MiSZle Application Help

Table of Contents

1 Main index

MiSZle Help Index

Getting Started

Matrix Algebra, An Introduction by Godwin Vickers

MiSZle Remote

Channel Guide

Commands

1.1 Title Bar

Title Bar

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

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

A title bar may contain the following elements:

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

1.2 Scroll bars

Scroll bars

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

<< Describe the actions of the various parts of the scrollbar, according to how they behave in your application. >>

1.3 Size

Size command (System menu)

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

After the pointer changes to the four-headed arrow:

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

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

Shortcut

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

1.4 Move

Move command (Control menu)

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

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

Shortcut

Keys: CTRL+F7

1.5 Minimize Command

Minimize command (application Control menu)

Use this command to reduce the MiSZle 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. MiSZle 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. MiSZle determines which window is previous according to the order in which you opened the windows.

Shortcut

Keys: SHIFT+CTRL+F6

1.9 Close

Close command (Control menus)

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

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

Shortcuts

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

1.10 Restore

Restore command (Control menu)

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

1.11 Switch to

Switch to command (application Control menu)

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

Shortcut

Keys: CTRL+ESC

Dialog Box Options

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

Task List

Select the application you want to switch to or close.

Switch To

Makes the selected application active.

End Task

Closes the selected application.

Cancel

Closes the Task List box.

Cascade

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

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

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

2 MiSZle Remote

MiSZle Remote

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

2.1 Channel Guide

Channel Guide

The six channels accessed via the MSZ remote:

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

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

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

MT: Random Mat-trap fractals

M1: Random Matrix -- traditional 3D quaternion and hypercomplex quaternion fractals

M2: Random Composite -- extended search through non-traditional formulas

2.2 New

New button

Use this button to open a new drawing window in MiSZle. 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 Undo

Undo button

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

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. Videos are limited to the standard 4/3-vga aspect or 1/1. The minimum size for an image is 40X30. Note: if the image is less than 100 width, the aspect must be 4/3 for solid guessing to work properly.

2.5 Color

Color button

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

2.6 Batch

Batch button

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

2.7 Fvr

FVR button

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

2.8 Draw button

Draw button

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

2.9 Abort button

Abort button

Use this command to stop drawing. Clicking inside a window's drawing area or close box (or the program close box) will also stop the drawing. Note: once a plot has started MiSZle 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.

2.10 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.

2.11 Scan

Scan button

This generates a quaternion Julia set from a formula's Mandelbrot 'P' space. Random points in a formula's current Mandelbrot space are scanned for an interesting Julia set. Rendering options are maintained in the current fractal. Equivalent to the 'F' hot key.

2.12 Rend

Rend button

The current coloring filter and lighting variables are applied. This allows you to see what the surface texture looks like before the fractal is finished drawing. Note: to randomize the coloring filter, click on the Rand Rend button or select Random Render from the Demo menu.

2.13 Help

Help button

Use this button to open the help index for MiSZle.

2.14 Channel J1

Random Julia (Channel J1 button)

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

2.15 Channel J2

Random Julia (Channel J2 button)

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

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

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

2.16 Channel S/B

Random Stalks and Bubbles (Channel S/B button)

A random Julia fractal is generated using one of Paul Carlson's orbit traps or bubble method.

2.17 Channel M1

Random Matrix (Channel M1 button)

A random 3D matrix fractal is generated. Like Random Julia, a set of formulas appropriate for 3D matrices is scanned to find an interesting Julia set, and then the parameters are adjusted to produce a 3D matrix image. The ranges are reset, Hj is set to 2.0, and the lighting is set for optimum viewing.

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

2.18 Channel M2

Random Composite (Channel M2 button)

A random 3D Julia fractal is generated using one of the composite Types. Two formulas are

selected at random and mixed using one of Types 2,3, 7 or 8. This option can be applied to all built-in matrix formulas.

2.19 Channel MT

Random Mat-Trap (Channel MT button)

A random mat-trap fractal is generated using one of Paul Carlson's orbit trap methods and a matrix image. A separate image is created for both matrix and orbit-trap formulas, in figures 1 and 2, then the figures are merged and drawn as one image.

2.20 Save

Save button

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

2.21 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.

2.22 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.)

2.23 Png

PNG radio button

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

2.24 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.

2.25 |||||

||||| button

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

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

2.26 >

> button

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

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

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

2.27 []

[ ] button

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

2.28 V

V button

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

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

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

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

3 File menu

File menu commands

The File menu offers the following commands:

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

Import

Load Palette [PQZ] [PL]	Load QuaSZ palette file.
Load Palette [MAP]	Load a Fractint map file.
Load Texture	Load QuaSZ texture file [QTX]

Export

Save Palette [PQZ]	Save current palette.
Save Texture	Save texture file [QTX].
Save as OBJ	Save polygonized matrix as Wavefront object.
Simplify Mesh	Simplify mesh.
Save as POV	Save polygonized matrix as a pov triangle object.
Smooth Triangles	Convert triangle mesh to smooth_triangle mesh.
Set Max Faces	Set target face size for simplify and smooth options.
Save as WRL	Save polygonized matrix as virtual reality file.
Save as DXF	Save polygonized matrix as AutoCad dxf

file.

Set Maximum Vertices Set maximum number of vertices allocated for M polygon.

Exit Exit MiSZle.

3.1 File New command

New command (File menu)

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

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

Shortcuts

Keys: CTRL+N

3.2 File Open command

Open command (File menu)

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

You can create new images with the New command.

Shortcuts

Toolbar: 

Keys: CTRL+O

3.2.1 File Open dialog box

File Open dialog box

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

File Name

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

List Files of Type

Select the type of file you want to open.

Drives

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

Directories

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

Network...

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

3.3 File Close command

Close command (File menu)

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

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

3.4 File Save command

Save command (File menu)

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

Shortcuts

Toolbar: 
Keys: CTRL+S

3.5 File Save As command

Save As command (File menu)

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

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

3.5.1 File Save As dialog box

File Save As dialog box

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

File Name

Type a new filename to save a drawing with a different name. MiSZle adds the extension .msz.

Drives

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

Directories

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

Network...

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

3.6 File Load Parameters command

Load Parameters command (File menu)

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

3.7 File Load Palettes command

Load Palettes command (File menu)

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

3.8 File Open [JPG] command

Open [JPEG] command (File menu)

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

3.9 File Open [PNG] command

Open [PNG] command (File menu)

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

3.10 File Save Parameters command

Save Parameters command (File menu)

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

3.11 File Save Palettes command

Save Palettes command (File menu)

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

3.12 File Save As [JPG] command

Save As[JPEG] command (File menu)

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

3.13 File Save As [PNG] command

Save As [PNG] command (File menu)

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

3.14 File Load Palette [PQZ] command

Load Palette [PQZ] command (File menu)

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

3.15 File Load Palette [MAP] command

Load Palette [MAP] command (File menu)

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

3.16 File Load Texture command

Load Texture command (File menu)

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

3.17 File Save Palette [PQZ] command

Save Palette [PQZ] command (File menu)

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

3.18 File Save Texture command

Save Texture command (File menu)

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

3.19 File Save Q Polygon [OBJ] command

Export -> Save as OBJ command (File menu)

Use this command to save a 3D matrix as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a matrix formula, and then writes the triangles to a Wavefront object file. The memory requirements for this routine are high, 20MB or more for a typical Julia set 3D matrix rendered at 320X240. The output file can be very large too, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object. Precision is set with the Steps variable in the Initial Values window, where precision=10/Steps.

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

3.20 File Simplify mesh command

Export -> Simplify Mesh command (File menu)

When this flag is set (default on) the object meshes are simplified using Garland's mesh-simplification algorithm before outputting to a Wavefront obj or POV mesh file, resulting in a much smaller export file. Set the number of facets in the target mesh file with the Set Max Faces command. You can set the resolution of the object as high as necessary (with the Params/Steps variable) to produce a finely detailed quaternion, but the output file remains about the same. Use the smoothing feature in Bryce to smooth the resulting object mesh.

3.21 File Save Q Polygon [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 matrix 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 matrix 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.

3.22 File Smooth command

Export -> Smooth command (File menu)

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

3.23 File Set Max Faces command

Export -> Set Max Faces command (File menu)

Set the number of facets in the target mesh file (obj or POV format.) Use the minimum number of faces to produce the quality of mesh desired. It is better to reduce faces to a minimum and do the smoothing in Bryce than to export an object at maximum resolution without mesh simplification. Object files load and smooth much faster in Bryce, and smoothing is usually necessary anyway to reduce jagged edges or blocky facets. (POV uses smooth_triangles to accomplish the same thing.) Up to 30X mesh reduction or more is possible with Garland's mesh-simplification algorithm.

3.24 File Save Q Polygon [WRL] command

Export -> Save as WRL command (File menu)

Use this command to save a 3D matrix as a true 3D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a matrix formula, and then writes the triangles to a virtual reality file. The memory requirements for this routine are high, 20MB or more for a typical Julia set 3D matrix rendered at 320X240. The output file can be very large too, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object. Precision is set with the Steps variable in the Initial Variables window, where precision=10/Steps.

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

3.25 File Save Q Polygon [DXF] command

Export -> Save as DXF command (File menu)

Use this command to save a quaternion as a true 3-D object. This uses John C. Hart's Implicit code (Quaternion Julia Set server) to polygonize a matrix formula, and then writes the triangles to an AutoCad dxf file. The memory requirements for this routine are 20MB or more for a typical Julia set quaternion rendered at 320X240. The output file can be very large too, 4MB or more, depending on the precision required. The higher the precision, the smoother the finished object becomes. Precision is set with the Steps variable in the Quaternion window, where precision=10/Steps.

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

3.26 File Set Max Vertices command

Set Max Indices (File menu)

Use this command to set the maximum number of indices that are allocated by the polygonizing routine. Default is 500,000 indices. Use less to limit the amount of memory used while polygonizing. Use more if necessary for higher resolution. Note: unless you have an application that can use very large object files, there's a limit to how much resolution is obtainable with the polygonizing routine. Bryce4 has problems with object files produced by MiSZle that are much larger than 2.5MB.

3.27 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.

3.28 File Exit command

Exit command (File menu)

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

Shortcuts

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

Keys: ALT+F4

4 Edit menu

Edit menu commands

The Edit menu offers the following commands:

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

4.1 Edit Undo command

Undo command (Edit menu)

Use this command to undo the last action. An image can be continued after an undo, if continue was enabled before the last action. Color-cycling is disabled after using Undo, though.

Shortcut

Keys: CTRL+Z

4.2 Edit Copy command

Copy command (Edit menu)

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

Shortcut
Keys: CTRL+C

4.3 Edit Clip command

Clip command (Edit menu)

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

Shortcut
Keys: CTRL+L

4.4 Edit Paste command

Paste command (Edit menu)

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

Shortcut
Keys: CTRL+V

4.5 Edit Copy Data command

Copy Data command (Edit menu)

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

Shortcut
Keys: CTRL+F

4.6 Edit Paste Data command

Paste Data command (Edit menu)

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

stored in the file is copied to palette 10(F11).

Shortcut
Keys: CTRL+R

4.7 Formula Window

Formula/Type Window

Fun #1 and Fun #2 are combo controls for selecting up to two formulas.. There is additionally a Type control, Latin control, a Column control and a Sign control that determine how the above formulas are processed. The 24 built-in Latin squares and eight sign matrices determine the matrix algebra used in processing the formulas. (For more details on the basis of matrix algebra and the use of the Column control, see Godwin's tutorial on Matrix Algebra.) The Custom Latin and Custom Sign buttons open editors for the Latin squares and sign matrices, so you can customize the matrix algebra. You can enter a value of 0-23 into the Latin control to use one of the built-in Latin squares, or click on the Custom Latin button and make up your own square. The Latin control will be set to 24, for custom. Similarly you can enter a value of 0-7 to use a built-in sign matrix, or click on Custom Sign, make your own, and the Sign control will be set to 8. Note: if you set the Latin or Sign controls to anything outside of their built-in ranges without using the custom editors, the default custom Latin square and sign matrix are 0 and 0 in their built-in lists. The Column control accepts values of 0-3 and determines which column is used in squaring the combined matrix.

The Type control accepts a value of 0 to 9. For a value of 0, the first formula is always used and the second formula is ignored. For a value of 1, the second formula is processed and the first formula is ignored.

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

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

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

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

Type 5 alternates between the two functions while iterating.

Type 6 takes the quotient of both functions.

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

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

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

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

About formula syntax: This applies if you elect to enter your own formula into one of the function boxes and use the parser to generate the plot. Up to 500 user-named-complex variables and constants may be included in a formula. A variable must begin with a letter and may contain numbers and letters only. A variable may be up to 9 characters long. A constant may be up to 20 digits long, including the decimal point. MiSZle uses syntax similar to Fractint's formula style with an initialization section, followed by the main formula, and an optional bailout routine. Bailout is otherwise handled by the 'Bailout' variable in the Initial Values window. Comments may be entered on the same line with a preceding ';'. A ':' terminates the initialization section. Multiple phrases may be entered in the main formula or initialization sections on the same line by using the terminator ',' between phrases. For a complete list of variables, operators and functions recognized by the parser, see Parser Information.

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

Click on the Okay button to use the formulas currently displayed in the window, or Cancel to exit the window without making any changes.

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

4.7.1 Custom Latin Square Editor

Custom Latin Square Editor

Here you enter a value of x, y, z or w into each of the 16 squares that make up the 4X4 Latin square. You can also use the row or column check boxes to swap one column in the square 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. Normally there is only one of each element (x, y, z and w) in any column or row, but

you can vary this to create even more algebraic possibilities.

4.7.2 Custom Sign Matrix Editor

Custom Sign Matrix Editor

Here you enter a value (-100 to 100) into each of the 16 squares that make up the 4X4 sign matrix. Each square represents an index to the "signs" of the composite quad matrix. Normally these are set at -1 or 1, but you can enter other magnitudes, even floating-point values to create unlimited algebraic possibilities. You can also 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.

4.8 Fractal Variables

Fractal Variables (Edit Menu)

The window opened varies with the fractal type selected, and contains most variables that MiSZle scales between key frames of an avi stream.

4.8.1 Initial Values Window

Initial Values Window

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

Composite type (Types 2,3,7 and 8 in the Edit/Formula window) 3D matrix formulas are supported. Type 2(IFS) and Type 3(Fun#1>Fun#2), work as for 2D fractals, while Types 7 and 8 work as follows: Fun#1 is iterated, followed by Fun#2(each function starts with initial q.) For Type 7, initial q is then selected from the highest composite q result (the higher of each hr, hi, hj and hk) of each function. Iteration continues to the loop-break point. For Type 8, initial q is selected from the lowest composite result of each function.

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

The Plane variables B (Back), F (Front), and P (Position) allow you to flatten part of the 3D matrix figure. For normal plotting, these variables default to 0. To have any effect on the image, the Back variable must be greater in value than the Front variable. The difference between back and front variables determines where the image is flattened. These variables

are limited to +/-9.99, with normal values being in the range –z to +z. The position variable sets the color of the flattened plane. Note: if you set the Back variable less than -z, and the front variable greater than or equal to –z, you can get a gradient in the background, depending on the lighting and coloring settings. This sometimes has the effect of placing part of the 3D matrix in a fog-like dimension.

4.8.2 Parameters Window

Parameters Window

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

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

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

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

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

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

Related Topic:
Initial Valuesdescribes the 3D matrix generator's data-collection window.

4.9 Size

Size (Edit menu)

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

4.10 Edit Mat-trap

Mat-Trap (Edit menu)

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

4.11 Ray-Tracing Window

Ray-Tracing Window

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

The Lighting variables shininess, highlight, gamma and ambient are used to adjust ambient light and highlights. The ranges for these variables appear beside their label. Decreasing the shininess value increases light reflected by the 3D matrix and the apparent sheen on the 3D matrix's surface. The ambient value controls the amount of ambient light that illuminates the 3D matrix. The highlight value increases or decreases the specular (Phong) highlighting, while the gamma value increases or decreases the intensity of the light source's illumination. Once a plot is started, the lighting variables and light point can be changed without redrawing the 3D matrix.

Click the Apply button to redisplay a plot after changing the lighting variables or light point. Click the Okay button to close the Ray-Tracing Window, applying new settings, if the variables were modified. Click on Cancel to revert to the state that existed when the ray-

tracing window was opened. Click on Defaults to set the lighting and viewpoint variables to the built-in defaults for these variables.

4.12 Edit Palette

Edit Palettes

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

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

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

Colors are shown in 8 groups of 29 colors, with four colors on the last row. This makes it easy to create divide-by-8, divide-by-4 and divide-by-2 palettes with 232 colors (Use a start color of 0 and an end color of 231.) With MiSZle, a palette is actually 60160 colors, with each succeeding color (except the last) followed by 255 colors that are evenly spread from one color to the next.

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

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

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

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

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

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

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

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

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

4.12.1 Reverse button

Reverse button

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

4.12.2 Neg Button

Neg button

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

4.12.3 Map Button

Map button

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

4.12.4 H/R Button

H/R button

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

4.12.5 Spread Button

Spread button

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

palette.

4.12.6 Copy Button

Copy button

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

4.12.7 SRG Button

SRG button

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

4.12.8 SRB Button

SRB button

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

4.12.9 Okay Button

Okay button

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

4.12.10 Reset Button

Reset button

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

4.12.11 Cancel Button

Cancel button

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

4.12.12 Red Slider

Red slider

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

4.12.13 Red edit box

Red edit box

Shows red/hue value of selected color index.

4.12.14 Green Slider

Green slider

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

4.12.15 Green edit box

Green edit box

Shows green/saturation value of selected color index.

4.12.16 Blue Slider

Blue slider

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

4.12.17 Blue edit box

Blue edit box

Shows blue/value magnitude of selected color index.

4.12.18 Smooth Button

Smooth palette button

Use to create a random palette with smooth color spreads.

4.12.19 Scramble

Scramble

Use to create a palette with random color indexes.

4.13 RGB Thresholds

RGB Thresholds (Edit menu)

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

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

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

These two tests can be combined with the Both option.

4.14 Edit Text command

Text (Edit menu)

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

4.15 Preferences

Preferences (Edit menu)

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

5 Image menu

Image menu commands

The Image menu offers the following commands:

Draw	Draw the picture.
Draw Composite	Draw composite from figures 1-4.
Plot To File	Plot large bitmap images directly to png file.
Plot Files In Directory	Disk render .fsz files in working directory.
Auto Redraw	Redraw image on command.
Auto Clear	Clear drawing area before new plot.

Auto Sound Alert	Enable or turn off sound alerts.
Auto Remote	Open remote automatically at startup.
Auto Time	Show time used to plot image.
Merge Sum	Merge current pixel color with previous color summing colors.
Merge And	Merge current pixel color with previous color anding colors.
Merge Or	Merge current pixel color with previous color oring colors.
Merge High	Merge current pixel color with previous color by choosing highest rgb.
Merge Low	Merge current pixel color with previous color by choosing lowest rgb.
Merge Back	Merge current pixel color with previous color by excluding background color.
Merge Diff	Merge current pixel color with previous color by using difference of colors.
Abort	Abort drawing.
Continue	Continue drawing.
Zoom	Zoom into rectangle.
New View on Zoom	New view on zoom.
Clone	Clone current view.
Pilot	Use Pilot to rotate, pan and zoom.
Scan	Scan Mandelbrot border for 3D matrix Julia set.
Dive	Peel off outer layer of 3D matrix.
Full Screen	View image full-screen.
Pilot	Use Pilot to rotate and alter key 3D matrix variables.
Reset->	Reset coordinates, current figure or all figures
Figure 1	Switch to figure one.
Figure 2	Switch to figure two.
Figure 3	Switch to figure three.
Figure 4	Switch to figure four.
Composite	Select figures to merge.
Count Colors	Count colors used in picture.

5.1 Image Draw command

Draw command (Image menu)

Use this command to draw or redraw the image for the current fractal variables. Clicking inside the draw window with the left-mouse button stops all plotting. Use the Continue command to restart plotting from the current column.

5.2 Image Draw Composite command

Draw Composite command (Image menu)

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

5.3 Plot to file

Plot to File

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

5.4 Plot Files in Directory

Plot Files in Directory

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

5.5 Image Redraw command

Auto Redraw command (Image menu)

With this command disabled (on by default), redraw does not occur except when the Draw command is executed, or Continue. Most of the time you want to see the results of changing a parameter or mapping option, so redraw occurs automatically with parameter or mapping changes. Sometimes you want to change more than one parameter before redrawing the image, or you may want to *continue* a mid-point displacement plot (after loading the file, this is the only way to recreate the plot.) So you need to turn this option off then.

5.6 Image Auto Clear command

Auto Clear command (Image menu)

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

5.7 Image Auto Alert command

Auto Sound Alert command (Image menu)

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

5.8 Image Auto Remote command

Auto Remote command (Image menu)

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

5.9 Image Auto Time command

Auto Time command (Image menu)

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

5.10 Image Merge Sum command

Merge Sum command (Image menu)

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

5.11 Image Merge And command

Merge And command (Image menu)

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

5.12 Image Merge Or command

Merge Or command (Image menu)

With this command enabled (off by default), current pixel color is not overwritten when a

new image is drawn. Instead the colors are merged using an oring algorithm. The auto-clear option must be disabled and solid-guessing off to choose this option. Useful to merge two or more separate fractal images/types with the initial image(s) "bleeding" through.

5.13 Image Merge High command

Merge High command (Image menu)

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

5.14 Image Merge Low command

Merge Low command (Image menu)

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

5.15 Image Merge Back command

Merge Back command (Image menu)

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

5.16 Image Merge Diff command

Merge Diff command (Image menu)

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

5.17 Image Abort command

Abort command (Image menu)

Use this command to stop drawing. Clicking inside a window's drawing area or close box (or

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

5.18 Continue Draw

Continue Draw (Image menu)

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

5.19 Zoom

Zoom (Image menu)

Turns on zoom mode, so that detail of the current plot may be magnified. Alternatively, just click inside any drawing window, move the mouse, and the zoom box will appear. Using the mouse, move the zoom box over the portion of the plot you wish to magnify. Hold the left mouse button to shrink the box or the right button to enlarge it. Use the left and right arrow keys to rotate the box counter-clockwise or clockwise. Use the up and down arrow keys to squash or expand the box, changing the aspect of the image. You start a zoom by pressing the space bar. You abort a zoom by clicking outside the main window or in the title bar, or by pressing the escape key. The program will begin a new plot at the new coordinates. You may zoom in by defining a box inside the current drawing area. You zoom out by drawing a box outside the current drawing area. The outer zoom limits are between -1000 and 1000. The precision is that of double precision (64 bits)
Note: Zooming in a three-dimensional plot is not supported, except for 3D matrix plots, nor is zooming on a random (midpoint displacement) fractal. Rotating a 3D matrix while zooming is possible, but inexact. Depending on the rotational angles set in the 3D matrix window, you may end up rotating on the z-axis, which may rapidly result in overshooting the area of zoom. So avoid large rotations combined with high levels of zoom. It is better to rotate at low zoom levels, and then zoom without rotating, for close-ups. If you change screen resolutions, you must redraw the bitmap image for a function before you can accurately zoom on it.

5.20 Image New View on Zoom command

New view on zoom (Image menu)

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

5.21 Image Clone

Clone (Image menu)

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

5.22 Pilot

Pilot (Image menu)

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

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

5.23 Scan

Scan (Image menu)

Equivalent to the Shift+G hot key. Enabled when the Type is Mandelbrot. A 3D matrix Julia set is generated in sector 2, using an iteration count of 10 and other parameters are changed temporarily to suit 3D matrix plots. (Z is set to 2.0, the rotational variables are reset and the light source is set to the default, if random lighting is enabled.) Once you find an interesting 3D matrix set using "G", like the J command another window is opened that sets the fractal parameters to those in the exploratory qjulia window. The parameters in the exploratory window revert to their original Mandelbrot settings.

5.24 Dive

Dive (Image menu)

Select Dive to go beneath the surface of a 3D matrix. Some 3D matrices have a smooth border that doesn't show the turbulence below the surface. Using the Dive option strips off the border layer to reveal what's underneath.

5.25 Full Screen

Full Screen (Image menu)

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

5.26 Reset

Reset

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

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

5.27 Figure #1

Figure #1

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

5.28 Figure #2

Figure #2

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

5.29 Figure #3

Figure #3

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

5.30 Figure #4

Figure #4

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

5.31 Image Composite command

Composite command (Image menu)

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

5.32 Count Colors

Count Colors

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

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

6 Type menu

Type menu commands

The Type menu offers the following commands:

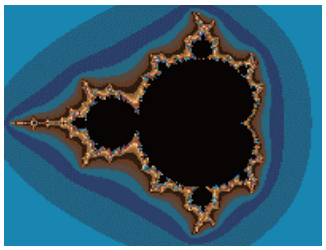
Mandelbrot	Mandelbrot set.
MandelbrotP	Mandelbrot set (orbit starts at pixel.)
Julia	Julia set.
Julia Tower	Julia tower of varying C.
3D Matrix	Set fractal type to 3-D matrix.
2D Matrix	Set fractal type to 2-D matrix.
Zero Init	Sets initial 'z' to zero for Mandelbrot sets.

6.1 Mandelbrot

Mandelbrot

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

of Z must be non-zero to generate anything.



Mandelbrot set

6.2 MandelbrotP

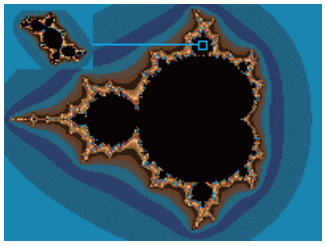
MandelbrotP

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

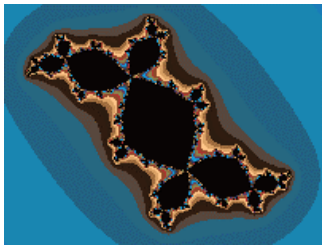
6.3 Julia

Julia

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



Julia from Mandelbrot

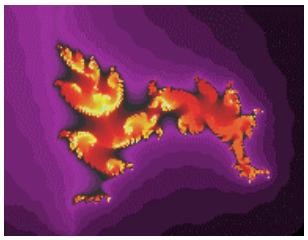


Julia set

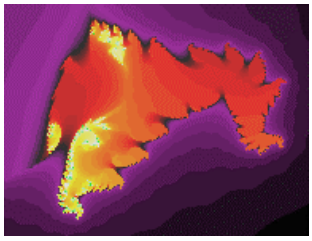
6.4 Julia Tower

Julia Tower

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



Original Julia set



Julia Tower

6.5 Type 2D Matrix command

2D Matrix (Type menu)

Use this command to set the fractal type to a 2D matrix. Variables that affect this fractal type are defined in the Edit/Fractal Variables window.

6.6 Type 3D Matrix command

3D Matrix (Type menu)

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

6.7 Type Zero Init command

Zero Init (Type menu)

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

7 Map menu

Map menu commands

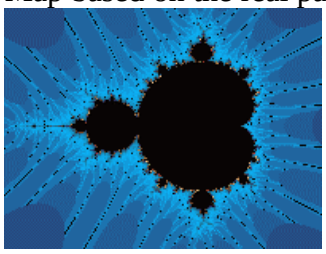
The Map menu offers the following commands:

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

7.1 Z-Real

Z-Real

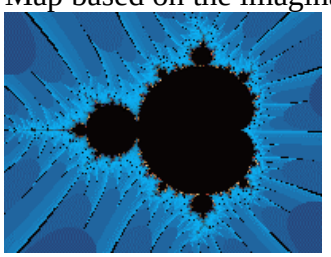
Map based on the real part of the complex number Z; used to map exponential Julia sets, etc.



7.2 Z-Imag

Z-Imag

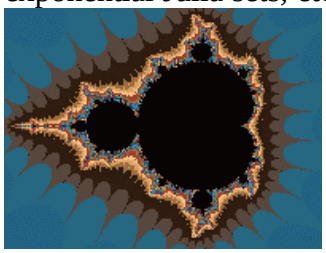
Map based on the imaginary part of the complex number Z.



7.3 Abs(Z-Real)

Abs(Z-Real)

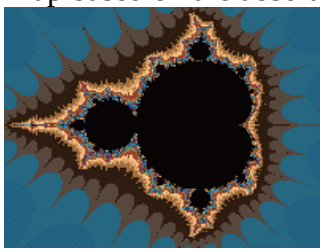
Map based on the absolute value of the real part of the complex number Z; used to map exponential Julia sets, etc.



7.4 Abs(Z-Imag)

Abs(Z-Imag)

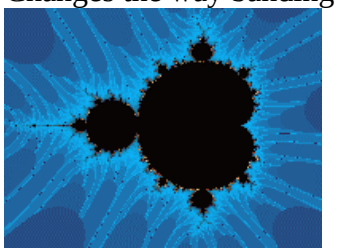
Map based on the absolute value of the imaginary part of the complex number Z.



7.5 Z-Real+Z-Imag

Z-Real + Z-Imag

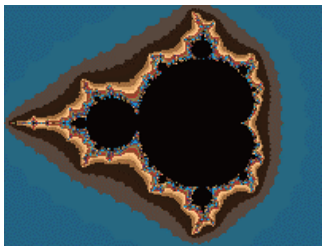
Map based on the sum of the real part and the imaginary part of the complex number Z. Changes the way banding appears in complex mappings.



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

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

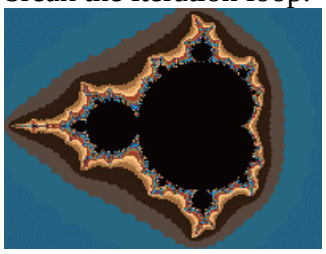
Map based on the absolute value of the real part plus the absolute value of the imaginary part of the complex number Z. Changes the way banding appears in complex mappings.



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

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

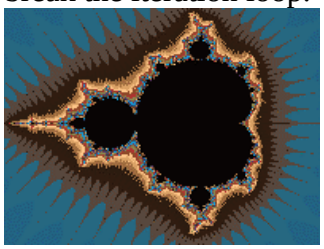
Map based on the greater of the absolute value of the real part or the imaginary part of the complex number Z. Works like a logical 'or', where either part of z must exceed zlimit to break the iteration loop. Changes the way banding appears in complex mappings.



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

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

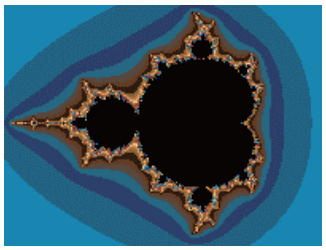
Map based on the lesser of the absolute value of the real part or the imaginary part of the complex number Z. Works like a logical 'and', where both parts of z must exceed zlimit to break the iteration loop. Changes the way banding appears in complex mappings.



7.9 Abs(Z)

Abs(Z)

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



8 Break menu

Break menu commands

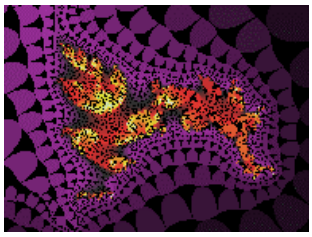
The Break menu offers the following commands:

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

8.1 Biomorph

Biomorph

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



8.2 Bioconvergence

Bioconvergence

This option relates the Biomorph method to convergence for convergent-type fractals

(Newton, Renormalization and Convergence.) A pseudo-biomorphic algorithm is applied to converging points.

8.3 Biomorph Off

Biomorph Off

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

8.4 Orbit traps

Orbit Traps

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

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

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

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

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

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

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

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

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

The Parametric Formula option allows you to enter your own parametric formula for an orbit trap in the Formula box in the Edit Formula window. This works for all built-in

formulas and fractal types except type 9(formula) and 5(random.) An example of how to specify this type of orbit trap is the following parametric formula:
z=0.3*(cos(pixel)^3+i*sin(pixel)^3).
Here, pixel is used to specify the polar angle of z (instead of its usual c or z-plane value), where pixel = atan(imag(z)/real(z)). Note that z is used for setting up the distance variable instead of x# in this case.

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

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

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

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

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

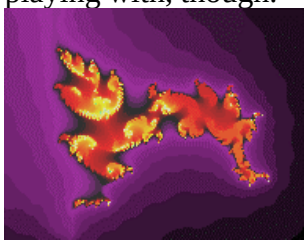
Use 3D matrix as Trap Color -- this option allows you to use a 3D matrix image to achieve true 3D effects in an orbit-trap. Start by defining an orbit trap image in figure 2. Define a 3D matrix image in figure 1. Then set the flag for "Use 3D matrix as Trap Color" and draw an initial "quat-trap" picture.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

transferred from figure 2. When you redraw the image, new orbit-trap values are transferred to figure 2. So if you decide to turn off this option to work on each image separately, both figures are up to date with the last quat-trap image drawn.

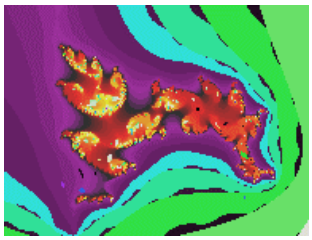
8.5 Renormalize

Renormalization

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



Original picture



With renormalization

8.6 Convergence

Convergence

With the Convergence flag set, the program does a convergence/periodic check on all points. This is similar to the convergence checks done with Newton and Renormalization, but also the orbits of each point are saved to determine if the orbit repeats. When an orbit repeats, the iteration loop is broken and the point colored according to its break time. Depending on the iteration limit, the last 200 points of each orbit are tracked for this check. This flag is mutually exclusive with the Newton, Renormalization, and Boundary Scan flags. May use

Newton display methods 3 and 6-8(alternate convergence tests), by setting the arg gadget to these values.

8.7 Period Check

Period Check

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

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

8.8 Default Function

Default Function

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

9 Render menu

Render menu commands

The Render menu offers the following commands:

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

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

9.1 Boundary-Scan

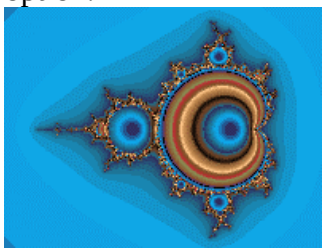
Boundary Scan

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

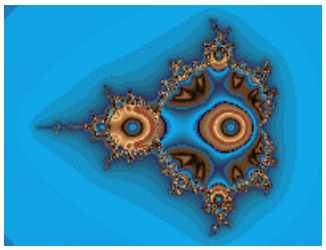
9.2 Level Curve

Level Curve

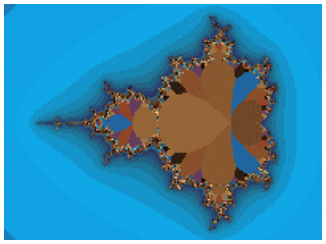
Level-curves map the set points based on how small the value of Z gets. This allows the inside of the complex set to be color-scaled. Log Map #1 produces colored bands on the inside of the complex set. Points are mapped according to what the value of z is at final iteration. Small Log #2 and Linear Map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



Log Map #1



Small Log #2



Indexed Log #3

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

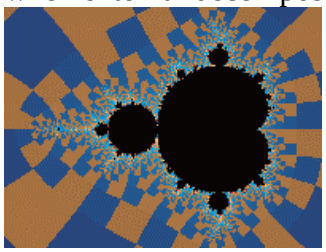
9.3 Decomposition

Decomposition

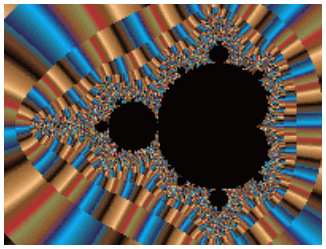
When a Decomposition flag is set, you have the option of performing either a binary or continuous decomposition. Toggle the External/Internal option for either an external or internal decomposition. The Angle-Only option excludes the Coloring-scaling options from consideration when plotting points derived from decomposition. It also excludes escape times in connection with the Angle-Iteration option on the Color-Scaling menu. An external decomposition decomposes points that are outside the complex set. An internal decomposition decomposes the complex set. For Mandelbrot/Julia curves, z-arg is broken into two parts for a binary decomposition. For Newton/Renormalization curves, the binary decomposition is also related to the number of solutions a formula has, if it supports mapping option 1. Continuous decomposition breaks z-arg into n parts, where n=angles(2-256), as set in the Continuous Decomposition window.

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

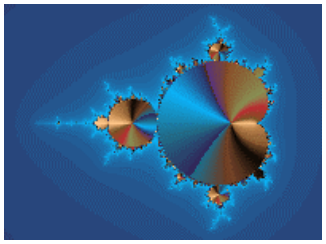
(Consult The Beauty of Fractals by Peitgen & Richter for a mathematical explanation of decomposition.) When Biomorph or Epsilon is decomposed, the tendrils or hairs are decomposed as external points. Use the Set Only flag to emphasize the tendrils and hairs when external decomposition is used.)



Binary Decomposition



Continuous External



Continuous Internal

9.4 Decomposition Off

Decomposition Off

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

9.5 Switch

Switch

When a Switch flag is set, you have the option of switching the real and imaginary parts of Z, or switching Z for C. The real part of Z is exchanged with the imaginary part of Z after each iteration. Using this technique with the Mandelbrot set produces a tricorn-like plot. When Z is switched for C, normally you get Mandelbrots from Julia sets and vice versa. This option

also offers an inverse to the Julia Tower, with glimpses into the elusive Mandel Tower, a sort of gateway to infinity.

9.6 Spin

Spin

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

9.7 Filter

Filter

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

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

9.8 HSV Filters

HSV Filters

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

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

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

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

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

9.9 Coloring Filter

Coloring Filter

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

The variable 'z' is equated to the zbuffer value in this case, so may also be included in the formula. Quaternions and ray-traced fractals normally use palette index one (the second index, zero being reserved for the background color) for their predominant color, with pixel intensities/colors affected by the lighting variables. When the coloring filter formula is defined, up to 235 colors can be used (the full palette) to create mixed textures.

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

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

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

the decimal point).

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

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

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

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

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

9.10 Surface Filter

Surface Filter

Here you define a 3D matrix surface filter based on a real function. Like a coloring filter, except that the formula is used to warp the 3D matrix's shape. The Magnify variable is used to control the intensity of the filter. Use the Preview button to see what the filter looks like with x and y ranges of -2pi to 2pi.
Use the Random Filter button to generate a random surface filter. The best surface filters will use the z value and one or both of the other variables (x or y.)

9.11 Anti-Alias

Anti-Alias

Applies a 2 to 1 or 4 to 1 averaging filter to every pixel plotted, to reduce jaggies and other high-frequency noise. This increases the processing time 4 to 8 times, so is mainly a final rendering method, not for general development use. Not available for most 3D-type fractals (for quaternions, only single-pass mode is supported -- or without ray-tracing, solid-guessing can be used), plot-to-file, midpoint-displacement, 3D backgrounds or with the tesseral solid-guessing method. Note: because of the lengthy time required for applying the anti-aliasing filter, and because anti-aliasing calculates different smoothing colors each time the palette is changed, all color-cycling and palette-switching hot keys are disabled with the anti-alias flag set.

9.12 Link Coloring To Pixel

Link Coloring To Pixel

Set coloring to match absolute coordinates of (3D matrix) image. This uses extra buffers to track a figure's texture, so that when you rotate it, the texture moves with the figure. Due the extra memory required for this command, you won't be able to open an image much larger than 1600X1200, unless you have more than 128MB of system memory.

9.13 Atan Coloring

Atan Coloring

Uses an Atan algorithm by David Makin to color a 3D matrix image. Link Coloring To Pixel is set with this option.

9.14 Bof60 Coloring

Bof60 Coloring

A variation of the Bof60 algorithm found in the classic Pietgen/Ricter text, The Beauty of Fractals, adapted by David Makin to color a 3D matrix image. Link Coloring To Pixel is set with this option.

9.15 Potential Coloring

Potential Coloring

The magnitude of z (at the 3D matrix border) is used to color the (3D matrix) image. Link Coloring To Pixel is set with this option.

9.16 Add Noise

Add Noise

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

9.17 Factors

Factors

Edit noise factors. The Blend variable determines how much noise is added to an image. The higher the blend, the more pronounced the noise appears. This also tends to darken an image, which can be compensated for by decreasing Gamma. The Grain variable determines the frequency of the noise. The higher the grain, the noisier the image appears. You can adjust how the noise maps to an image by changing the scale factors. Higher scale factors make the image noisier on the respective axis (x, y and z.) Additional variables affect the type and shaping of the noise data: Gaussian is an alternate form of noise, while Planet, Check, Tooth,

Barber and Wood apply a specific envelope to the noise. The Marble variable is used to introduce a low frequency or high frequency modulation on top of the noise. You can achieve marble-like textures by combining a high frequency marble value with a low frequency Blend value. The marble variable also adds a high-frequency bump map to the wood envelope.

The Surface Warp variable allows you to apply the same noise to a (3D matrix) figure's shape also, like a surface filter. Small values are best for creating realistic surface variations, like stone and wood grain.

9.18 Texture Scale

Texture Scale

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

9.19 Reset Noise Seed

Reset Noise Seed

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

10 Pixel menu

Pixel menu commands

The Pixel menu offers the following commands:

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

10.1 Phoenix

Phoenix

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

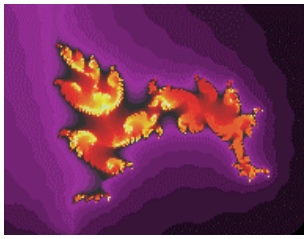
This option is normally used for mapping Phoenix curves (Shigehiro Ushiki), which are Julia-related curves based on the formula $f(z+1)=z^2+p+qz$. 'p' and 'q' are constants, and the 'z' term of 'qz' is actually the value of z^n-1, or the previous value of z before the current iteration. 'zn' is reserved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

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

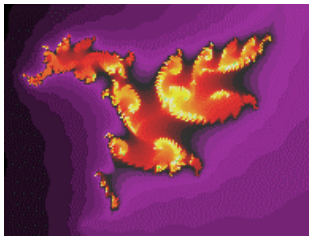
10.2 Invert

Invert

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



Original picture



Inverted

10.3 Invert Off

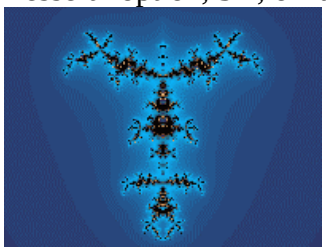
Invert Off

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

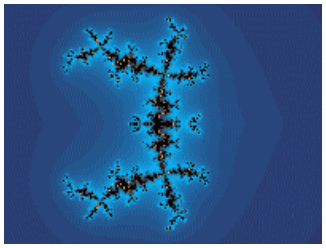
10.4 Symmetry

Symmetry

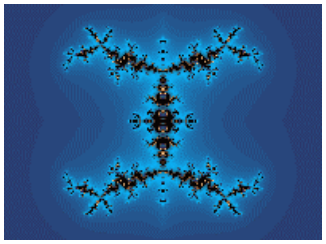
This produces a mirror image from left to right (vertical) or top to bottom (horizontal) or both (xy). You can zoom with symmetry, but the results will be uncertain if the zoom box is off-center on the window or if rotation is used. Symmetry has no effect when used with the Tesseral option, 3D, or random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



Vertical symmetry



Horizontal symmetry



XY symmetry

10.5 Fast

Fast

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

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

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

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

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

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

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

10.6 Solid Guessing

Solid Guessing

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

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

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

Solid-guessing is not available for 3D, midpoint displacement plots.

10.7 Tesseral

Tesseral

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

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

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

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

10.8 Segment

Segment

This option allows you to break a large bitmap (larger than would fit into memory) into tiles that can be reassembled later with a program such as Richard Paasen's Image Arithmetic. The Segments box defines how many tiles the image is broken into (1-225.) The number of tiles horizontally must equal the number of tiles vertically; for example, a 4X4 tiling would consist of 16 tiles total. Segments are numbered 0-(number of segments - 1), with 0 being the upper-left corner tile and following left to right in rows to the last tile at the lower-right corner.

10.9 Cliff's Slice

Cliff's Slice

With this option set, the 4D z-planes are rotated to match C. Pickover's 3D matrix examples in "Computers, Pattern, Chaos and Beauty Only affects 3D plots.

10.10 Torus

Torus

Pixels are mapped around a torus, and then expanded to fit the drawing area. A generalized

form of Earl Hinrichs' torus method, variables are provided for center x and center y to define the c and z radii and may both equal 0.0. Results will vary with the formula used, but resembles the warping effect found in hypercomplex images. Two versions of this method are provided: the Pixel method which uses pixel values to map the torus to the fractal space, and the Two-Pi method which uses an initial rectangle 2 pi by 2 pi to map the torus to a fractal image. With the Two-Pi method, when you zoom the rectangle's size and starting points are changed to match the zooming area. The rectangle's coordinates are saved with the fractal. If you turn off the torus flag after zooming and then reinitialize the torus flag, the rectangle reverts to a 2X2 area, so the image will change accordingly. Rotating is not supported for the Two-Pi method, but does work in a limited way with the Pixel method.

10.11 Torus Off

Torus Off

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

10.12 Use Stencil

Use Stencil

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

11 Color menu

Color menu commands

The Color menu offers the following commands:

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

11.1 Color Cycle command

Cycle command (Color menu)

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

11.2 Color-Scaling menu

Color-Scaling menu commands

The Color-Scaling menu offers the following commands:

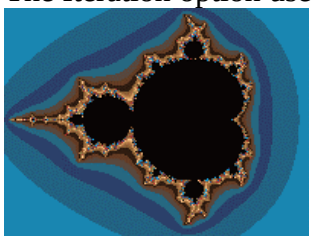
Escape->	Escape-time color scaling.
Level	Color scaling based on log(z).
Continuous Potential	Color scaling based on continuous potential.
Use Level	Use level curve option for all coloring.
Background	Set external points to background color.
Set Only	Plot points in complex set only.
Graded Palette->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

11.2.1 Escape

Escape

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

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



Escape-time coloring.

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

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

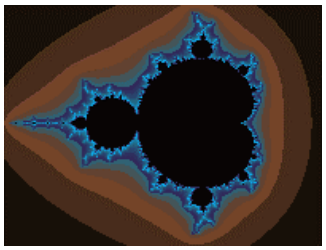
The Angle option use the absolute value of a point's exit angle (theta.) This is the atan method in Fractint.

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

11.2.2 Level

Level

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

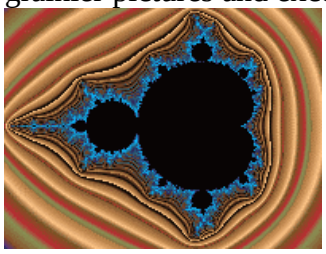


Level coloring.

11.2.3 Continuous Potential

Continuous Potential

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



Continuous-Potential coloring.

11.2.4 Use Level Curve

Use Level Curve

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

11.2.5 Background

Background

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

11.2.6 Set Only

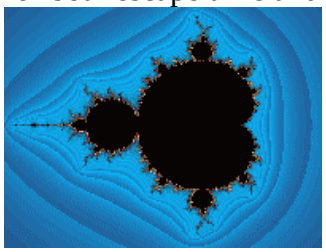
Set Only

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

11.2.7 Graded Palette

Graded Palette

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



Level-graded coloring.

11.2.8 Use Palette

Use Palette 1 for Background Filter

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

11.2.9 Ray Trace

Ray Trace

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

11.3 Palette menu

Palette menu commands

The Palette menu offers the following commands:

Palette #1-21 Use one of 21 palettes.

11.3.1 Palette 1-21 command

Palette command (Palette menu)

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

11.4 Color Divpal1 command

Divide by One Palette (Color menu)

Palette is not split before applying to pixel.

11.5 Color Divpal2 command

Divide by Two Palette (Color menu)

Palette is split into two parts before applying to pixel.

11.6 Color Divpal4 command

Divide by Four Palette (Color menu)

Palette is split into four parts before applying to pixel.

11.7 Color Divpal8 command

Divide by Eight Palette (Color menu)

Palette is split into eight parts before applying to pixel.

12 View menu

View menu commands

The View menu offers the following commands:

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

12.1 View Toolbar command

Toolbar command (View menu)

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

See Toolbar for help on using the toolbar.

12.1.1 toolbar

Toolbar

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

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

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

12.2 View Status Bar Command

Status Bar command (View menu)

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

See Status Bar for help on using the status bar.

12.2.1 status bar

Status Bar

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

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

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

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

13 Window menu

Window menu commands

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

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

13.1 Cascade

Cascade command (Window menu)

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

13.2 Tile

Tile command (Window menu)

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

13.3 Arrange Icons

Window Arrange Icons Command

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

13.4 Size DeskTop

Window Size DeskTop Command

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

13.5 1, 2, ...

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

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

14 A/V menu

A/V menu commands

The A/V menu offers the following commands:

Open AVI Stream	Open AVI file for writing and draw initial frame.
Write Frames	Write frames to AVI file.
AVI Variables	Edit AVI variables.
Close AVI Stream	Close an existing AVI stream.

View AVI	View an AVI animation file.
Start Midi	Enable midi sampling and set sample size.
End Midi	Disable midi sampling.
Save Midi	Save midi data to mdf and mid files.
Load Midi	Load midi variables from mdf file.
AVI Composite	Generate composite video.
AVI Object	Output video frames as obj file.
AVI WRL	Output video frames as wrl files.

14.1 Open Avi Stream

Open Avi Stream…

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

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

14.2 Write Frames

Write Frames…

With this option, frames are written to a stream based on the difference between the current key frame and the previous key frame. 15 frames make a second of animation. The first key frame is written when you open a stream. The next key frame is created each time you use this option. In between you can zoom or change Avi variables as much as necessary. The stream is only written to when this option is used. The last key frame is automatically saved after the 'tween' series is written. The number of frames may range from 1-1500 frames between keys. With a frame number of 1 only the key frames are written. This allows animation to be created that incorporates all scalable variables in MiSZle.

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

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

Notes: key frames are saved in parameter files (msz), with filenames of "bvf_image#_title.msz", where '#' is the number of the keyframe and 'title' is the name of the working fractal file. New: If you open a video stream after setting up a batch mode (Demo

menu), then the frames will be written as a series of bmp, obj or wrl files, depending on whether AVI Object or AVI WRL is also checked.

14.3 Avi Variables

Avi Variables…

This window contains all the major variables that MiSZle now scales between key frames of an avi stream. They are identical to some of the variables found in the Parameters, Formula and Epsilon windows, plus a few other windows. If you decide to change a variable not included in this window, no scaling occurs. A few exceptions are the newlimit, limit, invert, torus, 3D matrix variables and the palette indexes (if you change palettes between key frames.) These are scaled also. If you change the basic function type, formula or color-scaling method, morphing isn't supported for these sorts of changes.

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

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

14.4 Close Avi Stream

Close Avi Stream

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

14.5 View Avi

View Avi…

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

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

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

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

14.6 Start Midi..

Start Midi…

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

14.7 Stop Midi..

Stop Midi…

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

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

14.8 Save Midi

Save Midi…

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

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

14.9 Load Midi..

Load Midi…

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

14.10 Avi Composite

AVI Composite

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

14.11 AVI Object

AVI Object

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

14.12 AVI WRL

AVI WRL

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

15 Demo menu

Demo menu commands

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

Random Julia	Generate random Julia fractal.
Random Julia2	Generate random Julia fractal (includes composites).
Random Stalks and Bubbles	Generate random orbit-trap or bubble fractal.
Random 3D Matrix	Generate random 3D matrix fractal.
Random 3D Composite2	Generate random composite 3D matrix image.
Random Mat-Trap	Generate random mat-trap fractal.

Random Render	Select a random rendering.
Batch Mode	Repeat random fractal and save to file.

15.1 Random Julia

Random Julia (Demo menu)

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

15.2 Random Julia2

Random Julia2 (Demo menu)

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

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

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

15.3 Random Stalks and Bubbles

Random Stalks and Bubbles (Demo menu)

A random Julia fractal is generated using one of Paul Carlson's orbit traps or bubble method, or a custom orbit-trap formula.

15.4 Random 3D Matrix

Random 3D matrix (Demo menu)

A random 3D matrix fractal is generated. Like Random Julia, a set of formulas appropriate for 3D matrices is scanned to find an interesting Julia set, and then the parameters are adjusted to produce a 3D matrix image. The ranges are reset, Hj is set to 2.0, and the lighting is set for optimum viewing.

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

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

15.5 Random 3D Comosite

Random 3D Composite (Demo menu)

A random 3D Julia fractal is generated using one of the composite Types. Two formulas are selected at random and mixed using one of Types 2,3, 7 or 8. This option can be applied to all built-in matrix formulas.

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

15.6 Random Mat-Trap

Random Mat-Trap (Demo menu)

A random mat-trap fractal is generated using one of Paul Carlson's orbit trap methods and a 3D matrix image. A separate image is created for both 3D matrix and orbit-trap formulas, in figures 1 and 2, then the figures are merged and drawn as one image. See Orbit-Trap options for further details on quat-trap pictures.

15.7 Random Render

Random Render (Demo menu)

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

15.8 Batch Mode

Batch mode (Demo menu)

Here you set parameters for batching and saving random-generated images to disk. When the Repetitions value is non-zero, up to 1000 random images can be generated and saved to disk. Use a unique Filename to prevent batch files from overwriting existing image files. The Scan Limit directs the program on how many scans it makes through each formula before it skips to a new formula (if an interesting Julia fractal hasn't been found.)

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

16 Help menu

Help menu commands

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

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

16.1 Tip of the day

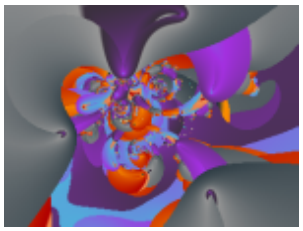
Tip of the day

Supplies extra information (in addition to this help file) that may be useful in some cases. Users are welcome to add to the file "tips.txt", if they have discovered an undocumented feature or method of obtaining certain results. These "tips" may become part of the next release of MiSZle (drop them in an email to terrywg@attbi.com)

16.2 Getting Started

Getting Started

Welcome to MiSZle!



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

The two methods of fractal composition that most fractal programs use are iterative and orbital. With the iterative method, which MiSZle uses, a formula is successively iterated until some criteria are met then the point is plotted. The process of iteration goes like this: starting from some initial value, a formula is evaluated, and then the result is used to reevaluate the formula. The formula is usually complex (based on complex variables of the form x+yi, where i=sqrt(-1)), the most commonly used formula being z^2+c. For two-dimensional fractals, the screen is mapped into ranges of the variables z or c. When the c variable is changed during iteration, the map produced is called the Mandelbrot set, after Benoit Mandelbrot. The z variable may be initialized at zero or some other value. When the z variable is changed during iteration, the c variable remaining fixed, a Julia set is formed. The Mandelbrot set is frequently considered a map of all Julia sets. If you choose a point for c anywhere on the border of the Mandelbrot set, a Julia set can be generated that looks very similar to the area it was taken from. The formula that came to be known as the Mandelbrot set, z^2+c, is only one of many complex formulas that can be used. MiSZle introduces a new method of defining generalized 2D and 3D fractal formulas -- matrix algebra -- which is the same algebra found in DynaMaSZ.

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

Redrawing the title picture: reload the title picture by using the New command on the File menu, and double click on title.msz. Click on the autoclear option in the Image menu to turn that option off. Use the Clear Screen command, shift-C to erase the screen. Change the image to Figure 2. Click on Continue in the Parameters window (or the Continue icon on the

toolbar) to redraw the sky. Change the image to Figure 1. Now use the Draw command on the Image menu to redraw the 3D matrix part of the title picture. Note: the 3D matrix used for the title picture was actually drawn at higher resolution than the image you see when redrawn. The default resolution (200 steps) gives you a quick look at a 3D matrix image when exploring, but you need to increase this to at least screen width (800 steps or more) to get a quality image.

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

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

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

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

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

16.3 Hyperalgebras

Tridimensional visualization of 4D Julia sets generated from Latin squares and sign rules

January 2002
Version 1.2
Godwin Vickers

1/ Introduction
2/ Latin squares
3/ Sign rules and sign matrices

1/ Introduction

This document shows a new method to define 4D algebras in R4 and higher dimensions. Iterative functions can be applied to these algebras and 4D Julia sets other than quaternionic Julia-sets can then be rendered and explored. The purpose of this document is to describe these new algebras and how they can be created from a set of rules that can be implemented in a specific algorithm. This is not a tutorial on 4D Julia sets or how to turn 4D Julia sets to 3D pictures (via Z-buffering, ray-tracing or radiosity). Plenty of tutorials explain step-by-step how to render 4D Julia sets, how to calculate the distance between a light source or a viewpoint and a 4D Julia set, how to calculate the normal at each point on the surface of a 4D Julia set and which theorems and formulas are involved in these processes. For more information check the links at the bottom of this document.

In the late 1970's Mandelbrot applied iterative functions to complex numbers and discovered mathematical objects which offered a potentially infinite level of detail at all scales, these details following a rule of self-symmetry as scale decreases or increases. Mandelbrot called these objects fractals. 2D fractals have been extensively visualized for the two past decades but 4D fractals are still hiding some unknown territories that we can now explore thanks to the combination of appropriate algorithms and the increasing speed of microprocessors.

Quaternions are naturally the start of this journey because they are a 4D generalization of complex numbers (or the complex plane). Quaternions are the only normed division algebra of order 4. In fact these algebras are all Clifford algebras and this is the way they are categorized:

Cl(0) 1 dimension R (real numbers)
Cl(1) 2 dimensions C (complex numbers)
Cl(2) 4 dimensions H (quaternions)
Cl(3) 8 dimensions O (octonions)

Once Cl(8) is reached, we obtain a 256-dimensional algebra that is the building block of all other Clifford algebras since they can all be calculated using Cl(8) using a periodicity theorem. All Clifford algebras are normalized.

The unit quaternion can be considered as a rotation in 4 dimensions and is defined by the following matrix:

x -y -z -w
y x w -z
z -w x y
w z -y x

Once you apply an iterative function to this matrix you can generate a 3D slice of your 4D Julia set. What we will show in this document is how you can modify the original quaternion matrix algebra and define a new set of 4D matrices which follow several building rules in order to explore new 4D Julia sets.

What we need to do first is dissociate the symbols from the signs so we can create a set of two matrices which, combined together, generate the initial matrix. Here are the two separate matrices:

x y z w
y x w z
z w x y
w z y x

and

+ - - -
+ + + -
+ - + +
+ + - +

(where '+' means '+1' and '-' means '-1')

Now that we have a matrix that represents the symbolic structure of the unit quaternion and a sign matrix that defines its sign rules we can analyse the two matrices and find the laws that create them. This reverse-engineering process leads not only to the rules of quaternion algebra but also to the rules of all other similarly built 4D algebras.

2/ Latin squares

The symbolic matrix we have extracted from our unit quaternion matrix is a Latin square. So in order to explore more algebras we'll define all the possible Latin squares of order 4 and keep the essential ones.

While magic squares are made of numeric values only, Latin squares are only made of symbolic values. Latin squares are to magic squares what algebra is to arithmetic.

The rules to construct a Latin square are the following:

-Once you define the order n (n being an integer) of your Latin square, n becomes the number of dimensions of your Latin square: this means there are n rows, n columns, and n different symbols.

-Each symbol must appear once and only once on each row and column of the Latin square.

There are exactly 4!*24 Latin squares of order 4 ((1*2*3*4)*24 = 24*24 = 24^2 = 576).

These 576 Latin squares are the result of the symbol permutations of the 24 Latin squares of order 4 starting with a first row in lexicographic order. This means that 24 essential Latin squares can generate 576 (24^2) Latin squares if you just permute their symbols.

Here are the 24 Latin squares of order 4 with their first row in lexicographic order:

```
x y z w  x y z w  x y z w  x y z w  x y z w  x y z w
z w y x  y w x z  w x y z  z w y x  y w x z  y x w z
w z x y  w z y x  z w x y  y x w z  z x w y  z w y x
y x w z  z x w y  y z w x  w z x y  w z y x  w z x y

x y z w  x y z w  x y z w  x y z w  x y z w  x y z w
w z y x  w z x y  w z x y  z w x y  w x y z  y x w z
z x w y  y x w z  z w y x  y x w z  y z w x  w z y x
y w x z  z w y x  y x w z  w z y x  z w x y  z w x y

x y z w  x y z w  x y z w  x y z w  x y z w  x y z w
z w x y  w z y x  y x w z  z w x y  z w x y  w z y x
w z y x  z w x y  w z x y  y z w x  w x y z  y x w z
y x w z  y x w z  z w y x  w x y z  y z w x  z w x y

x y z w  x y z w  x y z w  x y z w  x y z w  x y z w
y z w x  z x w y  w z y x  z x w y  y z w x  y x w z
w x y z  y w x z  y w x z  w z y x  z w x y  z w x y
z w x y  w z y x  z x w y  y w x z  w x y z  w z y x
```

For Latin squares of order 4, there are exactly 4 normalized Latin squares. The correct terminology for normalized Latin squares is 'reduced' or 'standardized': this means their first row and column is in lexicographic order.

3/ Sign rules and sign matrices

Let's analyse first the sign matrix we have extracted from our quaternion matrix. If we follow the rules of sign multiplication of real numbers we can find the sign multiplication rules of quaternions in R4 (+ * + = +, + * - = -, - * + = - and - * - = +). Note that '+' means '+1' and '-' means '-1'.

```
  + + + +

-  + - - -
-  + + + -
-  + - + +
-  + + - +
```

Row 1: + * - * - * - = -
Row 2: + * + * + * - = -

Row 3: + * - * + * + = -
Row 4: + * + * - * + = -

Column 1: + * + * + * + = +
Column 2: - * + * - * + = +
Column 3: - * + * + * - = +
Column 4: - * - * + * + = +

The product of each value of each row is always negative and the product of each value of each column is always positive. These are the simple rules of our quaternion sign matrix. If you replace each + with a - and each - with a + you also get a quaternion. It doesn't matter if rows are negative and if columns are positive as long as rows are always of the opposite sign of columns and that all rows are of the same sign.

All we actually need are two strings of 4 signs to define the sign rules. If 'r' stands for 'rows' and 'c' for 'columns', we can sum up quaternion sign rules like this:

r++++
c----

And since the relationship between the r sign string and the c sign string is "r is the opposite of c", we only need to define one string of signs because the second string will automatically be the opposite of the first one. So '++++' or '----' is enough to define the sign rules of quaternion algebra.

Sign rules and corresponding sign matrices of order 4:

Four pluses:

a + + + +

Three pluses:

b - + + +
c + - + +
d + + - +
e + + + -

Two pluses:

f + + - -
g - + + -
h + - + -
i - + - +
j + - - +

k - - + +

One plus:

l - - - +
m - - + -
n - + - -
o + - - -

No plus:
p - - - -

a & p are opposites: dual sign rule 1 (ap or pa): ++++ ---- or ---- ++++
b & o are opposites: dual sign rule 2 (bo or ob): -+++ +--- or +--- -+++
c & n are opposites: dual sign rule 3 (cn or nc): +-++ -+-- or -+-- +-++
d & m are opposites: dual sign rule 4 (dm or md): ++-+ --+- or --+- ++-+
e & l are opposites: dual sign rule 5 (el or le): +++- ---+ or ---+ +++-
f & k are opposites: dual sign rule 6 (fk or kf): ++-- --++ or --++ ++--
g & j are opposites: dual sign rule 7 (gj or jg): -++- +--+ or +--+ -++-
h & i are opposites: dual sign rule 8 (hi or ih): +-+- -+-+ or -+-+ +-+-

Overall 2^4 = 16 sign rules, and 16/2 = 8 dual (or opposite) sign rules.

Sign rule 1	**Sign rule 2**	**Sign rule 3**	**Sign rule 4**
+ + + + - - - -	- + + + + - - -	+ - + + - + - -	+ + - + - - + -
Sign matrix 1	Sign matrix 2	Sign matrix 3	Sign matrix 4
+ + + + - + - - - - + + + - - + - + + - + + - +	- + + + + + + + + - - + + + - - + + + - - + + +	+ - + + - - + + + + + + + + - - + + + - + - + +	+ + - + - - + + + - - + + + + + + + + - + + - +

Sign rule 5	**Sign rule 6**	**Sign rule 7**	**Sign rule 8**
+ + + - - - - +	+ + - - - - + +	- + + - + - - +	+ - + - - + - +
Sign matrix 5	Sign matrix 6	Sign matrix 7	Sign matrix 8

```
  + + + -          + + - -          - + + -          + - + -

- + + + -        - - + + +        + + + + +        - - + + +
- + + + -        - - + + +        - + + + -        + + + + +
- + + + -        + + + + +        - + + + -        - - + + +
+ + + + +        + + + - -        + - + + -        + + - + -
```

Overall there are 2^4 = 16 sign rules. If we discard the symmetries there are 16/2 = 8 sign rules left.

4/ Algebras defined as matrices

So far we have dissociated the original unit quaternion matrix in two separate matrices and then we found the laws to create these two matrices and all the other matrices that belong to their group.

Symbolic structure:
There are 576 = 4!*24 Latin squares of order 4.
If we discard permutations we obtain 24 Latin squares of order 4.
Note that among those 24 Latin squares some symmetries or rotations exist. I have read that there are 7 really essential Latin squares of order 4 from which you can calculate the 24 standardized Latin squares and all 576 Latin squares of order 4.

Sign rules:
There are 2^4 = 16 sign rules which are 8 pairs of symmetric sign rules.

If we now combine all the symbolic rules and all the sign rules just like we can combine the symbolic quaternion matrix and its sign matrix we obtain a group in R4 of all the matrices of order 4 which follow both sets of rules. The group we'll define here is composed of 24*8 = 192 algebras.

To combine a Latin square to a sign matrix you must not follow standard matrix multiplication. Instead you have to multiply each value of your Latin square to each corresponding value of your sign matrix.

For example:

```
x y z w        and        - + + +        gives        -x  y  z  w
z w x y                   + + + +                      z  w  x  y
w x y z                   - + + +                     -w  x  y  z
y z w x                   + - + -                      y -z  w -x
```

5/ Iterative functions

The iterative functions commonly used to create Julia sets are polynomials of order 2 and

above.

From now on we'll be dealing with standard inter-matrix operations. Addition and subtraction are simple: add or subtract each value of the first matrix to each corresponding value of the second matrix. Squaring a matrix or multiplying it to another matrix is the key thing to understand here:

-to square a matrix (or to multiply it by itself) you have to multiply all its values by its first column.

-to multiply a matrix M1 by another matrix M2 you need to define which matrix is first and which matrix is second because M1.M2 does not necessarily equal M2.M1.
M1.M2 is the first matrix multiplied by the first column of the second matrix.
M2.M1 is the second matrix multiplied by the first column of the first matrix.
This means that depending on which matrix is placed first the product of two matrices involves a full matrix and 1/n (with n being the order of the matrix) values of another matrix.

For example, if you want to apply the classic formula f(M)=M^2+c to the matrix below you need to follow the following procedure to square your original M1 matrix:

x -y z w
y x w z
z -w x -y
w -z -y x

multiplied with the first column of the matrix:

x
y
z
w

This gives :

x*x -y*y z*z w*w
y*x x*y w*z z*w
z*x -w*y x*z -y*w
w*x -z*y -y*z x*w

Your set of 4 discreet formulas is then (each row is a discrete function):

f(x) = x*x - y*y + z*z + w*w = x^2 - y^2 + z^2 +w^2
f(y) = y*x + x*y + w*z + z*w = 2xy + 2zw
f(z) = z*x - w*y + x*z - y*w = 2xz - 2yw
f(w) = w*x - z*y - y*z + x*w = 2xw - 2yz

If you want to multiply this matrix to the matrix M2 below,

```
-x  y  z  w
 w  z  y  x
-z  w  x  y
 y -x  w  z
```

Then M1.M2 =

```
x -y  z  w      *      -x      =      -x*x  w*(-y)  -z*( z)  y*( w)
y  x  w  z              w      =      -x*y  w*( x)  -z*( w)  y*( z)
z -w  x -y             -z      =      -x*z  w*(-w)  -z*( x)  y*(-y)
w -z -y  x              y      =      -x*w  w*(-z)  -z*(-y)  y*( x)
```

and M2.M1 =

```
-x  y  z  w      *      x      =      x*(-x)  y*( y)  z*z  w*w
 w  z  y  x             y      =      x*( w)  y*( z)  z*y  w*x
-z  w  x  y             z      =      x*(-z)  y*( w)  z*x  w*y
 y -x  w  z             w      =      x*( y)  y*(-x)  z*w  w*z
```

To add even more flexibility you can choose not to restrict yourself to the first column and use the second, third or fourth column of your 4x4 real matrix when performing inter-matrix multiplication. This can lead to very interesting Julia sets which you would not be able to see if you only used the first column.

To divide together arbitrary matrices of this group we need to define 1/M first. For transcendental functions (functions using trigonometry) we also need additional rules. This will require additional work.

Once we have a squared matrix f(M) we can iterate the function several times like this: f(f(f(f(f(f(f(M)))))))) for 7 iterations for instance. According to a pre-defined bailout value (4 is usually a good choice) we can then find which points belong to the set or not depending on their convergence, divergence or cyclic behaviour.

In terms of coordinates, we are iterating 4 discrete functions (f(x), f(y), f(z) and f(w), the four rows) of our squared matrix.

6/ Conclusion

This approach allows generalization to n-dimensional matrices. For each level of dimensions the same procedure can be followed: defining various groups of Latin squares of order n (among which are isomorphic and homeomorphic groups/classes, normalised groups, etc.), defining all sign rules or matrices of order n, combining both matrices to define an algebra and then applying an iterative function to visualize them. The bigger matrices become the more complex their structure can potentially be and they can therefore reveal some even more

interesting Julia sets in hyperspace. As computers get faster and faster it will be possible to slowly increase the number of dimensions at the same time so that bigger and bigger Julia sets are defined but the number of Latin squares for each number of dimensions increases exponentially and quickly reaches billions of billions.

Send your comments, questions or suggestions to:
ragnarok@hypercomplex.org

You can also visit www.hypercomplex.org for tutorials, software and galleries of rendered 4D Julia sets.

Links to 4D Julia sets tutorials:

http://skal.planet-d.net/quat/How.ang.html
http://home.hia.no/~fgill/quatern.html
http://astronomy.swin.edu.au/pbourke/fractals/quaternionjulia

16.4 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 MiSZle and various types of reference information.

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

16.5 Hot Keys

Hot keys

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

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

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

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

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

Shift-O -- Opens the Pilot window to adjust key 3D matrix parameters and redraw the figure. The current image is reduced to one quarter normal for faster redraw. Each click on a Pilot button increments or decrements a parameter.

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

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

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

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

Shift-U -- like "J" and "G" except that you choose the target type and beginning parameters. This is useful for exploring 3D Mandelbrot sets, insofar as you can scan the complex c planes for interesting variations. Also works for other multi-dimensional Mandelbrot formulas that are dependent on the c planes. Hint: turn off the Image/Auto-Redraw option after drawing the mapping picture. You can then set the target set (Mandelbrot, 3D matrix, etc.) before using this key sequence, without erasing the mapping picture.

Shift-D -- use the mouse to examine pixel depth in a drawing. By clicking with the left-mouse button on any area in the current fractal, the pixel's depth and bailout values are displayed in the status bar, along with the pixel's coordinates. Useful to locate Mandelbrot islands while zooming on an image. Click on the title bar or press Esc to exit this command.

Shift-F -- generate a Julia set from a formula's MandelbrotP space. Random points in a formula's current Mandelbrot space are scanned for an interesting Julia set (enabled on when a Mandelbrot type is selected from the Type menu).

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

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

16.6 Parser

Parser Information

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

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

sine z --- sin(z) or SIN(Z) ; where Z can be any complex expression
hyperbolic sine z --- sinh(z) or SINH(Z)
cosine z --- cos(z) or COS(Z)
hyperbolic cosine z --- cosh(z) or COSH(Z)
tangent z --- tan(z) or TAN(Z)
hyperbolic tangent z --- tanh(z) or TANH(Z)
cotangent z --- cotan(z) or COTAN(Z)
arctangent z -- atan(z) or ATAN(Z)
arcsine z -- asin(z) or ASIN(Z)
arccosine z -- acos(z) or ACOS(Z)
hyperbolic arctangent z -- atanh(z) or ATANH(Z)
hyperbolic arcsine z -- asinh(z) or ASINH(Z)
hyperbolic arccosine z -- acosh(z) or ACOSH(Z) -- implemented as asinh(z)/atanh(z)
vers z -- vers(z) or VERS(z) -- 1- cos(z)
covers z -- covers(z) or COVERS(z) -- 1- sin(z)
exp(z) or EXP(z) -- the exponential function
log(z) or LOG(Z) -- the logarithmic function -- implemented as ln(z+1)
Bessel -- bessel(z) or BESSEL(Z) -- first order Bessel function
Laguerre -- lag(z) or LAG(Z) -- 3rd order Laguerre function
Gaussian -- gauss(z) or GAUSS(Z) -- Gaussian probability function
absolute value of z --- abs(z) or ABS(Z) -- sqrt(x*x + y*y +z*z+w*w)
z squared --- sqr(z) or SQR(Z)
z cubed -- cube(z) or CUBE(Z)

square root of z --- sqrt(z) or SQRT(Z) -- implemented as sqrt(abs(z))
real part of z --- real(z) or REAL(Z)
imaginary part of z --- imag(z) or IMAG(Z)
modulus of z --- mod(z) or MOD(Z) or |z| -- (x*x + y*y +z*z+w*w)
conjugate of z -- conj(z) or CONJ(z) -- (x-y-z-w)
reciprocal of z -- rep(z) or REP(Z) -- inverts the matrix and returns the column one vector as a result
transpose -- trans(z) or TRANS(Z) -- transposes the matrix and returns its column one vector as a result
trace -- trace(z) or TRACE(Z) -- sums the diagonal elements of the matrix
rotate columns -- rotc(z) or ROTC(Z) -- rotate matrix columns in clockwise direction
rotate rows -- rotr(z) or ROTR(Z) -- rotate matrix rows in clockwise direction
reverse rotate columns -- rotc(z) or ROTC(Z) -- rotate matrix columns in counter-clockwise direction
reverse rotate rows -- rotr(z) or ROTR(Z) -- rotate matrix rows in counter-clockwise direction
swap columns -- swapc(z) or SWAPC(Z) -- swap outer and inner matrix columns
swap rows -- swapr(z) or SWAPR(Z) -- swap outer and inner matrix rows
swap adj columns -- saltc(z) or SALTC(Z) -- swap adjacent matrix columns
swap adj rows -- saltr(z) or SALTR(Z) -- swap adjacent matrix rows

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)

Notes:
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.

All transcendental and trig functions are implemented as power series. In the case of the log and inverse trig functions, since their domain is limited to -1<z<1 there is no guarantee that they will work correctly in all formulas. The results for all trig and transcendental functions are truncated to their column one vectors.

Math operators

+ --- addition
- --- subtraction
* --- multiplication
. -- matrix multiplication
/ --- division
% -- matrix 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)

Notes:
The multiplication operator '*' multiplies a matrix variable by the column vector of the following variable. Therefore, to match the built-in formulas, you need to enter --last-- column vectors such as c# and composite terms. For example, "c*cos(z)+c' would be entered 'z=cos(z)*c#+c#'. z^4 is implemented in the built-in formulas as sqr(sqr(z)). Other ways of entering z^4 such as z*z*z*z will give different results.

The '.' (dot) operator multiplies two terms as matrices, then uses the column one vector for the result.

The '/' division operator uses the column vector (specified by the column variable) as the divisor. The result is returned in the column one vector. The '%' operator is used to perform matrix division. The dividend is inverted, then multiplied by the divisor, using matrix multiplication, then the column one vector is used for the result.

The power operator works with whole number exponents only. If you use a quad variable as an exponent, the absolute integral value of the variable is used.

Constants and variables

complex constant --- c# or C#, read/write.
complex conjugate --- cc# or CC#, read-only.
cr -- the constant entered in the cr box in the Parameters window(use j# for parser)
ci -- the constant entered in the ci box in the Parameters window(use k# for parser)
e --- e or E -- 1e^1 -- 2.71828, read/write.
i --- i or I -- square root of -1,read/write.
iteration --- iter# -- iteration loop counter
j --- j# or J# -- real part of the complex constant, read-only.
k --- k# or K# -- coefficient of the imaginary part of the complex constant, read-only.
Note: j and k are the actual values of the complex constant terms as they are used in the iteration process, so will vary when the Mandelbrot option is used.
m --- m# or M# or pixel --a complex variable mapped to the pixel location as defined by the z coordinates entered in the Parameters window, read/write.
maxit -- the maximum number of iterations, as set in the Parameters window, read only
p --- p# or P# -- real constant used in phoenix maps; uses the real part of the complex constant when the Phoenix option is chosen, read-only.
p1 – the complex constant entered in the cr and ci gadgets, read-only.
pi --- pi or PI -- 3.14159, read/write.
q --- q# or Q# -- real constant used in phoenix maps; uses the imaginary part of the complex

constant when the Phoenix option is chosen, read-only
x --- x# or X# -- real part of Z, read/write.
y --- y# or Y# -- coefficient of the imaginary part of Z, read/write.
z --- z or Z -- function value at any stage of the iteration process, read/write.
zn# or ZN# -- the value of z at the previous stage of iteration, read-only.

16.7 Built-in Formulas

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

1	p0; z^2+c
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; cz^3+c
10	p9; z^2-cz^3+c
11	r0; z^3-z+c
12	r1; z^2+z^3+c
13	r2; z^3-zn+c
14	r3; z^4-z^2+c
15	r4; (z^2-c)(z+1)
16	r5; (z+j)(z+k)(z^2+1)+c
17	r6; (z+j)(z^2+z+k)+c
18	r7; (z-1)(z^2+z+c)
19	r8; z^5+c
20	r9; z^6+c
21	e0; z(z^5+c)
22	e1; z(z^6+c)
23	e2; z^7+c
24	e3; z(z^7+c)
25	e4; z^7-z^5+z^3-z+c
26	e5; z^3+j+kzn
27	e6; cz^2+zn+c
28	e7; z^4-c^4
29	e8; (z-c)(z+1)(z-1)+c
30	e9; z^2+c^2
31	s0; z^3+c/z+c
32	s1; 1/z^2+c
33	"2; z+z^3/c+c
34	s3; z^3/c+c
35	s4; 1/(z*z/c)+c
36	s5; 1/(z*z)-z+c

37 s6; (1+c)/(z*z)-z
38 s7; 1/(z*z*z/c)+c
39 s8; 1/(z*z*z)-z+c
40 s9; 1/z^2-cz-c
41 "t0; 1/z^2+exp(z)-c"
42 "t1; c*cos(z)+c"
43 "t2; z^3+sin(c)"
44 "t3; 1/z^3+sinh(z)-c"
45 "t4; z^3+tanz+c"
46 "t5; z^3+coshz+c"
47 "t6; z^2-bessel(z)+c"
48 "t7; lag(z)+c --- 3rd order Laguerre function
49 "t8; z^3+Gauss(z)+c -- Gaussian probability function
50 "t9; z^2+arcsin(z)*z-c

16.8 Bibliography

Bibliography

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

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

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

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

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

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

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

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

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

New York, 1988.

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

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

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

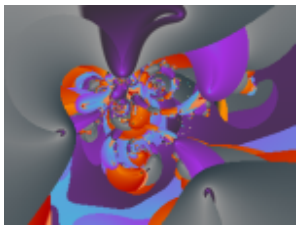
Wegner, Tim and Tyler, Bert: "Fractal Creations", Second Edition, Waite Group Press, Corte Madera, CA, 1993.

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

16.9 About MiSZle

About MiSZle

>>>>> MiSZle™ v3.052 ©2001 - 2004 by Terry W. Gintz



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

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

MiSZle uses the same built-in formulas and matrix math as DynaMaSZ. Therefore 3D matrix images can be exported to DynaMaSZ for additional processing by changing the data file extension from msz to dmz.

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

Memory requirements for MiSZle vary with the size of the drawing area MiSZle opens on, ranging from approximately 3 megabytes memory for a 640X480 area to 48 megabytes for a

2048X1536 area. Special routines have been added to reduce memory requirements for large bitmaps (up to 14400X10800) by writing these directly to a file instead of using a memory bitmap. For poster size pictures (up to 96000X72000), a segmenting option is available to divide pictures into up to 225 pieces. The segments can be tiled into one large bitmap using a program such as Richard Paasen's Image Arithmetic.

Acknowledgements: many thanks to Paul Carlson for providing me his algorithms for 3D-like fractals, and allowing me to incorporate his ideas into my programs. Also, special thanks to Ron Barnett for his help in setting up the animation routines, to Earl Hinrichs for sharing his unique programming methods on the fractal art and programmer's lists, to Frode Gill for his quaternion and ray-tracing algorithms, to Dirk Meyer for his Phong-shading algorithm, to Piet van Oostrum for his T2MF, without which the midi support would still be a far off dream, and to David Makin for sharing his ideas on quaternion colorings and 3D insights. Thanks also to Francois Guibert for his Perlin Noise example. The multi-windowing interface in MiSZle is courtesy of that extraordinary and prolific fractal programmer, Steven C. Ferguson, whose filters I have also included in MiSZle. Steve's contributions to the look and feel of previous versions of my programs have had a deep impact on my fractal imaging experiments.

For a short history of this program, see Chronology.

16.9.1 Chronology

Chronology

History of this program:

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

May 1990 -- ZPlot 1.3d -- added 3D projections for all formulas in the form of height fields.

May 1991 -- ZPlot 2.0 -- first 236-color version of ZPlot for Windows 3.0.

May 1995 -- ZPlot 3.1 -- ZPlot for Windows 3.1 -- 60 built-in formulas. Added hypercomplex support for most built-in formulas.

May 1997 -- ZPlot 24.02 -- first true color version of ZPlot -- 91 built-in formulas. Included support for 3D quaternion plots, Fractint par/frm files, Steve Ferguson's filters, anti-aliasing and Paul Carlson's orbit-trap routines.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

November 2000 -- Added octonion fractals to QuaSZ 1.01.

March 2001 -- Cubics 1.0 -- my first totally 3D fractal generator.

May 2001 -- QuaSZ 1.03 -- added Perlin noise and improved texture mapping so texture tracks with animations.

June 2001 -- Fractal Zplot 1.23 -- added Perlin noise and quat-trap method.

July 2001 -- QuaSZ 1.05 -- improved performance by converting many often-used dialogs to non-modal type.

October 2001 -- FraSZle 1.0, QuaSZ formula and algebra compatible version of Fractal Zplot

November 2001 -- DynaMaSZ 1.0, the world's first Dynamic Matrix Systems fractal generator

January 2002 -- MiSZle 1.1 -- generalized fractal generator with matrix algebra extensions

May 2002 -- DynaMaSZ SE 1.04 (unreleased version)-- scientific edition of DMZ, includes support for user-variable matrix dimensions (3X3 to 12X12)

January 2003 -- Pod ME 1.0 -- first stand-alone 3-D loxodromic generator, Hydra 1.0 -- first 3-D generator with user-defined quad types and Fractal Projector a Fractal ViZion-like version of DMZ SE limited to 3X3 matrices

May 2003 -- QuaSZ 3.052 -- added genetic-style function type and increased built-in formulas to 180. Other additions since July 2001: generalized coloring, support for external coloring and formula libraries, and Thomas Kroner's loxodromic functions.

May 2003 -- FraSZle and Fractal Zplot 3.052 -- added random 3D orbital fractals, new 3D export methods, upgraded most frequently-used dialogs to non-modal type and added genetic-style function type. FZ now based on FraSZle except for built-in formula list and Newton support.

Index

- 3 -

- A -

- B -

- C -

- D -

- E -

- F -

- H -

- I -

- M -

- P -

- R -

- S -

- T -

- V -