QuaSZ Application Help

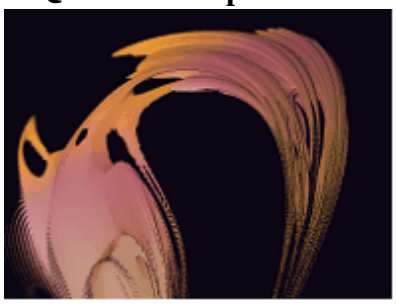
Table of Contents

1 Save As [FO6] -- x64 2

Save data in Fractal Orbits x64 compatible format. This allows you to transfer old zp data files to a file format that can be loaded by Fractal Orbits x64, the 64 bit version of Fractal Orbits.

2 Main Index

QuaSZ Help Index



Getting Started

QuaSZ Remote

Channel Guide

An Introduction To CQuat Fractals by Terry W. Gintz

Commands

File menu
Edit menu
Image menu
Type menu
Render menu
Pixel menu
View menu
Window menu
Video menu
Demo menu
Help menu

2.1 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

2.2 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.

2.3 Size (system) command

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.

2.4 Minimize (system) command

Minimize command (application Control menu)

Use this command to reduce the QuaSZ window to an icon.

Shortcut

Mouse: Click the minimize icon  on the title bar.
Keys: ALT+F9

2.5 Maximize (system) 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.

2.6 Move (system) command

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

2.7 Restore (system) command

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.

2.8 Close (system) command

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

2.9 Previous Window (system) command

Previous Window command (drawing Control menu)

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

Shortcut

Keys: SHIFT+CTRL+F6

2.10 Next Window (system) command

Next Window command (drawing Control menu)

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

Shortcut

Keys: CTRL+F6

2.11 Switch to (system) command

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.

2.12 Tutorial

An Introduction To CQuat Fractals By Terry W. Gintz

In the process of exploring all possible extensions to a fractal generator of this type, I considered using discrete modifications of the standard quaternion algebra to discover new and exciting images. The author of *Fractal Ecstasy* [6] produced variations of the Mandelbrot set by altering the discrete complex algebra of z^2+c. The extension of this to quad algebra was intriguing. There was also the possibility of different forms of quad algebra besides quaternion or hypercomplex types.

Having modeled 3D fractals with complexified octonion algebra, as described in Charles Muses' non-distributive algebra [7], it was natural to speculate on what shapes a "complexified" quaternion algebra would produce. Would it be something that was between the images produced with hypercomplex and quaternion algebra? Quaternion shapes tend to be composed of mainly rounded lines, and hypercomplex shapes are mainly square (see Figures 1 and 2.)

Figure 1. Quaternion Julia set of -1+0i

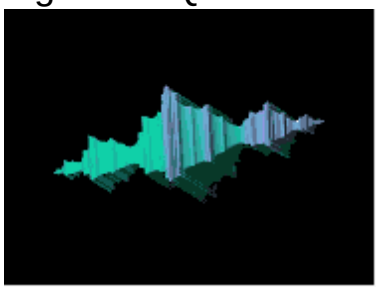


Figure 2. Hypercomplex Julia set of -1+0i

For those not familiar with the basics of hypercomplex and quaternion algebra, here are the algebraic rules that define how complex components interact with each other:

	i	j	k
i	-1	k	$-j$
j	k	-1	$-i$

k	*-j*	*-i*	1

Table 1 Hypercomplex variable multiplication rules

	i	*j*	*k*
i	-1	*k*	*-j*
j	*-k*	-1	*i*
k	*j*	*-i*	-1

Table 2 Quaternion variable multiplication rules

In both quaternion and hypercomplex algebra, $i^2=-1$. The hypercomplex rules provide for one real variable, two complex variables, (*i* and *j*) and one variable that Charles Muses refers to as countercomplex (*k*), since *k*k* = 1. It would appear from this that k = 1, but the rules in Table 1 show that k has complex characteristics. In quaternion algebra there is one real variable and three complex variables. In hypercomplex algebra, unlike quaternion algebra, the commutative law holds; that is, reversing the order of multiplication doesn't change the product. The basics of quaternion and hypercomplex algebra are covered in Appendix B of *Fractal Creations* [8]. One other concept important to non-distributive algebra is the idea of a "ring". There is one ring in quaternion and hypercomplex algebra (*i*,*j*,*k*). (There are seven rings in octonion algebra.) If you start anywhere in this ring and proceed to multiply three variables in a loop, backwards or forwards, you get the same number, 1 for hypercomplex, and 1 or -1 for quaternion, depending on the direction you follow on the ring. The latter emphasizes the non-commutative nature of quaternions. E.g. : using quaternion rules, *i*j*k* = *k*k* = -1, but *k*j*i* = *-i*i* = 1.

For "complexified" quaternion algebra, the following rules were conceived:

	i	*j*	*k*
i	-1	*-k*	*-j*
j	*-k*	1	*i*
k	*-j*	*i*	1

Table 3 CQuat variable multiplication rules

Note that there are two countercomplex variables here, *(j* and *k*). The commutative law holds like in hypercomplex algebra, and the "ring" equals -1 in either direction. Multiplying two identical quad numbers together, $(x+yi+zj+wk)(x+yi+zj+wk)$ according to the rules of the complexified multiplication table, combining terms and adding the complex constant, the following iterative formula was derived for the "complexified" quaternion set, q^2+c:

*x -> x*x - y*y + z*z + w*w + cx*
*y -> 2.0*x*y + 2.0*w*z + cy*
*z -> 2.0*x*z - 2.0*w*y + cz*
*w -> 2.0*x*w - 2.0*y*z + cw*

Just to get a feel for this new formula, a fairly basic constant, -1+0i, was used for the initial 3D test. The extraordinary picture "Equilibrium"(Figure 3) was the result.

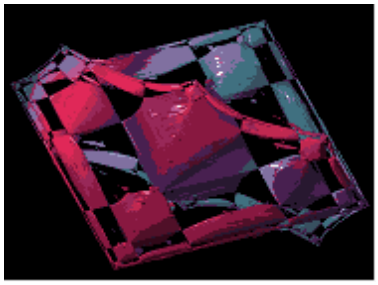


Figure 3. Equilibrium -- cquat Julia rendering of -1+0i

Being familiar with the quaternion and hypercomplex renditions of the Julia set -1+0i, it appeared that this image was a leap into hyperspace; the fractal seemed to literally expand in all directions at once. The next test used a Siegel disk constant, -.39054-.58679i, which Roger Bagula [9] had recently sent. The Siegel image (Figure 4) strongly suggested that cquats were indeed a new form of space-filling fractal.

Figure 4 Siegel -- cquat Julia rendering of -.39054-.58679i

Since then, Godwin Vickers has ported the cquat formula to the *Persistence of Vision Ray-tracer* [10], and verified that the equilibrium image wasn't just an artifact of QuaSZ. Nearly identical images have been obtained in POV, using Pascal Massimino's [11] custom formula algorithm for 3D Mandelbrot and Julia sets.

There remains the extension of cquat algebra to transcendental and exponential functions. Any ideas for this are welcome. The built-in formulas in QuaSZ have been revised to include cquat variations where possible.

References

1. Hamilton, W. R. (1969) *Elements of Quaternions, Vol. I and II*, reprinted by Chelsea Publishing Co.: New York
2. Mandelbrot, B. (1983) *The Fractal Geometry of Nature*, Freeman, San Francisco.
3. Norton, A. (1982) Generation and display of geometric fractals in 3D, Computer Graphics (ACM-SIGGRAPH) July 23(3): 41-50.
4. Vickers, Godwin, http://www.hypercomplex.org
5. Gintz, T. W. (1989-2003) *Fractal Zplot*, originally Zplot.
6. *Fractal Ecstasy* (1993) Deep River Publishing, Inc.
7. Muses, C., http://www.innerx.net/personal/tsmith/NDalg.html
8. Tim Wegner and Bert Tyler (1993) *Fractal Creations*, Waite Group Press: CA.
9. Bagula, R., http://home.earthlink.net/~tftn/
10. POV Team, *Persistence of Vision Ray-tracer*, Victoria, Australia, http://www.povray.org/
11. Massimino, P., http://skal.planet-d,net/quat/Compute.ang.html#JULIA

3 QuaSZ Remote

QuaSZ Remote

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

3.1 Channel Guide

Channel Guide

The eight channels accessed via the QuaSZ remote:

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

CP: Random Composite fractals

3.2 New button

New button

Use this button to open a new drawing window in QuaSZ. This is useful to view minor changes to a drawing. Use the Copy Data and Paste Data commands from the Edit menu to transfer current drawing parameters to the new window.

3.3 Undo button

Undo button

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

3.4 Size button

Size button

This allows you to set the drawing area for a picture, independent of the Windows screen size. It also shows which size is currently in use. The aspect for the drawing is based on the ratio of X (horizontal width) to Y (vertical height.) The size of an image can range in standard 4/3 and 1/1 aspects from 160X120 to 3564X2784 or you can choose a custom XY size. 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.

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

3.6 Batch button

Batch button

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

3.7 Text button

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

3.8 Pilot button

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

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

3.9 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 QuaSZ 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.

3.10 View button

View button

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

3.11 Scan button

Scan button

This generates a 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 key.

3.12 Rend button

Rend button

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, click on the Rand Rend button or select Random Render from the Demo menu.

3.13 Help button

Help button

Use this button to open the help index for QuaSZ.

3.14 Palette button

Palette button

Use the palette editor to modify the color palette in use.

3.15 Light button

Light button

Edit lightpoint and viewpoint variables

3.16 Formula button

Formula button

Use this button to change formulas or type.

3.17 Params button

Params button

Use this button to edit 3-D Parameters.

3.18 Q1 button

Random Quaternion (Channel Q1 button)

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 (if lighting is checked in the Random Batch window.)

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.

3.19 Q2 button

Random Quaternion2 (Channel Q2 button)

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 (if lighting is checked in the Random Batch window.)

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.

3.20 C1 button

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 (if lighting is checked in the Random Batch window.)

3.21 C2 button

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 (if lighting is checked in the Random Batch window.)

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.

3.22 CJ button

Random Cubic Julia (Channel CJ button)

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

3.23 O1 button

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.)

3.24 O2 button

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 (if lighting is checked in the Random Batch window.)

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

3.25 Cp button

Random Composite (Channel CP button)

This generates a random composite Julia quaternion set according to the options set in the Batch window. This is the same as the Demo/Random Composite command.

3.26 Random Render button

Random Render button

The current plot is ray-traced using random rendering values (randomizes coloring filter or generalized coloring parameters.) If the plot has been drawn in the current session (without using undo or reloading) then the ray tracing will take place in real time, else the plot will be redrawn and then ray-traced.

3.27 Save button

Save button

Use this button to save and name the active drawing. QuaSZ 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.

3.28 Load button

Load button

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

3.29 Bmp button

BMP button

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

3.30 Png radio button

PNG radio button

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

3.31 Jpg radio button

JPG radio button

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

3.32 ||||| button

||||| button

Through a series of windows, this allows you to name and open an avi animation stream and choose

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

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

3.33 > button

> button

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

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

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

3.34 [] button

[ ] button

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

3.35 V button

V button

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

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

Use the Open button to view a different avi file. Use the Save button to save the file in a different compression format. You must use a different name to save the file than the name that was used to

open it. Click on the left-mouse button or any key to abort a save operation.

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

4 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 Palette [PQZ]	Load palette file.
Load Texture	Load QuaSZ texture file [QTX]
Open [JPG]	Load jpeg.
Open [PNG]	Load png.
Open [JPG+QZB]	Load JPEG + Zbuffer file.
Open [QSZ-MAC]	Load text (platform-independent) data file.
Save Parameters	Save parameters for an opened drawing to a specified file name.
Save Par File	Save a parameters definition file.
Save Palette [PQZ]	Save palette to file.
Save Texture	Save texture file [QTX].
Save As [JPG]	Save in jpeg format.
Save As [PNG]	Save in png format.
Save As [JPG+QZB]	Save in JPEG format with Zbuffer.
Save As [QSZ-MAC]	Save data in text (platform-independent) format.
Save As [QS6] -- x64	Save data in QuaSZ x64 compatible file format

Import Options

Palette [MAP]	Load a Fractint map file.
Cubic Parameters [MAP]	Load parameter file created with Cubics
PodME Parameters [MAP]	Load parameter file created with PodME
Hydra Parameters [MAP]	Load parameter file created with Hydra

Export Options

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.
Set Max Faces	Set target face size for mesh-simplification option.
Save As [STL]	Save polygonized quaternion as STL solid file.
Save as [WRL]	Save polygonized quaternion as virtual reality file.

Save as [DXF]	Save polygonized quaternion as AutoCad dxf file.
Save as [PLY]	Save polygonized quaternion as Ply file.
Set Max Indices	Set maximum number of vertices allocated for Q polygon.
Exit	Exits QuaSZ.

4.1 New command

New command (File menu)

Use this command to create a new drawing window in QuaSZ. 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

4.2 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

4.2.1 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 QuaSZ stores the file that you want to open.

Directories

Select the directory in which QuaSZ stores the file that you want to open.

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

4.3 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:

4.4 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, QuaSZ 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

4.5 Save As command

Save As command (File menu)

Use this command to save and name the active drawing. QuaSZ 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.

4.5.1 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. QuaSZ adds the extension .qsz to

the data file.

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.

4.6 Load Other

4.6.1 Parameters command

Load Parameters command (File menu)

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

4.6.2 Par File

Load Par File command (File menu)

Opens a window that allows the user to select and convert Fractint-like par files that have been previously saved with QuaSZ. 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.

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

Note: though the QuaSZ par reader is technically compatible with most Fractint par files up to v19.3 (excluding non-deterministic fractal types), most of the built-in formulas in QuaSZ are different from Fractint's formula set, so the likelihood of a Fractint par file producing a useable image in QuaSZ is small. There is more chance that a Fractint formula that contains a simple user-defined formula (frm) will work in QuaSZ, since the QuaSZ parser automatically uses quad math for every operation in a user-defined formula.

Related Topics:
Save Par File has more information on saving par files.

4.6.3 Palette command

Load Palette command (File menu)

Use this command to load a palette file [.pqz]. The palette file contains a palette created previously with QuaSZ. You also have the option in the file descriptor box to select palette and coloring filter, to reload both palette and the coloring filter that was saved along with it.

4.6.4 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].

4.6.5 Open [JPG] command

Open [JPG] 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.

4.6.6 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.

4.6.7 Open [JPG+QZB] command

Open [JPG+QZB] command (File menu)

Use this command to load parameters, zbuffer file and a bitmap file that was saved in jpeg format. This option is provided so that you can continue a plot that was saved in another draw session. Note: the last files list doesn't keep track of images loaded in JPEG format.

4.6.8 Open [QSZ-MAC) command

Open [QSZ-MAC) command

Load text (platform-independent) data file. Useful for transferring data files between QuaSZ Mac and QuaSZ.
Incompatible with the binary data file with the same file extension [QSZ].

4.7 Save Other

4.7.1 Parameters command

Save Parameters command (File menu)

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

4.7.2 Par File

Save Par File command (File menu)

This option allows you to save QuaSZ parameters and most coloring 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 QuaSZ. The par definitions can be easily posted on the net and reloaded in QuaSZ 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 .qsz (data file). 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, QuaSZ 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.

Generalized coloring parameters and external rendering library functions are not saved in the par file.

4.7.3 Palette command

Save Palette command (File menu)

Use this command to save a palette for the current image in a palette file [.pqz]. Also saves the Coloring Filter used for surface mapping.

4.7.4 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].

4.7.5 Save As [JPG] command

Save As [JPG] 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.

4.7.6 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.

4.7.7 Save As [JPG+QZB] command

Save As [JPG+QZB] command (File menu)

Use this command to save the parameters, zbuffer and active bitmap in jpeg format. This option is provided so that you can continue a plot in another draw session. Not available for anti-aliased pictures. To enable this option the zbuffer must be created first by initializing a plot, using the Draw command or Auto-Draw option. Note: the last files list doesn't keep track of images saved in JPEG format.

4.7.8 Save As [QSZ-MAC) command

Save As [QSZ-MAC) command

Save data in text (platform-independent) format. Useful for transferring data files between QuaSZ Mac and QuaSZ. This command is disabled for images that use external formula libraries, Custom Quad, Dive, Loxodromic (front end), Pict-Trap coloring or external rendering libraries that Process Bailout, to ensure a minimum of compatability between programs. Some differences between rendering styles may be observed, but image shapes should be maintained.

4.7.9 Save As [QS6] -- x64

Save data in QuaSZ x64 compatible format. This allows you to transfer old qsz data files to a file format that can be loaded by QuaSZ x64, the 64 bit version of QuaSZ. Excluded are external rendering library data, since the 32 -bit plugins are incompatible with QuaSZ x64.

4.8 Import

4.8.1 Palette command

Import -> 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.

4.8.2 Cubic Parameters command

Import -> Cubic Parameters command (File menu)

Use this command to load a Cubics data file [.cbs]. The data file contains all variables to recreate

an image created previously with Cubics.

4.8.3 PodME Parameters command

Import -> PodME Parameters command (File menu)

Use this command to load a PodME data file [.pme]. The data file contains all variables to recreate an image created previously with PodME, except certain texture options such as external coloring libraries. The option "Pixel/Loxodromic/PodME Composites" is selected automatically with this command.

4.8.4 Hydra Parameters command

Import -> Hydra Parameters command (File menu)

Use this command to load a Hydra data file [.hda]. The data file contains all variables to recreate an image created previously with Hydra, except certain texture options such as external coloring libraries. The formula type is set to "custom quad", if applicable, and the Hydra formula set is used.

4.9 Export

4.9.1 Save As [OBJ] command

Export -> Save as [OBJ] 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 writes the triangles to a Wavefront object 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, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object becomes. Precision is set with the Steps variable in the Quaternion window, where precision=10/ Steps.

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.

4.9.2 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.

4.9.3 Save As [POV] command

Export -> Save as [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 precision=10/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.

4.9.4 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.

4.9.5 Set Max Faces command

Export -> Set Max Faces command (File menu)

Here you can set the target face size for a Wavefront object, if the Simplify option is selected. The face size can range from 1000 to 500,000 faces. This allows you to reduce the size of the Wavefront object file by a factor of 30 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).

4.9.6 Save As [STL] command

Export -> Save As [STL] 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 STL solid file. STL files are used with 3D printers and other machinery. 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, 2MB 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. See also the Simplify mesh command for ways to reduce object file size.

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.

4.9.7 Save As [WRL] command

Export -> Save as [WRL] 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 writes the triangles to a virtual reality 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, 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 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.

4.9.8 Save As [DXF] command

Export -> Save as [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 writes the triangles to an AutoCad 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, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object becomes. Precision is set with the Steps variable in the Quaternion window, where precision=10/ Steps.

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.

4.9.9 Save As [PLY] command

Export -> Save as [PLY] 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 writes the triangles to a Ply (polygon format) object 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, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object becomes. Precision is set with the Steps variable in the Quaternion window, where precision=10/Steps.

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.

4.9.10 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. Bryce4 has problems with object files produced by QuaSZ that are much larger than 2.5MB (on systems other than Win XP.)

4.10 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.

4.11 Exit command

Exit command (File menu)

Use this command to end your QuaSZ 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

5 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	Paste data from copy buffer.
Formulas/Type	Edit formula/type data.
Apply (Formula Library Functions (rll))	Apply external formula library function..
Formula Libraries Off	Disable external formula libraries.
Drawing Parameters	Edit drawing window parameters.
Size	Sets the image size.
Quaternion	Edit quaternion parameters.
Cubic Values	Edit cubic parameters.
Octonion Values	Edit octonion constants.
Ray-Tracing Variables	Edit lighting and viewpoint variables.
Palette Editor	Edit palette.
Text	Edit and add text to drawing.
Preferences	Startup preferences and defaults.

5.1 Undo command

Undo command (Edit menu)

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

Shortcut
Keys: CTRL+Z

5.2 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

5.3 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

5.4 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

5.5 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.qsz". The current palette for the view is also copied.

Shortcut
Keys: CTRL+F

5.6 Paste Data command

Paste Data command (Edit menu)

Use this command to paste the data in the file "c:\zcopy.qsz" to the active view.

Shortcut
Keys: CTRL+R

5.7 Formula Window

Formula/Type Window

Fun #1 and Fun #2 are combo controls for entering/selecting up to two formulas in the 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 is additionally a Type control, an Arg control, an 'S' control and an Arg Limit control that are used in various fractal types and formulas. (The Arg Limit variable is the same variable as the 4th Center variable in the Cubic editor window.) User-defined functions are supplied through list boxes f1-f4.

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

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

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

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

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

Type 5 alternates between the two functions while iterating.

Type 6 takes the quotient of both functions.

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

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

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

Type 10 uses both fun#1 and fun#2 in combination to produce Escher-like quaternion sets, loosely based on an algorithm from **The Science of Fractal Images**. Fun#1 sets the stage for the target set fun#2. Start with z=z*z for fun#1 and p0 for fun#2. Fun#1 may be any formula other than z=z*z, but may produce unsymmetrical tilings. Fun#2 can be any built-in formula. This results in a 3D tiling effect on the x/y axis that can be multiplied by increasing the number of terms in Fun #1, as in z=z*z*z.

Type 11 uses Fun#1 to produce "genetic" style fractals. Here you enter a character string or word that can contain any of 25 letters 'a' through 'y'. Each letter or "gene" references a built-in formula, as defined in the Genes window. For every iteration the program cycles through the character string and uses whatever formula is referenced by the characters. Notes: If Max. Iter is shorter than the gene string then only part of the string is used. For quaternions, which usually have a very short iteration cycle, changing Max. Iter can have a dramatic effect on the fractal output.

The 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.

LV1, LV2 and LV3 are used to vary the loxodromic formulas o1-o6, and LV3 to vary n1-n2. Each LVar has a different effect on a formula, and can range positive or negative. There are no

particular ranges for these variables, but start with small values less than 2 or greater than -2 to begin with.

The four buttons named Random fun#1, Random fun#2, Random f1-f4 and Random Formula are used to pick formulas/functions at random. Clicking on Random fun#1, a formula is chosen (from the first 100 built-in formulas) for fun #1. Clicking on Random fun#2, a formula is chosen for fun #2. Clicking on Random f1-f4 randomizes all four user-defined functions. Clicking on Random Formula, a random formula is generated in the Formula box and the Type is set to 9. The Level box determines the complexity of the formula generated.

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. QuaSZ 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 proceeding ';'. These are provided to allow QuaSZ users to more easily convert Fractint formula types to QuaSZ use. 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. Note: only a small subset of Fractint variables are supported by QuaSZ, so complicated Fractint formulas may not work, or could give misleading results.

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. This is 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 QuaSZ's startup directory.

Click on the Okay button to use the formulas currently displayed in the window, or Cancel to exit the window without making any changes. Click on Apply to apply a new formula, etc. without closing the Formula window.

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

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. To rotate the extra four octonion dimensions (E-K) use the following syntax:

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

To normalize the C and CC constants:
n -- normalize C and CC (default is un-normalized)
Alternate octonion initialization:
c -- set OE-OK to .01 at beginning of each iteration

5.8 Parameters Window

Drawing Parameters Window

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.

Select the Okay button to start a new plot from column 1. Select the Continue button to continue a plot at the row it left off, if it is not a complete drawing.

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

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

5.9 Formula Library Functions

5.9.1 Apply command

Apply (external formula function) command (Edit menu)

Use one of the external-plug-in formulas. The external formulas override any formula set in the Edit Formula/Type window.

5.9.2 Formula libraries off command

Formula Libraries off command (Edit menu)

Disable and deallocate the external formula function selected.

5.10 Size command

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 size of an image can range in standard 4/3 and 1/1 aspects from 160X120 to 3564X2784 or you can choose a custom XY size. 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.

5.11 Quaternion

5.11.1 Initial Quaternion Values

Initial Values for Quaternion Window

This is the data-collection window for the 3-D generator.

Min X, Max X, Min Y and Max Y are the spatial variables for framing the quad 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 quad 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 quad object. This has the effect of exposing the insides of a quad set.

Quad constants are cr, ci, cj and ck.

Three rotate variables determine the 3-D angle of rotation.

The Bailout variable can range from .000001 to 65000. For most escape-time formulas that have an attractive point at infinity, the bailout is set to a value 4 or greater. A larger value tends to make the figure smoother and fuller with some loss of detail. When the variable is set to a value less than one, then a bailout routine geared to convergent formulas such as built-in formulas q0-q9. The bailout uses Alessandro Rosa's "cutoff rate" routine to limit the basins of attraction (to viewable 3D areas) for formulas that use Newton's method, etc. This method also works for escape-time formulas, though slower and requiring more iterations to bring out detail.

Steps and Fine Tune are pitch adjustments that bear on the quality of the plot at the expense of lengthier calculations.

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

Use the Random Rotate button to set random values (0-360) for the Rotation variables.

When you click on Okay or Apply, the quaternion generator looks at the Fun#1 gadget in the Edit/

Formula window. If this contains a preset variable that function is iterated for its escape time, then the results are ray-traced in any 3-D object that may be created. If Fun#1 contains no preset variable, and the Type is set to 9, any custom formula (Fractint-style: as in 'z=z^3+c#') in the Formula box is used; else the original quaternion Julia formula (q^2+c) is iterated and ray-traced. Not all functions may produce a usable 3-D 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 Plane variables B (Back), F (Front), and P (Position) allow you to flatten part of the quaternion figure. For normal plotting, these variables default to 0. To have any effect on the image, the Back variable must be different in value from the Front variable. The difference between back and front variables determines where the image is flattened. These variables are limited to +/-9.99, with normal values being in the range –z to +z. The position variable sets the color of the flattened plane.

5.11.2 Cubic Values

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

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

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

9 -- compute M+ and M-, use M+ or difference of two if M+>M-

Args 3-9 affect only cubic Mandelbrots.

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

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

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

5.11.3 Octonion Values

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.

5.11.4 Ray-Tracing Variables

Ray-Tracing Window

The LightPoint 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 quaternion and the apparent sheen on the quaternion's surface. The ambient value controls the amount of ambient light that illuminates the quaternion. The highlight value increases or decreases the specular (Phong) highlighting, while the gamma 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 quaternion.

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.

5.12 Palette command

Palette command (Edit menu)

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

It is important to realize that palettes are software-simulated in QuaSZ (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 curent palette in QuaSZ. You can then save the palette in a .pqz file, or by saving the entire function and bitmap.

Colors are shown in 8 groups of 32 colors. With QuaSZ, 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 QuaSZ, colors do not cycle smoothly when you adjust the RGB/HSV sliders. This would be too slow with true color. The Map button is used to map color changes to an image after you are done adjusting the sliders. In the HSV mode, color spreads are based on HSV values instead of RGB values, which in some cases results in brighter color spreads.

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

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

Use Reverse to reverse the order of the colors in the palette. This affects only those colors in the start-color to end-color range.

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

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.)

5.12.1 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. This is useful for reversing divide-by-eight palettes, etc., for orbit-trap pictures that require a reversed palette.

5.12.2 Neg Button

Neg button

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

5.12.3 Map Button

Map button

In QuaSZ, 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.

5.12.4 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.

5.12.5 Spread Button

Spread button

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

5.12.6 Copy Button

Copy button

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

5.12.7 SRG Button

SRG button

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

5.12.8 SRB Button

SRB button

Use SRG to switch the red and blue components of all palette colors. This is for RGB mode only.

5.12.9 Okay Button

Okay button

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

5.12.10 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.

5.12.11 Cancel Button

Cancel button

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

5.12.12 Red Slider

Red slider

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

5.12.13 Green Slider

Green slider

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

5.12.14 Blue Slider

Blue slider

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

5.12.15 Red edit box

Red edit box

Shows red/hue value of selected color index.

5.12.16 Green edit box

Green edit box

Shows green/saturation value of selected color index.

5.12.17 Blue edit box

Blue edit box

Shows blue/value magnitude of selected color index.

5.12.18 Random Palette Button

Random palette button

Use to create a random palette. Fast way to define palettes.

5.13 Edit Text command

Text (Edit menu)

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

5.14 Preferences command

Preferences (Edit menu)

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

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

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

6 Image Menu

Image menu commands

The Image menu offers the following commands:

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

Show Dialogs	Show drawings for active drawing.
Abort	Abort drawing.
Continue	Continue drawing.
Zoom	Zoom into rectangle.
New View on Zoom	New view on zoom.
Clone	Clone current view.
Cycle Colors	Cycle colors.
Pilot	Use Pilot to rotate figure and alter key cubic variables.
Scan	Scan Mandelbrot border for quaternion Julia set.
Dive	Peel off outer layer of quaternion.
Ray Trace	Ray trace 3-D plot.
Reset->	Reset coordinates, current figure or all figures
Figure 1	Switch to figure one.
Figure 2	Switch to figure two.
Figure 3	Switch to figure three.
Figure 4	Switch to figure four.
Composite	Select figures to merge.

6.1 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.

6.2 Draw Composite command

Draw Composite command (Image menu)

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

6.3 Plot to file

Plot to File (Image menu)

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

6.4 Plot Files in Directory

Plot Files in Directory (Pixel menu)

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 (.qsz) in the working directory are enlarged to this resolution. Click on Okay to start. Note: the 3200X2400 bitmap size is suitable for 8 1/2X11 printouts at 320-720 dpi. The larger bitmap sizes are suitable for poster-size printouts. Merging, anti-aliasing and solid guessing are disabled when using this option.

6.5 Auto

6.5.1 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.

6.5.2 Clear command

Auto Clear command (Image menu)

With this command enabled (on by default), the drawing area is cleared before starting a new plot. You can turn off this option when you want to see the effect of minor changes to parameters, as they affect the plot pixel by pixel, or when setting up a multiple-layered fractal. Note: when you disable auto clear, no pre-image is drawn while the image is being calculated, and any background image (loaded through Open [Jpeg] or Open [Png]) is retained. You can use the shift-c command (hot keys) to clear the drawing area at any time.

6.5.3 Dust command

Auto Dust command (Image menu)

With this command enabled (on by default), single pixels and quaternion threads one pixel wide are removed after ray tracing. Sort of like the de-speckle filter in some image-processing programs.

6.5.4 Auto Alert command

Auto Sound Alert command (Image menu)

A sound clip is issued 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.

6.5.5 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.

6.5.6 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. QuaSZ 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 QuaSZ.

6.6 Merge Colors

6.6.1 Merge Sum command

Merge Sum command (Image menu)

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

6.6.2 Merge And command

Merge And command (Image menu)

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

6.6.3 Merge Or command

Merge Or command (Image menu)

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

6.6.4 Merge High command

Merge High command (Image menu)

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

6.6.5 Merge Low command

Merge Low command (Image menu)

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

6.6.6 Merge Back command

Merge Back command (Image menu)

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

6.6.7 Merge Diff command

Merge Diff command (Image menu)

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

6.7 Hide dialogs command

Hide dialogs command (Image menu)

Use this command to hide all open non-modal dialogs in the active window. This helps to de-clutter the screen and avoid confusion if two or more draw windows are open. Each draw window has its own dialogs. Disabled if no dialogs are open.

6.8 Show dialogs command

Show dialogs command (Image menu)

Use this command to show all open non-modal dialogs in the active window. This command restores any dialogs that may have been hidden by the Hide dialogs command. Each draw window has its own dialogs. Disabled if no dialogs are open or hidden. Note: you can restore individual dialogs by selecting the command that opened them originally.

6.9 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 QuaSZ 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.

6.10 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. Continue is disabled when an Image/Merge option is selected. Continue isn't available for 3-D images that have been reloaded from a previous drawing session.

6.11 Zoom command

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 up and down arrow keys to squash or expand the box, changing the aspect of the image. Use the Shift key to enlarge the zoom box X4 for quickly zooming outward. Use the Ctrl key to shrink the zoom box by 4. 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 fast-floating point (32 bits).

Note: As you zoom inward it will probably be necessary to increase the Steps variable (Initial Quaternion Values window) to see the finer detail and avoid a jagged look.

6.12 New View on Zoom command

New view on zoom (Image menu)

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

6.13 Clone command

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.

6.14 Color Cycle command

Cycle Colors command (Image menu)

Use this command to cycle colors when not plotting. Works with only palette-based coloring, and not with anti-aliasing. Undoing an action disables the cycle command until the image is redrawn.

6.15 Pilot command

Pilot (Image menu)

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

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

6.16 Scan command

Scan (Image menu)

This is 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. 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.

6.17 Dive command

Dive (Image menu)

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

6.18 Ray Trace command

Ray Trace (Image menu)

Uses Frode Gill's ray-tracer algorithm to add a light source to 3-D plots. Color palettes should be continuous (dark to light to dark) to take best advantage of this option. The light source parameters may be altered in the Ray-Tracing Variables window. This option is the default. Note: if you prefer to generate quaternions without ray tracing, deselect this flag before drawing the plot. The image draw will be the pre-image, using the existing palette, as is, without ray-tracing (Phong or shading.)

6.19 Full Screen command

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.

6.20 Reset ->

Reset (Image menu)

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

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

6.21 Figure

6.21.1 1

Figure #1 (Image menu)

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

6.21.2 2

Figure #2 (Image menu)

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

6.21.3 3

Figure #3 (Image menu)

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

6.21.4 4

Figure #4 (Image menu)

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

6.21.5 Composite command

Composite command (Image menu)

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

7 Type Menu

Type menu commands

The Type menu offers the following commands:

Mandelbrot0	Mandelbrot set (orbit starts at zero.)
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.
Custom Quad...	Set fractal type to user-generated quad type.

7.1 Mandelbrot0

Mandelbrot0 (Type menu)

Mandelbrots base their mapping on varying inputs of complex C, which corresponds to the min/max

values set in the Parameters window. With Mandelbrot0, the initial value of Z is set to zero.

7.2 MandelbrotP

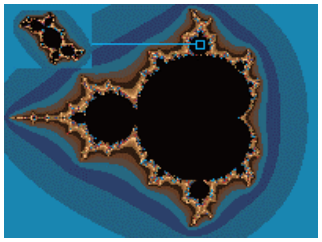
MandelbrotP (Type menu)

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. Usually used with cubic Mandelbrot formulas.

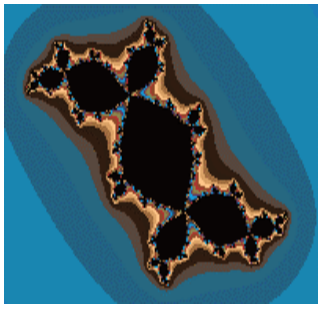
7.3 Julia

Julia (Type menu)

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

7.4 Quaternion command

Quaternion (Type menu)

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

7.5 Hypernion command

Hypernion (Type menu)

Use this command to set the fractal type to a hypercomplex 3-D quaternion. Variables that affect this fractal type are defined in the Edit/Quaternion window. A hypernion uses hypercomplex math to shape the 3-D object, which usually results in a squared-off shape, rather than the rounded shape of the typical quaternion. Cubic and octonion formulas can produce hypernion shapes that aren't like any standard formula.

7.6 Cubic command

Cubic (Type menu)

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

Cubic Mandelbrots quaternions use the S and Si variables to set the initial value and range of the 3rd dimension. Starting from an initial value of 'S', Si is normally set to 2*abs (S), when the Type is Mandelbrot. Si can also be set to center the 3rd dimension on something besides 0.0. Si has no affect in 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 (in the Formula window or Cubic Values 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.

7.7 Complexified Quaternion command

Complexified Quaternion (Type menu)

Use this command to set the fractal type to a 3-D 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 3-D 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.

7.8 Custom Quad command

Custom Quad (Type menu)

This command sets the fractal type to a custom type mapping. Here you can define your own quad math using a 3X3 sign matrix. The Hydra formula set is used for custom quad fractals. See the tutorial on cquat math for further information about the sign matrix and how it relates to quad math.

7.8.1 Custom Sign Matrix Editor

Custom Sign Matrix Editor

Here you enter a value (-1 or 1) into each of the squares that make up the sign matrix. The sign matrix determines the associative characteristics of the elements that make up the quad variable. You can use the row or column check boxes to swap one column in the matrix with another column, or one row with another row. Select on each side of the columns or rows which column or row is to be swapped with the other, then click on Swap Cols or Swap Rows. There is also a button to create a random sign matrix.

8 Render Menu

Render menu commands

The Render menu offers the following commands:

Coloring Filter	Define coloring filter.
Surface Filter	Define surface filter.
X Mapping Filter	Use bitmap file for quaternion rendering: X mapping.
Y Mapping	Use bitmap file for quaternion rendering: Y mapping.
Z Mapping	Use bitmap file for quaternion rendering: Z mapping.
Remap	Remap bitmap for quaternion rendering.
Anti-Alias	Use anti-aliasing, with 1X4 or 1X2 super-sampling.

Link Coloring To Pixel — Set coloring to match absolute coordinates of image.
Atan Coloring — Use Atan algorithm for coloring.
Bof60 Coloring — Use Bof60 algorithm for coloring.
Potential Coloring — Color by magnitude of z.
Filter — Choose an optional tail-end filter.
Orbit Traps — Set orbit trapping method.
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.
Generalized Coloring->
- Apply — Apply non-palette based coloring method.
- Blend->
  - linear scale — linear scale color blending.
  - average — average color blending.
  - subtractive — subtractive color blending.
  - sum of squares 1 — sum #1 color blending.
  - sum of squares 2 — sum #2 color blending.
  - sin #1 — sin color blending.
  - atan #1 — atan #1 color blending.
  - additive — additive color blending.
  - log — log color blending.
  - atan #2 — atan #2 color blending.
  - atan #3 — atan #3 color blending.
  - sin #2 — sin #2 color blending.
  - sin #3 — sin #3 color blending.
  - atan #4 — atan #4 color blending.
  - fractal dimension — fractal dimension color blending.
- Color Parameters — Color controls.
- RGB — RGB color mapping.
- RBG — RBG color mapping.
- GRB — GRB color mapping.
- GBR — GBR color mapping.
- BRG — BRG color mapping.
- BGR — BGR color mapping.
- Triangle Algorithm — Handle color overflow by triangle algorithm.
- Sine Algorithm — Handle color overflow by sine algorithm.
- Gray Scale — Gray scale mapping.
- Invert — Invert colors.
- Fractal Dimension — Apply fractal dimension algorithm to current blend option.

Rendering Library Functions (rll)->
- Apply — Apply external rendering function....
- Process Bailout — Process bailout from external rendering function.
- Rendering Libraries Off — Disable external rendering libraries.

Color Planes	Select color planes for external rendering library.

8.1 Coloring Filter command

Coloring Filter

Here you define a coloring 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-buffer variable and framing variables x and y. Sample function: .1*sin(z)+cos(x*x). The Magnify slider is used to control the intensity of the filter. Click on Apply to apply a new coloring formula without closing the window. Click on Okay to close the window and apply changes. Click on Cancel to close the window without applying changes. Use the Random Filter button to generate a random coloring filter. The best filters will use the z value and one of the other variables (x or y.)

Quaternions normally use palette index one (the second index, zero being reserved for the background color) for their predominant color, with pixel intensities/colors affected by the lighting variables. When the coloring filter formula is defined, up to 255 colors can be used (the full palette) to create mixed textures.

The trig and exponential functions translated include sine (sin), arc sine (asn), cosine (cos), arc cosine(acs), tangent (tan), hyperbolic tangent (th), hyperbolic sine (sh), hyperbolic cosine (ch), log (log), natural log (ln), power (pow), arc tangent (atn), absolute value (abs), exponential (exp) and square root (sqr.)

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

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

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

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

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

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

any of the math functions.

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

8.2 Surface Filter command

Surface Filter

Here you define a surface filter based on a real function. This is like a coloring filter, except that the formula is used to warp the quaternion's shape. The Magnify variable is used to control the intensity of the filter. Click on Apply to apply a new surface filter without closing the window. Click on Okay to close the window and apply changes. Click on Cancel to close the window without applying changes. Use the Random Filter button to generate a random surface filter. The best surface filters will use the z value and one or both of the other variables (x or y.)

8.3 Pic-Trap Coloring

8.3.1 X mapping option

X mapping

This option allows you to use a separate bitmap or picture to color a 3-D image. Enabled when a bitmap has been copied to the clipboard, each pixel of the image is replaced with a corresponding color in the clipboard image, depending on the mapping option used (x, y or z mapping.) This produces textured effects that add realism to the image. Notes: the internal structure of the 3-D image is used to determine each mapping algorithm, so the overall effect is to texture the image like a wood-grain rather than a decal. When you first enable this option, whatever is in the clipboard gets copied to a buffer file for rendering. To change the picture in the buffer, you need to change the clipboard image then use Remap to reinitialize the buffer. The clipboard image isn't saved with the data file, so you need to remember which bitmap file is used for the mapping, to redo a fractal like this later (you can sometimes leave a comment in the Fun#2 edit box in the Edit/Formula window for this purpose.).

8.3.2 Y mapping option

Y mapping

This option allows you to use a separate bitmap or picture to color a 3-D image. Enabled when a bitmap has been copied to the clipboard, each pixel of the image is replaced with a corresponding color in the clipboard image, depending on the mapping option used (x, y or z mapping.) This produces textured effects that add realism to the image. Notes: the internal structure of the 3-D image is used to determine each mapping algorithm, so the overall effect is to texture the image like a wood-grain rather than a decal. When you first enable this option, whatever is in the clipboard gets copied to a buffer file for rendering. To change the picture in the buffer, you need to change the clipboard image then use Remap to reinitialize the buffer. The clipboard image isn't saved with the

data file, so you need to remember which bitmap file is used for the mapping, to redo a fractal like this later (you can sometimes leave a comment in the Fun#2 edit box in the Edit/Formula window for this purpose.).

8.3.3 Z mapping option

Z mapping

This option allows you to use a separate bitmap or picture to color a 3-D image. Enabled when a bitmap has been copied to the clipboard, each pixel of the image is replaced with a corresponding color in the clipboard image, depending on the mapping option used (x, y or z mapping.) This produces textured effects that add realism to the image. Notes: the internal structure of the 3-D image is used to determine each mapping algorithm, so the overall effect is to texture the image like a wood-grain rather than a decal. When you first enable this option, whatever is in the clipboard gets copied to a buffer file for rendering. To change the picture in the buffer, you need to change the clipboard image then use Remap to reinitialize the buffer. The clipboard image isn't saved with the data file, so you need to remember which bitmap file is used for the mapping, to redo a fractal like this later (you can sometimes leave a comment in the Fun#2 edit box in the Edit/Formula window for this purpose.).

8.3.4 Remap command

Remap

This allows you to change the mapping buffer, after initializing an image with X, Y or Z mapping. To change the picture in the buffer, you need to change the clipboard image then use Remap to reinitialize the buffer.

8.4 Anti-Alias ->

Anti-Alias (Render menu)

Applies a 2 to 1 or 4 to 1 averaging filter to every pixel plotted, to reduce jaggies and other high-frequency noise. This increases the processing time 4 to 8 times, so is mainly a final rendering method, not for general development use. For quaternions, only single-pass mode is supported -- or without ray tracing, solid guessing can be used. Note: because of the lengthy time required for applying the anti-aliasing filter, and because anti-aliasing calculates different smoothing colors each time the palette is changed, all palette-switching hot keys and the light bulb button are disabled with the anti-alias flag set.

8.5 Link Coloring To Pixel command

Link Coloring To Pixel (Render menu)

Set coloring to match absolute coordinates of image. This uses extra buffers to track a figure's texture, so that when you rotate it, the texture moves with the figure. Used with the Atan Coloring, Bof60 Coloring, Potential Coloring, Filter, Orbit Traps and the Noise functions.

8.6 Atan Coloring command

Atan Coloring (Render menu)

Uses an Atan algorithm by David Makin to color an image.

8.7 Bof60 Coloring command

Bof60 Coloring (Render menu)

A variation of the Bof60 algorithm found in the classic Pietgen/Ricter text, The Beauty of Fractals, adapted by David Makin to color a 3-D image.

8.8 Potential Coloring command

Potential Coloring (Render menu)

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

8.9 Filter command

Filter (Render menu)

Based on Stephen C. Ferguson's filter algorithms in his program Iterations, this option allows you to choose one of 26 tail-end filters for surface rendering. Corresponds roughly to its effect on the basic Mandelbrot-squared set. The effect will vary with the formula and fractal type chosen.

The Magnify variable is used to intensify or de-intensify the effect of the filter. This value can range from 1-500 nominally. Click on Apply to apply a new filter without closing the window. Click on Okay to close the window and apply changes. Click on Cancel to close the window without applying changes.

8.10 Orbit traps ->

Orbit Traps (Render menu)

Orbit traps are used with the "Link Coloring to Pixel" option.

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.

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.

8.10.1 Orbit trap values

Orbit Trap Values (Render menu)

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.

8.11 Noise

8.11.1 Add Noise command

Add Noise (Render menu)

Add noise to image texture. A variation of Perlin's noise algorithm is used to add natural randomness to an image's coloring.

8.11.2 Factors command

Factors (Render menu)

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.) Additional variables affect the type and shaping of the noise data: Gaussian is an alternate form of noise, while Planet, Check, Tooth, Barber and Wood apply a specific envelope to the noise. The Marble variable is used to introduce a low frequency or high frequency modulation on top of the noise. You can achieve marble-like textures by combining a high frequency marble value with a low frequency Blend value. The marble variable also adds a high-frequency bump map to the wood envelope.

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.

8.11.3 Reset Noise Seed

Reset Noise Seed (Render menu)

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

8.12 Texture Scale command

Texture Scale (Render menu)

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. Click on Apply to apply changes without closing the window. Click on Okay to close the window and apply changes. Click on Cancel to close the window without applying changes.

8.13 Generalized Coloring

8.13.1 Apply command

Apply command (Render menu)

Use this command to switch to Steve Ferguson's generalized coloring mode. Images are colored via the selection in the Blend submenu and color-controls dialog, instead of the palette-based coloring filters. To switch back to palette mode, apply a coloring filter or one of the palette-based textures, such as Atan, Potential or one of the orbit traps.

8.13.2 Blend

8.13.2.1 linear scale command

linear scale command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.2 average command

average command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.3 subtractive command

subtractive command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.4 sum of squares 1 command

sum of squares 1 command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.5 sum of squares 2 command

sum of squares 2 command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.6 sin #1 command

sin #1 command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.7 atan #1 command

atan #1 command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.8 additive command

additive command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.9 log command

log command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.10 atan #2 command

atan #2 (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.11 atan #3 command

atan #3 (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.12 sin #2 command

sin #2 command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.13 sin #3 command

sin #3 command (Blend submenu)
This is color blending for the generalized coloring mode. selected formula is applied while mapping colors to pixels.

8.13.2.14 atan #4 command

atan #4 command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels.

8.13.2.15 fractal dimension command

fractal dimension command (Blend submenu)
This is color blending for the generalized coloring mode. The selected formula is applied while mapping colors to pixels. Uses an early version of Steve Ferguson's fractal dimension algorithm.

8.13.3 Color Parameters command

Color Parameters command (Render menu)

Use this command to adjust the color controls when in the generalized coloring mode.

8.13.4 RGB command

RGB command (Render menu)

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

8.13.5 RBG command

RBG command (Render menu)

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

8.13.6 GRB command

GRB command (Render menu)

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

8.13.7 GBR command

GBR command (Render menu)

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

8.13.8 BRG command

BRG command (Render menu)

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

8.13.9 BGR command

BGR command (Render menu)

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

8.13.10 Sine algorithm command

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 (non-palette mode only.)

8.13.11 Sawtooth algorithm command

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 (non-palette mode only.)

8.13.12 Gray Scale command

Gray Scale command (Render menu)

Use this command to color the active image with gray tones, if in the generalized coloring mode.

8.13.13 Invert command

Invert command (Render menu)

Use this command to invert image colors, if in the generalized coloring mode.

8.13.14 Fractal Dimension command

Fractal Dimension command (Render menu)

Generalized fractal dimension algorithm (S. Ferguson), for use with any blend option (Blend 14 is retained for compatibility with the previous QSZ version.)

8.14 Rendering Library Functions (rll)

8.14.1 Apply command

Apply (external rendering function) command (Render menu)

Use one of the TieraZon2-compatible-plug-in rendering methods. To use, this option automatically enables generalized coloring mode. There is a large potpourri of rendering methods (280+) in the TieraZon/Dofo-Zon Elite and Mark Townsend/Kerry Mitchell libraries that you can experiment with. Some have their own bailout routines that can distort the basic quaternion shape in interesting ways. For version 1.03 plugins, each method can have its own dedicated configuration box, containing as many variables as is needed to customize the function. Pre-1.03 plugins use a fixed configuration box, with only a scaling and orbit-trap variable. Note: selecting one of the palette-based rendering methods in the Render menu, such as Atan or Potential Coloring, or selecting a 2-D fractal Type, will disable and deallocate any external rendering function in use.

8.14.2 Process Bailout command

Process Bailout command (Render menu)

Deselect this option if you don't want to use an external rendering function's bailout routine. The bailout routine can sometimes distort a quaternion shape in undesirable ways, or you may like the coloring effect of the rendering method by itself.

8.14.3 Rendering libraries off command

Rendering Libraries off command (Render menu)

Disable and deallocate the external rendering function selected.

8.14.4 Color Planes command

Color Planes (Render menu)

Since most of the external coloring algorithms were originally written for 2-D fractals, there is the choice which two planes are used for coloring purposes. In the case of 3-D fractals, the X and Y planes don't always match the coloring algorithm to the fractal. With these options you can select which two planes the coloring algorithm uses or select the magnitude of all four planes.

9 Pixel Menu

Pixel menu commands

The Pixel menu offers the following commands:

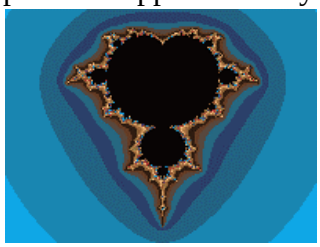
Phoenix	Phoenix orientation.
Invert	Invert image around circle.
Invert Off	Reset inversion flag.
Symmetry->	Horizontal, vertical or XY symmetry.

Switch Z For C	Switch z for c.
Loxodromic	Applies loxodromic function to front end of formula.
Solid-Guessing	Solid-guessing plotting mode.
Fast Quaternion>	Fast quaternion plot for q^2+c.

9.1 Phoenix option

Phoenix option (Pixel menu)

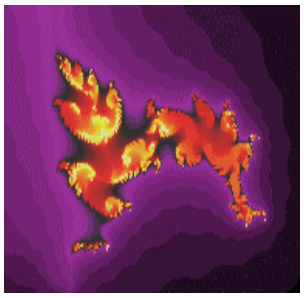
The Phoenix flag rotates the planes, so that the imaginary plane is mapped horizontally and the real plane is mapped vertically.



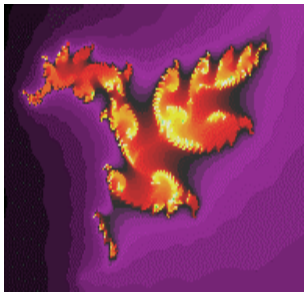
9.2 Invert command

Invert (Pixel menu)

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



Original picture



Inverted

9.3 Invert Off command

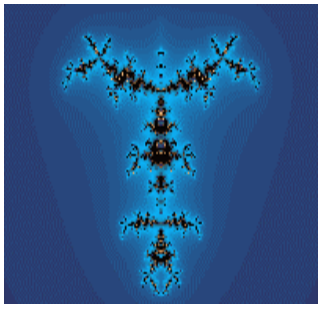
Invert Off (Pixel menu)

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

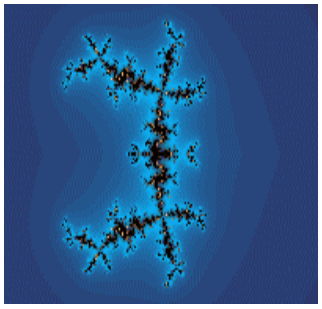
9.4 Symmetry ->

Symmetry (Pixel menu)

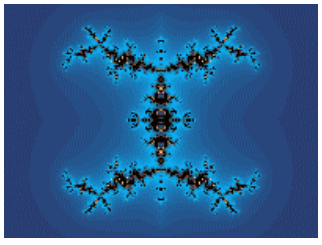
This produces a mirror image from left to right (vertical) or top to bottom (horizontal) or both (xy). You can zoom with symmetry, but the results will be uncertain if the zoom box is off-center on the window.



Vertical symmetry



Horizontal symmetry



XY symmetry

9.5 Switch command

Switch Z For C

When a Switch flag is set, you have switch Z for C. When Z is switched for C, normally you get Mandelbrots from Julia sets and vice versa.

9.6 Loxodromic ->

Loxodromic (Pixel menu)

Uses one of Thomas Kromer's loxodromic algorithms as a front-end process for any formula. There is a choice of the original loxodromic functions or the alternate functions, getatou, ventri, sinus, 'rings of fire', and teres. These functions produce a pronounced biomorphic effect on most formulas, increasing detail and realism. Also see built-in formulas L0-L4 and n0-n3. The option for PodME Composite is included for compatibility with PodME data files that use composite formulas (those that combine both Fun#1 and Fun#2). This adds the complex c after a composite formula has been calculated.
.

9.7 Solid Guessing

Solid Guessing (Pixel menu)

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

9.8 Fast Quaternion

Fast Quaternion (Pixel menu)

Fast Quaternion sets up the function data to optimize quaternion drawing (q^2+c only.) This option uses the qjulia set q^2+c directly in its iteration loop, which is 25-30% faster for generating quaternions than P0.

10 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.

10.1 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 QuaSZ, 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.

10.1.1 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 QuaSZ,

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

Click	To
	Open the QSZ Remote which contains shortcut buttons for many common tasks and options in QuaSZ
	Use Pilot to rotate figure, zoom and alter key cubic variables.
	Open a new drawing, based on the default template.
	Open an existing drawing. QuaSZ 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. QuaSZ displays the Save As dialog box.
	Set image size.
	Edit formula/type data.
	Edit fractal parameters.
	Edit ray-tracing variables.
	Edit palette.
	Draw image from current parameters.
	Continue drawing.
	Zoom into rectangle.
	Show picture full-screen.
	Reset coordinates.

 Draw Mandelbrot set

 Draw Julia set

 Display info about QuaSZ.

 Display QuaSZ's help index.

10.2 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.

10.2.1 status bar

Status Bar

The status bar is displayed at the bottom of the QuaSZ 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.

11 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.

11.1 Cascade command

Cascade command (Window menu)

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

11.2 Tile command

Tile command (Window menu)

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

11.3 Arrange Icons command

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.

11.4 Size Desktop command

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.

11.5 1, 2, ... command

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

QuaSZ 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.

12 Video Menu

Video menu commands

The Audio/Video menu offers the following commands:

Open AVI Stream	Open AVI file for writing and draw initial frame.
Write Frames	Write frames to AVI file.
Close AVI Stream	Close an existing AVI stream.
View AVI	View an AVI animation file.
AVI Composite	Generate composite video.
AVI Object	Output video frames as obj file.
AVI Wrl	Output video frames as wrl files.

12.1 Open Avi Stream command

Open Avi Stream…(Video menu)

Through a series of windows, this allows you to name and open an avi animation stream, set the frames per second, 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. You can also choose no compression for optimum view quality. (All compression methods degrade the original images, some more than others.) The first key frame in the stream is then drawn and written to the file.

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

12.2 Write Frames command

Write Frames…(Video menu)

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

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

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

Notes: key frames are saved in parameter files (qsz), with filenames of "bvf_image#_title.qsz", where '#' is the number of the keyframe and 'title' is the name of the working fractal file. If you open a video stream after setting up a batch mode (Demo menu), then the frames will be written as a series of bmp, obj, or wrl files, depending on whether AVI Object or AVI WRL is also checked.

12.3 Close Avi Stream command

Close Avi Stream (Video menu)

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

12.4 View Avi command

View Avi… (Video menu)

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.

12.5 Avi Composite option

AVI Composite (Video menu)

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

12.6 AVI Object option

AVI Object (Video menu)

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

12.7 AVI WRL option

AVI Wrl (Video menu)

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

13 Demo Menu

Demo menu commands

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

Random Quaternion	Generate random quaternion/hypernion fractal.
Random Quaternion2	Generate random quaternion/hypernion 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 Composite	Generate random composite quaternion/hypernion fractal.
Random Render	Apply random coloring filter.
Batch Mode/Random Setup	Set up initial values for batch mode/random commands.

13.1 Random Quaternion command

Random Quaternion (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, Z is set to 2.0, and the lighting is set for optimum viewing (if lighting is checked in the Random Batch window.)

Note: for some images a Z 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.

13.2 Random Quaternion2 command

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, Z is set to 2.0, and the lighting is set for optimum viewing (if lighting is checked in the Random Batch window.)

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 a Z 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.

13.3 Random Cubic Mandelbrot command

Random Cubic Mandelbrot (Demo menu)

A random cubic Mandelbrot fractal is generated. Like the inverse of a Random Quaternion Julia set, the essential cubic parameters are randomly adjusted to point into a four-dimensional formula G0-G9, 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 (if lighting is checked in the Random Batch window.)

13.4 Random Cubic Mandelbrot2 command

Random Cubic Mandelbrot2 (Demo menu)

A random cubic Mandelbrot fractal is generated. Like the inverse of Random Quaternion Julia set, the essential cubic parameters are randomly adjusted to point into a four-dimensional formula G0-G9, 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 (if lighting is checked in the Random Batch window.)

This option uses the cubic formulas G0-G9, with relaxed parameters to create cubic-Mandelbrot like fractals that may extend to six or more dimensions.

13.5 Random Cubic Julia command

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 Quaternion, 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, Z 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.

13.6 Random Octonion command

Random Octonion (Demo menu)

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

13.7 Random Octonion command

Random Octonion2 (Demo menu)

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

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

13.8 Random Composite command

Random Composite (Demo menu)

A random 3-D Julia fractal is generated using one of the composite Types. Two formulas are selected at random and mixed using one of Types 2-8. This option can be applied to all built-in formulas, including cubic Mandelbrot and octonion formulas.

13.9 Random Render command

Random Render (Demo menu)

A random coloring filter is applied. This changes the surface textures of the 3-D figure.

13.10 Batch Mode/Random Setup

Batch mode/Random Setup (Demo menu)

Here you set parameters for batching and saving random-generated images to disk. You can also customize random variables to direct how the random scanning process works. 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. You can also change the default directory for batch files, by clicking on the "…" button next to the default directory box. 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 3-D fractal hasn't been found.)

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

For quaternions and octonions, you have the option of selecting only quaternion types, only hypernion types (hypercomplex) or a mixture of quaternion/hypernion types.

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

Formula -- (default on) check to randomize built-in formula used
Lighting -- (default off) check to set default lighting
Symmetry -- (default off) check to randomize symmetry used
Rotation -- (default on) check to randomize camera angles
Coloring -- (default off) check to reset coloring parameters
Iteration -- (default off) check to randomize iterations
Z-Space -- (default on) check to set default z-space
Constants -- (default on) check to randomize the complex constants cj-cK
Custom Formula -- (default off) check to generate a random quaternion/hypernion formula
Slices -- (default off) check to randomize slice in Initial Values window
Genetic -- (default off) check to randomize genetic word strings and 'genes' in Formula window
S/Arglimit -- (default off) check to randomize 's' variable and arglimit variable

The Bounds variable (default 0) acts to delimit the boundary scan after finding a random Julia set. Since the scanning process is closely connected with the Mandelbrot set boundaries, most quaternions found this way are very connected/closed figures. The bounds variable adds a random distance from the Mandelbrot boundary to produce more open fractals. A good value to start with is 25 if you want to experiment with this option.

14 Help Menu

Help menu commands

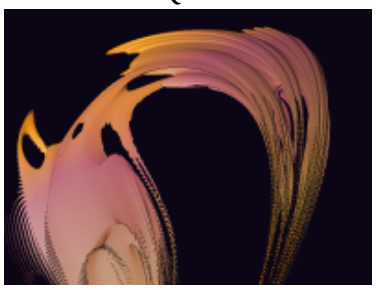
The Help menu offers the following commands, which provide you assistance with this application:

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

14.1 Getting Started

Getting Started

Welcome to QuaSZ!



This is a short tutorial that will cover basic commands and background material necessary for a new user to create an initial picture with QuaSZ. 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.

Start by resetting all Figures, using the Image/Reset All menu option. The drawing window is erased and a quaternion map of the Mandelbrot set is drawn. The 2D Mandelbrot set is frequently considered a map of all Julia sets. Here we are just interested in what 3D Julia sets can be generated using the 3D Mandelbrot as a rough map. (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 for quaternions. QuaSZ provides 200 such formulas.) Use the hot key combo shift-G to turn on a quaternion exploratory mode. The cursor changes and you can click on any portion of the draw window. If you left-click inside the draw window, the right-hand corner (in this case called sector 2) will be erased, and a miniature version of the quaternion that uses that space as its constants will be drawn. You can continue to click on areas of the draw window and see what quaternion lies there, or you can press the space bar to open a new window and draw the quaternion full-size. Press 'Esc' to exit this mode without creating a new window.

Once you've found and drawn a quaternion that looks good, you can change its palette or surface pattern. Use the palette editor to create a custom palette, or even create a random palette using the Random button. You can change the mapping of the palette by editing the Coloring Filter, or click on Demo/Random Render and let QuaSZ pick a coloring filter at random.
If you want to see the quaternion rotated at different angles, you can use the shift-R hotkey. This works like 'G' except that the cursor keys and Invert/Delete keys are used to rotate the image in 10-degree increments. See the hot key list for more information.

Other 3-D Types in QuaSZ include hypernions, cubic Mandelbrots and octonions. It's easy to switch from quaternion to hypernion using the Type menu. Most hypernions exhibit squared-off shapes rather than the round quaternion shapes. Cubic Mandelbrots are more complicated to set up. There are ten built-in formulas G0-G9 that work with the cubic Mandelbrot type (and ten for octonions, H0-H9.) Unless you have a background in cubic or octonion math, it's best to use the Demo/Random commands to start with. These automatically use a procedure like 'G' to scan through a cubic Mandelbrot/octonion space and pick out interesting areas to display. For a good preview of what QuaSZ is capable of, be sure to experiment with all the Demo/Random commands.

You'll probably be doing a lot of zooming and framing on your plots later, so we'll cover that briefly here. After the quaternion is finished (or as much of it is finished that you want to zoom in on), select the Zoom 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 can use the Shift key to expand the zoom box X 4. To exit this mode without zooming, click on the title bar of the draw window or press 'Esc'. Most of the time you'll be zooming in to reframe a quaternion, or zooming out to include parts of the quaternion that may appear outside the default z-space.

Special note: As you explore the many options included in QuaSZ 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.

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

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

14.2 Index command

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 QuaSZ 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

Shift-G -- 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 to Quaternion, else hypercomplex math is used for the Hypernion type. Once you find an interesting quaternion set using "G" (by clicking in any area around the Mandelbrot borders or anywhere in the draw window), press the space bar and 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. Click on the escape key to exit this mode without generating a quaternion set.

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

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

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

Shift-F -- generate a Julia quaternion set from a formula's MandelbrotP space. Random points in a

formula's current Mandelbrot space are scanned for an interesting Julia set.

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.

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 draw window to the current background color.

14.4 Parser Information

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 QuaSZ to represent standard function".

sine z --- sin(z) or SIN(Z) ; where Z can be any quad expression or variable
hyperbolic sine z --- sinh(z) or SINH(Z)
arcsine z --- asin(z) or ASIN(Z)
cosine z --- cos(z) or COS(Z)
hyperbolic cosine z --- cosh(z) or COSH(Z)
arccosine z --- acos(z) or ACOS(Z)
tangent z --- tan(z) or TAN(Z)
hyperbolic tangent z --- tanh(z) or TANH(Z)
arctangent z --- atan(z_) or ATAN(Z)
cotangent z --- cotan(z) or COTAN(Z)
arccotangent z --- acotan(z) or ACOTAN(Z)
e^z --- exp(z) or EXP(z) -- the exponential function
natural log of z --- log(z) or LOG(Z)
absolute value of z --- abs(z) or ABS(Z)
square root of z --- sqrt(z) or SQRT(Z)
z squared --- sqr(z) or SQR(Z)
real part of z --- real(z) or REAL(Z)
imaginary part of z --- imag(z) or IMAG(Z)
'j' or third component of z --- imaj(z) or IMAJ(Z)
'k' or fourth component of z --- imak(z) or IMAK(Z)

convert z-real to zi -- gami(z) or GAMI(Z)
convert z-real to zj -- jami(z) or JAMI(Z)
convert z-real to zk -- kami(z) or KAMI(Z)
modulus of z --- mod(z) or MOD(Z) or |z| -- (x*x + y*y)
conjugate of z -- conj(z) or CONJ(z) -- (x-yi)
flip(z) --- flip(z) or FLIP(Z) -- exchange real and imaginary parts of z (y+xi)
polar angle of z -- theta(z)
fn1(z) -- fn1(z) or FN1(z) -- user-defined function from f1 list box
fn2(z) -- fn2(z) or FN2(z) -- user-defined function from f2 list box
fn3(z) -- fn3(z) or FN3(z) -- user-defined function from f3 list box
fn4(z) -- fn4(z) or FN4(z) -- user-defined function from f4 list box

rfun/rend -- 'rfun' tells the parser to treat all variables as real numbers instead of quad variables. 'Rend' turns this option off. This is a useful speed-up when the formula consists mostly of discrete (real) variables instead of quad variables, which are combined to form a quad result. For example:

```
; Mandelbulb formula by Daniel White
n=l# ; arglimit value
rfun ; begin real values
z1=imaj(z) ; third component of z
r=sqrt(x#*x#+y#*y#+z1*z1+.0000001)
phi=n*asin(z1/r)
w=n*theta(z)
t2=cos(phi)
t1=r^n
x=t1*t2*cos(w)
y=t1*t2*sin(w)
z1=t1*sin(phi)
rend ; end real values
z=x+y*i+z1*j+c#
```

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.
arglimit --- l# or L# -- the constant entered in the Arglimit gadget, read-only.
e --- e or E -- 1e^1 -- 2.71828, read/write.
i --- i or I -- square root of -1, read-only. This is equivalent to the quad constant 0+1i+0j+0k.
iteration --- iter# -- iteration loop counter
j --- j or J -- third component of quad constant, read-only. This is equivalent to the quad constant 0 +0i+1j+0k.
k --- k or K -- fourth component of quad constant, read-only. This is equivalent to the quad constant 0+0i+0j+1k.
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# -- the real part of the complex constant, as entered in the cr box, read-only.
p1 – the quad constant entered in the cr, ci, cj and ck boxes, read-only.
pi --- pi or PI -- 3.14159, read/write.
q --- q# or Q# -- the imaginary part of the complex constant, [ci box] evaluated as a real number, 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+z^3/c+c
a2 -- tan(z)-(z^2+c)
a3 -- z^2+arcsinz-c

a4 -- cos(z)+csc(z)+c
a5 -- cos(z^3)+c
a6 -- z^7+c
a7 -- sin(z)+c^3
a8 -- z^3+zn+c
a9 -- Quaternion set -- q^3+c^3

b0 -- 1/z^2+exp(z)-c
b1 -- 1/z^3+logz-c
b2 -- z^3+j+kzn
b3 -- cz^2+zn+c
b4 -- sin(z)+kzn+j
b5 -- vers(z)+c
b6 -- vers(z)+covers(z)+c
b7 -- c*vers(z)+c
b8 -- vers(z)*covers(z)+c
b9 -- (foggy coastline #1)^2+c -- Barnsley

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

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

e0 -- (z+j)(z+k)(z^2+1)+c
e1 -- (z+j)(z^2+z+k)+c
e2 -- (z-1)(z^2+z+c)
e3 -- (z+j)(z+k)(z+1)+c
e4 -- chaos formula -- Barnsley

e5 -- snowflake IFS -- Barnsley
e6 -- cos(z)+c
e7 -- z^3+exp(z)+c
e8 -- (z-c)(z+1)(z-1)+c
e9 -- z^4-c^4

f0 -- z^2+z^3+sin(c)+cos(c)+c
f1 -- z^2*(1-cz^2)+c
f2 -- 1/(z*z/c)+c
f3 -- 1/(z*z)-z+c
f4 -- (1+c)/(z*z)-z
f5 -- (1+c)/(z*z/c)+c
f6 -- (1/c)/(z*z/c)+c
f7 -- 1/((z*z)/c)+(c/(1/z-c))
f8 -- 1/(z*z*z/c)+c
f9 -- 1/(z*z*z)-z+c

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

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

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

i6 -- cosh(z)*z^2-c^2
i7 -- 1/z^2-cz-c
i8 -- tan(z)-czc
i9 -- (tan(z)-1/z^3)/c^2

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

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

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

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

m7 -- (1/fn(z))^2+c
m8 -- (1/fn(z))^3+c
m9 -- fn(z)/(1-fn(z))+c

n0 -- Sinus by Thomas Kromer (fixed type)
n1 -- Sinus #2 by Thomas Kromer (fixed type) (Note 6)
n2 -- Rings of Fire by Thomas Kromer (fixed type) (Note 6)
n3 -- Teres by Thomas Kromer (fixed type)
n4 -- z^2+y+c
n5 -- z^2+y[n+1]+c
n6 -- z^2+zi+c
n7 -- z^2+zi[n+1]+c
n8 -- zr^2+3zi+c
n9 -- zr^2+4zi+c

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

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

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

q8 -- Newton's method applied to fn(z)+c
q9 -- Newton's method applied to fn(z)+fn(z)+c

r0 -- z^3+conj(z)c+c
r1 -- z^z+z^3+c
r2 -- z^3-z+c
r3 -- z^2+exp(z)+c
r4 -- (z^2-c)(z+1)
r5 -- cz^3+c
r6 -- z^2+cexp(z)+c
r7 -- sin(z)+cz^2+c
r8 -- (z-1)/c when x>=0; else (z+1)/cc -- Barnsley
r9 -- (z-1)/c when kx-jy>=0; else (z+1)/c -- Barnsley

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

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

u0 -- Mandelbulb (sine-based non-trig version) -- Note 8
u1 -- Mandelbulb (cosine-based non-trig version)
u2 -- Mandelbulb (sine-based trig version)
u3 -- Mandelbulb (cosine-based trig version)
u4 -- Tricorn Mandelbulb (sine-based non-trig version) -- Note 9
u5 -- Tricorn Mandelbulb (cosine-based non-trig version)
u6 -- Tricorn Mandelbulb (sine-based trig version)
u7 -- Tricorn Mandelbulb (cosine-based trig version)

Hydra Formulas:
p0 -- z^2+c
p1 -- cz(1-z)
p2 -- z(z-1)+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 -- z^2-z^3+c
p9 -- z^2-cz^3+c

r0 -- z^3+zc+c
r1 -- cz^2+zn+c
r2 -- z^3-z+c
r3 -- z^2+z+c
r4 -- (z^2-c)(z+1)
r5 -- cz^3+c
r6 -- z^2+cz+c
r7 -- z+cz^2+c
r8 -- z^3+j+kzn
r9 -- z^3+c^3

Note 1: For further info on Michael Barnsley's formulas, see his "Fractals Everywhere".

Note 2: The quaternion and hypernion types use the complex constant gadgets to input cr, ci, cj and ck. These may be zero when generating a Mandelbrot-like set of these functions. Quaternion 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 Z and 4th Dimension gadgets are used to input the z and w coefficients of the j and k planes. Z is typically set to 2.0. Values of 0 for Z, 4th Dimension, cj and ck result in a two-dimensional slice that matches the hypercomplex type. Higher values of z and w (as well as cj and ck) produce more pronounced asymmetry in the complex mapping.

Note 3: For the traditional cubic Mandelbrot (example in The Science of Fractal Images), Z, 4th Dimension, cj and ck (hypercomplex components of z and c) should be set to zero.

Note 4: Octonions have a form of xr+xi+xj+xk+xE+xI+xJ+xK. For these formulas C is the octonion constant (1,1,1,1,1,1,1,1) and CC is the octonion conjugate (1,-1,-1,-1,-1,-1,-1,-1). Additional options are entered via the Arg box in the Quaternion editor window. To rotate the extra four-octonion dimensions (E-K) use the following syntax:

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

To normalize the C and CC constants:

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

Alternate octonion initialization:

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

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

Note 5: the term 'fn(w)' represents any one of 54 user-defined functions chosen through the f1-f4 gadgets:

0: sin(w).	1: sinh(w).	2: cos(w).	3: cosh(w).
4: tan(w).	5: tanh(w).	6: exp(w).	7: ln(w).
8: w^c 9: w^z.	10: 1/w.	11: w^2.	12: w^3.
13: abs(w).	14: sqrt(w).	15: w.	16: conj(w).
17: csc(w).	18: csch(w).	19: sec(w).	20: sech(w).
21: cot(w).	22: coth(w).	23: cw.	24: 1.
25: arsin(w).	26: arcsinh(w).	27: arccos(w).	28: arccosh(w).
29: arctan(w).	30: arctanh(w).	31: arccot(w).	32: arccoth(w).
33: vers(w).	34: covers(w).	35: L_3(w): 3rd degree Laguerre polynomial.	
36: gamma(w): first order gamma function.		37: G(w): Gaussian probability function -- (1/sqr(2pi))*e^(.5w^2).	
38: c^(s+si).	39: zero.	40: w^(s+si).	41: |(wx)|+|(wy)|*i (abs).
42: wy+wx*i(flip).	43: conj(cos(w))--cosxx.	44: theta(w) -- polar angle(w).	
45: real(w).	46: imag(w)	47: loxodromic(w)	48: gedatou(w) 49: ventri(w)
50 sinus #1(w)	51 sinus #2(w)	52 rof(w)	53 teres(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 composite plots that use both fun#1 and fun#2, fun#1 takes its functions from f1 and f2 and fun#2 takes its functions from f3 and f4.

Note 6: For the loxodromic functions, Sinus #2 and Rings of Fire, the Arg Limit variable (in the Edit Formula/Type window) is used as an additional ingredient. Try values -1.5 to 1.5 for Sinus #2 and 1.5 or PI/2 for Rings of Fire.

Note 7: 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 8: 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.

Note 9: 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.

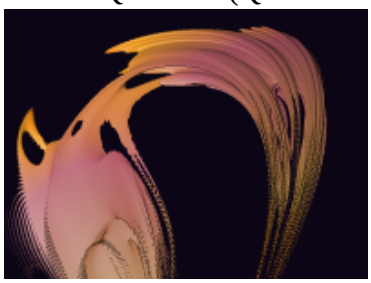
Wegner, Tim and Tyler, Bert: "Fractal Creations", Second Edition, Waite Group Press, Corte Madera, CA, 1993.

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

14.7 About QuaSZ

About QuaSZ

>>>>> QuaSZ™ (Quad Surface Zplot) v3.056 © 2000-2010 by Terry W. Gintz



QuaSZ graphs 3-D slices of formulas based on 4-D and 8-D complex number planes. QuaSZ currently supports quaternion, hypernion, cubic Mandelbrot and octonion renderings of the Mandelbrot set and Julia sets. 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. Random image generators and a random formula generator, batch mode and integrated video routines make the program easy for beginners and a powerful complement to advanced fractal artists.

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 composite formulas and IFS (Michael Barnsley).

200+ formulas have been hard-coded to reduce graphing time by 40 to 60 percent. Hypercomplex extensions (as described in Fractal Creations) are standard for all the formulas.

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

QuaSZ requires a true-color video adapter for best results. It works in 16-bit (high color) also.

Memory requirements for QuaSZ vary with the size of the drawing area QuaSZ opens on, ranging from approximately 3 megabytes memory for a 640X480 area to 48 megabytes for a 2048X1536 area. Special routines have been added to reduce memory requirements for large bitmaps (up to 14400X10800) by writing these directly to a file instead of using a memory bitmap.

Acknowledgements: A delight to include Thomas Kromer's loxodromic algorithms in the latest versions of QuaSZ. Special thanks to Frode Gill for his quaternion and ray-tracing algorithms, to Dirk Meyer for his Phong-shading algorithm, to Onar Aam for tutoring me on octonion algebra and its esoteric details, and to David Makin for sharing his ideas on quaternion colorings and 3-D insights. An honor to mention Stig Pettersson, for his ground-breaking work in cubic Mandelbrots, giving me clues to an historic fractal world I hardly knew existed...Thanks also to Francois Guibert for his Perlin noise example, and to Kerry Mitchell and Mark Townsend for allowing me to incorporate their coloring methods from Ultra Fractal. The multi-windowing interface in QuaSZ is courtesy of that extraordinary and prolific fractal programmer, Steven C. Ferguson. Steve's contributions to the look and feel of QuaSZ and previous versions of my program have had a deep impact on my fractal imaging experiments. I am indebted to Alessandro Rosa for carefully explaining the "cutoff rate" method to the QuaSZ group, and making 3D Newton/Halley pictures finally a reality.

For a short history of this program, see Chronology.

14.8 Chronology

Chronology

History of this program:

In September 1989, I first had the idea for a fractal program that allowed plotting all complex functions and formulas while attending a course on College Algebra at Lane College in Eugene, Oregon. In November 1989, ZPlot 1.0 was done. This Amiga program supported up to 32 colors, 640X400 resolution, and included about 30 built-in formulas and a simple formula parser.

May 1990 -- ZPlot 1.3d -- added 3-D 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 3-D quaternion plots, Fractint par/frm files, Steve Ferguson's filters, anti-aliasing and

Paul Carlson's orbit-trap routines.

June 1997 -- ZPlot 24.03 -- added Earl Hinrichs Torus method.

July 1997 -- ZPlot 24.08 -- added HSV filtering.

December 1997 -- Fractal Elite 1.14 -- 100 built-in formulas; added avi and midi support.

March 1998 --Split Fractal Elite into two programs, Dreamer and Medusa(multimedia.)

April 1998 -- Dofo 1.0 -- supports new Ferguson/Gintz plug-in spec.

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

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

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

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

April 1999 -- Split Dofo-Zon Elite into two programs: Fractal Zplot using built-in formulas and rendering methods, and Dofo-Zon to support only plug-in formulas and rendering methods.

May 1999 -- Fractal Zplot 1.18 -- added Phong highlights, color-formula mapping and random fractal methods.

June 1999 -- completed Fractal ViZion -- first version with automatic selection of variables/options for all fractal types.

July 1999 -- Fractal Zplot 1.19 -- added cubic Mandelbrot support to quaternion option; first pc fractal program to render true 3-D Mandelbrots.

September 2000 -- Fractal Zplot 1.22 -- added support for full-screen AVI video, and extended quaternion design options

October 2000 -- QuaSZ(Quaternion System Z) 1.00 -- stand alone quaternion/hypernion/cubic Mandelbrot generator

November 2000 -- Added octonion fractals to QuaSZ 1.01.

March 2001 -- Cubics 1.0 -- my first totally-3-D 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.
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 -- PodME 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.

July 2004 – Added the features of Hydra, Cubics and PodME to QuaSZ, now renamed "Quad Surface Zplot". Merged FraSZle with Fractal Zplot, and Fractal Projector with DynaMaSZ SE to form DynaMaSZ 2, including support for the original DynaMaSZ files.

Index

- 3 -

- B -

- C -

- D -

- E -

- F -

- H -

- I -

- P -

- R -

- S -

- T -

- V -