(Ln) -- rotates 3D curve's x and y (heading) vectors by n degrees.
^(Ln) -- rotates 3D curve's x and y (heading) vectors by -n degrees.
<(Ln) -- rotates 3D curve's y and z (heading) vectors by n degrees.
>(Ln) -- rotates 3D curve's y and z (heading) vectors by -n degrees.

b(n) -- set boundary limit for 3D curve (default is 0). Used preceeding a 'rp' command, to allow the repeat loop more flexibility in deciding when the beginning postition has been reached. Some 3D curves require a loosening of the boundary for the rp loop to work. Higher boundary values make the rp loop more effective, though too high a value can result in premature bailout. To eliminate endless rp loops, a failsafe bailout is set at 50000 reps.

i(n) -- skip n segments in 3D curve (default 0.) Use to speed up drawing by decreasing segment resolution. Sharp angles may appear as disks rather than a continuous form when segments are skipped.

l(n) -- set starting angle for 3D curve (default is 0.) Using 'l' and r(n) different part of the 3D curve can be drawn. The 'rp' command automatically sets l to 0.

P -- set center position for 3D curve. Uses the current turtle position as a center point for 3D curves

x{formula} -- set x formula for 3D curve. Can use any trig function, plus the angles, 'x', 'y' and 'z' and the loop variable 'd'. The default x formula is "sin(x*d)*cos(y*d)".

y{formula} -- set y formula for 3D curve. Can use any trig function, plus the angles, 'x', 'y' and 'z' and the loop variable 'd'. The default y formula is
"sin(x*d)*sin(y*d)"

w{formula} -- set z formula for 3D curve. Can use any trig function, plus the angles, 'x', 'y' and 'z' and the loop variable 'd'. The default z formula is "cos(x*d)".

u(n) -- sets the x angle for 3D curve.
v(n) -- sets the y angle for 3D curve.
k(n) -- sets the z angle for 3D curve.

Notes:

The color index is initialized at 2. Each index of the current palette is used by the ray-tracer as a separate palette, so up to 236 different objects can be modeled using up to 236 different colors. This works best when the index is a dark color. The color will be spread from dark to light when it is used during final ray-tracing.

The angles used in turning commands may be negative, as +(-25).

When F or a like drawing/movement command is followed by an argument, as in F(5), the turtle draws the absolute distance specified, in turtle units. This may be different from FFFFF, since the initial length of the draw vector is 100. An equivalent (and faster) draw command for FFFFF would therefore be F(500), assuming the length vector was not changed by one of the inc/dec commands. When the length vector of a line is decreased or increased, the width of the line is proportionately changed also (to retain compatibility with LParser files.)

Under the extended rule set, when line width is changed during drawing, Fractal Zplot attempts to smooth out width changes by introducing a gradient, or gradual width change, while drawing. If two or more consecutive commands are introduced that change the line width before drawing, only the last width change will have this gradient.

Syntax errors and assignment errors that occur during runtime are displayed in a message box, and the run is halted. The message box contains the characters that caused the syntax error, to aid troubleshooting.

16.7 Bibliography

Bibliography

Complex Mathematics
Churchill, Ruel.V. and Brown, James Ward: "Complex Variables and Applications", Fifth Edition, McGraw-Hill Publishing Company, New York, 1990.

Korn, Granino A. and Korn, Theresa M.: "Manual of Mathematics, McGraw-Hill Publishing Company, New York, 1967.

Fractal Theory
Barnsley, Michael: "Fractals Everywhere", Academic Press, Inc., 1988.

Devaney, Robert L.: "Chaos, Fractals, and Dynamics", Addison-Westley Publishing Company, Menlo Park, California, 1990.

Mandelbrot, Benoit B.: 'The Fractal Geometry of Nature", W.H.Freeman and Company, New York, 1983.

Peitgen, H.-O. and Richter, P.H.: 'The Beauty of Fractals", Springer-Verlag, Berlin Heidelberg, 1986.

Formulas and Algorithms
Burington, Richard Stevens: "Handbook of Mathematical Tables and Formulas", McGraw-Hill Publishing Company, New York, 1973.

Kellison, Stephen G.: "Fundamentals of Numerical Analysis", Richard D. Irwin,Inc. Homewood, Illinois, 1975.

Peitgen, Heinz-Otto and Saupe, Deitmar: "The Science of Fractal Images", Springer-Verlag, New York, 1988.

Pickover, Clifford A.: "Computers, Pattern, Chaos and Beauty", St. Martin's Press, New York, 1990.

Stevens, Roger T.: "Fractal Programming in C", M&T Publishing, Inc.,Redwood City, California, 1989.

Wegner, Tim, Tyler,Bert, Peterson, Mark and Branderhorst, Pieter: "Fractals for Windows", Waite Group Press, Corte Madera, CA, 1992.

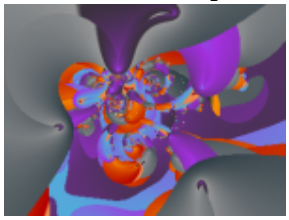
Wegner, Tim and Tyler, Bert: "Fractal Creations", Second Edition, Waite Group Press, Corte Madera, CA, 1993.

Whipkey, Kenneth L. and Whipkey, Mary Nell: "The Power of Calculus", John Wiley & Sons, New York, 1986.

16.8 About Fractal Zplot

About Fractal Zplot

>>>>> Fractal Zplot x64™ v1.000 ©2010 by Terry W. Gintz



Fractal Zplot x64 was rewritten to take advantage of the new generation of 64-bit computers now on the market. As such it is 30-40% faster than the 32 bit model, in 3D mode, and up to 3-4 times faster in 2D mode. Export models can be up to 10 million faces or more, unsimplified, limited only by the amount of system memory available. Caveats: the 64 bit model has a different structure alignment and larger pointer size, so data files (.zp) produced with the 32 bit model are incompatible with this version. There is, however, a transfer routine in the latest 32 bit version of Fractal Zplot that allows most of the parameters in older .zp files to be converted to a format Fractal Zplot x64 can load. Fractal Zplot x64 runs only on 64-bit Windows.

Fractal Zplot graphs formulas based on 4-D complex number planes. Fractal Zplot currently supports the Mandelbrot set, Julia sets, and Phoenix curves, with millions of mapping variations. Also included are midpoint displacement routines to enhance 3D plots. 3D plot types now include quaternion, hypernion, orbital fractals and lsystems, in addition to the original height-field type. The complex math functions supported include sin(z), sinh(z), z^z, e^z, z^n, sqrt(z), cos(z), cosh(z), tan (z), tanh(z), log(z), ln(z), n^z and others, including the gamma and Laguerre functions.

Up to two formulas for z using the above functions may be plotted, using traditional rules for generating Mandelbrot sets (Benoit B. Mandelbrot) and Julia sets (G. Julia.) Also, there are mapping options that use non-traditional methods, such as the epsilon-cross method (Clifford A. Pickover), renormalization and IFS (Michael Barnsley).

New to Fractal Zplot are the Mandelbulb formulas by Paul Nylander and Daniel White. Thanks guys for all the fish!

Since the formula parser is an interpreter, with its inherent lack of speed, 180 'popular' and unusual formulas have been hard-coded to reduce graphing time by 40 to 60 percent (plus the FraSZle formula set too!) Several of the built-in formulas are capable of producing graphs that the program could not generate through the parser alone, such as applying the Mandelbrot set to Newton's method for solving quadratic equations. Also included in the built-in functions are the Quaternion and hypercomplex sets of four dimensions. Hypercomplex extensions (as described in Fractal Creations) have been incorporated into most of the Mandelbrot and Julia functions except for the Quaternion set (this is itself a special 4D version of the Mandelbrot set z^2+c.), and the real Newton formulas and formulas f3-f9 by Sylvie Gallet.

Fractal Zplot requires a true-color video adapter for best results. It may work in 16-bit (high color), but this hasn't been tested thoroughly.

Memory requirements for Fractal Zplot vary with the size of the drawing area Fractal Zplot opens on, ranging from approximately 3 megabytes memory for a 640X480 area to 48 megabytes for a 2048X1536 area. Special routines have been added to reduce memory requirements for large bitmaps (up to 14400X10800) by writing these directly to a file instead of using a memory bitmap. For poster size pictures (up to 96000X72000), a segmenting option is available to divide pictures into up to 225 pieces. The segments can be tiled into one large bitmap using a program such as Richard Paasen's Image Arithmetic.

Acknowledgements: many thanks to Paul Carlson for providing me his algorithms for 3D-like

fractals, and allowing me to incorporate his ideas into Fractal Zplot. Also, special thanks to Ron Barnett for his help in setting up the animation routines, to Earl Hinrichs for sharing his unique programming methods on the fractal art and programmer's lists, to Frode Gill for his quaternion and ray-tracing algorithms, to Dirk Meyer for his Phong-shading algorithm, to Piet van Oostrum for his T2MF, without which the midi support would still be a far off dream, and to David Makin for sharing his ideas on quaternion colorings and 3D insights. Thanks also to Francois Guibert for his Perlin Noise example. A special mention for Stig Pettersson, for his ground breaking work in cubic Mandelbrots, giving me clues to an historic fractal world I hardly knew existed. The multi-windowing interface in Fractal Zplot is courtesy of that extraordinary and prolific fractal programmer, Steven C. Ferguson, whose filters I have also included in FZ. Steve's contributions to the look and feel of Fractal Zplot and previous versions of my program have had a deep impact on my fractal imaging experiments. Remercia te très beaucoup, to Jean-Pierre Louvet, for his timely suggestions for improving ZPlot 24, Fractal Elite and Dofo-Zon. I am indebted to Alessandro Rosa for carefully explaining the "cutoff rate" method to the QuaSZ group, and making 3D Newton/Halley pictures finally a reality.

For a short history of this program, see Chronology.

Disclaimer: This program/programmer has no relation, now or ever, to the EIS program ZPlot by Scribner Associates, Inc. of Southern Pines, NC, USA. Fractal ZPlot is capable of a lot, but it was never designed for use with frequency analyzers, potentiostats or galvanostats...

16.8.1 Chronology

Chronology

History of this program:

In September 1989, I first had the idea for a fractal program that allowed plotting all complex functions and formulas while attending a course on College Algebra at Lane College in Eugene, Oregon. In November 1989, ZPlot 1.0 was done. This Amiga program supported up to 32 colors, 640X400 resolution, and included about 30 built-in formulas and a simple formula parser.

May 1990 -- ZPlot 1.3d -- added 3D projections for all formulas in the form of height fields.

May 1991 -- ZPlot 2.0 -- first 236-color version of ZPlot for Windows 3.0.

May 1995 -- ZPlot 3.1 -- ZPlot for Windows 3.1 -- 60 built-in formulas. Added hypercomplex support for most built-in formulas.

May 1997 -- ZPlot 24.02 -- first true color version of ZPlot -- 91 built-in formulas. Included support for 3D quaternion plots, Fractint par/frm files, Steve Ferguson's filters, anti-aliasing and Paul Carlson's orbit-trap routines.

June 1997 -- ZPlot 24.03 -- added Earl Hinrichs Torus method.

July 1997 -- ZPlot 24.08 -- added HSV filtering.

December 1997 -- Fractal Elite 1.14 -- 100 built-in formulas; added avi and midi support.

March 1998 --Split Fractal Elite into two programs, Dreamer and Medusa(multimedia.)

April 1998 -- Dofo 1.0 -- supports new Ferguson/Gintz plug-in spec.

June 1998 -- Dofo-Zon -- redesigned multi-window interface by Steve Ferguson, and includes Steve's 2D coloring methods.

August 1998 --Dofo-Zon Elite -- combination of Fractal Elite and Dofo-Zon

October 1998 -- Dofo-Zon Elite v1.07 -- added orbital fractals and IFS slide show.

November 1998 -- Dofo-Zon Elite v1.08 -- added lsystems.

April 1999 -- Split Dofo-Zon Elite into two programs: Fractal Zplot using built-in formulas and rendering methods, and Dofo-Zon to support only plug-in formulas and rendering methods.

May 1999 -- Fractal Zplot 1.18 -- added Phong highlights, color-formula mapping and random fractal methods.

June 1999 -- completed Fractal ViZion -- first version with automatic selection of variables/options for all fractal types.

July 1999 -- Fractal Zplot 1.19 -- added cubic Mandelbrot support to quaternion option; first pc fractal program to render true 3D Mandelbrots.

September 2000 -- Fractal Zplot 1.22 -- added support for full-screen AVI video, and extended quaternion design options.

October 2000 -- QuaSZ (Quaternion System Z) 1.00 -- stand alone quaternion/hypernion/cubic Mandelbrot generator

November 2000 -- Added octonion fractals to QuaSZ 1.01.

March 2001 -- Cubics 1.0 -- my first totally-3D fractal generator.

May 2001 -- QuaSZ 1.03 -- added Perlin noise and improved texture mapping so texture tracks with animations.

June 2001 -- Fractal Zplot 1.23 -- added Perlin noise and quat-trap method.

July 2001 -- QuaSZ 1.05 -- improved performance by converting many often-used dialogs to non-modal type.

October 2001 -- FraSZle 1.0, QuaSZ formula and algebra compatible version of Fractal Zplot

November 2001 -- DynaMaSZ 1.0, the world's first Dynamic Matrix Systems fractal generator

January 2002 -- MiSZle 1.1 -- generalized fractal generator with matrix algebra extensions

May 2002 -- DynaMaSZ SE 1.04 (unreleased version)-- scientific edition of DMZ, includes support for user-variable matrix dimensions (3X3 to 12X12)

January 2003 -- Pod ME 1.0 -- first stand-alone 3-D loxodromic generator, Hydra 1.0 -- first 3-D generator with user-defined quad types and Fractal Projector a Fractal ViZion-like version of DMZ SE limited to 3X3 matrices

May 2003 -- QuaSZ 3.052 -- added genetic-style function type and increased built-in formulas to 180. Other additions since July 2001: generalized coloring, support for external coloring and formula libraries, and Thomas Kroner's loxodromic functions.

May 2003 -- FraSZle and Fractal Zplot 3.052 -- added random 3D orbital fractals, new 3D export methods, upgraded most frequently-used dialogs to non-modal type and added genetic-style function type. FZ now based on FraSZle except for built-in formula list and Newton support.

Index

- 3 -

- A -

- B -

- C -

- D -

- E -

- F -

- H -

- I -

- M -

- P -

- R -

- S -

- T -

- V -