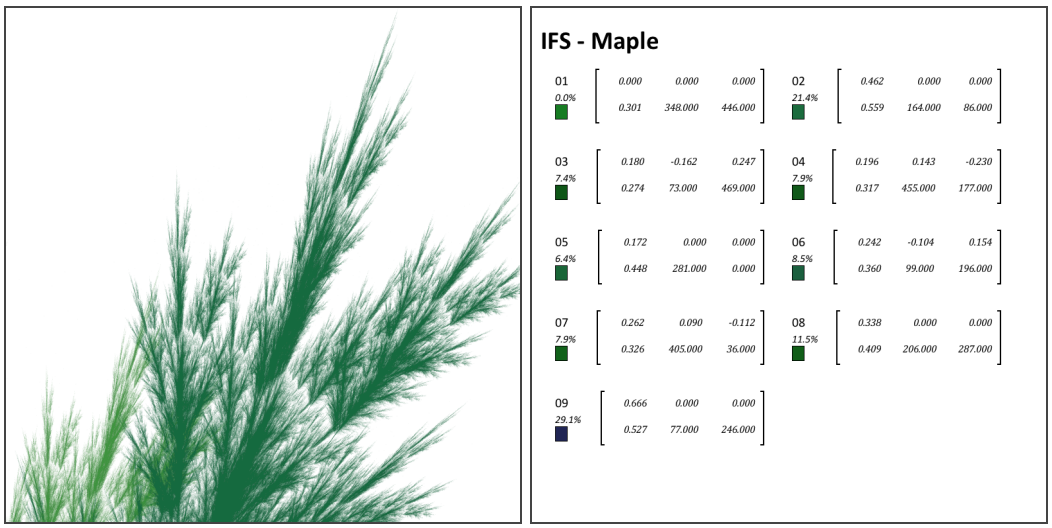
IFS Explorer 1.0.5

Iterated Function System Explorer

An interactive **Iterated Function System** explorer. This program allows you to explore a vast world of complex and beautiful fractal images produced by simple mathematical rules, through a process of trial and error, experimentation and exploration. See the *References* section below for more details on the mathematics and concepts behind these systems.

IFS Explorer provides an interactive UI to create and manipulate a set of affine transforms, which are then iterated randomly to produce an image. The Java `AffineTransform` class is used to represent and plot the transforms. The systems can be saved and loaded as XML files, and rendered images can be exported as PNG graphics files.



*Screenshots for the **Viewer** and **Details** modes.*

Program requirements

- Java 1.7.0 Runtime Environment
- Windows, Linux or OSX Operating System
- Maven and 1.7.0 JDK for building *(Optional)*

Instructions

Either build the program using Maven or extract one of the packaged distributions. These can be downloaded from GitHub as either `.tar.gz` or `.zip` archives. Then run the relevant script for your operating system, using the optional flags `-f` or `--fullscreen` to

specify full-screen mode.

Enable colour rendering with the `-c` or `--colour` flag. This will use a fixed list of colours for the transforms. To use a colour palette taken from a sample image specify `-p` or `--palette`. The image used can be specified in the `explorer.palette` system property and can be *abstract*, `wave` or `car`.

```
$ ./bin/explorer.sh [-f] [-c] [-p] [ifs.xml]
$ ./bin/explorer.command [-f] [-c] [-p] [ifs.xml]
C> .\bin\explorer.cmd [-f] [-c] [-p] [ifs.xml]
```

Configuration

- `explorer.grid.min` - Minimum grid spacing - *10*
- `explorer.grid.max` - Maximum grid spacing - *50*
- `explorer.grid.snap` - Snap-to-grid spacing- *5*
- `explorer.palette.file` - Colour palette image source - *abstract*
- `explorer.palette.size` - Number of colours in palette - *64*
- `explorer.palette.seed` - Random seed used for choosing colours - *0*
- `explorer.window.width` - Width of main window - *600*
- `explorer.window.height` - Height of main window - *600*
- `explorer.debug` - Enable debugging mode - *false*

The configuration properties listed should be added to the `JAVA_OPTS` variable bwfore running the program, as ilustrated below.

```
$ export JAVA_OPTS="-Dexplorer.palette.file=wave \
> -Dexplorer.palette.seed=1234"
$ ./bin/explorer.sh --colour --palette ./data/spikes.xml
```

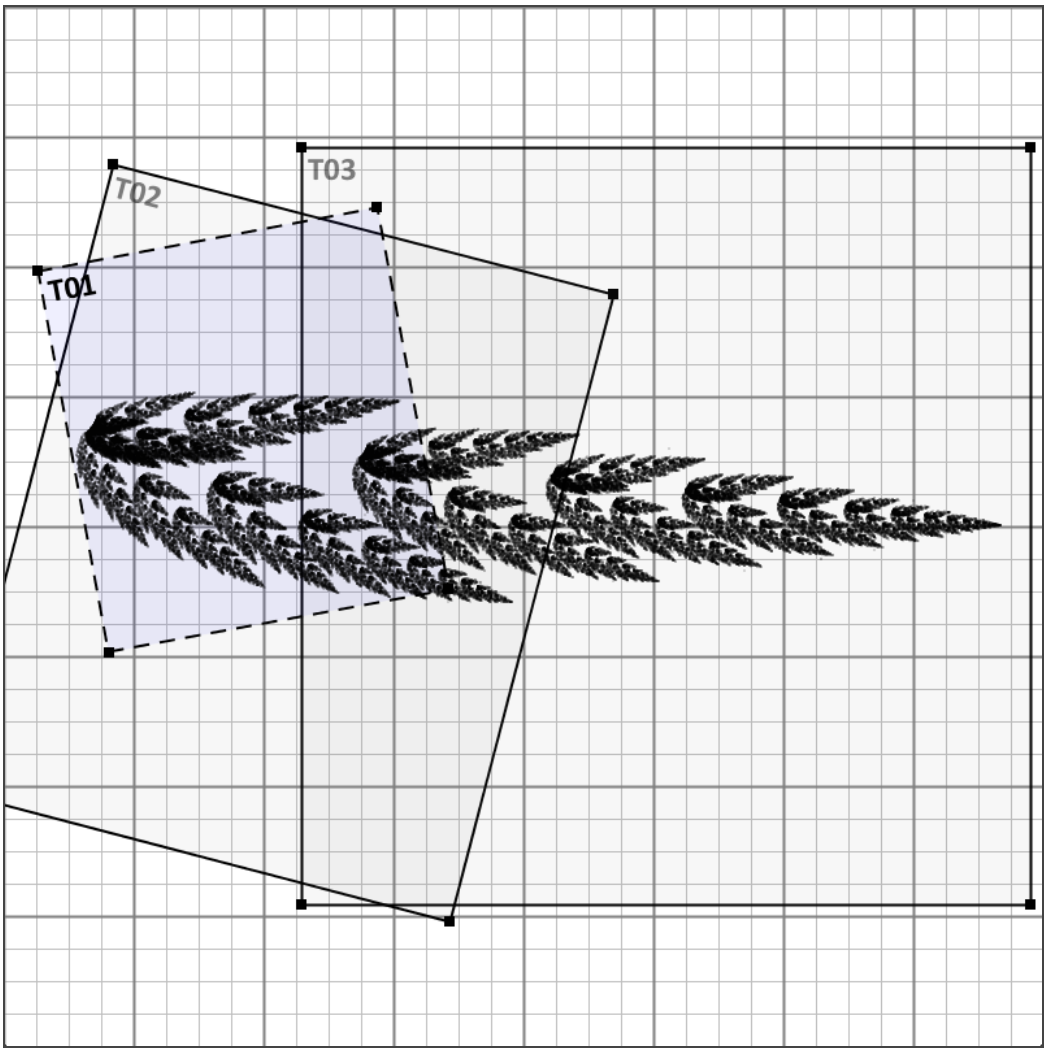
Usage

A series of example XML data files for IFS transforms are provided in the `data` directory of the distribution. Some of these can be found in the *The Computational Beauty of Nature* and *Superfractals* books, details of which are available in the *References* section. To load these examples, specify the path to the file as the last argument on the command-line. Sample images for these transforms have also been provided, so be sure to use the `.xml` file, not the `.png`.

Once the program starts, it will diplay an empty grid in the **Editor** mode. The menus provide access to standard operations, including the ability to switch modes. The keyboard can also be used to toggle between modes using the *Tab* key.

A live view of the generated IFS is displayed at low resolution to give an idea of the

eventual rendered image of the fractal. This capability is unique among IFS generating applications, and allows much more creativity and hels reach the desired final image much more quickly.



*Screenshot of the **Editor** mode with the live IFS view*

In the **Editor** mode, clicking and dragging the mouse will create a new transform which can then be moved, resised and rotated (with ***Shift*** held down) as desired. In this mode, the *Delete* or *Backspace* key will delete the selected transform and the ***Left*** or ***Right*** arrow keys will rotate it by ninety degrees.

When in **Viewer** mode the *Space* key will pause and continue the rendering process. To zoom into the fractal select a specific area using the left mouse button, or use the *z* and *Z* keys to to zoom in and out of the centre of the image by a factor of two. Note that when zoomed in at high magnification the iterative process can take a very long time until the rendered fractal becomes visible.

The **Details** mode shows the actual affine transforms and their matrix coefficients for the system as a scrollable list.

TODO

- Properties editor for transform
  - Matrix coefficients *or* rotation/displacement/scale values
- Printing
- Preferences dialog
- Better operating system integration
  - Native full-screen mode
  - Support native windowing system features
- Add skew transform
- Better UI for rotations
- Improve documentation
- JNLP Web Start mechanism
- JavaFX UI

References

1. **Iterated Function System**; *http://en.wikipedia.org/wiki/Iterated_function_system*; *Wikipedia*
2. **Affine Transform**; *http://en.wikipedia.org/wiki/Affine_transformation*; *Wikipedia*
3. **Construction of fractal objects with iterated function systems**; *http://www-users.cs.umn.edu/%7Ebstanton/pdf/p271-demko.pdf*; *Demko, Stephen and Hodges, Laurie and Naylor, Bruce; SIGGRAPH Computer Graphics, Volume 19, Number 3, 1985*
4. **Superfractals: patterns of Nature**; *http://www.amazon.co.uk/SuperFractals-Michael-Fielding-Barnsley/dp/0521844932*; *Barnsley, Michael F; Cambridge University Press; 7 Sep 2006; ISBN 978-0521844932*
5. **The Computational Beauty of Nature: Computer Explorations of Fractals, Chaos, Complex Systems and Adaptation**; *http://www.amazon.co.uk/The-Computational-Beauty-Nature-Explorations/dp/0262561271*; *Flake, Gary W; MIT Press; 1 Mar 2000; ISBN 978-0262561273*
