Genetic Algorithm

Artificial Intelligence Project 2 Report

Contributors:

Anmol Arora (130050027)

Pranjal Khare (130050028)

Aman Goel (130050041)

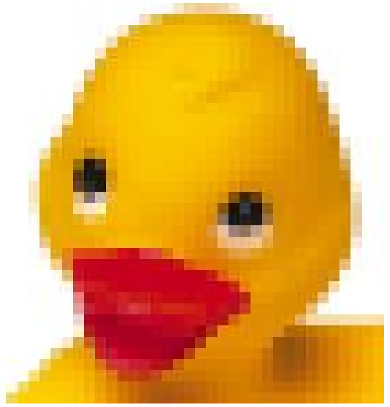
Problem Statement

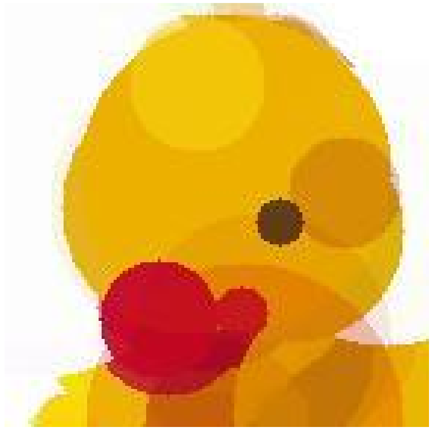
- The basic idea is to learn and understand the working of a ***Genetic Algorithm***
- We have programmed a Genetic Algorithm in Java that takes an image as input and approximates the image using circles/rectangles of different dimensions and colors

Outcomes of the project

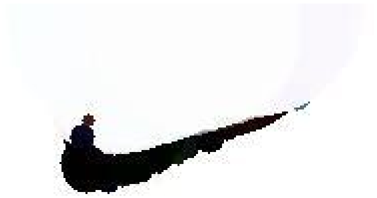
- We ran our algorithm on several different types of images, specifically ***logos*** of various famous companies
- We have shown the stage-wise progress of the algorithm by ***generating the intermediate images***
- We have then used software to convert those image sequences to videos
- Final image generated is quite a good approximation of the original image and the algorithm performed reasonably well after setting appropriate parameters
- The algorithm ***takes significant time*** to run (~20 hours for 100k generations) because it is highly computationally expensive

Screenshot - duck

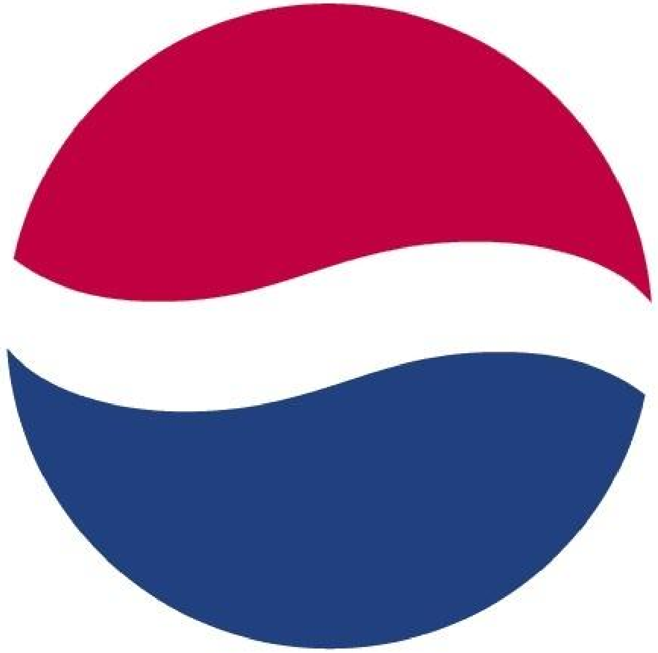


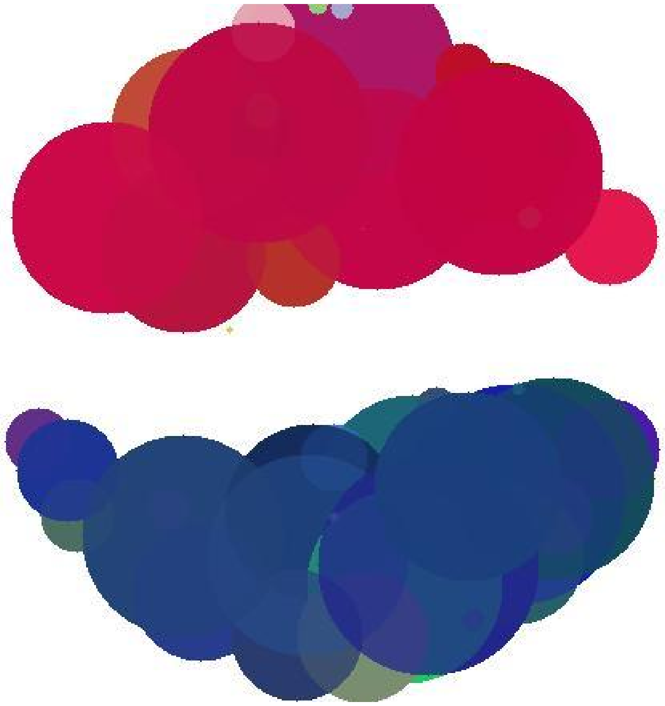


Screenshot - Nike logo

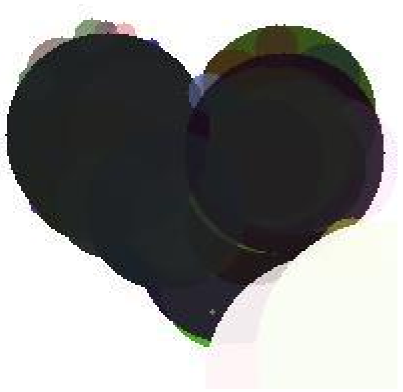


Screenshot - Pepsi logo

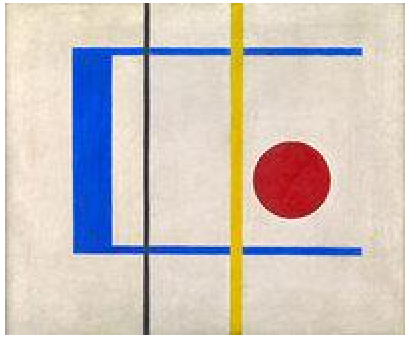


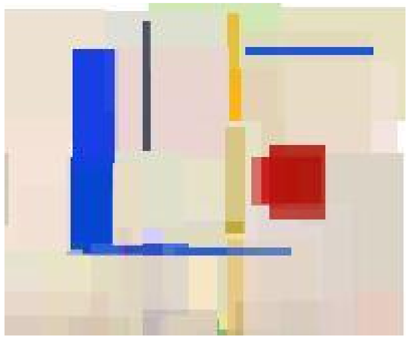


Screenshot - Black Heart

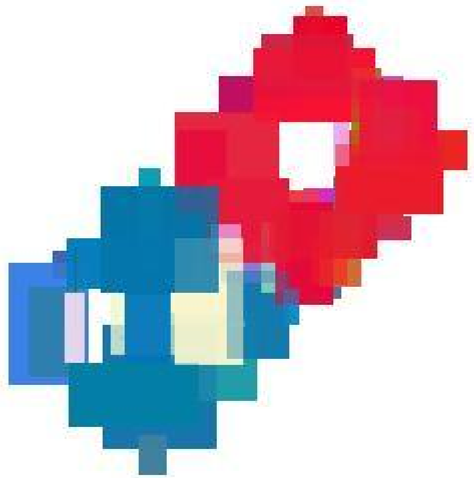


Screenshot - abstract art



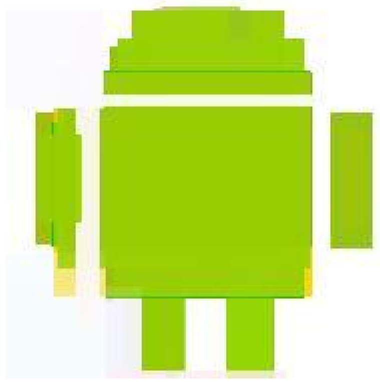


Screenshot - Dominos logo

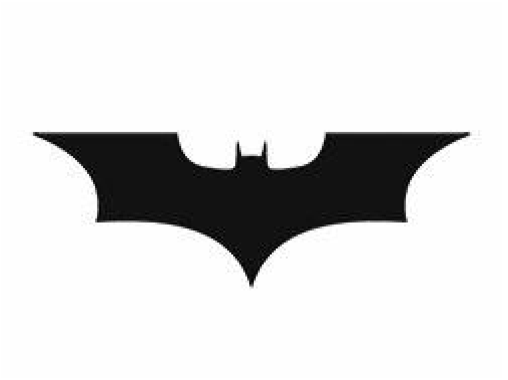


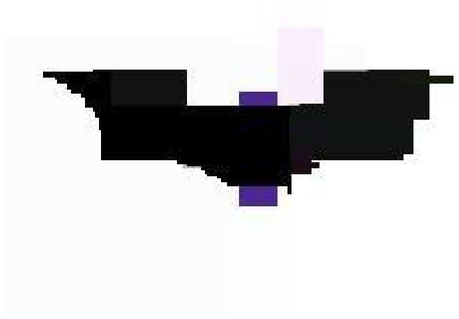
Screenshot - Apple logo

Screenshot - Android logo

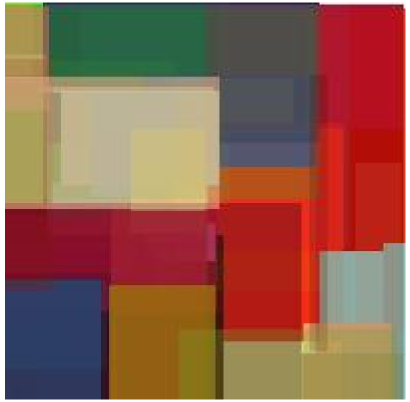


Screenshot - Batman



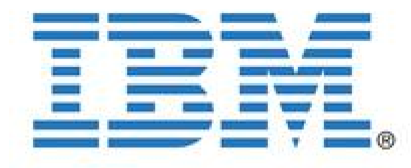


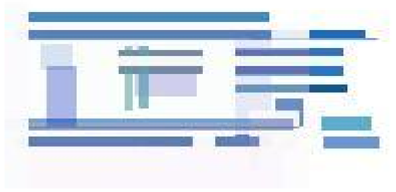
Screenshot - Block Art



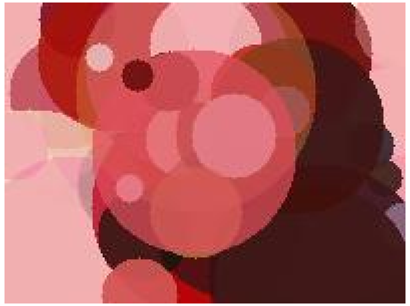
Screenshot - National Flag

Screenshot - IBM logo





Screenshot - Angelina Jolie (complex image)



Code Overview

The most important aspect of our code is the fitness function. The fitness function is defined as:

```
ans += ((Math.pow(2,abs(m.getRed() - n.getRed())/255.
0) + Math.pow(2,abs(m.getGreen() - n.getGreen())/255.
0) + Math.pow(2,abs(m.getBlue() - n.getBlue())/255.0)) )
/6.0;

this.fitness = 100000.0/ans;
```

We have used Java libraries to convert the input image to 2D array and also used Java libraries to convert the output 2D array to an image

Our program involves several parameters:

- POPSIZE = 15;
- MAXGENS = 150000;
- PXOVER = 0.8;
- PMUTATION = 0.1;
- circleCount = 50;
- RADIUSLIMIT=175;
- PMUTATIONCOLOR = .2;

The following is the order of function calls:

- selector();
- crossover();
- mutate();
- evaluate();
- elitist();

We have described these functions in our previous report in detail

Contribution of individual members

The 3 members worked as a team and contributed almost equally.

Below are individual team member contributions:

- Aman Goel (130050041) - 1/3rd
- Anmol Arora (130050027) - 1/3rd
- Pranjal Khare (130050028) - 1/3rd

Learning from Project

- We have learned the various aspects of Genetic Algorithm and the intricacies involved - crossover, mutation, elitism, etc.
- It is important to fine tune the parameters of the algorithm appropriately to make it work properly.

Suggestions for future work

- Instead of using just circles/rectangles, we can generalize the algorithm for various types of polygons. The number of sides of the polygon can be varied randomly. This will give more flexibility to the algorithm.
- We can apply concepts of parallel computing to make the algorithm run faster. This way, we can run it for more number of generations to get even better results in the same time. Moreover, we can run the algorithm on higher resolution images.
- We can try different variants of the fitness functions which may help us to converge faster towards our target image.