CS480/580 Introduction to Artificial Intelligence

## Assignment 2

Total Points: 100

Due Date: 10/12/2021

**Genetic Algorithm: Mimic an image with rectangles**

The design of this assignment is inspired by the following blog.

<https://rogerjohansson.blog/2008/12/07/genetic-programming-evolution-of-mona-lisa/>

In this assignment, you are asked to implement your own genetic algorithm to use a set of semi-transparent rectangles (say 100) to mimic an arbitrary image. Each rectangle is represented by 4 points in 2D space, (x0, y0), (x0, y1), (x1, y0), (x1, y1) and a color of (r, g, b). An individual can be represented as a fixed-length array of the rectangles.

To implement the genetic algorithm, you need to design the following components:

1. Fitness function: You are required to implement a fitness function to measure how well your individual is matching with the target image. You can simply sum the RGB value differences of each pixel in your generated pixel with respect to the pixel in the target image or you can design your own fitness function.
2. Selection: You are required to implement selection method to select individuals for reproduction.
3. Genetic operation: You need to implement crossover of two individual and mutation of a single individuals.

Then, you can assemble the above components into genetic algorithm iterations.

**Analysis**

The parameters (population size, crossover rate, mutation rate, etc.), the implementation of fitness function, the genetic operators can have significant impact to the performance of the genetic algorithm. You are required to submit a written report showing analysis on how different parameters have impact on the convergence.

**Hints**

Always start from the simplest setup. Attempt a small image at first. Genetic algorithm on a large image takes time.

Good data structure is important.

**What to Hand in**

1. Well documented codes implementing the genetic algorithm. A README file should provide instructions on how to compile and execute the code. If you use codes available in the internet, you need to specify and give credit to the authors. Provide detailed description of the implementation of your genetic algorithm.
2. For your best implementation, provide the initial image, the final image, and two in-between images for four target images you select to show the evolution process.
3. Analyze the results using different parameters in the genetic algorithm and how they impact the performance of the genetic algorithm.
4. Submit an image of your photo and an mimicked photo with rectangles using your genetic algorithm.

Please turn in the program and the analysis before the assignment due date.

Bonus: (10 pts)

Bonus will be given for students who exhibit novel implementations, for examples, fitness function, representations, crossover and mutation, etc.

**Answers:**

1. The program can parse all the files in the /data directory. It can also get the pixel data, such as the RGB values of each rectangle.
2. There is a class for a Rectangle object, that has three numbers as attributes. Each number represents one of the RGB spectrum. I did create a Coordinate class that has X and Y coordinates, but I decided not to include it in my Rectangle class as I initially planned. This was because I could not figure out how to implement that coordinate class while also comparing rectangles.
3. The program can create rectangles with randomly generated values, as well as compare those values with the values of the original rectangles.
4. I tried implementing the fitness(), selection(), and mutation() functions, but I was unable to finish them.
5. My plans for the fitness() function was to find the differences between the rectangles and add them to a special numpy array. That numpy array would be used for the selection() function, which would then lead to the mutation() function.
6. I had an idea that my mutation() function would also created a empty numpy array, which was to be converted into an image. The array would have a mixture of both the original rectangles and the generated rectangles with a small difference.