CS 351: Design of Large Programs

Project 2:
The Triangle Genome

Parallel Genetic Algorithm for Image Compression

Instructor: **Joel Castellanos**
e-mail: joel@unm.edu
Web: http://cs.unm.edu/~joel/

Lab Instructor**: Torin Adamson**
e-mail: wolfcoder@dreamincode.net

9/19/2014

Image from Geneffects

Search Algorithms

- Name some.
- How do they work?
- Computer Scientists have been working on this for a long time. Is this a problem that is solved or an active field of research?
- If it is active, what type of results do people working on it publish?
- If something is an "active" field, then there are many funding sources. Who would fund this?

2

Project 2: The Triangle Genome

- Based on Roger Aliasing's 2008 Work: Genetic Programming Evolution of Mona Lisa.
- Find a set of overlapping semi-transparent triangles that form a good likeness of a target image while using a number of triangles that is small compared to the number of pixels in the image.

3

Project 2: Requirements Overview

- Teams of 3 advanced, amazing super java coders.
- Must use Hill Climbing and Genetic Algorithms to find a "good" set overlapping, semi-transparent triangles to encode a target image.
- Must have GUI with specific specifications.
- Must make good use of 8 core processor.
- Must have complete set of unit tests.
- Must have extensive performance analysis.

4

Search Algorithm

- In computer science, a search algorithm finds an item with specified properties among a collection.
- When "*Specified Properties*" includes "*best*" the search must include *all items in the collection*.
- Items may be:
  - Stored individually as records in a database.
  - Elements of a search space defined by a mathematical formula or procedure.
- Algorithms for these problems include:
  - Brute-force Search.
  - Heuristics that try to exploit partial knowledge about structure of the space.

5

Big O: Brute Force for Triangle Genome

A simple brute force search of the Triangle Genome for the 450×363 pixel Mona Lisa is simply:

$n = 200 \times (450 \times 363) \times (450 \times 363\text{-}1) \times (450 \times 363\text{-}2) \times 255 \times 255 \times 255 \times 255$

$n = 3.69 \times 10^{27}$

Thus, if you had a computer that could evaluate fitness in 1 millisecond, the solution can be found in just:

$$= 3.69 \times 10^{27} \times \left(\frac{1\ s}{1000\ ms}\right)\left(\frac{1\ min}{60\ s}\right)\left(\frac{1\ hr}{60\ min}\right)\left(\frac{1\ day}{24\ hr}\right)\left(\frac{1\ year}{365.25\ d}\right)$$

$$= 1.17 \times 10^{17}\ years$$

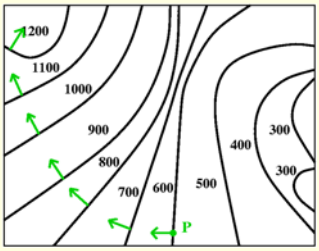
$$= 116{,}798{,}926{,}014{,}151{,}000\ years$$

Age of the Universe: $13{,}750{,}000{,}000\ years$

6

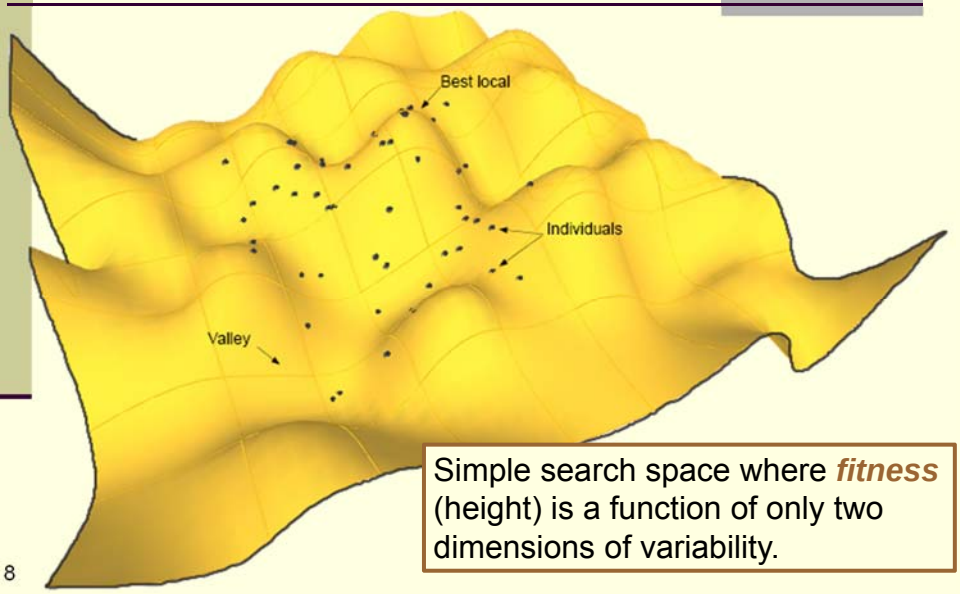
Local Search Methods

- ***Local search methods*** attempt to find a global solution by examining a particular item and some ***neighborhood*** around that item.
- The elements of the search space are viewed as the vertices of a graph, with edges defined by a set of heuristics applicable to the case.
- The space is scanned by moving from solution to solution, according to some heuristic:
  - Steepest Ascent.
  - Best-First Criterion.
  - Stochastic.



7

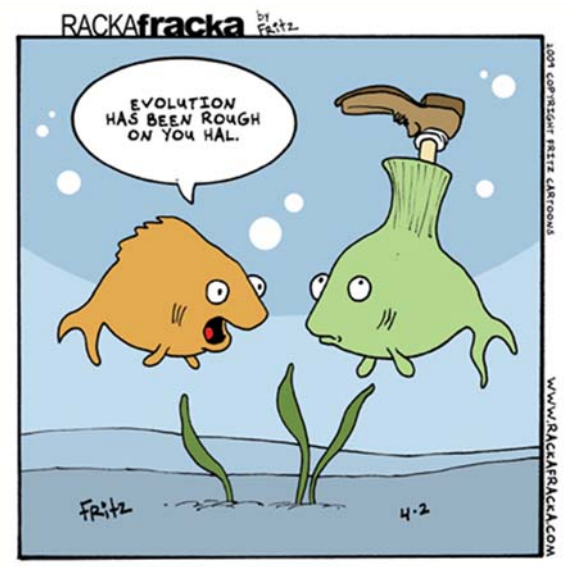
Search Space with *Local Maxima*



Simple search space where ***fitness*** (height) is a function of only two dimensions of variability.

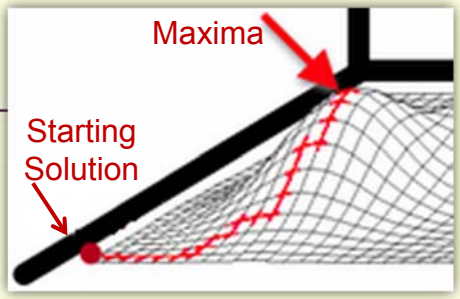
8

A Problem of Local Maxima



9

Hill Climbing



- In computer science, ***hill climbing*** is a mathematical optimization technique which belongs to the ***local search*** family.
- Start with an ***arbitrary solution*** to a problem.
- Attempt to find a better solution by incrementally changing a ***single element*** of the solution.
- If the change produces a better solution, then that ***same change is made again***. Otherwise a different change is made.
- Repeat until no further improvements can be found.

10

Pure Hill Climbing in Triangle Genome

- Each genome is a "solution" within the search space.
- Each genome has 200 triangles $\times$ 10 genes = 2000 values
- One Generation:
  - If the last change was an improvement, do it again. otherwise...
    - Choose a value with probability 1/2000.
    - Choose a direction ($\pm$) with 50% probability (if on a boundary, move toward center).
    - Modify chosen value in the chosen direction by 1 unit.
  - Evaluate fitness of "child".
- On average, pure hill climbing reaches a normalized fitness of 75 in 40,000 generations on the 3 provided large images.

11

Hill Climbing Mutation Types

- In pure hill climbing, a mutation means to ***incrementally change a single element of the solution***.
- Heuristics that try to exploit partial knowledge about structure of the space may define different ***types*** of mutation. In the Triangle Genome, these might be:
  - Mutate single gene (one of the 2000 values in a genome).
  - Mutate triangle vertex (x_i, y_i).
  - Mutate triangle color (r, g, b, a).
  - Mutate triangle location / orientation / size.
  - Mutate triangle order.

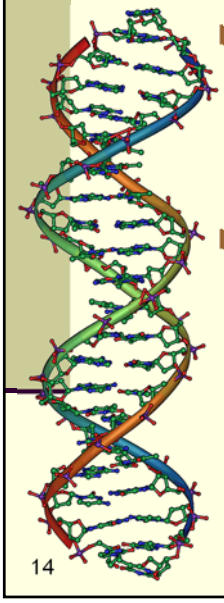
12

Adaptive Hill Climbing

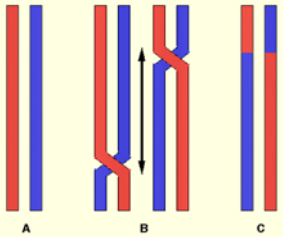
- Start with an arbitrary mutation step size, S.
- Let each mutation be a random step, s_i, from 1 to S in a random direction, $\pm$ (or toward/away form center).
- Select a random *type* of mutation to a random *item*.
- When a mutation yields an improvement:
  1) Increase the probability of selecting that mutation (same type, same item, same direction).
  2) Adapt the step size:
     - If $s_i < S/2$, then decrease S.
     - If $s_i \geq S/2$, then increase S.
- Adaptive Rate: As slow as needed to observe trends.

13

Genetic Algorithm

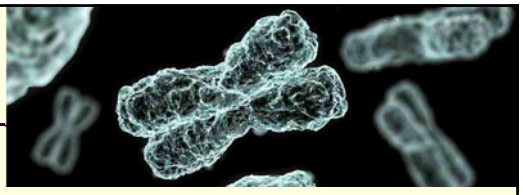


- A genetic algorithm (GA) is a search technique used in computing to find exact or approximate solutions to optimization and search problems.
- Genetic algorithms are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology:
  - Inheritance
  - Mutation
  - Selection
  - Crossover



14

GA Chromosomes



- In genetic algorithms, a chromosome is a set of parameters which define a proposed solution to the problem that the genetic algorithm is trying to solve.
- The chromosome is often represented as an array of characters, although a wide variety of other data structures are also used.
- *Each parameter* that makes up the chromosome is called a *gene*.
- The *full set* of genes in a solution is called the solution's *chromosome*, *DNA* or the *genome*.

15

GA Genome Design Example 1

- Suppose the problem is to find the integer value of x between 0 and 255 that provides the maximal result for $f(x) = 536x - x^2$.
- Possible solutions are the integers from 0 to 255, which can all be represented as 8-digit binary strings.
- Thus, we might use an 8-digit binary string as our genome. If a given genome in the population represents the value 155, its genome would be 10011011.
- For this problem, the binary representation is a better genome than the base 10 string which is too short and had different constraints on different digits.

16

GA Genome Design Example 2

- A more realistic problem we might wish to solve is the travelling salesman problem.
- In this problem, we seek an ordered list of cities that results in the shortest trip for the salesman to travel.
- Suppose there are six cities, which we'll call A, B, C, D, E, and F.
- A good design for our genome might be the ordered list we want to try.
- An example genome we might encounter in the population might be DFABEC.

17

How does Natural Selection Find Solutions?

The population or individual does not "want" or "try" to evolve, and natural selection cannot try to supply what an organism "needs."

Natural selection just selects among whatever variations exist in the population. The result is evolution.

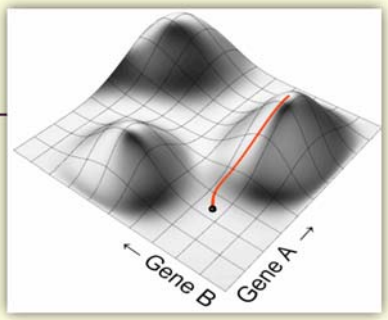
18

Natural Selection Conundrums

- How does natural selection explain the existence of altruism?
- Why do men have nipples?
- Why does the variation between creatures seem continuous within a species, yet seem to make large jumps between species?
- It seems that many improvements require ***many genes to be changed in a single step***, yet the probability simultaneous events decreases exponentially.

19

Fitness Function



- A ***fitness function*** is a particular type of objective function that quantifies the optimality of a solution (a genome) in a genetic algorithm so that each genome may be ranked against all the other genomes.
- A good fitness function correlates closely with the algorithm's goal, and yet may be computed quickly. Speed of execution is very important, as a typical genetic algorithm must be iterated many, many times in order to produce a usable result for a non-trivial problem.

20

Fitness Function in Triangle Genome

- Most obvious:
  - Pixel-by-Pixel, Euclidian distance between colors:

$$= \sum_{x=0}^{width} \sum_{y=0}^{y=height} \sqrt{(r_{x,y} - \hat{r}_{x,y})^2 + (g_{x,y} - \hat{g}_{x,y})^2 + (b_{x,y} - \hat{b}_{x,y})^2}$$

- Other ideas:
  - Replace RGB color space with HSB.
  - Sum a multi-resolution comparison with each resolution weighted to normalize. For example, a 50×50 image has ¼ the number of pixels as a 100×100 image, so multiply the sum of the errors of the smaller image by 4.



21

Fitness Function Optimizations

- Do not evaluate square root in:

$$\sqrt{(r - \hat{r})^2 + (g - \hat{g})^2 + (b - \hat{b})^2}$$

- Replace full fitness with differential fitness (often only part of the child will differ from a parent).
- Fitness is a sum of differences. Thus, quit as soon as fitness of part of image is worse.
- Evaluate fitness low resolution to high resolution.
- For cash coherency, be sure to visit pixels in the order they are stored in Java's bufferedImage.
- For pixel color, do not create Color objects, use int rgb.

22

GA Selection

Selection is the stage of a genetic algorithm in which individual genomes (full solution sets) are chosen from a population for breeding (recombination or crossover).

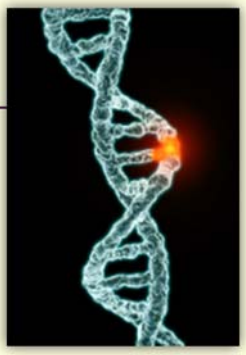
Genomes which are more optimal (higher fitness), are more often allowed to reproduce and mix their datasets by any of several techniques such as:

- Tournament Selection
- Fitness Proportionate Selection
- One parent always from best group, other parent with equal chance of being anywhere in group.

Every genome in the population must have ***some chance*** of breeding or it is pointless to keep it in the population.

23

GA Mutation → Hill Climbing



- The purpose of mutation in GAs is to allow the algorithm to avoid local minima by preventing the population of genome from becoming too similar to each other, thus slowing or even stopping evolution.
- The classic example of a mutation operator involves a probability that an arbitrary bit in a genetic sequence will be changed from its original state.
- While this is a classic textbook example, in practice it is unacceptably slow.
- Most genetic algorithms replace purely random mutation with hill climbing

24

Fitness Proportionate Selection

(suggestion, not requirement)

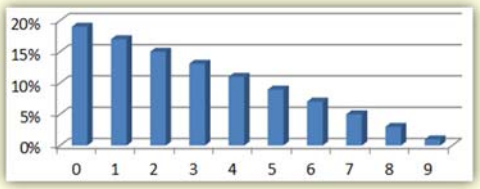
Consider a ridiculously small GA with a population of 10 that has been sorted from most fit in element 0 to least fit in 9.

```
java.util.Random rand = new java.util.Random();
int r = rand.nextInt(10);
```

Will produce a **uniform distribution** where each integer from 0 through 9 is equally probable.

```
r = (int)(10.0*Math.abs(
     rand.nextDouble() - rand.nextDouble()));
```

Will produce a ***right-triangular distribution***:



25

Genome Representation

Each image is generated from DNA composed of:

1. An *ordered* list of some number of triangles.
2. Each triangle has ten dimensions: Three vertices in 2D space a uniform color in 4D color space (red, green, blue and transparency):

$$(x_1, y_1, x_2, y_2, x_3, y_3, r, g, b, t)$$

26

Unit of Selection: The Gene

- A unit of selection is a biological entity within the hierarchy of biological organization (e.g. genes, cells, individuals, groups, species) that is subject to natural selection.
- For several decades there has been intense debate among evolutionary biologists about the extent to which evolution has been shaped by selective pressures acting at these different levels.
- In computer science, most genetic algorithms use the ***gene*** as the only unit of selection.

27

Size of the Initial Population

- In a genetic algorithm, the initial population should be large enough to contain *most* of the genetic material required for a good solution.
- Thus, the unit of selection (the gene) must be small.
- If the gene is a triangle, then the initial population for a 512×413 image would need to be on the order of: $(512 \times 413)^3 \times 256^4 = 4.0 \times 10^{25}$
- By contrast, if a gene is one of the 10 properties of a triangle, then for a 512×413 image, less than 500 triangles can be constructed to include all possible genes: (i.e. (0,0,0,0,0,0,0,0,0,0), (1,1,1,1,1,1,1,1,1,1), ...).
- The multitude of combinations of these genes is achieved through breading and crossover.

28

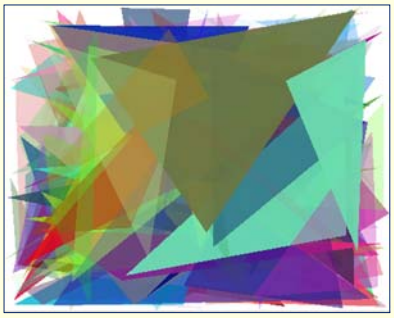
Seeding Initial Population

- Genetic algorithms are slow. Too slow to be useful if not ***well tuned to the specific problem***.
- Triangles are drawn from element 0 through element 199. Seed the first n triangles as large triangles hoping to capture low frequency information and the last m triangles be small hoping to capture high frequency information.
- Start with a few special triangles (maybe 4 triangles that totally tile the window from edge to edge.
- Sample the image for some common colors.

29

Poor Starting Triangles

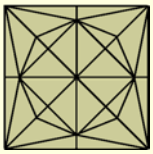
- This 450×363 image shows a sequence of 200 triangles where the *x*-coordinate of each vertex is a random number, uniformly distributed between 0 and 449.
- Notice that the triangles are piled up in the middle.
- This is because an edge pixel only gets color when a vertex is on that edge. By contrast, an internal pixel gets color whenever vertices surround it.

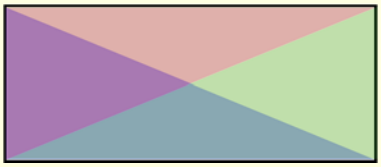


30

Selection Initial Population

- In a genetic algorithm, a good initial population is critical.
- The initial population should contain significant randomness, but not necessarily be totally random. For example:
  - Make one chromosome of the initial population that is systemically constructed by a set of overlapping triangles that cover the whole image.
  - Sprinkle throughout the 200 triangles of every chromosome a set of four predetermined triangles that cover the whole image, each having a color equal to the most common red, most common green and most common blue in the target image.





Population Hierarchy

Tribe: Contains 50 to 2000 or so genomes.

Genome: Contains 200 triangles.

- A genome is the set of parameters used to create one image.
- Each genome has a fitness.
- Genomes are selected from the population for breading.

Triangle: A triangle is a part of a genome.

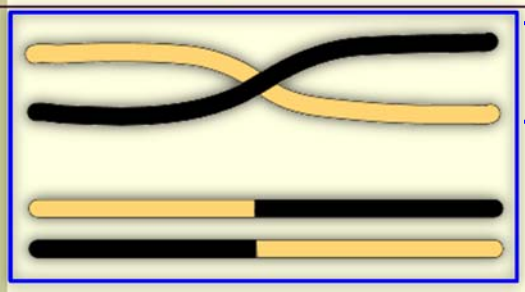
- Each triangle contains 10 genes (x_1, y_1, x_2, y_2, x_3, y_3, r, g, b, t).

Gene: Each coordinate of each vertex, each color value of a triangle.

32

Crossover: Key Component in a GA

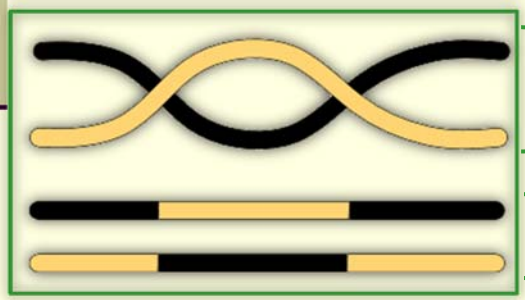
Crossover points can be ***anyplace*** along the genome



Parents

Children

Single Point Crossover



Parents

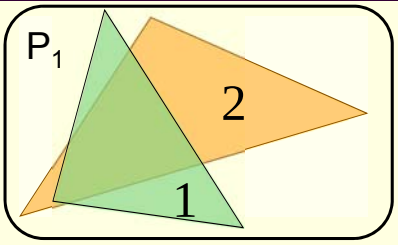
Children

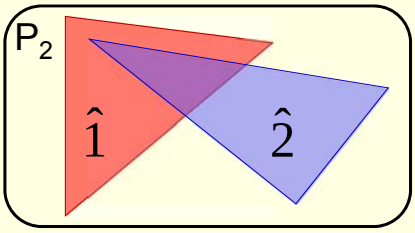
Double Point Crossover

33

Single Point Crossover

Each genome in this example has just 2 triangles.





Triangles 1 & $\hat{1}$ *Triangles* 2 & $\hat{2}$

P_1 $[(x_1^1 y_1^1 x_2^1 y_2^1 x_3^1 y_3^1 r^1 g^1 b^1 t^1), (x_1^2 y_1^2 x_2^2 y_2^2 x_3^2 y_3^2 r^2 g^2 b^2 t^2)]$

P_2 $[(\hat{x}_1^1 \hat{y}_1^1 \hat{x}_2^1 \hat{y}_2^1 \hat{x}_3^1 \hat{y}_3^1 \hat{r}^1 \hat{g}^1 \hat{b}^1 \hat{t}^1), (\hat{x}_1^2 \hat{y}_1^2 \hat{x}_2^2 \hat{y}_2^2 \hat{x}_3^2 \hat{y}_3^2 \hat{r}^2 \hat{g}^2 \hat{b}^2 \hat{t}^2)]$

$[(x_1^1 y_1^1 x_2^1 y_2^1 x_3^1 \hat{y}_3^1 \hat{r}^1 \hat{g}^1 \hat{b}^1 \hat{t}^1), (\hat{x}_1^2 \hat{y}_1^2 \hat{x}_2^2 \hat{y}_2^2 \hat{x}_3^2 \hat{y}_3^2 \hat{r}^2 \hat{g}^2 \hat{b}^2 \hat{t}^2)]$

34

Diversity

- A genetic algorithm will stall without a diverse population.
- Tips on maintaining diversity:

a) Reject any child with genes identical to either parent.

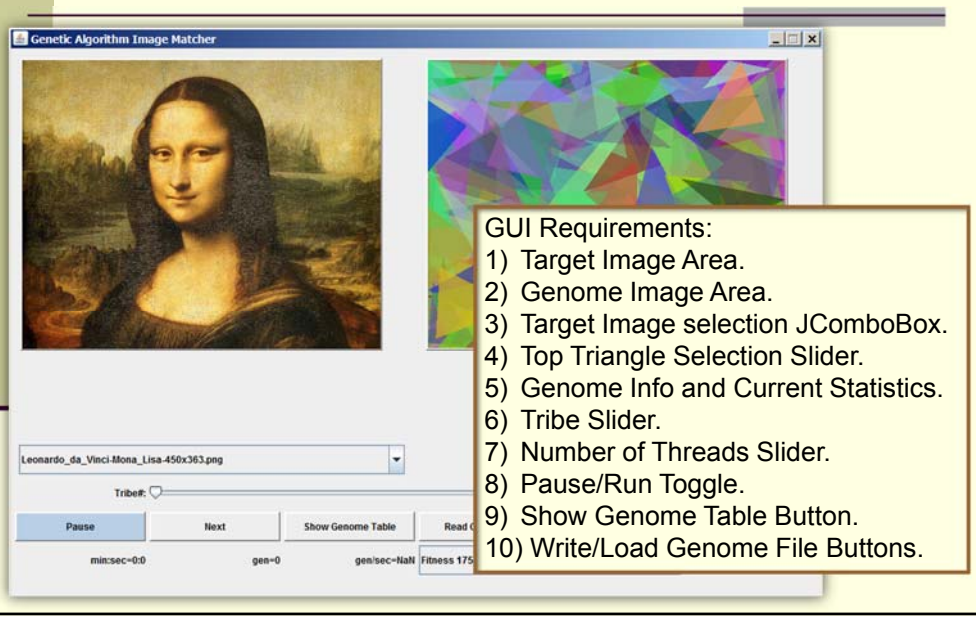
b) Avoid inbreeding (breeding pairs of parents with similar genomes).

c) Rather than picking a random location for crossover, count the number of genetic differences. Then, crossover after a random number of difference between 1 and n-1.

d) Modify the base fitness function to value diversity.

35

Graphical User Interface

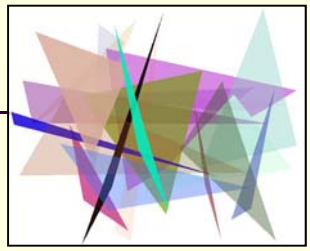


GUI Requirements:
1) Target Image Area.
2) Genome Image Area.
3) Target Image selection JComboBox.
4) Top Triangle Selection Slider.
5) Genome Info and Current Statistics.
6) Tribe Slider.
7) Number of Threads Slider.
8) Pause/Run Toggle.
9) Show Genome Table Button.
10) Write/Load Genome File Buttons.

After 1 hour on an Intel quad core i7

37

Milestone 1 Due Tue, Sept 30



- Create a GUI Frame (or frames) with:
  - Control Panel: Working buttons for Start/Pause, Reset, Next, and Previous (one step only), Sleep Slider, and JComboBox that selects 1 of 3 target images.
  - Target Image Panel
  - Genetic Algorithm Image Panel
- Each generation, generate 200 random triangles (random, 2D coordinates of each vertex, random red, blue, green, and transparency of each triangle).
- Display (somewhere in the GUI) calculated fitness value at each generation. For this milestone, the fitness function is the sum of the errors in three-color space of each pixel.

38

Milestone 1: Grading

7 points: JComboBox Loads and displays the 3 target images.

7 points: Next button shows 200 triangles with random vertices, random RGB color and random transparency.

7 points: At each generation, the fitness value is correctly calculated and displayed.

7 points: Sleep slider, Pause, Back, and Reset buttons work.

7 points: Code is well commented, clear an easy to read.

39

Image 1: Leonardo da Vinci: Mona Lisa

Use the 512 × <512 Pixel source images posted in the website.

Large areas of similar color.

40

Image 2: Monet: Poppies, near Argenteuil

512×384

The small red poppies will be hard for overlapping triangles.

41

Image 3: Hokusai: The Great Wave off Kanagawa

42

Image 4: (Not part of milestone 1)

- Add an image of your choosing for image 4.
- It could be a photograph or painting.
- When choosing an image find one your algorithm does particularly well with, yet will impress the casual observer.

43