EVOLUTIONARY ALGORITHM

An Approach For Template Matching*

Muhammad Bilal †

Sunday 31st October, 2021

Acknowledgement

I would like to express my special thanks of gratitude to my honorable teacher Dr. Junaid Akhtar who provided me a golden opportunity to perform this wonderful project and gain the experience of what AI is? This helped me in doing a lot of Research and i came to know about so many new things. I am really thankful to them.Secondly, i would also like to thanks the TA Ali Raza Khan also my friends who helped me a lot in finalizing this project within the limited time frame.

*AI [CS304]

†Namal University Mianwali(BSCS201919)

Abstract

This report is about solving the Template Matching Problem using a natural phenomenons i.e Natural selection and Evolution. Natural selection is the process through which populations of living creatures adjust and acclimatize. Individuals in a populace are naturally divergent, implying that there exists diversity in the nature. This variation implies that some members of the population have some traits more fitted to the current environment than others. Individuals with such traits that give them some benefit to survive in the environment are more likely to take part in reproduction passing their adaptive traits on to their offspring. Over the time, these beneficial traits become more common in the populace. Through process of natural selection, beneficial traits are passed on to next generations. This process leads to speciation, where one species give rise to a new and very unique variety of species. It is one of the processes that introduces the concept of evolution and assists us in understanding the core concept of diversity of life on Earth. Using this phenomenons and Darwin's Theory of Natural Selection and Evolution we will try to model an application to solve Template Matching problem.

Table of Contents

1 Roadmap

- Problem Statement.
- Natural Selection And Evolution
- Darwin's Theory
- Modeling
- Application

2 Problem Statement

The problem in hand is that we have two pictures and we are required to build an application which can process these images and finds the second image(Template Image) in the original first image. The application is to be designed and build in Python. To solve the problem, we utilizes an algorithm which is basically inspired from the Darwin's Theory of Natural Selection and Evolution. In this project, we will try to map our problem on the phenomena of Diversity of Life and try to figure out the solution using a model which is derived from Darwin's Theory.

3 Natural Selection and Evolution

When we look around us in the environment the thing which we noticed the most is the diversity in the everything. None of the two things looks exact similar there is some variation in species. Formally, a species is defined as a group of possibly interbreeding organisms. The difference in species may be of traits, shapes, or environment. The variation exists due to genetics, species and the ecosystem or the habitat and give rises to Biodiversity. Biodiversity is an important aspect in the world because it enables the survival and sustainability of living things in an environment. The theory of evolution

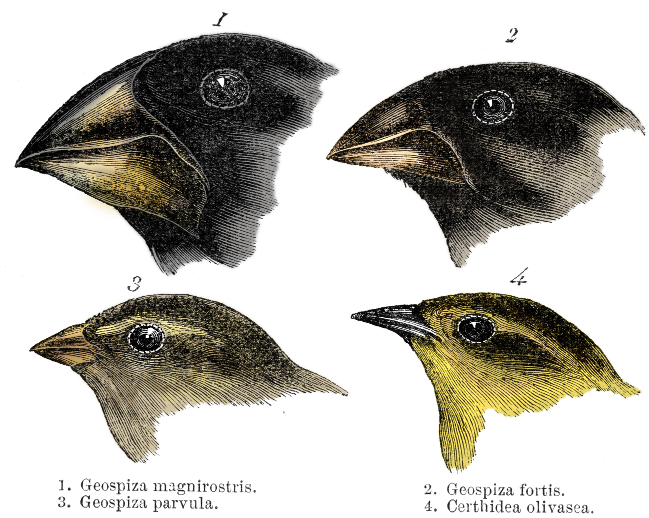

Figure 1: Finches with different beaks adapted due to different environmental conditions.

by natural selection explains how a species can change over time. All species from the bacteria on our skin to the birds outside have evolved from one species to another. Although living things seem to remain the same from generation to generation, this is not the case because evolution continues. Evolution is a process by which the characteristics of a species change over time, which can eventually lead to the birth of new generations.

Figure 2: Evolution of humans.

4 Darwin's Theory of Evolution by Natural Selection

A theory is an idea, concerning how something in nature works, that has gone through complete testing, through perceptions and trials intended to prove the idea right or wrong. With regards to the evolution of life, different philosophers and researchers, including an eighteenth-century English doctor named Erasmus Darwin, proposed various ideas of what later had become evolutionary theory. But, evolution didn't reach the status of being a scientific theory until Darwin's grandson, the most famous Charles Darwin, published his popular book On the Origin of Species. Darwin suggested that evolution happens due to a phenomenon called natural selection. Natural selection, one of the most important theories of biology, was proposed by the Charles Darwin in the 19th century.

The theory can be used to explain an imbalance between a population, a particular feature of the body, or a phenotype. In the theory of natural selection, Individuals who are better equipped to survive and physically regenerate produce more offspring than who lack this type of fitness because those individuals do not reach the age where they can reproduce or have fewer children than those fittest. Natural selection is sometimes summed up as "Survival of the fittest" because the most suitable individual - the ones that are best suited to their environment - are the ones that are most successfully reproduced and passed on their traits to the next generation. The evolutionary theory provides a very fascinating essence i.e. we can think of monkeys trying teaching their community that they are evolved from human.

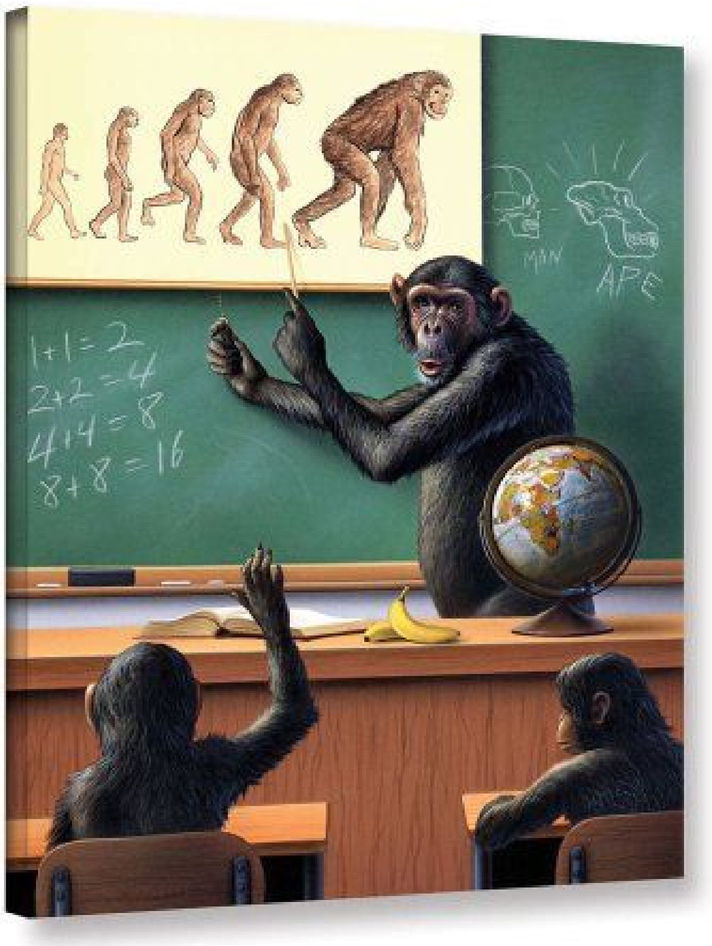

Figure 3: Evolutionary concept of Monkeys

5 Modeling

Being a computer scientist it is our duty to find solution to our problem by studying a natural reality and the theory and the theory associated to it. After reading and analyzing the both section we build a model to solve the problem which is purely inspired from the theory. Here the problem which we want to solve is Template Matching. Template matching refers to image processing where we find similar templates in a source image by comparing it with a sample templates. To solve the problem we will try to design a model which will be fully inspired from Darwin's Theory of Evolution by Natural Selection.

5.1 Evolutionary Theory

In the process of Evolution by Natural Selection, nature preserves the fittest individual w.r.t adaption to their living environment. This slow but continuous process of evolution give rise to ever-divergent variation in the population, which w.r.t time appears as a different species.

5.2 Computational Model

From the theory we can clearly see that there are some specific processes which are happening again and again. These continuous processes will take us to the most fittest individual. In our context the most fittest individual will be the template in source image which matches the given template image. The model from this theory will be as follow:

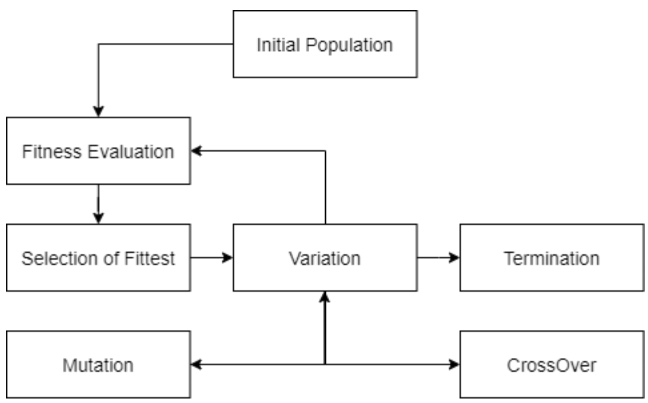

Figure 4: Flow Diagram of Model

5.3 Pseudo Code

The Pseudo Code of our computational model will be as follows:

1. *Initialize a random population of some size.*
2. *Map each individual from population to its fitness value.*
3. *Select the best fittest individual from the population.*
4. *Bring variation in the population using crossover and mutation operator.*
5. *Check if we have landed at our desired fitness then stops otherwise sent the descendent individual for the evaluation of fitness and repeat the process.*

5.4 Flow Chart of designed Computational Model

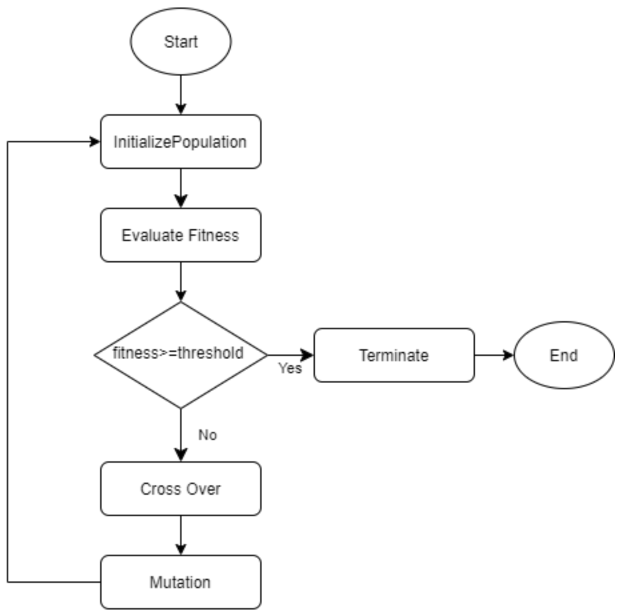

Figure 5: Flow Chart of Model

6 Application

In this section of report we will be discussing the actual problem which was given to us and describe the overall procedure how we tackle the overall problem and how we design the program using the model defined above which help us in solving the template matching problem.

We were given two images "groupGray.jpg" and "boothiGray.jpg" with dimensions (512,1024) and (35,29) respectively. We treated the first image as our source image and second image was treated as template image. We used image module from **Matplotlib** library to read images and the function which is being used to read images id *imread()*. Both images were read and stored in a numpy array. There are total six functions which are used to implement the evolutionary algorithm which are as follow:

- **InitializePopulation:** This function will generate a random population of coordinates containing randon row and column.
- **Fitness Evaluation:** This function will consume the population generated in above function and map each coordinate on its fitness value.
- **Selection:** This function will get the population and its mapped fitness value and return the individuals who will survive and reproduce for next generation.
- **CrossOver:** This function will bring variation in the population using crossover operator and return the reproduced individuals.
- **Mutation:** This function will bring variation in the reproduced descendants but the variation rate in this function is only 0.02 and we only flip a single bit while mutating.
- **Termination:** This function will determines whether the stopping criteria is fulfilled or not which in our case is as follow:
    1. When we reached our threshold value which in our case is 0.85.
    2. When the number of generation cross 10000.

7 Experiments

After translating the function described above in code, I started to do experiments. The population size for each population is 100 and total number of generations are 10,000 and the threshold value or the stopping values for our program is 0.85. Whenever the fitness value touches or crosses the threshold value our loop will be terminated.

7.1 Experiment 1:

In experiment 1, the mutation and cross over rate is 1 for each member of population during each generation. The result is as follow:

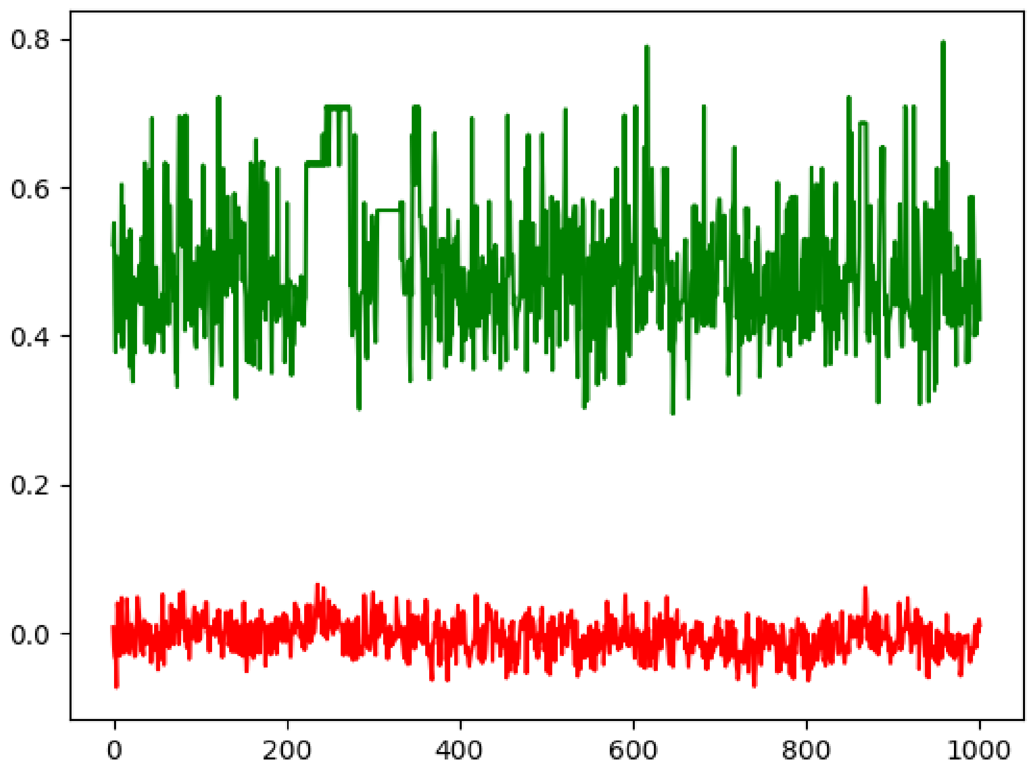

Figure 6: Red=Average Fitness, Green=Max Fitness

The average and the max fitness both graphs are in random behavior there is no linear behavior in any graph.

7.2 Experiment 2:

In experiment 2, the mutation and cross over rate is still 1 for each member of population during each generation but I selected the best fittest in each generation and without applying crossover and mutation passed it to next generation. The result is as follow:

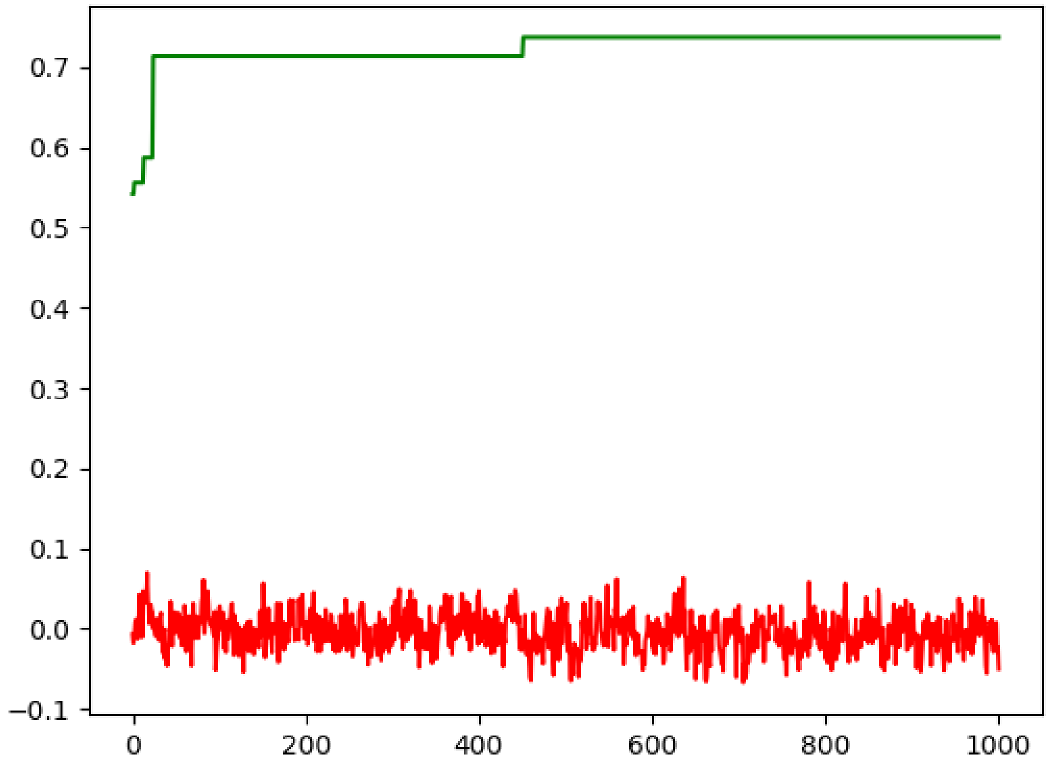

Figure 7: Red=Average Fitness, Green=Max Fitness

The max fitness is now in linear behavior and increasing with time but the average fitness graph is still not increasing with time. But we can see that still our max fitness is not touching the threshold value.

7.3 Experiment 3:

In experiment 3, I introduced the factor of probability for mutation which is 0.02 but cross over rate is still 1 for each member of population during each generation. And I selected the best fittest in each generation and without applying crossover and mutation passed it to next generation. The result is as follow:

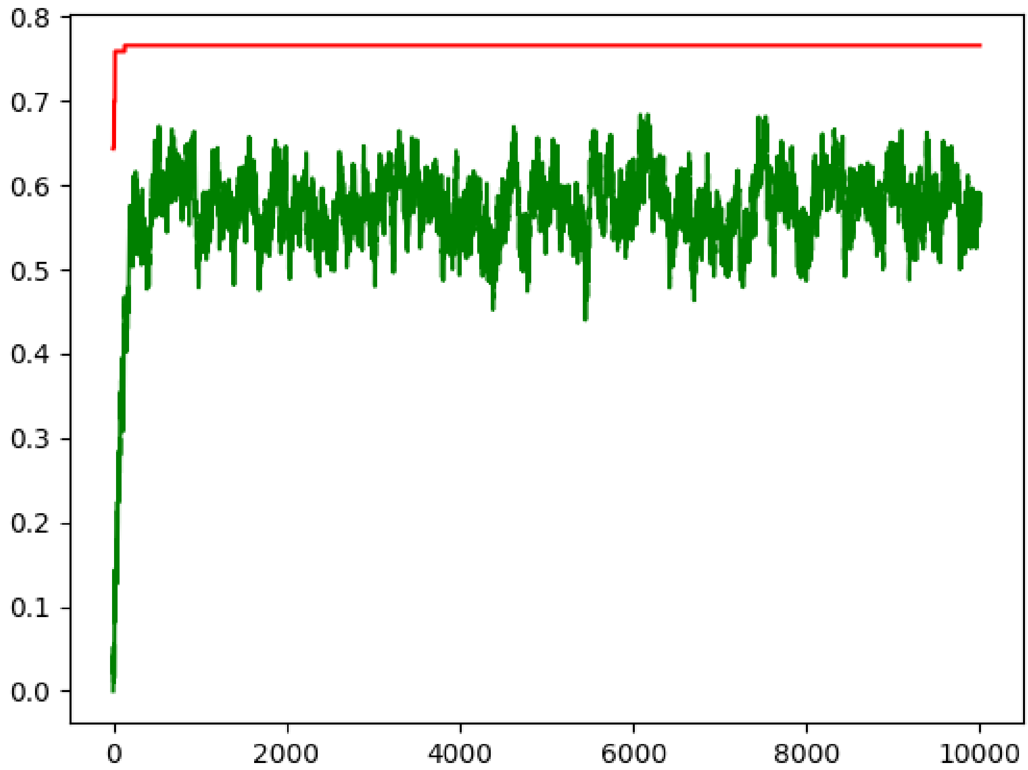

Figure 8: Red=Average Fitness, Green=Max Fitness

The max fitness is in linear behavior and increasing with time also the average fitness graph is increasing with time and increase the probability that we will found the smaller image in bigger image.

7.4 Experiment 4:

In experiment 4, I just use cross over and there is no mutation rate for each member of population during each generation. The result is as follow:

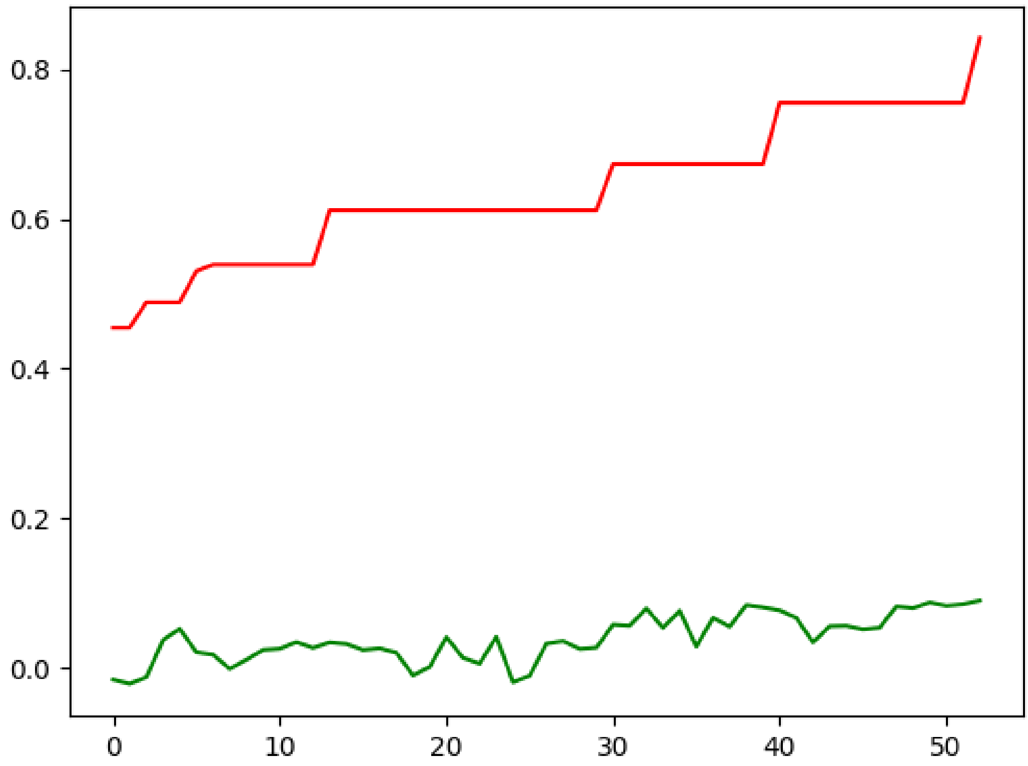

Figure 9: Red=Average Fitness, Green=Max Fitness

The max fitness is in linear behavior and increasing with time also the average fitness graph is increasing with time and increase the probability that we will found the smaller image in bigger image.

8 Conclusion

We used evolutionary algorithm to solve template matching problem.Through experiments we come to know that mutation is a major factor which determine the fitness value. Because the crossover bring minor changes in while doing variation but mutation brings higher changes. Moreover, the project helped us in understanding the underlying concept of how scientist solve problems using natural phenomenons. Here are the sample images and the outcome we received after running our code.

Figure 10: Original Large Image

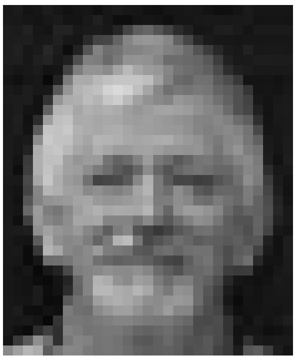

Figure 11: Small Template Image

Figure 12: Output Result of the Code

9 Code

```
import matplotlib.pyplot as plt
import matplotlib.image as img
import numpy as np
import matplotlib.patches as patches
from numpy.core.numeric import binary_repr
import random


def flip(prob):
    check_flip = np.random.binomial(n=1, p= prob)
    if check_flip==1:
        return True
```

```
    else:
        return False

def show_output(image1,population):
    fig, ax = plt.subplots()
    plt.gray()
    ax.imshow(image1)
    for i in population:
        print(i)
        rect = patches.Rectangle(i, 29, 35, linewidth=1, edgecolor='r', facecolor='no
        ax.add_patch(rect)
    plt.show()



def initialize_pop(row,column,sizeofPop):
    population = [(np.random.randint(0,row), np.random.randint(0,column)) for _ in ra
    return population

def correlation_coefficient(img1, img2):
    numerator = np.mean((img1 - img1.mean()) * (img2 - img2.mean()))
    denominator = img1.std() * img2.std()
    if denominator == 0:
        return 0
    else:
        result = numerator / denominator
        return result

def fittness_check(image1,image2,population):
    fitness_values = []
    for i in population:
```

```
        x = i[0]
        y= i[1]
        image3 = image1[y:y+35,x:x+29]
        corVal = correlation_coefficient(image3,image2)
        fitness_values.append(corVal)
    return fitness_values

def selection(fitness_values,population):
    combine = zip(fitness_values,population)
    combine = sorted(combine,reverse=True)
    ranked_pop = [x for y,x in combine]
    avg_max_fitness = [np.average(fitness_values),combine[0][0]]
    ranked_pop[-1] = ranked_pop[0]
    return ranked_pop,avg_max_fitness

def cross_over(ranked_pop):
    for i in range(1,len(ranked_pop)-1,2):
        first_x = binary_repr(ranked_pop[i][0],10)
        first_y = binary_repr(ranked_pop[i][1],9)
        first = str(first_x)+str(first_y)
        second_x = binary_repr(ranked_pop[i+1][0],10)
        second_y = binary_repr(ranked_pop[i+1][1],9)
        second = str(second_x)+str(second_y)
        point = random.randint(0,19)
        first = list(first)
        second = list(second)
        for index in range(point,len(first)-1):
            first[index],second[index] = second[index],first[index]
        first = "".join(first)
        second = "".join(second)
```

```
        first_x = int(first[0:10],2)
        first_y = int(first[10:],2)
        second_x = int(second[0:10],2)
        second_y = int(second[10:],2)
        ranked_pop[i] = (first_x,first_y)
        ranked_pop[i+1] = (second_x,second_y)

    cross_pop = ranked_pop
    return cross_pop

def mutation(cross_pop):
    for i in range(len(cross_pop)):
        is_run = flip(0.02)
        if is_run or cross_pop[i][0]>995:
            while True:
                pop_x = str(binary_repr(cross_pop[i][0]))
                pop_x = list(pop_x)
                point = random.randint(0,len(pop_x)-1)
                if pop_x[point] =='0':
                    pop_x[point] = '1'
                else:
                    pop_x[point] = '0'
                pop_x = ''.join(pop_x)
                pop_x = int(pop_x,2)
                cross_pop[i] = (pop_x,cross_pop[i][1])
                if pop_x < 996:
                    break
        if is_run or cross_pop[i][1]>477:
            while True:
                pop_y = str(binary_repr(cross_pop[i][1]))
```

```
                pop_y = list(pop_y)
                point = random.randint(0,len(pop_y)-1)
                if pop_y[point] =='0':
                    pop_y[point] = '1'
                else:
                    pop_y[point] = '0'
                pop_y = ''.join(pop_y)
                pop_y = int(pop_y,2)
                cross_pop[i] = (cross_pop[i][0],pop_y)
                if pop_y < 478:
                    break
    population = cross_pop
    return population

def check_termination(fitness_values,population):
    fitness_results = sorted(zip(fitness_values,population),reverse=True)
    match = []
    for val in fitness_results:
        if val[0] >= 0.83:
            match.append(val[1])
    if len(match)>0:
        return match
    else:
        return match



run_count = 0
best_fitness = 0
max_fitness = []
```

```
avg_fitness = []
fitness_match = []
im1 = np.array(img.imread("D:\Semester 5\AI\Course\groupgray.jpg"))
im2 = np.array(img.imread("D:\Semester 5\AI\Course\\boothigray.jpg"))
pop = initialize_pop(995,477,100)
while(True):
    fitness_values = fittness_check(im1,im2,pop)
    if run_count<=10000 and len(fitness_match) ==0:
        result = check_termination(fitness_values,pop)
        fitness_match.extend(result)
    else:
        show_output(im1,fitness_match)
        break
    ranked_pop,min_max = selection(fitness_values,pop)
    avg_fitness.append(min_max[0])
    max_fitness.append(min_max[1])
    evolved_pop = cross_over(ranked_pop)
    pop = mutation(evolved_pop)
    run_count += 1

plt.figure(2)
plt.plot(max_fitness,'r')
plt.plot(avg_fitness,'g')
plt.show()
```

10 Reference

1. https://www.khanacademy.org/science/ap-biology/natural-selection/natural-selecti
a/darwin-evolution-natural-selection

2. https://www.nationalgeographic.org/encyclopedia/theory-evolution/

3. https://www.livescience.com/474-controversy-evolution-works.html

4. https://hips.hearstapps.com/hmg-prod.s3.amazonaws.com/images/four-or-the-species
jpg

5. https://www.yourgenome.org/facts/what-is-evolution