

Fractal Orbits IM Application Help

© 2010-2013 ... Mystic Fractal

Table of Contents

Foreword	0
1 Main Index	1
1 Title Bar	1
2 Scroll bars	2
3 Size	2
4 Move	2
5 Minimize Command	3
6 Maximize Command	3
7 Next Window	3
8 Previous Window	3
9 Close	3
10 Restore	4
11 Switch to	4
2 Fractal Orbits Remote	5
1 New	5
2 Undo	5
3 Size	5
4 Color	5
5 Batch	5
6 Fvr	5
7 Draw button	6
8 Abort button	6
9 Scan	6
10 Rend	6
11 View	6
12 Help	7
13 Pal	7
14 Gen	7
15 Form	7
16 Channel Guide	7
17 Channel ST	8
18 Channel BB	8
19 Channel NT	8
20 Channel HA	8
21 Channel Q1	8

22 Channel Q2	9
23 Channel C1	9
24 Channel C2	9
25 Channel CJ	9
26 Channel O1	10
27 Channel O2	10
28 Channel QT	10
29 Channel MC	10
30 Channel JC	11
31 Random Render	11
32 Save	11
33 Load	11
34 Bmp	11
35 Png	11
36 Jpg	11
37 	12
38 >	12
39 □	12
40 V	12

3 File menu 13

1 File New command	14
2 File Open command	14
File Open dialog box	14
3 File Close command	15
4 File Save command	15
5 File Save As command	15
File Save as dialog box	15
6 File Load Parameters command	16
7 Load Par File	16
8 File Load Palettes command	16
9 File Open [JPG] command	17
10 File Open [PNG] command	17
11 File Save Parameters command	17
12 Save Par File	17
13 File Save Palettes command	18
14 File Save As [JPG] command	18
15 File Save As [PNG] command	18
16 File Load Palette [PQZ] command	18
17 File Load Palette [MAP] command	18

18	File Load Texture command	19
19	File Save Palette command	19
20	File Save Texture command	19
21	File Save QuaSZ command	19
22	File Save Q Polygon[OBJ] command	19
23	Simplify option	20
24	File Save Q Polygon[POV] command	20
25	Smooth option	20
26	File Save Q Polygon[WRL] command	20
27	File Save Q Polygon[DXF] command	21
28	File Set Max Vertices command	21
29	Export Setup command	21
30	File 1, 2, 3, 4, 5, 6 command	21
31	File Exit command	22
4	Edit menu	22
1	Edit Undo command	22
2	Edit Copy command	23
3	Edit Clip command	23
4	Edit Paste command	23
5	Edit Copy Data command	23
6	Edit Paste Data command	24
7	Formula Window	24
8	Fractal Variables	26
	Parameters Window	26
	Quaternion Window	28
9	Cubic Values Window	29
10	Octonion Values Window	30
11	Edit Quat-trap	30
12	Size	31
13	Ray-Tracing Window	31
14	Edit Palette	31
	Reverse button	33
	Neg Button	33
	Map Button	33
	H/R Button	33
	Spread Button	33
	Copy Button	33
	SRG Button	33
	SRB Button	33
	Okay Button	34
	Reset Button	34
	Cancel Button	34

Red Slider	34
Red edit box.....	34
Green Slider	34
Green edit box.....	34
Blue Slider	34
Blue edit box.....	34
Smooth Button	35
Scramble	35
15 Edit Text command	35
16 Preferences	35
5 Image menu	35
1 Image Draw command	36
2 Plot to file	36
3 Plot Files in Directory	36
4 Image Redraw command	37
5 Image Auto Clear command	37
6 Image Auto Alert command	37
7 Image Auto Remote command	37
8 Image Auto Time command	37
9 Image Abort command	37
10 Continue Draw	38
11 Zoom	38
12 Image New View on Zoom command	38
13 Image Clone	39
14 Scan	39
15 Full Screen	39
16 Pilot	39
17 Reset Ranges	40
18 Reset Figure	40
6 Type menu	40
1 Mandelbrot	40
2 MandelbrotP	41
3 Julia	41
4 Type Quaternion command	42
5 Type Hypernion command	42
6 Type Cubic command	42
7 Type Complexified Quaternion command	43
8 Type 2D Quaternion Map command	43
9 Type 2D Hypercomplex Map command	43
10 Type 2D Cubic Map command	43

11	Type 2D Complexified Map command	44
12	Type Cloud command	44
7	Map menu	45
1	Z-Real	45
2	Z-Imag	45
3	Abs(Z-Real)	46
4	Abs(Z-Imag)	46
5	Z-Real+Z-Imag	46
6	Abs(Z-Real)+Abs(Z-Imag)	46
7	>Abs(Z-Real) or Abs(Z-Imag)	47
8	<Abs(Z-Real) or Abs(Z-Imag)	47
9	Abs(Z)	47
8	Render menu	48
1	Phoenix	48
2	Orbit traps	49
	Orbit trap values	50
3	Newton Set	50
4	Newton Off	50
5	Renormalize	51
6	Symmetry	51
7	Palette-based Coloring Options	52
	2D Palette-based Coloring Options	52
	Log Map	54
	Small Log	55
	Indexed Log.....	55
	Linear Map.....	55
	Indexed Linear.....	55
	Bubble	55
	Use Level	55
	Iteration	55
	Log Palette.....	55
	Continuous Potential.....	55
	Angle	56
	Angle-Iteration.....	56
	QFactor	56
	Bubble Extensions.....	56
	Biomorph	56
	Decomposition.....	56
	Set Only	57
	Filter	57
	3D Palette-Based Coloring Options	57
	Function box.....	59
	Magnify value box.....	59
	Magnify Slider.....	59
	Atan Coloring.....	59

Bof60 Coloring.....	59
Potential Coloring.....	59
Distance Coloring.....	59
Random Filter button.....	59
Random Filter Complexity box.....	59
8 Palette 1-21 command	60
9 Divide by 1 palette command	60
10 Divide by 2 palette command	60
11 Divide by 4 palette command	60
12 Divide by 8 palette command	60
13 Divide by 16 palette command	60
14 Generalized Coloring	60
Blend buttons	61
Red/Grn/Blu controls	61
RGB button	61
RBG button	61
GRB button	61
GBR button	61
BRG button	61
BGR button	61
Sine algorithm button	61
Sawtooth algorithm button	62
Gray Scale button	62
Invert button	62
Fractal Dimension button	62
15 Add Noise	62
16 Factors	62
17 Reset Noise Seed	63
18 Texture Scale	63
9 View menu	63
1 View Toolbar command	63
toolbar	63
2 View Status Bar Command	64
status bar	64
10 Window menu	65
1 Cascade	65
2 Tile	65
3 Arrange Icons	65
4 Size DeskTop	65
5 1, 2,	66
11 Video menu	66
1 Write Avi Video	66
2 Add Key Frame	66

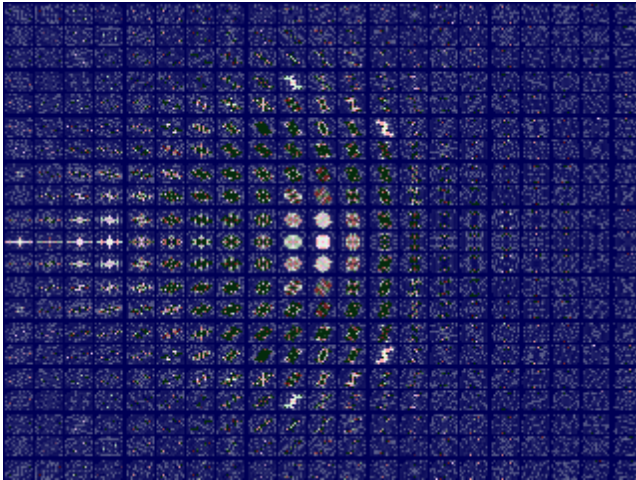
3	Reset Frames	67
4	Edit Frames	67
5	Save Frames [FOV]	67
6	Load Frames [FOV]	67
7	View Avi	67
12	Exploratory menu	68
1	BBM 20	68
2	BBM 5	68
3	BBM 1	68
13	Demo menu	68
1	Random Stalks	69
2	Random Bubbles	69
3	Random Newton	69
4	Random Halley	69
5	Random Quaternion	69
6	Random Quaternion2	70
7	Random Cubic Mandelbrot	70
8	Random Cubic Mandelbrot2	70
9	Random Cubic Julia	71
10	Random Octonion	71
11	Random Octonion2	71
12	Random Quat-Trap	71
13	Random MandelCloud	72
14	Random JuliaCloud	72
15	Random Render	72
16	Batch Mode/Random Setup	72
14	Help menu	73
1	Getting Started	73
2	Index	76
3	Hot Keys	76
4	Parser	78
5	Built-in Formulas	80
6	Bibliography	95
7	About Fractal Orbits	96
	Chronology	97

Index

100

1 Main Index

Fractal Orbits IM Help Index



[Getting Started](#)

[Fractal Orbits Remote](#)

[Channel Guide](#)

Commands

[File menu](#)

[Edit menu](#)

[Image menu](#)

[Type menu](#)

[Map menu](#)

[Render menu](#)

[View menu](#)

[Window menu](#)

[Video menu](#)

[Demo menu](#)

[Help menu](#)

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 Orbits 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 Orbits 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 Orbits 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 Orbits Remote

Fractal Orbits Remote

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

2.1 New

New button

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

2.2 Undo

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.

2.3 Size

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. 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

Color button

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

2.5 Batch

Batch button

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

2.6 Fvr

FVR button

The parameter editor varies with the fractal type selected (2D or 3D), and contains many of the variables that Fractal Orbits 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 to restart plotting from the current column.

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 Orbits 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

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.

2.10 Rend

Rend button

For quaternion fractals, 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

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

Help button

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

2.13 Pal

Palette Coloring button

Apply Palette-based coloring method.

2.14 Gen

[Gen]eralized Coloring button

Apply generalized coloring method.

2.15 Form

Formula button

Open Formula/Type editor window.

2.16 Channel Guide

Channel Guide

The fourteen channels accessed via the Fractal Orbits remote:

ST: Random Stalks -- based on Paul Carlson's orbit trap methods, etc.

BB: Random Bubbles -- based on Paul Carlson's Bubble method

NT: Random Newton -- based on Newton's method

HA: Random Halley -- based on Halley's method

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

MC: Random Mandel Cloud fractals

JC: Random Julia Cloud fractals

2.17 Channel ST

Random Stalks (Channel ST button)

A random Julia fractal is generated using one of Paul Carlson's orbit trap methods.

2.18 Channel BB

Random Bubbles (Channel BB button)

A random Julia fractal is generated using Paul Carlson's bubble method.

2.19 Channel NT

Random Newton (Channel NT button)

A random Julia fractal is generated using Newton's method.

2.20 Channel HA

Random Halley (Channel HA button)

A random Julia fractal is generated using Halley's method.

2.21 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, H_j is set to 2.0, and the lighting is set for optimum viewing.

Note: for some images an h_j 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.

2.22 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, H_j 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 h_j 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 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.

2.24 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.

2.25 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, H_j 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.

2.26 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.)

2.27 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 O_i) to create octonion fractals that may extend to eight dimensions.

2.28 Channel QT

Random Quat-Trap (Channel QT button)

A random quat-trap fractal is generated using one of the [orbit trap](#) methods and a quaternion formula. This is a hybrid fractal combining aspects of both orbit trap and 3D quaternion. The orbit trap part controls the overall shape of the image generated, while the quaternion part adds depth and lighting to the final image.

2.29 Channel MC

Random Mandel Cloud (Channel MC button)

A random Mandelbrot Cloud fractal is generated using any of the built-in formulas or a custom formula.

2.30 Channel JC

Random Julia Cloud (Channel JC button)

A random Julia Cloud fractal is generated using any of the built-in formulas or a custom formula.

2.31 Random Render

Random Render

The rendering options for the current fractal are randomized. Does not affect formula or range variables. For quaternions, a random coloring filter is applied.

2.32 Save

Save button

Use this button to save and name the active drawing. Fractal Orbits 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.

2.33 Load

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.

2.34 Bmp

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.)

2.35 Png

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.

2.36 Jpg

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.

2.37 ||||

Write Avi button

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 frames in the frame buffer are then written to the avi stream and the stream closed.

2.38 >

Add Key Frame button

Fractal Orbits uses a frame buffer to compose an animation. You add key frames to the buffer with this command. Each key frame is identical to the active image. Change variables between key frames to create the illusion of motion or morphing. You can edit the frames with the [frame editor](#).

2.39 []

[] button

Opens the frame editor window so you can edit frames in the video buffer by using any of the other editor windows. The Move button allows you to move a frame from one spot in the buffer to another. You can change the frame image being edited by using the Frame slider or Edit box. After changing frames, use the Preview button to display the current frame being edited. The Delete button allows you to delete all but two of the frames, the minimum number of frames to create a movie. (If you want to delete all the frames, use the [Video/Reset Frames](#) command.)

2.40 V

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 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 as OBJ	Save polygonized quaternion as Wavefront object.
Simplify	Simplify mesh.
Save as POV	Save polygonized quaternion as a pov triangle object.
Smooth	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.

Export Setup	Edit/View parameters for exporting meshes.
Exit	Exits Fractal Orbits.

3.1 File New command

New command (File menu)

Use this command to create a new drawing window in Fractal Orbits. 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 Orbits stores the file that you want to open.

Directories

Select the directory in which Fractal Orbits 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 Orbits 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 Orbits 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 Orbits adds the extension .fo6.

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 [.fo6]. The data file contains all variables to recreate an image created previously with Fractal Orbits.

3.7 Load Par File

Load Par File

Opens a window that allows the user to select and convert Fractint/Fractal Orbits parameter files to Fractal Orbits 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 Orbits 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 Orbits, 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 Orbits 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 Orbits 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 Orbits (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 [.fo6].

3.12 Save Par File

Save Par File

This option allows you to save Fractal Orbits 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 Orbits. The par definitions can be easily posted on the net and reloaded in Fractal Orbits 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 .fo6. 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 Orbits 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.

Compatibility issues: Since Fractal Orbits has many options not found in Fractint, no attempt was

made to make the Fractal Orbits par format compatible with Fractint's. The object was to make Fractal Orbits picture parameters more accessible to Fractal Orbits users. As formula files add another unnecessary layer to the parameter puzzle, Fractal Orbits'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.

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 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.

3.17 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.

3.18 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].

3.19 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].

3.20 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].

3.21 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.

3.22 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. The memory requirements for this routine are high, 20MB or more for a typical Julia set quaternion rendered at 320X240. 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 Quaternion window, where precision=10/Steps.

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.

3.23 Simplify option

Export -> Simplify option (File menu)

With this option selected (default on) if the Save as OBJ command is executed the resulting polygon mesh is simplified according to Garland's QSlim algorithm before being output as a Wavefront obj file.

3.24 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. 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 $\text{precision} = 10/\text{Steps}$.

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.

3.25 Smooth option

Export -> Smooth option (File menu)

With this option selected (default on) smooth normals are calculated for the POV triangle mesh and the polygonized object is output as a POV mesh with smooth_triangles. Effective when the Save as POV command is executed.

3.26 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. The memory requirements for this routine are high, 20MB or more for a typical Julia set quaternion rendered at 320X240. 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 Quaternion window, where $\text{precision} = 10/\text{Steps}$.

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.

3.27 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 $\text{precision} = 10/\text{Steps}$.

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.

3.28 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 Fractal Orbits that are much larger than 75MB (2 million faces) on Windows 7.

3.29 Export Setup command

Export Setup command (File menu)

Here you can set the target face size for a Wavefront object. The face size can range from 1000 to 5,000,000 faces. This allows you to reduce the size of the Wavefront object file by a factor of 10 or more and still retain the essential image detail. Use smoothing in Bryce to eliminate most of the triangle artifacts. Before exporting the polygon as an obj file, it helps to make the mesh resolution as high as practical by increasing the Steps size in the initial Quaternion values window to a suitable value. This varies with the complexity of the quaternion figure. The face size limits the object file size by reducing faces on the polygon until the face limit is reached, so you never export a polygon with more than n-size faces. It's possible that there could be fewer faces than the face limit, and in that case no mesh reduction is performed. But usually you'll see a dramatic reduction in object faces (and obj file size) if the Steps size is set to a value greater than 200 (the startup default).

3.30 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.

3.31 File Exit command

Exit command (File menu)

Use this command to end your Fractal Orbits 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.
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.

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.fo6". 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.fo6" 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 9. 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.

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 Newton 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.)

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 Orbits 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 line with a preceding ';'. Some variables such as 'pixel' and p1 are named after Fractint's predefined variables. These are provided to allow Fractal Orbits users to more easily convert Fractint formula types to Fractal Orbits use. However, Fractal Orbits 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 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 Orbits'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 x64. You can then export any quaternion done with Fractal Orbits as a QuaSZ parameter file [QS6] and import it into QuaSZ x64 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 Orbits 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 Orbits 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.

The Starting cr, Starting ci, Increment cr and Increment ci are used with BBM 5 and BBM 20, image sets of 25 and 441 respectively. The Julia or Mandelbrot sets are initialized with incremented complex constants to form either Julius sets (BBM 20) or sets of 25 images.

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 Orbits'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.

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 (P0-N9) 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 greater in value than 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. Note: if you set the Back variable less than $-z$, and the front variable greater than or equal to $-z$, you can get a gradient in the background, depending on the lighting and coloring settings. This sometimes has the effect of placing part of the quaternion in a fog-like dimension.

The Starting cr, Starting ci, Increment cr and Increment ci are used with BBM 5 and BBM 20, image sets of 25 and 441 respectively. The Julia or Mandelbrot sets are initialized with incremented complex constants to form either Julius sets (BBM 20) or sets of 25 images.

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 $2*\text{abs}(S)$, 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)

This window allows you to edit all paramters that determine the overall shape of the quat-trap fractal. This is a hybrid fractal combining aspects of both orbit trap and 3D quaternion. The orbit trap part controls the overall shape of the image generated, while the quaternion part adds depth and lighting to the final image.

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 Palettes

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

It is important to realize that palettes are software-simulated in Fractal Orbits (since 24-bit color supports no hardware palettes), so color-cycling and palette switching are not fast operations as with a 256-color system that supports palettes.

There are copy and spread options to smooth or customize the existing palettes in Fractal Orbits. 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, with four colors on the last row. This makes it easy to create divide-by-16, divide-by-8, divide-by-4 and divide-by-2 palettes with 256 colors. With Fractal Orbits, 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 Orbits, 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.

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 Orbits, 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 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 cross hair. 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 Orbits's startup directory. Title text can also be edited (as a single line only) in the Edit/Formula window.

4.16 Preferences

Preferences (Edit menu)

Each time you use the Reset command, Fractal Orbits 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 Orbits saves the last changes you make to these options. If you want to go back to the initial settings (the way Fractal Orbits was packaged originally) you can click on the Reset Defaults button. This restores the data, Auto variables and random setup defaults.

5 Image menu

Image menu commands

The Image menu offers the following commands:

[Draw](#) Draw the picture.

Plot To File	Plot large bitmap images directly to png file.
Plot Files In Directory	Disk render .fo6 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.
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.
Full Screen	View image full-screen.
Pilot	Use Pilot to rotate 3D figure and alter quaternion variables.
Reset Ranges	Reset coordinates of active figure.
Reset Figure	Reset current figure.

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.

5.2 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.

5.3 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 (.fo6) 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.

5.4 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. So you need to turn this option off then.

5.5 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. You can use the shift-c command ([hot keys](#)) to clear the drawing area at any time.

5.6 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.

5.7 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.

5.8 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 Orbits 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 Orbits.

5.9 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 Orbits 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.

5.10 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. Not available with cloud-type fractals.

5.11 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)

Zooming with cloud-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 Image/Reset Ranges is selected. 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 cloud fractals.

5.12 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.

5.13 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.

5.14 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.

5.15 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.

5.16 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.

5.17 Reset Ranges

The Reset Ranges command resets the real Z and imaginary Z ranges in the Parameters or Quaternion window (to ± 2.0 and $\pm 2 \times \text{screen aspect}$.) 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. For Cloud-type fractals, this command resets the magnify mode to unmagnified.

5.18 Reset Figure

Reset Figure

Reset the current figure 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.

6 Type menu

Type menu commands

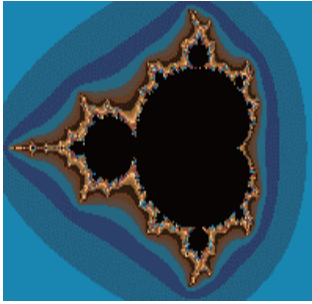
The Type menu offers the following commands:

Mandelbrot	Mandelbrot set.
MandelbrotP	Mandelbrot set (orbit starts at pixel.)
Julia	Julia set.
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.
Cloud	Set fractal type to orbital fractal.

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, C_r and C_i 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

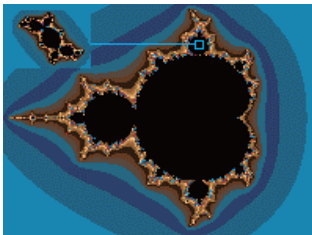
MandelbrotP

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.

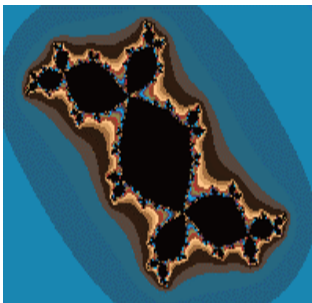
6.3 Julia

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 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.

6.5 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.

6.6 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 $2 * \text{abs}(S)$, 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.

6.7 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.

6.8 Type 2D Quaternion Map command

2D Quaternion Map (Type menu)

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

6.9 Type 2D Hypercomplex Map command

2D Hypercomplex Map (Type menu)

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

6.10 Type 2D Cubic Map command

2D Cubic Map (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 $2*\text{abs}(S)$, 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 2D Complexified Map command

2D Complexified Map (Type menu)

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

6.12 Type Cloud command

Cloud (Type menu)

The cloud fractal type plots points based on the iterative value of each z. Points that overlap have their color indexes incremented. This produces images that resemble fractal clouds. The Steps and Iterations variables in the Parameters window are used to control the clouds resolution. The cloud can be based on any built-in formula or a custom formula, with either the Mandelbrot or Julia type selected. The Mandelbrot type tends to produce more flowery fractals, however.

7 Map menu

Map menu commands

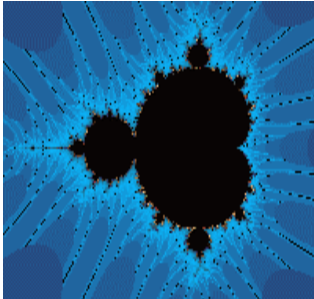
The Map menu offers the following commands:

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

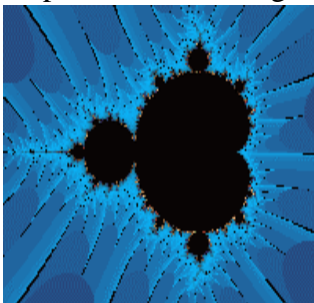
Map based on the real part of the complex number Z; used to map exponential Julia sets, etc.



7.2 Z-Imag

Z-Imag

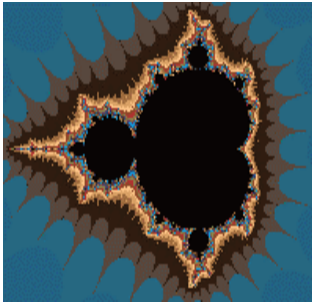
Map based on the imaginary part of the complex number Z.



7.3 Abs(Z-Real)

Abs(Z-Real)

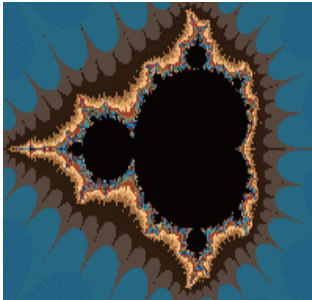
Map based on the absolute value of the real part of the complex number Z ; used to map exponential Julia sets, etc.



7.4 Abs(Z-Imag)

Abs(Z-Imag)

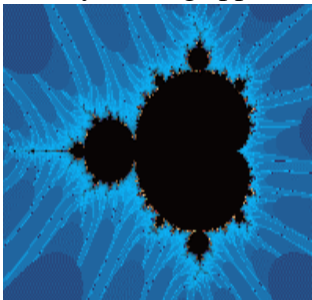
Map based on the absolute value of the imaginary part of the complex number Z .



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.

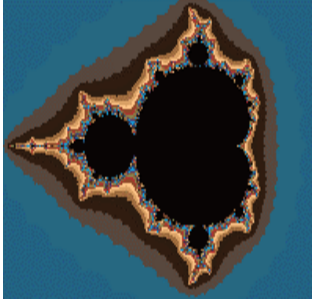


7.6 Abs(Z-Real)+Abs(Z-Imag)

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

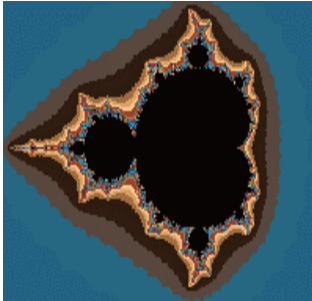
complex number Z . Changes the way banding appears in complex mappings.



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

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

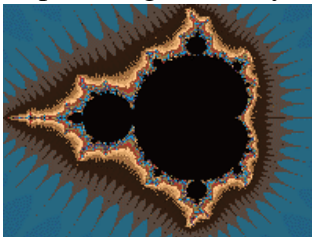
Map based on the greater of the absolute value of the real part or the imaginary part of the complex number Z . Works like a logical 'or', where either part of z must exceed $zlimit$ to break the iteration loop. Changes the way banding appears in complex mappings.



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

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

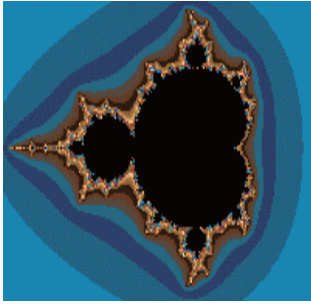
Map based on the lesser of the absolute value of the real part or the imaginary part of the complex number Z . Works like a logical 'and', where both parts of z must exceed $zlimit$ to break the iteration loop. Changes the way banding appears in complex mappings.



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 Orbits uses only the 'sum'(modulus of z) for break-point tests.) The standard method of mapping Julia and Mandelbrot sets.



8 Render menu

Render menu commands

The Render menu offers the following commands:

Phoenix	Phoenix Curve.
Orbit Traps	Set orbit trapping method.
Newton->	Define Newton method or clear Newton flag.
Renormalization	Renormalization.
Symmetry->	Horizontal, vertical or XY symmetry.
Palette-Based Coloring Options	
Generalized Coloring	
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.

8.1 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 Orbits 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'.

8.2 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 Astroid method (Paul Carlson) uses the atan function to create an orbit trap based on exit angles.

The Chains method is a variation of the Rings method, where the ring stalks have a non-uniform width.

The Bullet method is based on the magnitude of z, where points are colored when z is less than epsilon.

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

To produce the maximum 3-D effects (as Phil Pickard and Paul Carlson do) with these options, Level Curve Linear Map 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.

To automate the process of producing Paul Carlson's 3-D like fractals, a button has been added to this window for 'Carlson extensions'. This sets the Set Only flags as well as the Level Curve Linear Map and the cutoff value, and sets the Mapping to $\langle \text{Abs}(Z\text{-Real}) \text{ or } \text{Abs}(Z\text{-Imag}) \rangle$. An exclude value of 2 is also used. The Mapping and exclude values are extra parameters that Paul uses in some of his formulas, and may be omitted in other cases.

8.2.1 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.

8.3 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 Renormalization flag, 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. In 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.

8.4 Newton Off

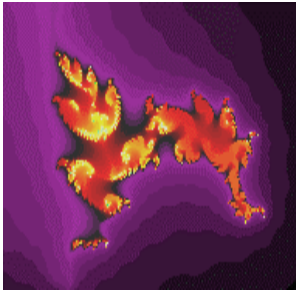
Newton Off

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

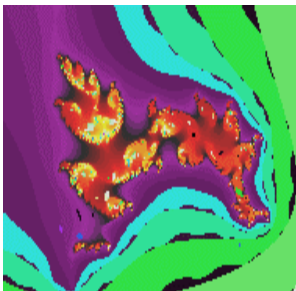
8.5 Renormalize

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

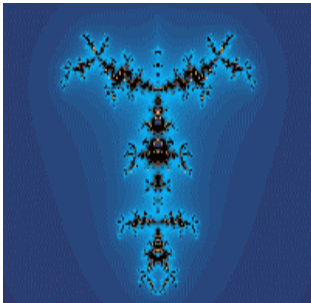


With renormalization

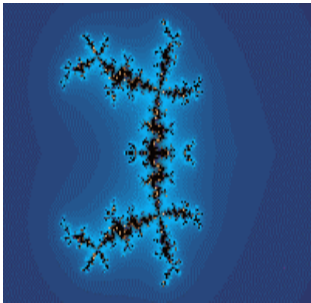
8.6 Symmetry

Symmetry

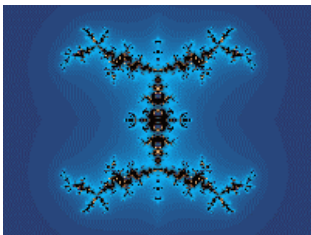
This produces a mirror image from left to right (vertical) or top to bottom (horizontal) or both (xy). You can zoom with symmetry, but the results will be uncertain if the zoom box is off-center on the window or if rotation is used. Symmetry has no effect when used with the Tesseral option, 3D, or random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



Vertical symmetry



Horizontal symmetry



XY symmetry

8.7 Palette-based Coloring Options

Switch to palette-based coloring. There are different options for 2D and 3D fractals.

8.7.1 2D Palette-based Coloring Options

2D Palette-based Coloring Options

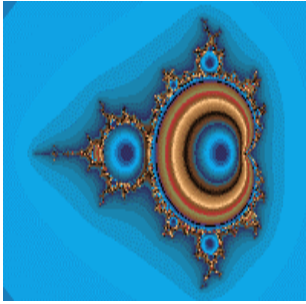
Inside Coloring Options

Level Curves

Level-curves map the set points based on the value of Z . This allows the inside of the complex set to be color-scaled.

Log Map

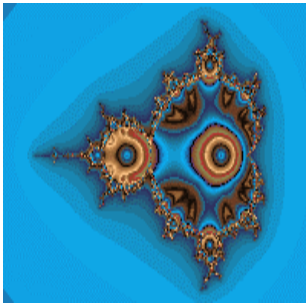
produces colored bands on the inside of the complex set. Points are mapped according to what the value of z is at final iteration.



Log Map

Small Log

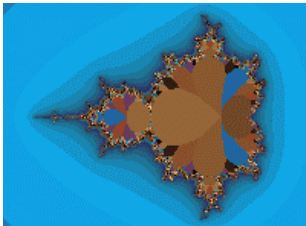
produces circular patterns inside the complex set. Points are mapped according to the smallest value z gets during iteration.



Small Log

Indexed Log

is mapped according to the escape time it takes z to reach its smallest value.

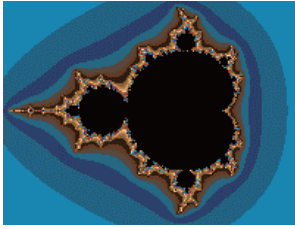


Indexed Log

Linear Map is mapped like Log Map (with the mapped value of the function at its final iteration applied to the color palette) and produces 3D-like effects with orbit-trap renderings.

Outside Coloring Options**Iteration**

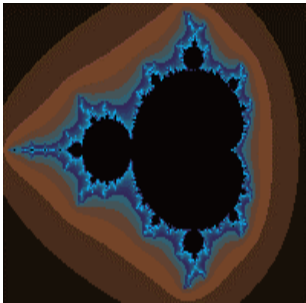
The Iteration option uses the point's escape time.



Escape-time coloring.

Log Palette

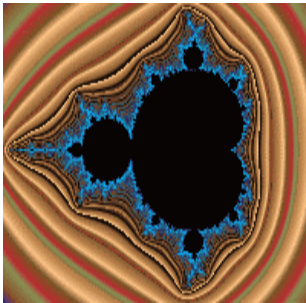
A point is colored based on its logarithmic escape. The QFactor controls the smoothness of the coloring. A small number from 10-20 works best here..



Log coloring.

Continuous Potential

A point is colored based on its continuous potential (when it blows up.) The QFactor controls the smoothness of the coloring. A higher number from 2000-20000 works best here.



Continuous-Potential coloring.

Angle uses the absolute value of a point's exit angle (theta.) This is the atan method in Fractint. Angle-Iteration uses the angle formed by the difference between a point's last two exit values and subtracts the point's escape time. This is Paul Carlson's atan method.

Set Only

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

8.7.1.1 Log Map

Log Map produces colored bands on the inside of the complex set. Points are mapped according to what the value of z is at final iteration.

8.7.1.2 Small Log

Small Log produces circular patterns inside the complex set. Points are mapped according to the smallest value z gets during iteration.

8.7.1.3 Indexed Log

Indexed Log is mapped according to the escape time it takes z to reach its smallest value.

8.7.1.4 Linear Map

Linear Map is mapped like Log Map (with the mapped value of the function at its final iteration applied to the color palette) and produces 3D-like effects with orbit-trap renderings.

8.7.1.5 Indexed Linear

Indexed Log is mapped according to the escape time it takes z to reach its smallest value.

8.7.1.6 Bubble

Bubble 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. Cutoff needs to be a small negative value, usually less than one. 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. Note: the magnitude of the cutoff variable needs to be increased or decreased proportionately by the number of palette splits to keep the "bubbles" the same size. The color ranges in the palette should be graded from light to dark to highlight the bubble centers.

8.7.1.7 Use Level

Use Level -- All points (inside and outside) are colored according to the Level curve selected or Potential Linear if no level curve is selected.

8.7.1.8 Iteration

Iteration option uses the point's escape time to color points outside the Mandelbrot or Julia set.

8.7.1.9 Log Palette

Log Palette -- A point is colored based on its logarithmic escape. The QFactor controls the smoothness of the coloring. A small number from 10-20 works best here..

8.7.1.10 Continuous Potential

Continuous Potential -- A point is colored based on its continuous potential (when it blows up.) The QFactor controls the smoothness of the coloring. A higher number from 2000-20000 works best here.

8.7.1.11 Angle

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

8.7.1.12 Angle-Iteration

Angle-Iteration uses the angle formed by the difference between a point's last two exit values and subtracts the point's escape time. This is Paul Carlson's atan method.

8.7.1.13 QFactor

QFactor

The QFactor controls the smoothness of the coloring. A small number from 10-20 works best for Log Palettes, while higher number 2000-20000 is best for Continuous Potential.

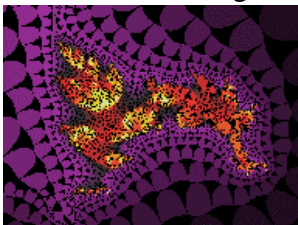
8.7.1.14 Bubble Extensions

The **Bubble Extensions** button selects the Bubble map and Use Level coloring methods, and sets the Cutoff variable to a small negative value. It also selects a Divide-by-two palette if a split palette has not already been chosen. (The Palette may need to be redesigned in the Palette editor to match the split factor.) This automates the process of producing Paul Carlson's 3-D bubble-like fractals.

8.7.1.15 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.)



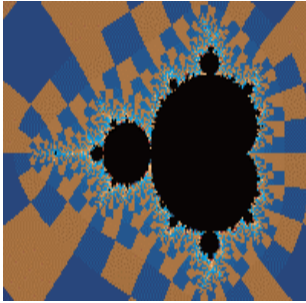
8.7.1.16 Decomposition

Decomposition

When a Decomposition option is set, you have the option of performing either an external or internal binary decomposition. For Mandelbrot/Julia curves, z-arg is broken into two parts.

(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.)



External Decomposition

8.7.1.17 Set Only

Set Only plots all outside points in the background color.

8.7.1.18 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. 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.

8.7.2 3D Palette-Based Coloring Options

3D Palette-based Coloring Options

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. This is a sample function: $\sin(x+y)+\cos(x*x)$. The Magnify slider is used to control the intensity of the filter. Use the Random Filter button to generate a coloring filter with random functions. The level box controls the complexity of the random filter.

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.

Atan Coloring

Uses an algorithm by David Makin to color a quaternion image. The angle of each point (as it escapes the quaternion border) is used to color the image.

Bof60 Coloring

A variation of the Bof60 algorithm found in the classic Pietgen/Richter text, *The Beauty of Fractals*, adapted by David Makin, is used to color a quaternion image. The smallest magnitude of z (found while calculating the quaternion border) is used to render the image.

Potential Coloring

The escape value of z (at the quaternion border) is used to color the image.

8.7.2.1 Function box

The **Function box** is used to enter a coloring filter, up to 80 characters. Use the Random Filter button to generate a coloring filter with random functions.

8.7.2.2 Magnify value box

The **Magnify value box** is used to enter the magnify value, instead of using the slider. Value can range from 1-100.

8.7.2.3 Magnify Slider

The **Magnify slider** is used to control the intensity of the filter. Value can range from 1-100.

8.7.2.4 Atan Coloring

Atan Coloring uses an algorithm by David Makin to color a quaternion image. The angle of each point (as it escapes the quaternion border) is used to color the image.

8.7.2.5 Bof60 Coloring

Bof60 Coloring uses a variation of the Bof60 algorithm found in the classic Pietgen/Richter text, The Beauty of Fractals, adapted by David Makin, is used to color a quaternion image. The smallest magnitude of z (found while calculating the quaternion border) is used to render the image.

8.7.2.6 Potential Coloring

Potential Coloring uses the escape value of z (at the quaternion border) to color the image. .

8.7.2.7 Distance Coloring

Distance Coloring uses the distance of z from zero (at the quaternion border) to color the image.

8.7.2.8 Random Filter button**Random Formula button**

Use this button to create a random coloring formula whose complexity is controlled by the Level box.

8.7.2.9 Random Filter Complexity box

The Random Filter Complexity box controls the length of the random filter generated by the Random Filter button.

8.8 Palette 1-21 command

Palette command (Palette menu)

Switch to palette #.

8.9 Divide by 1 palette command

Divide by One Palette (Palette-based coloring menu)

Palette is not split before applying to pixel.

8.10 Divide by 2 palette command

Divide by Two Palette (Palette-based coloring menu)

Palette is split into two parts before applying to pixel.

8.11 Divide by 4 palette command

Divide by Four Palette (Palette-based coloring menu)

Palette is split into four parts before applying to pixel.

8.12 Divide by 8 palette command

Divide by Eight Palette (Palette-based coloring menu)

Palette is split into eight parts before applying to pixel.

8.13 Divide by 16 palette command

Divide by Sixteen Palette (Palette-based coloring menu)

Palette is split into sixteen parts before applying to pixel.

8.14 Generalized Coloring

Use this command to switch to Steven Ferguson's generalized coloring mode. Images are colored using algorithmic methods that address the whole rgb spectrum, instead of the palette-based coloring methods. To switch back to palette mode, apply one of the palette-based options. Use the Apply button to change the current coloring options for the active figure, if the figure has been drawn recently, without closing the window. Click on Okay to apply new generalized options and redraw the active figure, or Close to close the window without applying any additional changes.

8.14.1 Blend buttons

Various formulas are applied while mapping colors to pixels.

8.14.2 Red/Grn/Blu controls

Red/Grn/Blu controls

Use these sliders and edit boxes adjust the color emphasis or red/green/blue mixture of an image.

8.14.3 RGB button

RGB button

Use red/green/blue mapping for pixels.

8.14.4 RBG button

RBG button

Use red/blue/green mapping for pixels.

8.14.5 GRB button

GRB command (Render menu)

Use this command to use green/red/blue mapping, if in the generalized coloring mode.

8.14.6 GBR button

GBR command (Render menu)

Use green/blue/red mapping for pixels.

8.14.7 BRG button

BRG command (Render menu)

Use blue/red/green mapping for pixels.

8.14.8 BGR button

BGR command (Render menu)

Use blue/green/red mapping for pixels.

8.14.9 Sine algorithm button

Sine Algorithm command (Render menu)

When color values exceed the range of rgb components, the values are scaled with Steven C.

Ferguson's sine algorithm.

8.14.10 Sawtooth algorithm button

Triangle Algorithm command (Render menu)

When color values exceed the range of rgb components or palette indexes, the values are scaled with a triangle algorithm, or linear ramp.

8.14.11 Gray Scale button

Gray Scale button

Color the active image with gray tones.

8.14.12 Invert button

Invert command (Render menu)

Invert image colors.

8.14.13 Fractal Dimension button

Fractal Dimension command (Render menu)

Generalized fractal dimension algorithm (S. Ferguson), for use with any blend option.

8.15 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.

8.16 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.

8.17 Reset Noise Seed

Reset Noise Seed

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

8.18 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.

9 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.

9.1 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 Orbits, 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.

9.1.1 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 Orbits,

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 Orbits

	Open an existing drawing. Fractal Orbits 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 Orbits displays the Save As dialog box.
	Draw Mandelbrot set
	Draw Julia set
	Zoom into rectangle.
	Set image size.
	Edit palette.
	Edit formula/type data.
	Edit fractal parameters.
	Draw image from current parameters.
	Continue drawing.
	Reset coordinates.
	Show picture full-screen.
	Display info about Fractal Orbits.
	Display Fractal Orbits's help index.

9.2 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.

9.2.1 status bar

Status Bar



The status bar is displayed at the bottom of the Fractal Orbits 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.

10 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.

10.1 Cascade

Cascade command (Window menu)

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

10.2 Tile

Tile command (Window menu)

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

10.3 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.

10.4 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.

10.5 1, 2, ...

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

Fractal Orbits 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.

11 Video menu

Video menu commands

The Video menu offers the following commands:

Write Avi Video	Write video buffer to file.
Add Key Frame	Add image to video buffer.
Reset Frames	Reset frame buffer.
Edit Frames	Edit frames in frame buffer.
Load Frames [FOV]	Load frame buffer.
Save Frames [FOV]	Save frame buffer.
View AVI	View an AVI animation file.

11.1 Write Avi Video

Write Avi Video

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 frames defined by the frame buffer are then plotted and written to the avi stream and the stream closed. Variables between each key frame are interpolated and frames added to the avi file to give the illusion of animation.

11.2 Add Key Frame

Add Key Frame

Fractal Orbits uses a frame buffer to compose an animation. You add key frames to the buffer with this command. Each key frame is identical to the active image. Change variables between key frames to create the illusion of motion or morphing. You can edit the frames with the [frame editor](#).

11.3 Reset Frames

Reset Frames

Delete the current frame buffer. The number of video frames is reset to zero.

11.4 Edit Frames

Edit Frames

When the frame editor window is open you can edit the frames in the video buffer by using any of the other editor windows. The Move button allows you to move a frame from one spot in the buffer to another. You can change the frame image being edited by using the Frame slider or Edit box. After changing frames, use the Preview button to display the current frame being edited. The Delete button allows you to delete all but two of the frames, the minimum number of frames to create a movie. (If you want to delete all the frames, use the [Video/Reset Frames](#) command.) The Spread variable determines how many frames are generated between key frames. A higher value produces a smoother video, but also adds to the file size.

11.5 Save Frames [FOV]

This command saves the current frame buffer in a [fov] file. A file requester is opened that allows you to choose the location and name of the frame library. The frame buffer files can also be used as image libraries, similar to Fractint's par and frm formats. The frames contain all the information to reproduce an image at any supported size.

11.6 Load Frames [FOV]

Load a frame buffer that has been previously saved by Fractal Orbits. The buffer replaces any existing frame buffer.

11.7 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.

12 Exploratory menu

Exploratory menu commands

The Exploratory menu offers the following commands:

BBM 20	Julius set of 441 images
BBM 5	Set of 25 images in one window
BBM 1	Display single image.

12.1 BBM 20

BBM 20

This option displays the Julius Ruis set of 441 images in one window (BBM 20). These can be either Mandelbrot or Julia set type fractals. In the [Fractal Parameters editor](#) window you define your own ranges for Starting 'c', as well as incremental values. Not available with the Cloud Type.

12.2 BBM 5

BBM 5

This option displays a set of 25 images in one window (BBM 5). These can be either Mandelbrot or Julia set type fractals. In the [Fractal Parameters editor](#) window you define your own ranges for Starting 'c', as well as incremental values. Not available with the Cloud Type.

12.3 BBM 1

BBM 1

Use this option to display a single image in the draw window, based on the parameters set in the [Fractal Parameters editor](#) window.

13 Demo menu

Demo menu commands

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

Random Stalks	Generate random orbit-trap fractal.
Random Bubbles	Generate random bubble fractal.
Random Newton	Generate random Julia fractal using Newton method.
Random Halley	Generate random Julia fractal using Halley's method.
Random Quaternion	Generate random quaternion/hypernion fractal.

Random Quaternion2	Generate random quaternion/hyperion fractal(extended formula search).
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 Mandelcloud	Generate random Mandelbrot Cloud fractal
Random Juliacloud	Generate random Julia Cloud fractal
Random Render	Select a random rendering.
Batch Mode/Random Setup	Repeat random fractal (batch mode) and randomizing setup.

13.1 Random Stalks

Random Stalks (Demo menu)

A random Julia fractal is generated using one of Paul Carlson's orbit trap methods.

13.2 Random Bubbles

Random Bubbles (Demo menu)

A random Julia fractal is generated using Paul Carlson's bubble method.

13.3 Random Newton

Random Newton (Demo menu)

A random Julia fractal is generated using Newton's method.

13.4 Random Halley

Random Halley (Demo menu)

A random Julia fractal is generated using Halley's method.

13.5 Random Quaternion

Random Quaternion (Demo menu)

A random quaternion/hyperion fractal is generated. 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.

13.6 Random Quaternion2

Random Quaternion2 (Demo menu)

A random quaternion/hypernion fractal is generated. 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.

13.7 Random Cubic Mandelbrot

Random Cubic Mandelbrot (Demo menu)

A random cubic Mandelbrot fractal is generated. 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.

13.8 Random Cubic Mandelbrot2

Random Cubic Mandelbrot2 (Demo menu)

A random cubic Mandelbrot fractal is generated. 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.

13.9 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 (G_0 and G_1) 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, H_j 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.

13.10 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 H_0 - H_9 , 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.

13.11 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 H_0 - H_9 , 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 H_0 - H_9 , with random dimensional switching (one of O_E - O_K for O_i) to create octonion fractals that may extend to eight dimensions.

13.12 Random Quat-Trap

Random Quat-Trap (Demo menu)

A random quat-trap fractal is generated using one of the [orbit trap](#) methods and a quaternion formula. This is a hybrid fractal combining aspects of both orbit trap and 3D quaternion. The orbit trap part controls the overall shape of the image generated, while the quaternion part adds depth and lighting to the final image.

13.13 Random MandelCloud

Random MandelCloud

A random Mandelbrot Cloud fractal is generated using any of the built-in formulas or a custom formula.

13.14 Random JuliaCloud

Random JuliaCloud

A random Julia Cloud fractal is generated using any of the built-in formulas or a custom formula.

13.15 Random Render

Random Render (Demo menu)

The rendering options for the current fractal are randomized. Does not affect formula or range variables. For quaternions, a random coloring filter is applied.

13.16 Batch Mode/Random Setup

Batch mode/Random Setup (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.)

There are radio boxes that allow you to customize how random variables are processed to create new 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 c_j - c_K
- Orbit Traps -- (default on) check to randomize orbit trap types and variables

For 3D quaternions you have the option of selecting only quaternion types, only hypernion types (hypercomplex), only cquat types (complexified quaternion) or a mixture of types.

14 Help menu

Help menu commands

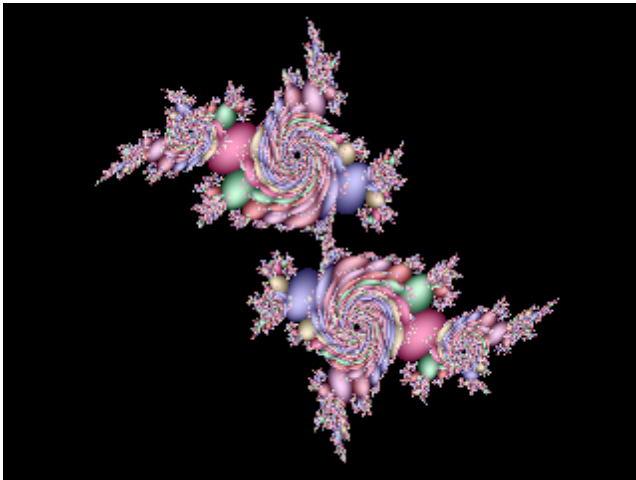
The Help menu offers the following commands, which provide you assistance with this application:

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

14.1 Getting Started

Getting Started

Welcome to Fractal Orbits IM!



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 Orbits. 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.

Jules Ruis developed the Awareness Governance Model "BBM" (in Dutch being the abbreviation of Bewustzijns Besturings Model), used in Training Interaction Management. The BBM-model is an organizational model that makes a distinction on three levels of awareness: Basic Awareness (on the level of each person individual), Business Awareness (on the level of the business or organisation in which that person works) and Global Awareness (on the level of the region/country in which that person lives). Fractal Imaginator was the first Windows program to use BBM as a business tool. MiSZle IM now uses BBM as an exploratory tool only.

Using [BBM 20](#) and [BBM 5](#), MiSZle IM enables you to discover all kind of interesting and beautiful pictures.

At the opening screen you see a Julius set of a fractal formula. This shows a general map ([BBM 20](#)) of the Julia basins that make up the Mandelbrot set of this formula. Double click with the left-mouse button on any of the Julia sets shown and that image will be displayed at the resolution of the draw window. After the Julia set is finished drawing (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, or use the Shift key to zoom out by 50% with each keystroke. You can also rotate the zoom box by using the left and right arrow keys (2-D only). 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.

If you want to zoom into a portion of the Julius set itself, right click with the mouse cursor on an area of the J-set. The cursor changes to a box which you can manipulate as if you were zooming into a single image as explained above.

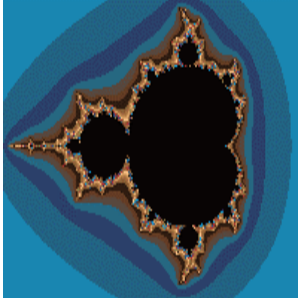
The two methods of fractal composition that most fractal programs use are iterative and orbital. With the iterative method, which is the primary method that Fractal Orbits uses, a formula is successively iterated until some criteria are met then the point is plotted. The process of iteration goes like 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 Orbits provides 180 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 Figure 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 Orbits's main window will be erased and a small plot of the Mandelbrot set will be drawn in the center of the drawing window.



If the Auto-Alert option is enabled (on by default), Fractal Orbits 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. 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.

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 Orbits:

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 Orbits** 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 Orbits allows you to **Undo** the last command in most cases. However this is mostly a failsafe command, as it 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 Orbits**.

14.2 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 Orbits** 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.

14.3 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-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-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.

14.4 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 Orbits to represent standard function".

sine z --- $\sin(z)$ or $\text{SIN}(Z)$; where Z can be any complex expression

hyperbolic sine z --- $\sinh(z)$ or $\text{SINH}(Z)$

arcsine z --- $\text{asin}(z)$ or $\text{ASIN}(Z)$

cosine z --- $\cos(z)$ or $\text{COS}(Z)$

hyperbolic cosine z --- $\cosh(z)$ or $\text{COSH}(Z)$

arccosine z --- $\text{acos}(z)$ or $\text{ACOS}(Z)$

tangent z --- $\tan(z)$ or $\text{TAN}(Z)$

hyperbolic tangent z --- $\tanh(z)$ or $\text{TANH}(Z)$

arctangent z --- $\text{atan}(z)$ or $\text{ATAN}(Z)$

cotangent z --- $\text{cotan}(z)$ or $\text{COTAN}(Z)$

arccotangent z --- $\text{acotan}(z)$ or $\text{ACOTAN}(Z)$

e^z --- $\text{exp}(z)$ or $\text{EXP}(z)$ -- the exponential function

natural log of z --- $\log(z)$ or $\text{LOG}(Z)$

absolute value of z --- $\text{abs}(z)$ or $\text{ABS}(Z)$

square root of z --- $\text{sqrt}(z)$ or $\text{SQRT}(Z)$

z squared --- $\text{sqr}(z)$ or $\text{SQR}(Z)$

real part of z --- $\text{real}(z)$ or $\text{REAL}(Z)$

imaginary part of z --- $\text{imag}(z)$ or $\text{IMAG}(Z)$

modulus of z --- $\text{mod}(z)$ or $\text{MOD}(Z)$ or $|z|$ -- $(x^2 + y^2)$

conjugate of z -- $\text{conj}(z)$ or $\text{CONJ}(z)$ -- $(x - yi)$

flip(z) --- $\text{flip}(z)$ or $\text{FLIP}(Z)$ -- exchange real and imaginary parts of z (y+xi)

polar angle of z -- $\text{theta}(z)$

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 --- π 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.

14.5 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 $\sin z-c$. 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: $(z/e)^z*\sqrt{2*\pi*z}+c$.
 $c7$ -- 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 -- $z^s / \text{limit} + c$: anti-derivative of z^n ; s may be complex using the si variable as its imaginary component.

d1 -- Sterling expansion of gamma function: $(z/e)^z * \sqrt{2 * \pi / z} + c$.

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 $zreal = \text{abs}(zreal)$ (Paul Carlson's "alien" Julia set).

d5 -- z^2 ; where $zreal = \text{abs}(zreal) - cr$, $zimag = zimag - ci$ (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 $\log z - c$.

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) -- $\text{limit} * fn(z) + s + si$, 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 -- $(z^2-z^2(2-s))/s+c$ -- 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 \sin^3z-c
q5 -- Newton's method applied to $\sin z+\cos z-c$
q6 -- Newton's method applied to $\exp(z)\sin z-c$
q7 -- Newton's method applied to $\exp(\sin z)-c$
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 $z^3+\text{conj}(z)c$ -- 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 -- $c\exp(z)$ -- exponential Julia curve, etc. with additional plane checking when real value of Z exceeds 50. If $\cos(\text{imag-Z}) \geq 0$, 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 $\sin z-c$.
s1 -- $\text{sexpz}+c$ -- 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 $\ln(z)+c$.
 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 $z(z^{\text{limit}}-1)$ (Julia set only).

t0 -- Newton/Halley map of $z(z^{\text{limit}}-c)$ (Julia or Mandelbrot set ; display method 2 default; $\text{limit} \geq 1.0$; use 1.0 for initial z with Mandelbrot0 type, or use MandelbrotP type.).
 t1 -- Newton/Halley map of Chebyshev function $\cos(n \cdot \arccos x)$.
 t2 -- Newton/Halley map of Hermite polynomial: $16x^4-48x^2+12$.
 t3 -- alternate Newton/Halley map of $\tan z-c$ (Julia or Mandelbrot set.) the twist is in the second derivative of the Halley type.
 t4 -- Newton/Halley map of $z^{\text{limit}}-c$ (Julia or Mandelbrot set ; display method 2 default; $\text{limit} \geq 1.0$; use 1.0 for initial z with Mandelbrot0 type, or use MandelbrotP type.).
 t5 -- $\ln(\ln(z))+c$ -- user-defined complex set. When the first function is z^2 and the second function is $\text{conj}(z)$, 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)

FraSZle/QuaSZ formulas
 a0 -- spiral network -- C. Pickover
 a1 -- $z+z^3/c+c$
 a2 -- $\tan(z)-(z^2+c)$
 a3 -- $z^2+\arcsin z-c$
 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 -- $1/z^3+\log z-c$

b2 -- $z^3+j+kzn$
 b3 -- cz^2+zn+c
 b4 -- $\sin(z)+kzn+j$
 b5 -- $\text{vers}(z)+c$
 b6 -- $\text{vers}(z)+\text{covers}(z)+c$
 b7 -- $c*\text{vers}(z)+c$
 b8 -- $\text{vers}(z)*\text{covers}(z)+c$
 b9 -- $(\text{foggy coastline \#1})^2+c$ -- Barnsley

 c0 -- $(\text{foggy coastline \#2})^2+c$ -- Barnsley
 c1 -- $\sin(\text{vers})+c$
 c2 -- $\text{csec}(z^2)+c$
 c3 -- $z^3+\text{sech}(z^2)+c$
 c4 -- $c*z^{(c-1)}+z^2$
 c5 -- $\cos(-1/w^2)+c$
 c6 -- $(z/e)^z*\text{sqr}(2*\text{pi}*z)+c$
 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 -- $z^3*\text{sqr}(2*\text{pi}/z)+c$
 d2 -- $z^2-\text{csc}(h)\cot(h)+c$
 d3 -- $(1/(\sin(z)+c))^3$
 d4 -- z^2-c ; where $x=\text{abs}(\text{real}(z))$, $y=\text{imag}(z)$ -- Paul Carlson
 d5 -- z^2 ; where $x=\text{abs}(\text{real}(z))-cr$, $y=\text{imag}(z)-ci$ -- Paul Carlson
 d6 -- $c*\cos(z)+c$
 d7 -- $z*\cos(z)+c$
 d8 -- $z^2+z/\sin(z)+c$
 d9 -- $z^2+z^\text{pi}+c$

 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 -- $fn(z) + fn(z) + c^n$
 L7 -- $cfn(z) + c^n$
 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 \sin^3z-c
 q5 -- Newton's method applied to $\sin z+\cos z-c$
 q6 -- Newton's method applied to $\exp(z)\sin z-c$
 q7 -- Newton's method applied to $\exp(\sin z)-c$
 q8 -- Newton's method applied to $fn(z)+c$
 q9 -- Newton's method applied to $fn(z)+fn(z)+c$

r0 -- $z^3+\text{conj}(z)c+c$
 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 -- $z^2 + c \exp(z) + c$
 r7 -- $\sin(z) + cz^2 + c$
 r8 -- $(z-1)/c$ when $x \geq 0$; else $(z+1)/c$ -- Barnsley
 r9 -- $(z-1)/c$ when $kx - jy \geq 0$; else $(z+1)/c$ -- Barnsley

 s0 -- $\sin(z) - c^2$
 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 -- $z^3 + \tan z + c$
 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 $\sin z$, $\exp z$, $\log z$ and $\tan z$, or $\text{fn}(z)$, 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 $\sin z$, $\tan z$, $\log z$, $\exp z$, and $\text{fn}(z)$): ---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 -- $(1/\sqrt{2\pi}) * e^{-.5w^2}$.

38: $c^{(s+si)}$. 39: zero. 40: $w^{(s+si)}$. 41: $|(wx)| + |(wy)| * i(abs)$.

42: $wy + wx * i(flip)$. 43: $\text{conj}(\cos(w)) - \cos xx$. 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 ($z^z * z^{(z-1)}$) 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 $\text{conjugate}(h)=hr-hi-hj+hk$. 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 $\text{conjugate}(h)=hr-hi-hj-hk$.

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), h_j , h_k , c_j and c_k (hypercomplex components of z and c) should be set to zero when used with the quaternion type.

Cubic Mandelbrots quaternions use the S and S_i variables (in the New Formula window) to set the initial value and range of the 3rd dimension. Starting from an initial value of ' S ', S_i is normally set to $2 * \text{abs}(S)$, when the Type is MandelbrotP or Julia Tower. S_i can also be set to center the 3rd dimension on something besides 0.0. S_i 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 S_i 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 S_i 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 S_i also.

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

In addition, the Arg value (entered in NF window) has the following affect on cubic Mandelbrots:

0 -- compute M_+ , using h_j for z space
 1 -- compute M_+ , using greater of S or h_j for z space
 2 -- compute M_- , using greater of S or h_j 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

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 $\text{PI}/2$ 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:

```

OI    --    rotate OE-OK to OI-OE
OJ    --    rotate OE-OK to OJ-OI
OK    --    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: z^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.

14.6 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.

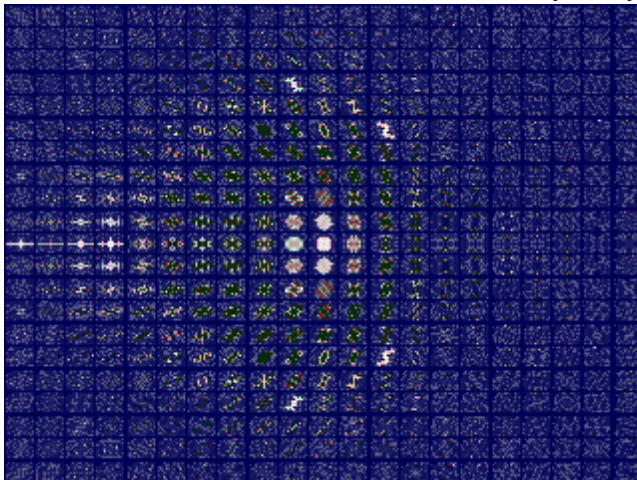
Whipkey, Kenneth L. and Whipkey, Mary Nell: "The Power of Calculus", John Wiley & Sons,

New York, 1986.

14.7 About Fractal Orbits

About Fractal Orbits

>>>>>Fractal Orbits IM™ v1.000 ©2010 by Terry W. Gintz



Fractal Orbits IM was written to take advantage of the new generation of 64-bit computers now on the market. As such it is up to 3 times faster than the 32 bit model. Export models can be up to 10 million faces or more, unsimplified, limited only by the amount of system memory available. Fractal Orbits IM runs only on 64-bit Windows.

Fractal Orbits graphs formulas based on 4-D complex number planes. Fractal Orbits currently supports the Mandelbrot set, Julia sets, and Phoenix curves, with millions of mapping variations. 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).

Since the formula parser is an interpreter, with its inherent lack of speed, 200+ 'popular' and unusual formulas have been hard-coded to reduce graphing time by 40 to 60 percent (plus the FraSZle/QuaSZ 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 Orbits requires a true-color video adapter for best results.

Acknowledgements: many thanks to Paul Carlson for providing me his algorithms for 3D-like fractals, and allowing me to incorporate his ideas into Fractal Orbits. Also, special thanks to Ron Barnett for his help in setting up the animation routines, to Frode Gill for his quaternion and ray-tracing algorithms, to Dirk Meyer for his Phong-shading algorithm, 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 Orbits is courtesy of that extraordinary and prolific fractal programmer, Steven C. Ferguson, whose filters I have also included in FO. Steve's contributions to the look and feel of Fractal Orbits and previous versions of my program have had a deep impact on my fractal imaging experiments.

For a short history of the programs that preceded Fractal Orbits, see [Chronology](#).

14.7.1 Chronology

Chronology

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/frn 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 -- Dofu 1.0 -- supports new Ferguson/Gintz plug-in spec.

June 1998 -- Dofu-Zon -- redesigned multi-window interface by Steve Ferguson, and includes Steve's 2D coloring methods.

August 1998 --Dofu-Zon Elite -- combination of Fractal Elite and Dofu-Zon

October 1998 -- Dofu-Zon Elite v1.07 -- added orbital fractals and IFS slide show.

November 1998 -- Dofu-Zon Elite v1.08 -- added lsystems.

April 1999 -- Split Dofu-Zon Elite into two programs: Fractal Zplot using built-in formulas and rendering methods, and Dofu-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 Mandelbrot.

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/hyperion/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

- 2 -

2Dcolorbutton: angle 56
 2Dcolorbutton: angle-iteration 56
 2Dcolorbutton: biomorph 56
 2Dcolorbutton: bubble extensions 56
 2Dcolorbutton: button 55
 2Dcolorbutton: continuous potential 55
 2Dcolorbutton: decomposition 56
 2Dcolorbutton: indexed linear 55
 2Dcolorbutton: indexed log 55
 2Dcolorbutton: iteration 55
 2Dcolorbutton: linear map 55
 2Dcolorbutton: log map 54
 2Dcolorbutton: log palette 55
 2Dcolorbutton: qfactor 56
 2Dcolorbutton: set only 57
 2Dcolorbutton: small log 55
 2Dcolorbutton: use level 55

- 3 -

3Dcolorbutton: atan coloring 59
 3Dcolorbutton: bof60 coloring 59
 3Dcolorbutton: distance coloring 59
 3Dcolorbutton: filter complexity box 59
 3Dcolorbutton: function box 59
 3Dcolorbutton: magnify slider 59
 3Dcolorbutton: magnify value box 59
 3Dcolorbutton: potential coloring 59
 3Dcolorbutton: random filter 59

- B -

break: newton 50
 break: orbit traps 49
 break: renormalization 51
 button: [] 12
 button: |||| 12
 button: > 12
 button: abort 6
 button: batch 5
 button: bmp 11

button: channel BB 8
 button: channel c1 9
 button: channel c2 9
 button: channel cj 9
 button: channel HA 8
 button: channel JC 11
 button: channel MC 10
 button: channel NT 8
 button: channel o1 10
 button: channel o2 10
 button: channel Q1 8
 button: channel Q2 9
 button: channel qt 10
 button: channel ST 8
 button: color 5
 button: draw 6
 button: Form 7
 button: fr 5
 button: Gen 7
 button: help 7
 button: jpg 11
 button: load 11
 button: new 5
 button: Pal 7
 button: png 11
 button: rend 6
 button: save 11
 button: scan 6
 button: size 5
 button: undo 5
 button: V 12
 button: view 6

- C -

color: 2d palette-based coloring options 52
 color: 3D palette-based coloring options 57
 color: blue edit box 34
 color: blue slider 34
 color: cancel button 34
 color: copy button 33
 color: divide by eight palette 60
 color: divide by four palette 60
 color: divide by one palette 60
 color: divide by sixteen palette 60
 color: divide by two palette 60
 color: edit palette 31
 color: fractal dimension 62

color: green edit box 34
 color: green slider 34
 color: h/r button 33
 color: invert 62
 color: map button 33
 color: neg button 33
 color: okay button 34
 color: pixel 61, 62
 color: rand button 35
 color: red edit box 34
 color: red slider 34
 color: reset button 34
 color: reverse button 33
 color: spread button 33
 color: srb button 33
 color: srg button 33

- D -

demo: batch mode 72
 demo: random coctonion 71
 demo: random cubic julia 71
 demo: random cubic mandelbrot 70
 demo: random newton 69
 demo: random octonion 71
 demo: random quaternion 69
 demo: random quaternion2 70
 demo: random quat-trap 71
 demo: random render 72
 demo: random stalks 69

- E -

edit: clip 23
 edit: copy 23
 edit: copydata 23
 edit: cubic values 29
 edit: formula 24
 edit: fractal variables 26
 edit: octonion parameters 30
 edit: parameters 26, 28
 edit: paste 23
 edit: pastedata 24
 edit: preferences 35
 edit: quatrap 30
 edit: ray-tracing variables 31
 edit: size 31

edit: text 35
 edit: undo 22
 exit 22

- F -

files: load jpeg 17
 files: load palette 18
 files: load palettes 16
 files: load par 16
 files: load parameters 16
 files: load png 17
 files: load texture 19
 files: managing 14, 15, 21
 files: save palette 19
 files: save palettes 18
 files: save par 17
 files: save parameters 17
 files: save q polygon 19, 20, 21
 files: save quasz parameters 19
 files: save texture 19
 files: set max vertices 21
 files: simplify 20
 files: snooth 20
 files: write jpeg 18
 files: write png 18

- H -

help: about fractal orbits 96
 help: bibliography 95
 help: built-in formulas 80
 help: channels 7
 help: chronology 97
 help: hot keys 76
 help: parser info 78
 help: remote 5
 help: tutorial 73

- I -

image: abort 37
 image: auto alert 37
 image: auto remote 37
 image: auto time 37
 image: clear 37
 image: clone 39

image: continue draw 38
image: draw 36
image: mesh setup 21
image: pilot 39
image: redraw 37
image: reset 40
image: scan 39
image: show picture 39
image:new view on zoom 38

- M -

map: <abs(z-real) or abs(z-imag) 47
map: >abs(z-real) or abs(z-imag) 47
map: abs(z) 47
map: abs(z-imag) 46
map: abs(z-real) 46
map: abs(z-real)+abs(z-imag) 46
map: z-imag 45
map: z-real 45
map: z-real+z-imag 46

- P -

palettes: selecting 60
pixel: phoenix 48
pixel: symmetry 51

- R -

render: add noise 62
render: factors 62
render: filter 57
render: orbit trap values 50
render: reset noise seed 63
render: texture scale 63

- S -

status bar 64

- T -

toolbar 63
type: 2d complexified quaternion 44
type: 2d cubic 43

type: 2d hypercomplex 43
type: 2d quaternion 43
type: complexified quaternion 43
type: cubic 42
type: hypernion 42
type: julia 41
type: mandelbrot 40, 41
type: orbits 44
type: quaternion 42

- V -

video: avi variables 67
video: open avi stream 66
video: view avi 67
video: write frames 67