

Moquela Application Help

© 2004 ... Mystic fractal

Table of Contents

Foreword	0
1 Main Index	5
1 Title bar	5
2 Size	5
3 Scroll bars	6
4 Move	6
5 Minimize command	6
6 Maximize command	7
7 Next Window	7
8 Previous Window	7
9 Close	7
10 Restore	8
11 Switch to	8
12 Tutorial	8
2 Moquela Remote	12
1 Channel Guide	12
2 New	12
3 Abort button	13
4 Size	13
5 Batch	13
6 View	13
7 Figure	13
8 Light	14
9 Draw Figure	14
10 Help	14
11 Formula	14
12 Params	14
13 Q1 button	14
14 Q2 button	14
15 C1 button	15
16 C2 button	15
17 CJ button	15
18 O1 button	15
19 O2 button	15
20 Cp button	16

21	Save	16
22	Load	16
23	Bmp	16
24	Png	16
25	Jpg	16
3	File menu	17
1	New command	17
2	Open command	17
	Open dialog box	18
3	Close command	18
4	Save command	18
5	Save As command	19
	Save As dialog box	19
6	File Open [JPG] command	19
7	File Save Bitmap As [JPG] command	19
8	File Open [PNG] command	20
9	File Save Bitmap As [PNG] command	20
10	Import	20
	Cubic Parameters command	20
	PodME Parameters command	20
	Hydra Parameters command	20
11	Export (copy)	21
	File Save Model [OBJ] command	21
	Save As [POV] command	21
	Set Max Faces command	21
	Set Max Vertices command	22
12	File 1, 2, 3, 4, 5, 6 command	22
13	File Exit command	22
4	Edit menu	22
1	Copy Data command	23
2	Paste Data command	23
3	Size	23
4	Formula Window	24
5	Initial Quaternion Values Window	24
6	Cubic Values	24
7	Octonion Values	25
8	Figure	26
	Colors	26
	RGB	26
9	Lighting	26

5 Image menu	27
1 Draw command	27
2 Make command	27
3 Abort command	28
4 Auto Redraw command	28
5 Auto Alert command	28
6 Auto Remote command	28
7 Picture	28
8 Pilot	29
9 Texture	29
texture	29
10 Figure #	29
11 Composite command	29
12 Unitize	29
13 Center	30
14 Use Offsets	30
6 Type menu	30
1 Mandelbrot	30
2 Julia	30
3 Quaternion command	31
4 Hypernion command	31
5 Cubic command	32
6 Complexified Quaternion command	32
7 Custom Quad command	33
Custom Sign Matrix Editor	33
7 Pixel Menu	33
1 Switch command	34
2 Loxodromic ->	34
8 View menu	34
1 Toolbar command	34
toolbar	34
2 Status Bar Command	35
status bar	35
9 Window menu	36
1 Cascade	36
2 Tile	36
3 Arrange Icons	36

4	Size Desktop	36
5	1, 2,	37
10	Demo menu	37
1	Random Quaternion	37
2	Random Quaternion2 command	38
3	Random Cubic Mandelbrot command	38
4	Random Cubic Mandelbrot2 command	38
5	Random Cubic Julia command	39
6	Random Octonion command	39
7	Random Octonion2 command	39
8	Random Composite command	40
9	Batch Mode	40
11	Help menu	41
1	Getting Started	41
2	Index	42
3	Hot Keys	42
4	Parser Information	43
5	Built-in Formulas	45
6	Bibliography	51
7	About Moquela	52
	Chronology	53
	Index	56

1 Main Index

Moquela Help Index

[Getting Started](#)

[Moquela Remote](#)

[Channel Guide](#)

[An Introduction To CQuat Fractals by Terry W. Gintz](#)

Commands

[File menu](#)

[Edit menu](#)

[Image menu](#)

[Type menu](#)

[Pixel menu](#)

[View menu](#)

[Window menu](#)

[Demo menu](#)

[Help menu](#)

1.1 Title bar

Title Bar

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

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

A title bar may contain the following elements:

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

1.2 Size

Size command (System menu)

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

arrow keys.



After the pointer changes to the four-headed arrow:

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

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

Shortcut

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

1.3 Scroll bars

Scroll bars

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

1.4 Move

Move command (Control menu)

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



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

Shortcut

Keys: CTRL+F7

1.5 Minimize command

Minimize command (application Control menu)

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

Shortcut

Mouse: Click the minimize icon  on the title bar.


Keys: ALT+F9

1.6 Maximize command

Maximize command (System menu)

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

Shortcut

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

1.7 Next Window

Next Window command (drawing Control menu)

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

Shortcut

Keys: CTRL+F6

1.8 Previous Window

Previous Window command (drawing Control menu)

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

Shortcut

Keys: SHIFT+CTRL+F6

1.9 Close

Close command (Control menus)

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

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



Shortcuts

Keys: CTRL+F4 closes a drawing window
ALT+F4 closes the application

1.10 Restore**Restore command (Control menu)**

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

1.11 Switch to**Switch to command (application Control menu)**

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

Shortcut

Keys: CTRL+ESC

Dialog Box Options

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

Task List

Select the application you want to switch to or close.

Switch To

Makes the selected application active.

End Task

Closes the selected application.

Cancel

Closes the Task List box.

Cascade

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

Tile

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

Arrange Icons

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

1.12 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.)

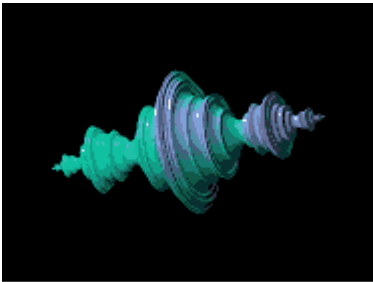


Figure 1. Quaternion Julia set of $-1+0i$

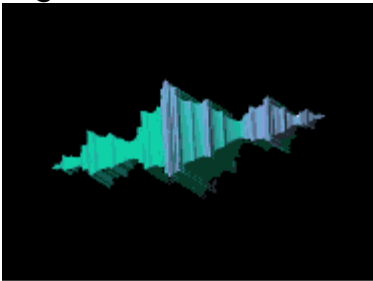


Figure 2. Hypercomplex Julia set of $-1+0i$

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	$-j$

$$j \quad -k \quad -1 \quad i$$

$$k \quad j \quad -i \quad -1$$

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:

$$i \quad j \quad k$$

$$i \quad -1 \quad -k \quad -j$$

$$j \quad -k \quad 1 \quad i$$

$$k \quad -j \quad i \quad 1$$

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 :

$$x \rightarrow x*x - y*y + z*z + w*w + cx$$

$$y \rightarrow 2.0*x*y + 2.0*w*z + cy$$

$$z \rightarrow 2.0*x*z - 2.0*w*y + cz$$

$$w \rightarrow 2.0*x*w - 2.0*y*z + cw$$

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.

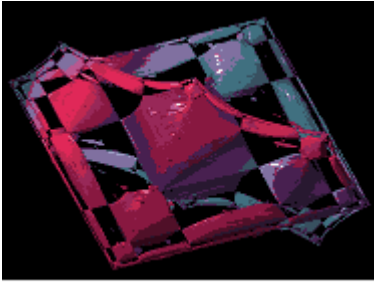


Figure 3. Equilibrium -- cquat Julia rendering of $-1+0i$

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.

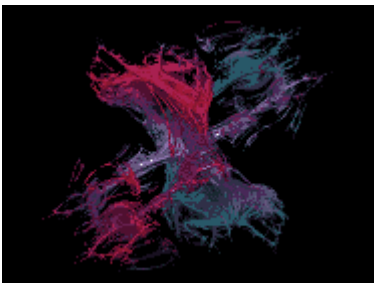


Figure 4 Siegel -- cquat Julia rendering of $-.39054-.58679i$

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/quar/Compute.ang.html#JULIA>

2 Moquela Remote

Moquela Remote

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

2.1 Channel Guide

Channel Guide

The eight channels accessed via the Moquela 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

2.2 New

New button

Use this button to open a new drawing window in Moquela. 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.

2.3 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 Moquela continues to draw the image for that window regardless of which drawing window has the input focus, until done or aborted. You can open and close other drawing windows without affecting the current drawing, but only one drawing is active at any time.

2.4 Size

Size button

This allows you to set the drawing area for a picture, independent of the Windows screen size. It also shows which size is currently in use. The aspect for the drawing is based on the ratio of X (horizontal width) to Y (vertical height.) The 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. Note: the resolution of the figure model doesn't change when you increase the picture size, but it does give you an idea of how rough the object file is and how large it can be scaled in Bryce without additional smoothing. Model resolution is controlled by the number of Steps in the [Initial Quaternion Values](#) window.

2.5 Batch

Batch button

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

2.6 View

View button

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

2.7 Figure

Figure button

Use this button to edit parameters for the current figure.

2.8 Light

Light button

Edit lighting variables and position light source

2.9 Draw Figure

Figure button

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

2.10 Help

Help button

Use this button to open the help index for Moquela.

2.11 Formula

Formula button

Use this button to change the quaternion formula.

2.12 Params

Params button

Use this button to edit 3-D parameters for quaternions.

2.13 Q1 button

Random Quat (Channel Q1 button)

A random quad model is generated. A set of formulas appropriate for quaternions is scanned to find an interesting Julia set.

2.14 Q2 button

Random Quaternion2 (Channel Q2 button)

A random quad model is generated. A set of formulas appropriate for quaternions is scanned to find an interesting Julia set. 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.

2.15 C1 button

Random Cubic Mandelbrot (Channel C1 button)

A random cubic Mandelbrot model is generated. Like the inverse of Random Julia, the essential cubic parameters are randomly adjusted to point into the four-dimensional formula G_0 , and then the Mandelbrot set is scanned to find an interesting area to zoom into.

2.16 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 G_0 , and then the Mandelbrot set is scanned to find an interesting area to zoom into. This option uses the cubic formulas G_0 and G_1 , with relaxed parameters to create cubic-Mandelbrot like fractals that may extend to six or more dimensions.

2.17 CJ button

Random Cubic Julia (Channel CJ button)

A random cubic Julia model (the Julia analog of a cubic Mandelbrot model) is generated. The essential cubic parameters are randomly adjusted to point into a four-dimensional formula. Like Random Julia, a set of formulas (G_0 and G_1) appropriate for cubic Julias is scanned to find an interesting Julia set, and then the parameters are adjusted to produce the cubic image. Note: This is a quasi-Julia approximation that doesn't follow traditional cubic Mandelbrot theory. The "quaternions" produced by this method do exhibit characteristics of cubic Mandelbrots, but here I am more interested in esthetics than mathematical conformity.

2.18 O1 button

Random Octonion (Channel O1 button)

A random octonion Julia model is generated. The essential octonion parameters are randomly adjusted to point into an eight-dimensional formula H_0 - H_9 , and then the octonion Mandelbrot set is scanned to find an interesting area to zoom into.

2.19 O2 button

Random Octonion2 (Channel O2 button)

A random octonion Julia model is generated. The essential octonion parameters are randomly adjusted to point into an eight-dimensional formula H_0 - H_9 , and then the octonion Mandelbrot set is scanned to find an interesting area to zoom into. This option uses the octonion formulas H_0 - H_9 , with random dimensional switching (one of O_i - O_k for O_i) to create octonion fractals

that may extend to eight dimensions.

2.20 Cp button

Random Composite (Channel CP button)

This generates a random composite Julia quad model according to the options set in the Batch window. This is the same as the [Demo/Random Composite](#) command.

2.21 Save

Save button

Use this button to save and name the active drawing. Moquela displays the Save As dialog box so you can name your drawing. To save a drawing with its existing name and directory, use the File/Save command.

2.22 Load

Load button

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

2.23 Bmp

BMP button

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

2.24 Png

PNG radio button

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

2.25 Jpg

JPG radio button

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

3 File menu

File menu commands

The File menu offers the following commands:

New	Creates a new drawing.
Open	Opens an existing drawing.
Close	Closes an opened drawing.
Save	Saves an opened drawing using the same file name.
Save As	Saves an opened drawing to a specified file name.
Open [JPEG]	Load jpeg.
Save AS [JPEG]	Save in jpeg format.
Open [PNG]	Load png.
Save As [PNG]	Save in png format.
Import Options	
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 Model[OBJ]	Save model for current figure.
Save as [POV]	Save polygonized quaternion as a pov triangle object.
Set Max Faces	Set target face size for mesh-simplification option.
Set Max Indices	Set maximum number of vertices allocated for Q polygon.
Exit	Exits Moquela.

3.1 New command

New command (File menu)

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

You can open an existing data/image file with the [Open command](#).

Shortcuts

Keys: CTRL+N

3.2 Open command

Open command (File menu)

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

You can create new images with the [New command](#).

Shortcuts

Toolbar: 
Keys: CTRL+O

3.2.1 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:
<< List your application's file types here. >>

Drives

Select the drive in which Moquela stores the file that you want to open.

Directories

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

Network...

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

3.3 Close command

Close command (File menu)

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

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



3.4 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, Moquela displays the [Save As dialog box](#) so you can name your drawing. If you want to change the name and directory of an existing drawing before you save it, choose the [Save As command](#).

Shortcuts

Toolbar: 
Keys: CTRL+S

3.5 Save As command

Save As command (File menu)

Use this command to save and name the active drawing. Moquela displays the [Save As dialog box](#) so you can name your drawing.

To save a drawing with its existing name and directory, use the [Save command](#).

3.5.1 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. A filename can contain up to eight characters and an extension of up to three characters. Moquela adds the extension you specify in the Save File As Type box.

Drives

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

Directories

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

Network...

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

3.6 File Open [JPG] command

Open [JPEG] command (File menu)

Use this command to load parameters, foreground object 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.

3.7 File Save Bitmap As [JPG] command

Save As [JPEG] command (File menu)

Use this command to save the parameters, foreground object 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.

3.8 File Open [PNG] command

Open [PNG] command (File menu)

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

3.9 File Save Bitmap As [PNG] command

Save As [PNG] command (File menu)

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

3.10 Import

3.10.1 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.

3.10.2 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.

3.10.3 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.

3.11 Export (copy)

3.11.1 File Save Model [OBJ] command

Save Model [OBJ] command (File menu)

Use this command to save the model vertices and vertex normals of the current foreground figure in a Wavefront object file. Since each model generated by the Random Quaternion command is saved initially as "test.obj" (if not in batch mode) it is essential to save them under different names to avoid overwriting their file when the next model (foreground object) is generated. Vertex normals are calculated automatically for every model created with Moquela. This command acts as an export function for programs such as Amorphium 1.0 or Bryce.

Note: when you save the entire scene using one of the "Save all" commands in the File menu, the object file is saved again with the prefix chosen for the data and bitmap files.

3.11.2 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.

3.11.3 Set Max Faces command

Export -> Set Max Faces command (File menu)

Here you can set the target face size for a Wavefront object. 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).

3.11.4 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. Bryce 4 has problems with object files produced by QuaSZ that are much larger than 2.5MB (on systems other than Win XP.)

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

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

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

3.13 File Exit command

Exit command (File menu)

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

Shortcuts

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



Keys: ALT+F4

4 Edit menu

Edit menu commands

The Edit menu offers the following commands:

Copy Data	Copy fractal data to buffer.
Paste Data	Paste data from copy buffer.
Size	Sets the image size.
Formula	Select formula for foreground figure.
Initial Values	Edit quad parameters.
Cubic Values	Edit cubic parameters.
Octonion Values	Edit octonion constants.
Figure	Edit current figure.
Lighting	Edit lighting parameters.

4.1 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.Moquela".

Shortcut

Keys: CTRL+F

4.2 Paste Data command

Paste Data command (Edit menu)

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

Shortcut

Keys: CTRL+R

4.3 Size

Size command (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. Note: the resolution of the figure model doesn't change when you increase the picture size, but it does give you an idea of how rough the object file is and how large it can be scaled in Bryce without additional smoothing. Model resolution is controlled by the number of Steps in the [Initial Quaternion Values](#) window.

4.4 Formula Window

Formula Window

Formula is a combo control for selecting the function used to generate the current figure.

The button named Random fun#1 is used to pick a formula at random. Clicking on Random fun#1, a formula is chosen from the 100 built-in formulas.

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.

Hint: when you change formulas, most likely a blank window will be generated when you Apply the change. You need to execute the [Random Quaternion](#) command to locate suitable complex constants to form the current figure model. If you deselect the Formula box in the Batch window, Moquela will use the current formula to generate a new quad figure.

4.5 Initial Quaternion Values Window

Initial Quaternion Values Window

This defines some of the characteristics of the quad figure generator.

Complex constants are cr, ci, cj and ck. Bailout and iterations can also be altered to change the current figure. A larger value for the bailout variable (up to 65000) can result in fuller, more complete figures, whereas a bailout of 4 (the default) may produce a choppier but more open figure.

Figure precision is set with the Steps variable, where $\text{precision} = 10/\text{Steps}$.

Note: the complex constants, and other variables here are automatically set by the Demo/Random Quaternion and other Demo commands. They can be modified slightly here to produce variations on the original object. Since there are limits to what can be mapped by the routine that produces the 3-D model, it's possible to over-modify these variables and the routine will fail. Select Cancel to undo changes.

4.6 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 $2 \cdot \text{abs}(S)$, when the Type is Mandelbrot. Si can also be set to center the 3rd dimension on something besides 0.0. Si has no affect in 2-D hypercomplex mode. (In 2-D mode, the S variable acts as one of the two fixed dimensions, along with the 4th Center variable.)

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

In addition, the Arg value has the following affect on cubic Mandelbrots:

- 0 -- compute M+, using Z for z space
- 1 -- compute M+, using greater of S or Z for z space
- 2 -- compute M-, using greater of S or Z for z space
- 3 -- compute M+ and M-, use lesser of M+/M- for pixel depth
- 4 -- compute M+ and M-, use greater of two for pixel depth
- 5 -- compute M+ and M-, use difference of two for pixel depth
- 6 -- compute M+ and M-, use sum of two for pixel depth
- 7 -- compute M+ and M-, use vector magnitude of two for pixel depth
- 8 -- compute M+ and M-, use intersection of two (CCL)
- 9 -- compute M+ and M-, use M+ or difference of two if $M+ > M-$

Args 3-9 affect only cubic Mandelbrots.

For args 0-9, a second argument determines the plane that is used for the fourth dimension:

- b -- b-imag (default)
- B -- b-real
- a -- a-imag
- A -- a-real

A third argument also works for args 0-9:

- j -- alternate cubic Mandelbrot mapping

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

4.7 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.

4.8 Figure

Figure command (Edit Menu)

Use this window to edit the current figure. Use the Palette button to edit the color indexes for the selected figure.

Figure Parameters

xOffset -- the origin of the figure on the x axis

yOffset -- the origin of the figure on the y axis

zOffset -- the origin of the figure on the z axis

Width -- the figures width or radius.

Height -- the figure's unit height.

Depth -- the figure's unit depth.

Spin X -- rotate the figure around the x-axis.

Spin Y -- rotate the figure around the y-axis.

Spin Z -- rotate the figure around the z-axis.

4.8.1 Colors

Colors command (Figure window)

The colors editor is used to modify the color for the current figure and the background color. Color index 2 (the figure color) is only used when a figure has no texture bitmap, or uses a built-in texture such as Check, else white is used as the underlying color for textured figures.

Click on one of the two (small) color boxes to select that color for editing. Use the [RGB](#) slider controls to edit the color.

Use Reset to reset the colors to the values they were when the colors editor was first opened.

4.8.1.1 RGB

RGB

Red, green and blue components.

4.9 Lighting

Lighting command (Edit Menu)

Use this command to adjust lighting parameters and position the light source(s). Parameter boxes are provided for controlling the [RGB](#) ambient intensity of figures. The ambient range is 0-1.0 for each ambient control. Higher values of ambiance produce bright figures with

light shading. Default values are (.2,.2,.2) and (.6,.6,.6) for figure ambiance and model ambiance, respectively. In two-lights mode, another light is positioned at (-Light X, Light Y, Light Z).

Shininess or reflective quotient of the foreground object is also adjusted here. The "shine" is more visible when you use a built-in texture, such as Check. The lower the Shininess factor the more light is reflected off the foreground object.

5 Image menu

Image menu commands

The Image menu offers the following commands:

Draw Figure	Draw the active figure.
Make Figure	Convert image data to model.
Abort	Abort drawing.
Auto Sound Alert	Enable or turn off sound alerts.
Auto Remote	Open remote automatically at startup.
Picture	View image full-screen.
Pilot	Use Pilot to rotate and alter key form variables.
Texture	Apply built-in texture to figures or load bitmap file for texture.
Figure 1-10	Switch to figure # (1-10).
Composite	Select figures to merge into one composite model.
Unitize	Unitize figures in composite.
Center	Center figures in composite.
Use Offsets	Use figure offsets for composite figures.

5.1 Draw command

Draw Figure command (Image menu)

Use this command to draw or redraw the current figure. Left-click in the draw window or [Esc] to halt a drawing.

5.2 Make command

Make command

Use this command to remake the current figure or composite model. Left-click in the draw window or [Esc] to halt this operation.

5.3 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 Moquela continues to draw the image for that window regardless of which drawing window has the input focus, until done or aborted. You can open and close other drawing windows without affecting the current drawing, but only one drawing is active at any time.

5.4 Auto 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. Most of the time you want to see the results of changing a parameter, so redraw occurs automatically with parameter changes. Sometimes you want to change more than one figure before making a composite model. So you may want to turn this option off then.

5.5 Auto Alert command

Auto Sound Alert command (Image menu)

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

5.6 Auto Remote command

Auto Remote command (Image menu)

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

5.7 Picture

Picture command (Image Menu)

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

5.8 Pilot

Pilot

Use the Pilot window to rotate and alter key figure parameters interactively. The current figure can also be adjusting using the keyboard:

5.9 Texture

Texture

Set the [texture](#) for the current figure. You can choose one of the simple built-in textures like Check, or load a bitmap to wrap around the figure.

5.9.1 texture

Textures are normally bitmaps that wrap around or imprint a figure.

5.10 Figure

Figure

Switch to Figure # (1-10). Current settings are saved for the previous figure. Moquela supports up to 10 figures in the same data file. You can combine these figures into one model using the [Image/Draw Composite](#) or [Image/Make](#) commands.

5.11 Composite command

Composite command (Image menu)

Opens the Composite Figure window, where you can define a set of figures to merge into one model. You can define up to ten figures as part of the composite. Use the drop-down box to select "include" for each figure you want to merge into the composite model or "none" to exclude that figure (default is "none.")

5.12 Unitize

Unitize (Image menu)

With this option selected (default on) each figure in the composite is unitized before merging. This ensures that all figures are at the same relative scale and depth. Otherwise one figure may predominate if it is much larger or closer than another. Models are displayed after unitizing them to fit in the display window, so it would be hard to tell the actual size of a model that has been polygonized. The size and location of the model can vary according to the bailout, precision and iterations specified in the fractal data.

5.13 Center

Center (Image menu)

With this option selected (default on) each figure is centered around zero in the composite model. Otherwise the position of each figure may vary with the fractal data that is used.

5.14 Use Offsets

Use Offsets

With this option selected (default off) the offsets used to display a figure are used to position the figure inside the composite model. Otherwise all figures appear in the composite as they would be displayed in the OpenGL window with offsets of 0,0,0.

6 Type menu

Type menu commands

The Type menu offers the following commands:

Mandelbrot	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.

6.1 Mandelbrot

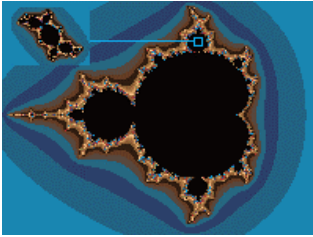
Mandelbrot (Type menu)

Mandelbrots base their mapping on varying inputs of complex C , which corresponds to the min/max values set in the Parameters window. With Mandelbrot, the initial value of Z is set to the value of the pixel being iterated. Usually used with cubic Mandelbrot formulas.

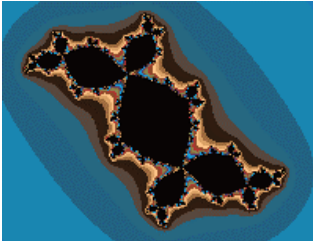
6.2 Julia

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

6.3 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 Values window. A figure that was created using a different type will be recreated using the polygonizing routine.

Note: When changing from one type to another, or just repeating the same type with the same quality, the polygonizing routine may generate a slightly different object from the original. The routine uses a bit of randomizing to map out the object. In extreme cases, some formulas can occasionally produce very different objects each time the same data is mapped.

6.4 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 Values 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. A figure that was created using a different type will be recreated using the polygonizing routine.

Note: When changing from one type to another, or just repeating the same type with the same quality, the polygonizing routine may generate a slightly different object from the original. The routine uses a bit of randomizing to map out the object. In extreme cases, some formulas can occasionally produce very different objects each time the same data is mapped.

6.5 Cubic command

Cubic (Type menu)

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

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

6.6 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 Values window. A cquat 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. A figure that was created using a different type will be recreated using the polygonizing routine.

Note: When changing from one type to another, or just repeating the same type with the same quality, the polygonizing routine may generate a slightly different object from the original. The routine uses a bit of randomizing to map out the object. In extreme cases, some formulas can occasionally produce very different objects each time the same data is mapped.

See the [tutorial](#) on cquat math for further information about complexified quaternions.

6.7 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.

6.7.1 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.

7 Pixel Menu

Pixel menu commands

The Pixel menu offers the following commands:

[Switch Z For C](#)
[Loxodromic](#)

Switch z for c.
Applies loxodromic function to front end of formula.

7.1 Switch command

Switch Z For C

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

7.2 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 added compatibility with PodME data files that use composite formulas.

8 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.

8.1 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 Moquela, 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.











8.1.1 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 Moquela,

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

Click	To
	Opens or brings the Moquela Pilot to the front.
	Open an existing drawing. Moquela 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. Moquela displays the Save As dialog box.
	Set image size.
	Draw figure from current parameters.
	Draw scene from current parameters.
	Edit lighting variables.
	Show picture full-screen.
	Display info about Moquela.
	Display Moquela's help index.

8.2 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.

8.2.1 status bar

Status Bar



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

9 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.

9.1 Cascade

Cascade command (Window menu)

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

9.2 Tile

Tile command (Window menu)

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

9.3 Arrange Icons

Window Arrange Icons Command

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

9.4 Size Desktop

Window Size DeskTop Command

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

9.5 1, 2, ...

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

Moquela 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.

10 Demo menu

Demo menu commands

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

Random Quaternion	Generate random quaternion/hyper-nion fractal.
Random Quaternion2	Generate random quaternion/hyper-nion 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/hyper-nion fractal.
Batch Mode	Set up initial values for batch mode/random commands.

10.1 Random Quaternion

Random Quaternion (Demo menu)

An object file based on a random quaternion fractal is generated. A set of formulas appropriate for quaternions is scanned to find an interesting Julia set, and then the parameters are adjusted or randomized (as directed by options in the Batch mode/Random setup window) to produce a quaternion, hyper-nion or cquat object. The a polygonizing routine is used to map and save the object in a Wavefront obj file.

Note: When batch mode is set, each object of the batch is generated and saved sequentially under a different file name, otherwise the object is saved as "tempxyz[figure#].obj" in the default objects directory. So when batch mode is off you need to save each object (that you intend to keep) under a different name before using this command again for the same figure.

10.2 Random Quaternion2 command

Random Quaternion2 (Demo menu)

A random quad model is generated. 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.

A set of formulas appropriate for quaternions is scanned to find an interesting Julia set, and then the parameters are adjusted or randomized (as directed by options in the Batch mode/Random setup window) to produce a quaternion, hypernion or cquat object. The polygonizing routine is used to map and save the object in a Wavefront obj file.

Note: When batch mode is set, each object of the batch is generated and saved sequentially under a different file name, otherwise the object is saved as "tempxyz[figure#].obj" in the default objects directory. So when batch mode is off you need to save each object (that you intend to keep) under a different name before using this command again for the same figure.

10.3 Random Cubic Mandelbrot command

Random Cubic Mandelbrot (Demo menu)

A random cubic Mandelbrot model 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 polygonizing routine is used to map and save the object in a Wavefront obj file.

Note: When batch mode is set, each object of the batch is generated and saved sequentially under a different file name, otherwise the object is saved as "tempxyz[figure#].obj" in the default objects directory. So when batch mode is off you need to save each object (that you intend to keep) under a different name before using this command again for the same figure.

10.4 Random Cubic Mandelbrot2 command

Random Cubic Mandelbrot2 (Demo menu)

A random cubic Mandelbrot model 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.

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. The polygonizing routine is used to map and save the object in a Wavefront obj file.

Note: When batch mode is set, each object of the batch is generated and saved sequentially under a different file name, otherwise the object is saved as "tempxyz[figure#].obj" in the default objects directory. So when batch mode is off you need to save each object (that you intend to keep) under a different name before using this command again for the same figure.

10.5 Random Cubic Julia command

Random Cubic Julia (Demo menu)

A random cubic Julia model (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. 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.

The polygonizing routine is used to map and save the object in a Wavefront obj file.

Note: When batch mode is set, each object of the batch is generated and saved sequentially under a different file name, otherwise the object is saved as "tempxyz[figure#].obj" in the default objects directory. So when batch mode is off you need to save each object (that you intend to keep) under a different name before using this command again for the same figure.

10.6 Random Octonion command

Random Octonion (Demo menu)

A random octonion Julia model 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 polygonizing routine is used to map and save the object in a Wavefront obj file.

Note: When batch mode is set, each object of the batch is generated and saved sequentially under a different file name, otherwise the object is saved as "tempxyz[figure#].obj" in the default objects directory. So when batch mode is off you need to save each object (that you intend to keep) under a different name before using this command again for the same figure.

10.7 Random Octonion2 command

Random Octonion2 (Demo menu)

A random octonion Julia model 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. 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. The polygonizing routine is used to map and save the object in a Wavefront obj file.

Note: When batch mode is set, each object of the batch is generated and saved sequentially under a different file name, otherwise the object is saved as "tempxyz[figure#].obj" in the default objects directory. So when batch mode is off you need to save each object (that you intend to keep) under a different name before using this command again for the same figure.

10.8 Random Composite command

Random Composite (Demo menu)

A random 3-D Julia model 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. The polygonizing routine is used to map and save the object in a Wavefront obj file.

Note: When batch mode is set, each object of the batch is generated and saved sequentially under a different file name, otherwise the object is saved as "tempxyx[figure#].obj" in the default objects directory. So when batch mode is off you need to save each object (that you intend to keep) under a different name before using this command again for the same figure.

10.9 Batch Mode

Batch mode/Random Setup (Demo menu)

Here you set parameters for batching and saving random-generated figures 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 figures can be generated and saved to disk. Use a unique Filename to prevent batch files from overwriting existing image files. The Scan Limit directs the program on how many scans it makes through each formula before it skips to a new formula (if an interesting 3-D fractal hasn't been found.)

You have the option of selecting only quaternion types, only hypernion types (hypercomplex) or a mixture of quaternion/hypernion or cquat types for the figure. With the commands Random Quaternion or Random Quaternion2, if the quad type is selected, only quad type figures will be generated. The quad type is not used with the Random Cubic Mandelbrot, Random Octonion or Random Composite commands.

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

Formula -- (default on) check to randomize built-in formula used

Max Iter/Bail -- (default on) check to set default iterations and bailout at 8 and 4 respectively

Constants -- (default on) check to randomize the complex constants c_j and c_k

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.

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

11.1 Getting Started

Getting Started

Welcome to Moquela!



This is a short tutorial that will cover basic commands and information necessary for a new user to create an initial picture with Moquela. For help on any menu command, press F1 while the command is highlighted.

Moquela uses the OpenGL library to display 3-D figures based on quad or octonion math.

Try one of the Random commands in the Demo menu to construct a figure automatically. A set of formulas is scanned and the result polygonized into a Wavefront object, then reloaded into Moquela for display as an OpenGL model. You can set particular aspects of the object created using the [Batch mode/Random setup window](#).

Figures have their own bitmap [textures](#) loaded with the [Texture/Bmp option](#) in the Image menu. Moquela also provides the option of a built-in texture (like Check) for the figure, if you elect not to use a bitmap texture.

The [Pilot](#) contains buttons to move, rotate and size a figure. This can be accessed by the Spiderman icon in the toolbar or from the Image menu. There are also keyboard shortcuts for each button on the Pilot.

This completes the Getting Started tutorial. Be sure to read the [Edit/Figure Lighting](#), [Initial Quaternion Values](#) and [Hot Keys](#) sections for additional info.

11.2 Index

Index command (Help menu)

Use this command to display the opening screen of Help. From the opening screen, you can jump to step-by-step instructions for using Moquela 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.

11.3 Hot Keys

Hot keys

up arrow --- rotate form 5 degrees around x axis (Spin X)
down arrow --- rotate form -5 degrees around x axis (Spin X)
left arrow -- rotate form 5 degrees around y axis (Spin Y)
right arrow -- rotate form -5 degrees around y axis (Spin Y)
y -- rotate form 5 degrees around z axis (Spin Z)
b -- rotate form -5 degrees around z axis (Spin Z)
l -- increment size by .1
s -- decrement size by .1
w -- increment Width by .1
n -- decrement Width by .1
z -- increment Height by .1
x -- decrement Height by .1
i -- increment yOffset by .025
m -- decrement yOffset by .025
j -- decrement xOffset by .025
k -- increment xOffset by .025

u -- increment zOffset by .025
 n -- decrement zOffset by .025

(Global)

Ctrl-N -- new window
 Ctrl-O -- open file
 Ctrl-S -- save file
 Ctrl-C -- copy bitmap to clipboard
 Ctrl-L -- clip part of bitmap to clipboard
 Ctrl-V -- paste from clipboard
 Ctrl-H -- Help index

11.4 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 Moquela to represent standard function".

sine z --- $\sin(z)$ or $\text{SIN}(Z)$; where Z can be any complex expression
 hyperbolic sine z --- $\sinh(z)$ or $\text{SINH}(Z)$
 arcsine z --- $\text{asin}(z)$ or $\text{ASIN}(Z)$
 cosine z --- $\cos(z)$ or $\text{COS}(Z)$
 hyperbolic cosine z --- $\cosh(z)$ or $\text{COSH}(Z)$
 arccosine z --- $\text{acos}(z)$ or $\text{ACOS}(Z)$
 tangent z --- $\tan(z)$ or $\text{TAN}(Z)$
 hyperbolic tangent z --- $\tanh(z)$ or $\text{TANH}(Z)$
 arctangent z --- $\text{atan}(z)$ or $\text{ATAN}(Z)$
 cotangent z --- $\text{cotan}(z)$ or $\text{COTAN}(Z)$
 arccotangent z --- $\text{acotan}(z)$ or $\text{ACOTAN}(Z)$
 e^z --- $\exp(z)$ or $\text{EXP}(z)$ -- the exponential function
 natural log of z --- $\log(z)$ or $\text{LOG}(Z)$
 absolute value of z --- $\text{abs}(z)$ or $\text{ABS}(Z)$
 square root of z --- $\text{sqrt}(z)$ or $\text{SQRT}(Z)$
 z squared --- $\text{sqr}(z)$ or $\text{SQR}(Z)$
 real part of z --- $\text{real}(z)$ or $\text{REAL}(Z)$
 imaginary part of z --- $\text{imag}(z)$ or $\text{IMAG}(Z)$
 modulus of z --- $\text{mod}(z)$ or $\text{MOD}(Z)$ or $|z|$ -- $(x*x + y*y)$
 conjugate of z -- $\text{conj}(z)$ or $\text{CONJ}(z)$ -- $(x-yi)$
 flip(z) --- $\text{flip}(z)$ or $\text{FLIP}(Z)$ -- exchange real and imaginary parts of z $(y+xi)$
 polar angle of z -- $\text{theta}(z)$
 fn1(z) -- $\text{fn1}(z)$ or $\text{FN1}(z)$ -- 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

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

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

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

Math operators

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

Constants and variables

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

z coordinates entered in the Parameters window, read/write.

maxit -- the maximum number of iterations, as set in the Parameters window, read only

p --- $p\#$ or $P\#$ -- real constant used in phoenix maps; uses the real part of the complex constant when the Phoenix option is chosen, read-only.

$p1$ -- the complex constant entered in the cr and ci gadgets, read-only.

pi --- pi or PI -- 3.14159, read/write.

q --- $q\#$ or $Q\#$ -- real constant used in phoenix maps; uses the imaginary part of the complex constant when the Phoenix option is chosen, read-only

x --- $x\#$ or $X\#$ -- real part of Z , read/write.

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

z --- z or Z -- function value at any stage of the iteration process, read/write.

$zn\#$ or $ZN\#$ -- the value of z at the previous stage of iteration, read-only.

11.5 Built-in Formulas

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

- 1 $p0; z^2+c$ (Note 2)
- 2 $p1; cz(1-z)$
- 3 $p2; z(z-1/z)+c$
- 4 $p3; cz^2-c$
- 5 $p4; z^2+cz^2+c$
- 6 $p5; z^3+c$
- 7 $p6; ((z^2)*(2z+c))^2+c$
- 8 $p7; z^2+j+kzn$
- 9 $p8; (x^2-y^2-j,2xy-k)$ when $x>0$; else $(x^2-y^2-c+jx,2xy+kx-k)$ -- Barnsley (Note 1)
- 10 $p9; z^2-cz^3+c$
- 11 $r0; z^3+\text{conj}(z)c+c$
- 12 $r1; z^z+z^3+c$
- 13 $r2; z^3-z+c$
- 14 $r3; z^2+\exp(z)+c$
- 15 $r4; (z^2-c)(z+1)$
- 16 $r5; cz^3+c$
- 17 $r6; z^2+c\exp(z)+c$
- 18 $r7; \sin(z)+cz^2+c$
- 19 $r8; (z-1)/c$ when $x>=0$; else $(z+1)/cc$ -- Barnsley
- 20 $r9; (z-1)/c$ when $kx-jy>=0$; else $(z+1)/c$ -- Barnsley
- 21 $e0; (z+j)(z+k)(z^2+1)+c$
- 22 $e1; (z+j)(z^2+z+k)+c$
- 23 $e2; (z-1)(z^2+z+c)$
- 24 $e3; (z+j)(z+k)(z+1)+c$
- 25 $e4; \text{chaos formula}$ -- Barnsley
- 26 $e5; \text{snowflake IFS}$ -- Barnsley

27	e6; $\cos(z)+c$
28	e7; $z^3+\exp(z)+c$
29	e8; $(z-c)(z+1)(z-1)+c$
30	e9; z^4-c^4
31	s0; $\sin(z)-c$
32	s1; $z^4+\exp(z)+c$
33	s2; $(c(z^2+1)^2)/(z^2-1)$
34	s3; $z^2+\tan(z)+c$
35	s4; strange attractor IFS (s=1.4142) -- Barnsley
36	s5; z^5-c^4
37	s6; composite function $cz-c/z$ && z^2+c
38	s7; $1/z^2+c$
39	s8; $(z^3+3(c-1)z+(c-1)(c-2))$
40	s9; $z(z^5+c^2)$
41	t0; $z(z^6+c^2)$
42	t1; $z^7-z^5+z^3-z+c$
43	t2; z^4-z^2+c
44	t3; $z^3+\tan z+c$
45	t4; z^2+z^c+c
46	t5; $z^3+c/z+c$
47	t6; discretizes Volterra-Lotka equations via modified Heun algorithm
48	t7; z^3+c^z+c
49	t8; Quaternion set -- q^2+c^2
50	t9; z^2+cz^2+c
51	a0; spiral network -- C. Pickover
52	a1; $z+z^3/c+c$
53	a2; $\tan(z)-(z^2+c)$
54	a3; $z^2+\arcsin z-c$
55	a4; $\cos(z)+\csc(z)+c$
56	a5; $\cos(z^3)+c$
57	a6; z^7+c
58	a7; $\sin(z)+c^3$
59	a8; z^3+zn+c
60	a9; Quaternion set -- q^3+c^3
61	b0; $1/z^2+\exp(z)-c$
62	b1; $1/z^3+\log z-c$
63	b2; $z^3+j+kzn$
64	b3; cz^2+zn+c
65	b4; $\sin(z)+kzn+j$
66	b5; $\text{vers}(z)+c$
67	b6; $\text{vers}(z)+\text{covers}(z)+c$
68	b7; $c*\text{vers}(z)+c$
69	b8; $\text{vers}(z)*\text{covers}(z)+c$
70	b9; (foggy coastline #1)^2+c -- Barnsley
71	c0; (foggy coastline #2)^2+c -- Barnsley
72	c1; $\sin(\text{vers})+c$
73	c2; $c\sec(z^2)+c$

- 74 $c3; z^3 + \operatorname{sech}(z^2) + c$
75 $c4; c \cdot z^{(c-1)} + z^2$
76 $c5; \cos(-1/w^2) + c$
77 $c6; (z/e)^z \cdot \operatorname{sqr}(2 \cdot \pi \cdot z) + c$
78 $c7; 1/8(63z^5 - 70z^3 + 15z) + c$
79 $c8; (z^2 + e^{(-z)}) / (z+1) + c$
80 $c9; z^2 \cdot \exp(z) + c$
81 $d0; (z^3) / c + c$
82 $d1; z^3 \cdot \operatorname{sqr}(2 \cdot \pi / z) + c$
83 $d2; z^2 - \operatorname{csc}(h) \cot(h) + c$
84 $d3; (1 / (\sin(z) + c))^3$
85 $d4; z^2 - c; \text{ where } x = \operatorname{abs}(\operatorname{real}(z)), y = \operatorname{imag}(z) \text{ -- Paul Carlson}$
86 $d5; z^2; \text{ where } x = \operatorname{abs}(\operatorname{real}(z)) - cr, y = \operatorname{imag}(z) - ci \text{ -- Paul Carlson}$
87 $d6; c \cdot \cos(z) + c$
88 $d7; z \cdot \cos(z) + c$
89 $d8; z^2 + z / \sin(z) + c$
90 $d9; z^2 + z^\pi + c$
91 $f0; z^2 + z^3 + \sin(c) + \cos(c) + c$
92 $f1; z^2 \cdot (1 - cz^2) + c$
93 $f2; 1 / (z \cdot z / c) + c$
94 $f3; 1 / (z \cdot z) - z + c$
95 $f4; (1 + c\#) / (z \cdot z) - z$
96 $f5; (1 + c) / (z \cdot z / c) + c$
97 $f6; (1/c) / (z \cdot z / c) + c$
98 $f7; 1 / ((z \cdot z) / c) + (c / (1/z - c))$
99 $f8; 1 / (z \cdot z \cdot z / c) + c$
100 $f9; 1 / (z \cdot z \cdot z) - z + c$
101 $g0; z^3 - 3c^2z + s \text{ -- cubic Mandelbrot (Note 3)}$
102 $g1; z^3 - 3sz + c \text{ -- alternate cubic Mandelbrot}$
103 $g2; z^3 - 3c^2z^2 + s \text{ -- cubic Mandelbrot variant}$
104 $g3; z^3 - 3sz^2 + c \text{ -- alternate cubic Mandelbrot variant}$
105 $g4; z^3 - 3c^2/z + s \text{ -- cubic Mandelbrot variant}$
106 $g5; z^3 - 3s/z + c \text{ -- alternate cubic Mandelbrot variant}$
107 $g6; z^3 - 3c^3 \cdot z + s \text{ -- cubic Mandelbrot variant}$
108 $g7; z^3 - 3s/z^2 + c \text{ -- alternate cubic Mandelbrot variant}$
109 $g8; z^3 - 3c^2 + z + s \text{ -- cubic Mandelbrot variant}$
110 $g9; z^3 - 3s + z + c \text{ -- alternate cubic Mandelbrot variant}$
111 $h0; (O \cdot C^{-1})(C \cdot O) + c \text{ -- octonion set (Note 4)}$
112 $h1; cO \cdot CC(1 - C \cdot O) \text{ -- octonion set}$
113 $h2; O \cdot C(O - CC \cdot O) + c \text{ -- octonion set}$
114 $h3; cO^2 + O^2 \cdot CC - c \text{ -- octonion set}$
115 $h4; O^2 \cdot CC + O^2 \cdot C + c \text{ -- octonion set}$
116 $h5; O^3 \cdot CC + c \text{ -- octonion set}$
117 $h6; O^3 \cdot CC + O^3 \cdot C + c \text{ -- octonion set}$
118 $h7; O^4 \cdot CC + c \text{ -- octonion set}$
119 $h8; cO^3 \cdot CC + c \text{ -- octonion set}$
120 $h9; O^2 \cdot CC + O^3 \cdot C + c \text{ -- octonion set}$

121	i0; $2*z*c\#\cos(\pi/z)$ -- Godwin Vickers
122	i1; $2*z*c\#\sin(\pi/z)$ -- Godwin Vickers
123	i2; $2*z*c\#\tan(\pi/z)$ -- Godwin Vickers
124	i3; $1/z^3+\sin(z)*c^2$
125	i4; $\cosh(z)/c*z+c^2$
126	i5; $\exp(z)/z^3-c$
127	i6; $\cosh(z)*z^2-c^2$
128	i7; $1/z^2-cz-c$
129	i8; $\tan(z)-czc$
130	i9; $(\tan(z)-1/z^3)/c^2$
131	j0; $\cosh(z)*\cos(z)+1/c$
132	j1; $z^4/(z^3-c^2)$
133	j2; $1/z^2-\tan(z)+c$
134	j3; $\tan(z)/z^3+c$
135	j4; $1/z^3-cz+1/c$
136	j5; $z^3+\tan(z)*c$
137	j6; $1/z^3-\tan(z)-c$
138	j7; $\tan(z)-\sin(z)+c$
139	j8; $1/z^2-\tan(z)+1/c$
140	j9; $z^4+\sin(z)/c$
141	k0; Mandelbrot set(sine variation)
142	k1; $z^{1.5}+c$ -- Godwin Vickers
143	k2; $(z^2-z^(2-s))/s+c$ -- Escher set by Roger Bagula
144	k3; $z^2+z/(z +c)$ -- Roger Bagula
145	k4; quantum set -- S.M. Ulam
146	k5; prey predator #1 -- Roger Bagula
147	k6; prey predator #2 -- Roger Bagula
148	k7; Klein group #1 -- Roger Bagula
149	k8; Klein group #2 -- Roger Bagula
150	k9; Klein group #3 -- Roger Bagula
151	L0; Loxodromic by Thomas Kroner (fixed type)
152	L1; squared loxodrome
153	L2; Gedatou by Thomas Kroner (fixed type)
154	L3; Ventri by Thomas Kroner (fixed type)
155	L4; squared gedatou
156	L5; $fn(z)-cfn(z)$ (Note 5)
157	L6; $fn(z)+fn(z)+c''$
158	L7; $cfn(z)+c''$
159	L8; $fn(z)+cfn(z)+1''$
160	L9; $fn(z)+c$
161	m0; $fn(fn(z))+c$
162	m1; $fn(z)+fn(c)$
163	m2; $fn(z)+zn+c$
164	m3; $cfn(z)+zn$
165	m4; $fn(z)+kzn+j$ -- generalized phoenix curve
166	m5; $fn(z)*fn(z)+c$
167	m6; $fn(1/(fn(z)+c))$

168	m7; $(1/fn(z))^2+c$
169	m8; $(1/fn(z))^3+c$
170	m9; $fn(z)/(1-fn(z))+c$
171	n0; Sinus by Thomas Kromer (fixed type)
172	n1; Sinus #2 by Thomas Kromer (fixed type) (Note 6)
173	n2; Rings of Fire by Thomas Kromer (fixed type) (Note 6)
174	n3; Teres by Thomas Kromer (fixed type)
175	n4; z^2+y+c
176	n5; $z^2+y[n+1]+c$
177	n6; z^2+zi+c
178	n7; $z^2+zi[n+1]+c$
179	n8; $zr^2+3zi+c$
180	n9; $zr^2+4zi+c$
181	o0; $z^2+timewave[n]+c$ -- Note 7

Hydra Formulas:

1	p0; z^2+c
2	p1; $cz(1-z)$
3	p2; $z(z-1)+c$
4	p3; cz^2-c
5	p4; z^2+cz^2+c
6	p5; z^3+c
7	p6; $((z^2)*(2z+c))^2+c$
8	p7; $z^2+j+kzn$
9	p8; z^2-z^3+c
10	p9; z^2-cz^3+c
11	r0; z^3+zc+c
12	r1; cz^2+zn+c
13	r2; z^3-z+c
14	r3; z^2+z+c
15	r4; $(z^2-c)(z+1)$
16	r5; cz^3+c
17	r6; z^2+cz+c
18	r7; $z+cz^2+c$
19	r8; $z^3+j+kzn$
20	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:

- OI -- rotate OE-OK to OI-OE
- OJ -- rotate OE-OK to OJ-OI
- OK -- rotate OE-OK to OK-OJ

To normalize the C and CC constants:

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

Alternate octonion initialization:

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

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

Note 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: |
| arccoath(w). | | | |
| 33: vers(w). | 34: covers(w). | 35: L3(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: |
| i*(abs(wy)+wx) . | | | |
| 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 $\text{PI}/2$ 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.

11.6 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.

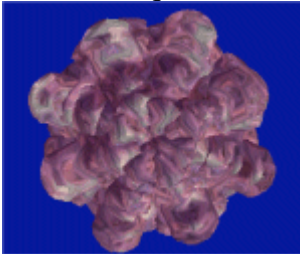
Wegner, Tim and Tyler, Bert: "Fractal Creations", Second Edition, Waite Group Press, Corte Madera, CA, 1993.

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

11.7 About Moquela

About Moquela

>>>> Moquela™ v3.056 ©2004 by Terry W. Gintz



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

Moquela uses the OpenGL for Windows 3D graphics library from Silicon Graphics.

Memory requirements for Moquela vary with the size of each drawing window Moquela opens, ranging from approximately 2.4 megabytes additional memory for each 640X480 window to an additional 24 megabytes for a 2048X1536 window. A minimum of 256MB system memory is recommended.

Acknowledgements: The polygonizing routine used to convert quaternions into Wavefront models was originally written by John C. Hart. Moquela's built-in texture types were originally created by Linas Vepstas in 1994 to illustrate his OpenGL extrusion library. Wavefront is a trademark of Alias|Wavefront, a division of Silicon Graphics Limited. The Wavefront object loader in Moquela uses the GLM library written by Nate Robins.

11.7.1 Chronology

Chronology

History of this program:

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

May 1990 -- ZPlot 1.3d -- added 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 -- Dofu 1.0 -- supports new Ferguson/Gintz plug-in spec.

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

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

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

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

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

May 1999 -- Fractal Zplot 1.18 -- added Phong highlights, color-formula mapping and

random fractal methods.

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

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

- B -

button: abort 13
button: batch 13
button: bmp 16
button: channel c1 15
button: channel c2 15
button: channel cj 15
button: channel o1 15
button: channel o2 15
button: channel Q1 14
button: channel Q2 14
button: Cp 16
button: draw 14
button: figures 13
button: formula 14
button: info 14
button: jpg 16
button: load 16
button: new 12
button: params 14
button: png 16
button: save 16
button: size 13
button: view 13
button:Light 14

- C -

color: edit palette 26

- D -

definition: texture 29
demo: batch mode 40
demo: random coctonion 39
demo: random composite 40
demo: random cubic julia 39
demo: random cubic mandelbrot 38
demo: random octonion 39
demo: random quaternion 37
demo: random quaternion2 38

- E -

edit: copydata 23
edit: cubic values 24
edit: figure list 26
edit: formula 24
edit: lighting 26
edit: octonion parameters 25
edit: parameters 24
edit: pastedata 23
edit: size 23
exit 22

- F -

files: import cubic parameters 20
files: import hydra parameters 20
files: import podme parameters 20
files: load jpeg 19
files: load png 20
files: managing 17, 18, 19, 22
files: save model 21
files: save q polygon 21
files: set max faces 21
files: set max vertices 22
files: write jpeg 19
files: write png 20
formula: custom sign matrix 33

- H -

help: about Moquela 52
help: bibliography 51
help: built-in formulas 45
help: channels 12
help: chronololgy 53
help: hot keys 42
help: parser info 43
help: remote 12
help: tutorial 8, 41

- I -

image: abort 28
image: auto alert 28
image: auto remote 28

image: composite 29
image: draw 27
image: figure 29
image: pilot 29
image: redraw 28
image: show picture 28
image: texture 29

- R -

render: loxodromic 34
render: switch 34
rgb define 26

- S -

status bar 35

- T -

toolbar 34
type: complexified quaternion 32
type: cubic 32
type: custom quad 33
type: hypernion 31
type: julia 30
type: mandelbrot 30
type: quaternion 31