Evolvotron User Manual

Evolvotron is interactive "generative art" software to evolve images/textures/patterns through an iterative process of random mutation and user-selection driven evolution.

On starting the application, a grid of images is displayed. Resize or maximise the application if you like, but the more pixels have to be calculated, the slower it will be. (For the default 2D image mode, you will need a fast machine or patience. For the optional animation mode, you will need both.)

Simply repeat the following until bored:

- Click (singleclick) on an image you like to spawn the next generation of its mutant offspring.
- Wait until variations on it are regenerated in sufficient detail that you can decide which one you like best again.

IMPORTANT: Initially you should select images with some sort of variation. If you select a uniform image, you may get stuck in a degenerate zone with little to mutate and therefore little chance of escape to more interesting images. You can always reset/restart from the "File" menu (the difference is that "reset" also resets the mutation parameters to their default values). Selecting one of the "warp" options from a context menu (right-click on an image) can also help by introducing an additional opportunity for mutation on subsequent spawns.

Note that various spirals, grids and tiles, although complex looking, are actually implemented by a single function node and may leave you stuck too.

Command Line Options

The following are equivalent:

- evolvotron --grid 12x8
- evolvotron --grid=12x8
- evolvotron -g 12x8

General Options

- -a, --autocool
  Enable autocooling by default, and cause resets of mutation parameters to re-enable autocooling if it was disabled.

- -F, --fullscreen
  Start in "fullscreen" mode (NB for Qt on X11 this means a screen-filling borderless/undecorated window is used; it's not simply maximising the window, and it's not the sort of framebuffer hijacking used by SDL games). The Qt documentation claims some window managers may not be entirely cooperative with this (in which case sorry, you're on your own). evolvotron actions which bring up dialog boxes (e.g save) seem to generally behave fairly sensibly but child windows (e.g enlargements or dialogs) can show some "interesting" behaviour. Fullscreen mode can be toggled within the application using "F" key. The "Esc" key will also exit it.

- -g, --grid *cols*x*rows*
  Sets size of the grid of image display cells in the main application area (defaults to 5x6)

- -h, --help
  Print summary of command line options and exit.

- -j, --jitter
  Enable sample jittering. Samples will be made at a random position within a pixel instead of on a regular grid, providing some antialiasing.

- -m, --multisample *multisample grid*
  Enables additional antialiasing passes. Specifying 2 or 3 will provide an additional pass with 2x2 or 3x3 samples per pixel. Specifying 4 (of higher) will provide a 2x2 and a final 4x4 pass. Specifying 1 provides the default behaviour of one sample per pixel. For best rendering quality also specify -j.

- -M, --menuhide
  Start with menu and status bar hidden. Nice with --fullscreen. Hiding can be toggled within the application using ctrl-m. The Esc key will also bring them back.

- -p, --spheremap
  Images are produced by sampling the underlying 3D function on the latitude-longitude grid of a sphere. The resulting images should be usable as spheremaps/spherical environment maps. Animations vary the radius of the sphere. NB when in spheremap mode, middle mouse button adjustments do not yet behave like you'd expect.

- -S, --startup *function_filename*
  Specify a function filename (evolvotron's XML format) to load on startup (or reset). The option can be provided multiple times, and this is also the interpretation of any positional arguments. Startup functions are placed in the grid from left to right, top to bottom.

- -U, --shuffle
  Use in conjunction with -S / --startup options, to display the specified functions in random order, both on application startup and on each reset of the application.

Animation Options

- -f, --frames *frames*
  Number of frames in animations (defaults to 1 i.e no animation)

- -l, --linear
  Vary z linearly with time rather than sinusoidally. Sinusoidal variation generally looks better when animations are "bounced" forwards and backwards, but this option is useful when generating slices to use as volumetric textures.

- -s, --fps *framerate*
  Rate at which frames are displayed per second (integer). (Defaults to 8).

Power-user & Debug Options

Note that the usual Qt/X11 options (for example, -geometry *width*x*height* option to set on-screen size in pixels) are processed and removed before evolvotron options are checked.

- -D, --debug
  Puts the certain aspects of the app into a more debug oriented mode. Currently (ie this may change) it simply changes function weightings so virtually all function nodes are FunctionNoiseOneChannel. By itself this is a pretty pointless thing to do, but in conjunction with the -X options it's useful for examining the behaviour of specific functions.

- -E, --enlargement-threadpool
  Use a separate thread pool for computing enlargements. Using this option ensures computation of enlargements continue to make some progress even while the main grid is being actively worked on. However, this will be at the expense of main grid rendering performance. Without this option, enlargements' final high-resolution renderings are invariably lower priority than computation for images in the main grid. See also the -N option to control the priority of threads in this pool.

- -n, --nice *niceness*
  Sets additional niceness (relative to the main application thread) of the compute (rendering) thread(s). It's useful to run compute threads at a slightly lower priority ("nice 4" is the default without this option) than the main (GUI) part of the program (but note that this means other normal/lowish priority tasks running on your machine may slow evolvotron down a bit more than expected).

- -N, --Nice *enlargement niceness*
  Sets additional niceness (relative to the main application thread) of the compute thread(s) computing enlargements (default value is 8). Only effective in conjunction with -E option.

- -t, --threads *threads*
  Sets number of compute threads. If this is not specified, then as many compute threads are created as there are processors on the system (unless this cannot be discovered in which case only a single compute thread is created). Non-linux builds will likely not include code to determine processor count (suitable patches gratefully received).

- -u, --unwrapped
  Modifies -F behaviour so that the specified "favourite" function is NOT wrapped by space/colour transforms. NB For functions without leaf nodes or parameters (e.g FunctionSphericalToCartesian) this doesn't leave any scope for variation or future mutation. Function name recognition is case sensitive. Example: `evolvotron -F FunctionKaleidoscope -u`

- -v, --verbose
  Verbose mode, writes various things to application stderr. This is primarily intended to assist debugging. This option used to be useful for getting a list of supported function names for use with the -F option, but those can also be inspected via the Settings dialogs.

- -x, --favourite *functionname*
  Force a specific "favourite" function type to be used at the top level of all function trees. The specified function is still wrapped by spatial and colour warping functions which may disguise it considerably. A list of all the function names understood by evolvotron is output during app startup when the -v option is specified. Function name recognition is case sensitive. Example: `evolvotron -F FunctionSpiralLinear`

Mouse Control

Left-click

A left-click on an image in the main window spawns the mutant offspring of that image to all the other (non-locked) displays in the grid.

Right-click Context Menu

Right clicking on an image gets you a few more options:

- "Respawn" regenerates just the current image from whatever it was spawned from (and using recolour or warp, if that's what was used to produce it). The main use of this is to make your grid of images look nice for screendumps, by regenerating any which aren't up to scratch. NB May not work as expected after an "undo".
- "Spawn" is the same as clicking an image. It generates mutated images to all unlocked images in the grid.
- "Recolour" to produce different coloured variants of the selected image
- "Warp"'s sub-options produce variants of the image which have been zoomed/rotated/panned.
- "Lock" to prevent an image from being overwritten by spawns from other images (select again to toggle).
- "Enlarge" to produce a blow-up of the image in a single window. Submenu items select either a freely resizable window or a scrollable view of a fixed size image. If the application is running in fullscreen mode (NB this is NOT the same as a simply "maximised" window) then the enlarged image will also be fullscreen (the "Resizeable" mode is probably what you want in this case as the image will automatically be rendered at the correct resolution).
- "Save image" to save the image in a file (.ppm or .png). You generally want to save an enlarged image: if you save a small image from the grid, the size you see on the screen is the size you get in the file. Save isn't allowed until the full resolution image has been generated; if you try to save too early a dialog box will be displayed telling you to try again later.
- "Save function" to store the function to an XML file.
- "Load function" to load a stored function from an XML file. NB if the file was saved from a different version numbered evolvotron, a warning message will be generated. Save/load of functions is an experimental feature and you should not count on future versions of evolvotron being able to load files saved from old versions, or producing the same image from a loaded function. Attempting to load functions from later versions into earlier versions is even less likely to succeed.
- "Simplify" prunes the function tree of redundant branches where possible (the same action can be applied to all images from the main "Edit" menu). This doesn't change the appearance of the image, but may make it recompute faster.

- "Properties" brings up a dialog box containing some information about the image (e.g the number of function nodes it contains).

Middle Mouse Button

[NB This feature will probably only be of practical use to those with high-end machines].

You can use the middle mouse button to drag-adjust individual images. This is useful for "final composition" type tweaks, e.g centering an image's most interesting feature, or just for satisfying your curiosity about what's off the edge of the image.

It also works on enlarged images, although it's virtually unusable without a bit of practice on smaller, faster ones (just boldly make the adjustment you want, release the button... and wait).

Changes made can be rolled-back on the main Edit/Undo menu item, one drag-action at a time.

An unmodified middle-mouse drag pans the image around following the mouse motion.

A SHIFT-middle drag zooms the image in and out with scaling proportional to the distance from the centre of the image. Beware of generating huge zooms by clicking too near the centre of the image.

An ALT-SHIFT-middle drag is similar but anisotropic: the scaling may be different in X and Y. Warning: this technique is very sensitive and can be quite tricky to use! In particular, if you initially click near the centre axes of the image the zoom factor can be HUGE, so the best way to start using this is to click about halfway on a diagonal between the image centre and a corner and gently move in and out radially. Dragging from one side of the image to the other flips it over (the degenerate case of infinite zoom at the centre is handled cleanly I think). If it all goes horribly wrong, undo and try again.

A CTRL-middle drag rotates the image about its centre.

A CTRL-ALT-middle drag shears the image (the best way to see what this does is to click in the corner of an image and move the mouse horizontally or vertically).

Keyboard Control

There are some keyboard shortcuts.

Main Window

- "r"/"t"/"x" perform various starts of reset/restart.

- "q" quits the application.

- "u" (and also Ctrl-z) does an undo.

- "f": full-screen mode (on X11, Qt does this by asking the window manager for a screen-filling undecorated window, and the documentation contains some dire warnings about problems with broken window managers). See also "-F" command line option.

Fullscreen mode propagates to enlarged image display windows. NB The application may completely disappear from the screen for a brief interval while switching mode.

- "m" : hides status and menu-bar hiding, which can be nice when in full-screen or window-maximised mode. See also "-M" command line option. Also note that while the menu bar is hidden, most of these keyboard shortcuts won't function as they're tied to the menu system.

- Esc : exits full-screen and/or menu-hiding mode, putting the application into its normal default state.

Enlargement Windows

The image display windows created by selecting "Enlarge" from a context menu also have a couple of keyboard operations:

- "f" : [NB only available with fullscreen build option] toggles full-screen mode. When returning to normal mode, if the main app window was fullscreen then it will also drop back to normal mode.

- Esc : [NB only available with fullscreen build option] completely closes a fullscreen-mode enlargement window.

Gui Elements

Main Menu Bar

- File menu: Items to restart, reset and quit the application. The difference between restart and reset is that reset sets the mutation parameters back the their default values. The "randomize function weights" version of restart scrambles the relative probability of the various function types (if you think evolvotron just keeps generating the same kinds of images give it a try). The "restart with specifc function" item duplicates the functionality of the "-x" and "-X" command-line options.
- Edit menu: "Undo" lets you undo certain actions: e.g spawn, middle-button adjustment, simplification and lock/unlock. There is a large but limited number of levels of undo. "Simplify" is of curiosity value only: it prunes redundant branches from images ("junk DNA"); this may help them recompute faster but at the cost of there being less mutatable material.
- Settings menu: "Mutations" brings up a dialog to modify the amount of change spawned images are subject to. (See "advanced usage" below.) "Functions" brings up a dialog to modify the relative probability of functions being used. By default all functions are equally likely except for iterative functions and fractals, which are almost but not completely supressed. But if you think there are too many spirals or grids (or not enough fractals) then this is the place to adjust the frequency with which they appear. If the software was built with the fullscreen option, that can also be controlled from this menu. "Favourite" brings up a dialog which allows you to select a specific function type to always be used as the root node of any new functions. The function can be wrapped by some other random stuff, or unwrapped. See also the -X and -x command line options.
- Help menu: Items to bring up documentation, and the usual "About" box (which includes the license).

Status Bar

An area on the status bar shows how many compute "tasks" are outstanding (or "Ready" when there are none). When two task totals are reported, the first is for the main grid and the second for any enlargements being computed. Each "task" is the recomputation of an image at some resolution. Tasks are prioritised by their number of pixels (small image implies higher priority). This is why, if the main grid is still recomputing, recalculations of enlargements will appear to freeze after they have reached a certain resolution, at least until other lower resolution tasks have completed.

The status bar also provides some control over the "autocool" mechanism which reduces mutation strength with time. See the advanced usage section below.

Tips

- Don't start a session with any preconceived ideas about the kind of image you want to get out of it. You will be disappointed.
- I get the best results when I click the image which most immediately catches my eye as they start appearing. If you stop to think about it too much then things seem to go downhill.
- If you seem to be just getting the same old spirals and grids all the time, stop clicking on spirals and grids! (The same goes for random mush).
- Don't get too hung up on using the warp and middle-mouse drag adjustments every iteration... use those tools for final polishing of your masterpiece.
- You can quickly cycle through a lot of initial images (until you find one with real potential) by bashing on Ctrl-r to repeatedly restart.
- To add variety to an image's mutations, nudge it with a small middle-mouse drag. This introduces a top level transform, and therefore more parameters to be varied.
- Enlargements take a long time to complete their final high-resolution rendering pass (especially with multisampling enabled). Most convenient practice seems to be to go away and leave them to complete, then come back and save them later. Continuing to click away on the main grid effectively starves them of CPU, unless the -E command-line option is used.

Animation

As of version 0.2.0 evolvotron contains some experimental support for generation of animations (although so far the results have been pretty disappointing IMHO, but it's still early days).

NB THIS IS EVEN MORE COMPUTATIONALLY AND MEMORY INTENSIVE THAN THE STATIC IMAGE MODE.

Simply supply a -f *frames* command line option and evolvotron will generate animated sequences with the specified number of frames. These will be displayed at the frame rate specified by the optional -s *framerate* option (default 8). So "evolvotron -s 24" will generate 3 second long animations. Animations reverse direction at each end to avoid a sudden jump.

If you save an animation as PPM or PNG, multiple files will be saved with .fnnnnnn (where nnnnnn is the zero-filled frame number) inserted in each filename before the filetype qualifier.

For example, if you enter foo.ppm as the filename to save, files foo.f000000.ppm, foo.f000001.ppm... will be saved. If you have the ImageMagick tools you can convert these to an animated GIF playing at approx. 8 frames per second with:

```
convert -delay 12 foo.f??????.ppm foo.gif
```

Advanced Usage

Evolvotron's idea of an image is a function which converts XYZ co-ordinates to an RGB colour (however we can only display a 2D plane for now so the input Z is fixed to zero, or varied with time when animating).

The image functions are constructed from trees of function nodes. (In the mathematical expression 1+(2*x) the "+" and the "*" would be function nodes.) Evolvotron's functions tend to correspond to geometric or colour-space operations or anything else which can be applied to a 3D vector.

By mutating the structure of the function tree (adding random branches, for example) and the values of the constant embedded within it, the image can be changed.

The mutation parameters are under control from the dialogs accessible via the Settings menu, and the "autocool" mechanism exposed in the status bar also has some influence.

There are two kinds of mutation: perturbations to the magnitude of constants, and structural mutations which rearrage the function tree of an image. Four types of structural mutations are currently implemented:

- replacement of a function branch by a new random stub (a "Glitch" mutation).
- a random shuffle of a node's sub-nodes
- insertion of random nodes between a node and it's sub-nodes
- the substitution of a node with one of a different type, with sub-nodes unaffected where possible).

The probability (per function node) of these mutations is controlled from spinboxes on the "Mutation Parameters" dialog (expressed as chances-in-a-hundred), as is the size of perturbations to constants.

It is useful to think of the perturbations to constant parameters as being a thermal effect (hence the "heat" and "cool" buttons), while structural alterations are more drastic and are caused by high energy gamma rays or something (hence "irradiate" and "shield" buttons to adjust the probability of structual mutations).

So why would you want to change the mutation parameters from the initial defaults ? Basically, if you're getting too much variation in spawned images (this tends to happen after many generations of images, by which time the function trees have grown quite large and therefore are experiencing a lot of mutations) then cool and/or shield. If all the images look too similar, heat and/or irradiate.

The "autocool" mechanism (enabled from the statusbar or mutation parameters dialog) automatically reduces the strength of mutations from the base values with successive generations. The cooling can be cancelled by disabling autocooling or pressing the "Reheat" button to zero the number of generations counted for cooling. The effect of the cooling is a compound halving of the mutation strength after some number of generations (this number is the "half-life" controllable from the mutation parameters dialog). Note that if autocooling is enabled then eventually, after a number of iterations more than many multiples of the half-life has passes, spawned images will differ very little from their parents (hence the need for "reheat").

There is also a dialog accessible from "Functions..." on the "Settings" menu. This allows control over the relative proportions in which functions occur. There is a tab showing the relative weighting of all functions (log-2 scale: each tick halves the probability of the function occurring), and additional tabs for various classifications of function for quicker access. The "Randomize" button on each tab assigns random weightings and helps increase the uniqueness of your session.

The "Functions" dialog also has a "Dilution" tab which allows the average function-tree branching ratio to be controlled (by changing the proportion of trivial zero-branch functions added): note that using a high branching ratio results in very complex images which will take a long time to compute, while reducing the ratio results in simple, boring images.

3 types of function node are considered fundamental: constant nodes (which return a constant), identity nodes (which return their position argument) and transform nodes (which transform their positon argument by random parameters). On the "Dilution" tab of the "Functions" dialog there are two slider controls to affect things related to these:

- "proportion constant" controls the proportion of diluting fundamental nodes which are constants. Changing this from its default value of 0.5 doesn't actually seem to have much effect.
- "proportion transforms" sets the proportion of non-constant nodes diluting which are transforms (as opposed to identity nodes). The main effect of this is that images are less commonly obviously centred on the origin or aligned with the axes. I think this is a good thing, so the value is at 1.0 by default.

Other Executables

This release also builds some other command-line driven (non-GUI, non-interactive) utilities. Consult the man pages for full details.

evolvotron_render reads a XML function description from its standard input and renders it to the file specified.

evolvotron_mutate reads an XML function description from its standard input and outputs a mutated version. A command line option allows the "genesis" situation of creating a random function description with no input.

Examples

Evolving and mutating on the command line:

```
evolvotron_mutate -g | tee fn.xml | evolvotron_render /tmp/xxx.ppm ;
display /tmp/xxx.ppm
```

```
cat fn.xml | evolvotron_mutate | evolvotron_render -j -m 4
/tmp/xxx.ppm ; display /tmp/xxx.ppm
```

Animating a function ani.xml saved from evolvotron in animation mode:

```
cat ani.xml | evolvotron_render -f 100 -v -s 256 256 ani.ppm ; animate
```

```
ani.f??????.ppm
```

Future Developments

Please check the TODO file first before you send me suggestions!

Please don't ask me to port evolvotron to proprietary platforms. You are of course Free under the terms of the GPL to do so yourself, but please read http://www.fefe.de/nowindows/ first.

Thanks

To those who have contributed feedback, suggestions and patches:

- Dmitry Kirsanov
- Jonathan Melhuish
- Karl Robillard
- Linc Davis
- Paolo Greppi
- Marcin Wojtczuk
- Michael Sterrett
- Massimiliano Guastafierro
- Goetz Waschk
- Forrest Walter

And to the anonymous Linspire reviewer who perhaps came up with the best summary of evolvotron yet: "Fascinating. Utterly pointless, but fascinating."

The friezegroups wouldn't have been possible without http://michaelshepperd.tripod.com/resources/groups.html

Thanks to www.di.fm, www.somafm.com and Trance4Ever for music to code to.

Thanks especially to a SIGGRAPH conference panel many years ago (likely including Karl Sims, the pioneer in this area) who first got me interested in this stuff.

Why ?

I have always admired those who have the skill to wield a pen or paintbrush and fill a sheet of paper or a canvas with some striking image from their imagination. Unfortunately I lack the patience to learn such skills, and probably the necessary manual dexterity and imagination too.

Evolvotron, and several predecessors developed on and off over a decade since I first came across the idea, are an attempt to compensate for this using the skills I do have i.e some mathematical sensibility and the ability to write working code (well, sometimes). If you like an image it produces, then as far as I'm concerned that's as satisfying a result as if you liked something I'd drawn myself.

Tim Day