Triangle Genome Documentation

Weston Ortiz
Juan Somarriba Jarque
Niranjan Humagain

Genome Representation

The genome is represented by 200 triangles (defined in Constants.java) each with 10 genes, which are represented by 3 arrays, two size 3 int arrays for x and y points of vertices, and a size 4 array for the color values of red, green, blue, alpha.

Fitness

Fitness is calculated using Euclidian distance squared by default, there is an option in the source code to use either manhattan distance or distance squared.

Fitness is calculated by getting a raster of both the target image and the genome image and then fitness compares all pixel values in that raster.

There is some code which attempted to do partial calculation of fitness by only calculated the fitness of the bounding rectangle of the triangle for the current gene that was being changed, but there is a bug in that code where the fitness will be better on the bounding rectangle but not the actual image.

Always done on an off screen buffered image.

Hill Climbing

Hill climbing is done in two ways, the first is basic hill climbing where a delta of magnitude 1 is randomly chosen for a randomly chosen gene and if that delta improves the gene will continue to move along that direction for a genome until no more improvement is made. After two successful climbs along a delta that delta will be doubled, if it can no longer move along that delta or if the delta is too large the delta is halved.

The other kind of hill climbing is a random gene is chosen and then a random value is set for that gene, if that value improves the genome then a delta is calculated as the new value minus the old value and it continues to move along that delta in the same manner as the other method of hill climbing.

Starting Triangles

Starting triangles are randomly colored, there are first 4 triangles created to cover the whole image.

Then triangles are randomly created until all triangles are made, right now they use a method which creates triangles with a fixed alpha of a little less than half.

Crossover

There are two means of crossover in the project, single point crossover and uniform crossover.

Single point crossover chooses a random distance within the hamming distance of a genome and crosses over at the gene that corresponds to that distance.

Uniform crossover randomly chooses a gene from either of the parents for each gene.

Both of the methods in this project create two children, where for every gene that is chosen in one child the other child will have the other parents gene.

Selection and Breeding

Breeding is done after a set number of generations have occurred, there is a multiplier which is used to calculate when breeding should occur, this multiplier is multiplied by the number of tribes to get how many generations should occur before breeding. When the number of generations is reached the main thread pauses all Tribes, gathers breeders from the tribes and then adds all breeders to all tribes.

Breeders are chosen as the best genome in each tribe (as this was a requirement) and then random genomes from the rest of the tribe until each tribe gives the initial population size of breeders.

Children are only added if they are not that similar to another genome in the tribe, if they are similar then the better of the two keeps that position in the tribe.

Similarity is chosen by Hamming distance and a constant called DIVERSITY METRIC in Constants.java, which is set to GENE_COUNT/3

Implementation

The triangle genome was implemented as a main thread called TriangleGenome with controls a thread for each tribe.

The main thread TriangleGenome synchronizes data with the GUI from the Tribes. It also controls when breeding occurs.

Each tribe will periodically update their data for TriangleGenome to update the global data

Each tribe does it's own hill climbing and fitness calculations.

Reading/Writing genome files

Genomes can be saved as either ASCII txt files which are easily readable or XML files

Genomes can also be read from either file format.

Shortcomings

No accumulated statistics.

Could not get sub fitness calculation to work.

Tests are lacking.

Next does not start the thread for one generation but instead just performs one generation step manually for each tribe.

References

Some ideas were taken from these references
http://blog.nihilogic.dk/2009/01/genetic-mona-lisa.html and
http://alteredqualia.com/visualization/evolve/

The most significant would be using random mutation to get good results and lower starting transparency for triangles. Mona lisa's face image is also taken from the second link.

Extra Image

There are two extra images, the mona lisa face does particularly well because it is a small image which allows for fast drawing and many generations per second. The other weren’t tried as much, the great wave and the poppy fields don’t seem to do as well as mona lisa.