Artificial Intelligence

NAMAL INSTITUTE

Report: Template Matching Problem

Instructor: DR. JUNAID AKHTAR

Group Members: Muhammad Rehan Ali,
Syed Uzair Ul Hassan, Usama Ali
& Muhammad Musawar Baig

A) Natural Phenomenon:

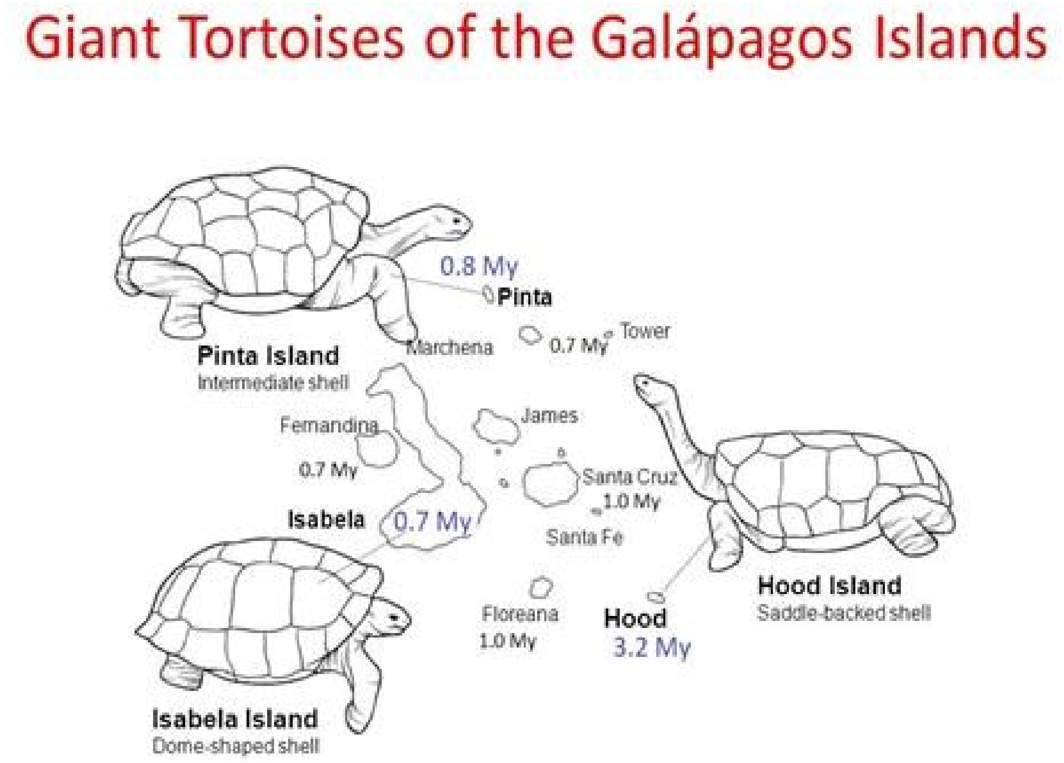
When we observe nature, We find out that nature is diverse and there is a variety of species. A huge number of species are on a single pattern. To name a few, We have 315 kinds of hummingbirds, 200 known species of Monkeys, 1000 types of bats, and at least 350,000 known species of beetles. Now, one might ask **Why is there so much diversity and variety in nature?**

B) Theory:

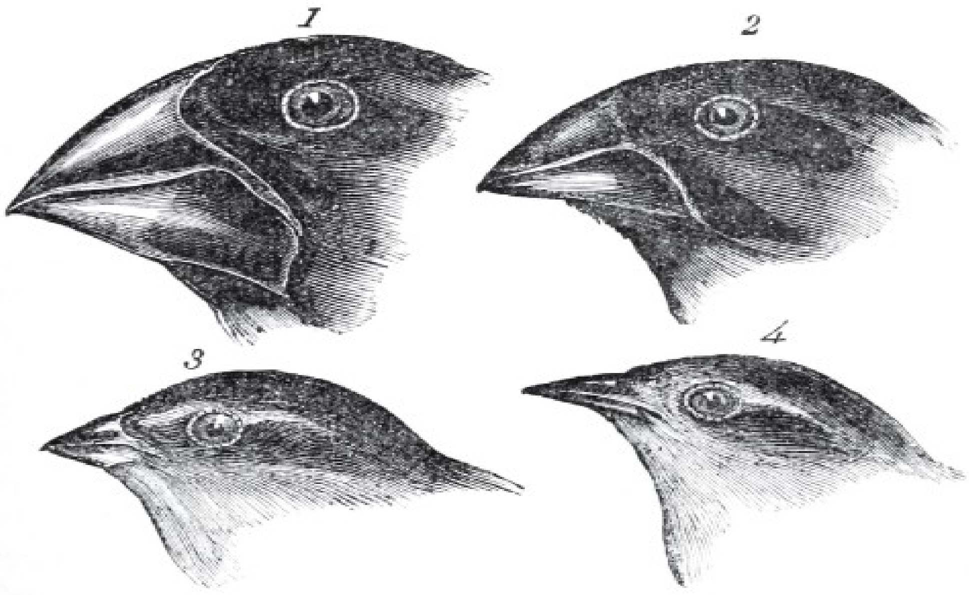
- ### Background:

Why there is so much diversity and variety in nature? Charles Darwin has provided an answer to this question. When in 1831 Darvin was on the surveying ship Beagle, He observed dazzling wonders of nature.

Darwin claimed that he knew from the shape of the tortoise's shell, which island it had come from. If it had a rounded front, it came from a well-watered island where it fed on lush green plants whereas one from a drier island had a peek at the front, which enables it to reach higher vegetation.



Darwin also noticed interesting differences in the same kind of finches that he found in the Galapagos Islands. They were similar except for there beaks you can also observe this from the following figure:



1. Geospiza magnirostris
2. Geospiza fortis
3. Geospiza parvula
4. Certhidea olivacea

Finches from Galapagos Archipelago

Those finches whose survival depends on insects have small beaks and the ones whose survival depends on nuts, they have heavy beaks. The small beak is to eat insects and the big heavy beak is to crack nuts.

From the above example of finches and tortoises, We can say that the variations between different species can happen due to the environment to which they are exposed. It can happen in a long time that one species with gradual changes has turned itself into a very different form and we might compel it to even categorize it as a different species.

Darwin called this idea of gradual change “Evolution by Natural Selection”.

- **Conclusion:**

Evolution by Natural Selection is the process by which nature selects the individuals that are best suited for a particular environment. Natural Selection is the reason for diversity and the facinating variety of life on our planet Earth.

Natural Selection is responsible for small changes like the change in size, some minor characteristics or color, etc. The idea of these small changes is called "microevolution ". The term "macroevolution" which happens over the vast length of geological time is also attributed to Natural Selection that states small accumulated changes can create new species.

C) Computational Model(for template matching or face recognation):

The following are the steps that we will follow to solve our template matching problem:

1)Representation: Images are represented in the form of matrices and these matrices are filled with the number and these numbers correspond to all the individual pixels of the image.

2)Initial Population: We will randomly select a pre-defined number of points from our search space or the bigger image. All of these represent the location of our potential solutions in the bigger image. In the next step, we will be checking how good or bad is each of these solutions.

3)Fitness Function: Our fitness function will take three inputs(Bigger Image, Smaller Image, and the Initial Population). It will give us the fitness value. This value will tell us how fit is our potential solution by comparing it with the smaller image. The value will range between 0 to 1. 1 means 100% match and 0 means 0% or no match etc.

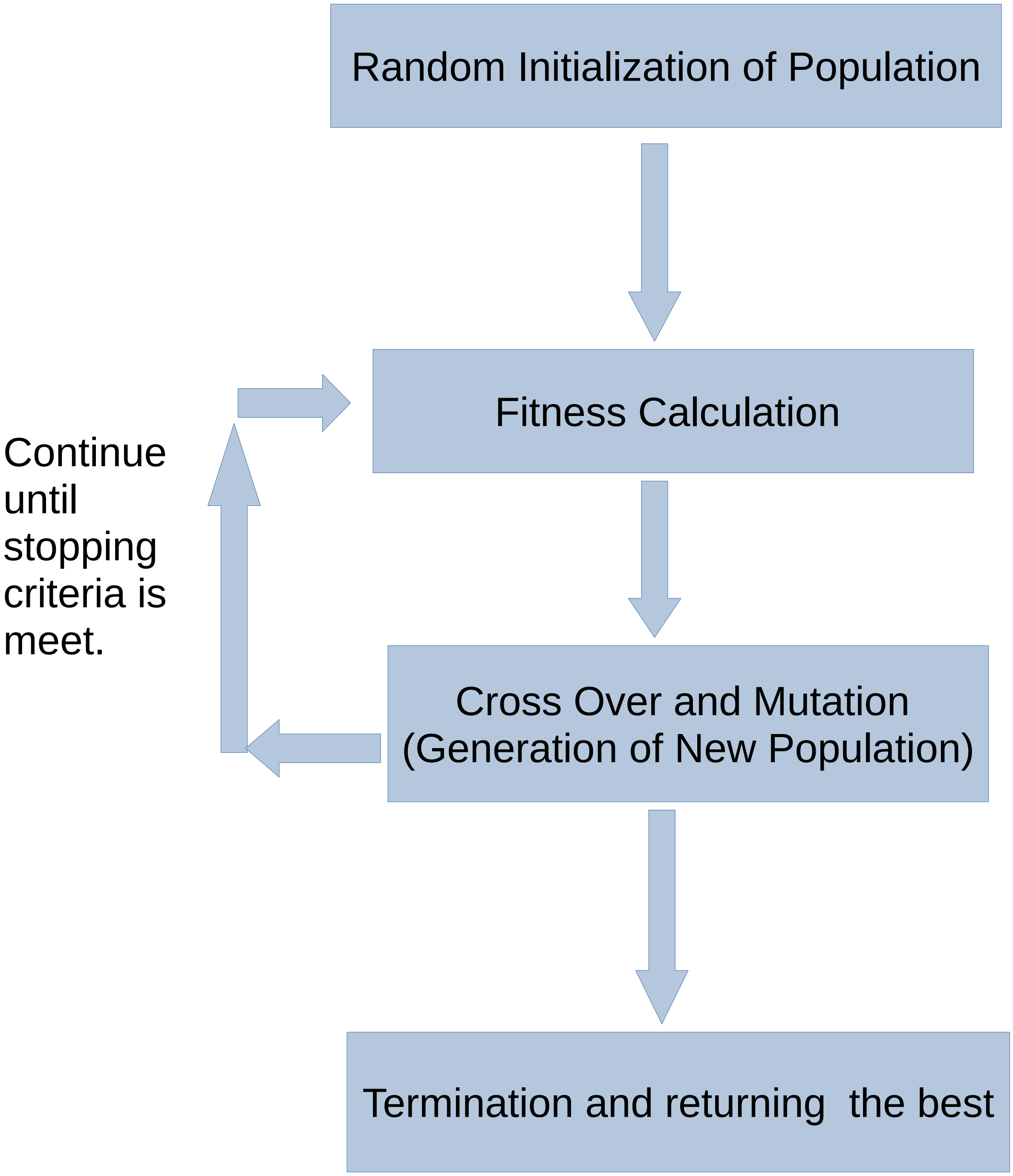
4)Sorting and Saving the Fittest Solution: We will sort our potential solutions based on their fitness value and save the fittest solution of the whole generation. Here we are artificially selecting the best fit and in the case of the theory of evolution, this was done naturally by Natural Selection.

5)CrossOver and Mutation: After saving the fittest solution from our sorted Array, We will cross the pairs in our population to create a new population. We will cross 1st solution and 2nd solution, then 2nd solution and 3rd solution,…, then lastly 2nd last solution with the last solution in our ranked population. Here a new population will be generated.

By doing CrossOver and Mutation our new population is actually being evolved from the older population. This same concept of evolution was given in our Evolutionary theory for living species.

6) After creating a new population, We will jump on Step 3 of fitness calculation and continue this process until a certain threshold is met. A threshold hold could be a matching percentage based on the output of fitness function or a particular number of generations.

The following figure represents the main stages of our Evolutionary Algorithm:


Random Initialization of Population
Fitness Calculation
Continue until stopping criteria is meet.
Cross Over and Mutation (Generation of New Population)
Termination and returning the best

The above are the main stages of our Evolutionary Algorithm.

D) Application

• Problem

We are supposed to solve an image processing or pattern recognition problem using our computational model. e.g, In our case, We are finding the location of a small image(Fig A) in the bigger image(Fig B). These images can be any other object or patterns like maps, tables, windows, and doors, etc.

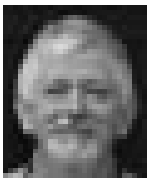


Figure A

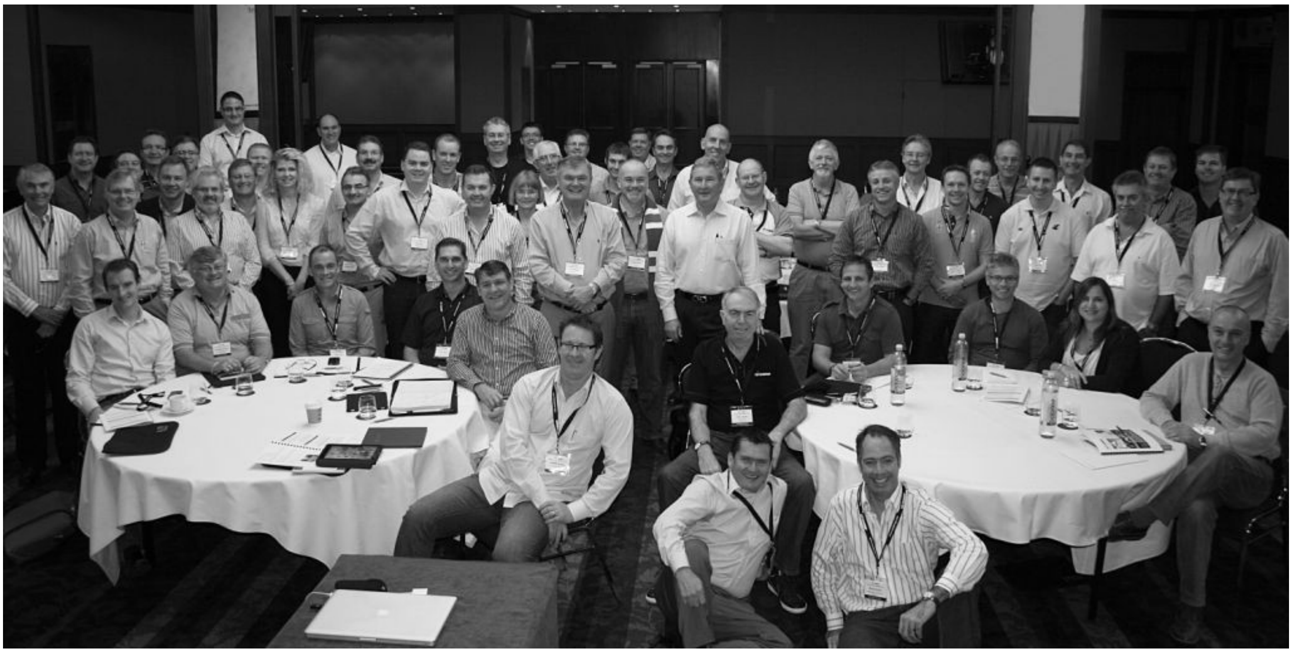
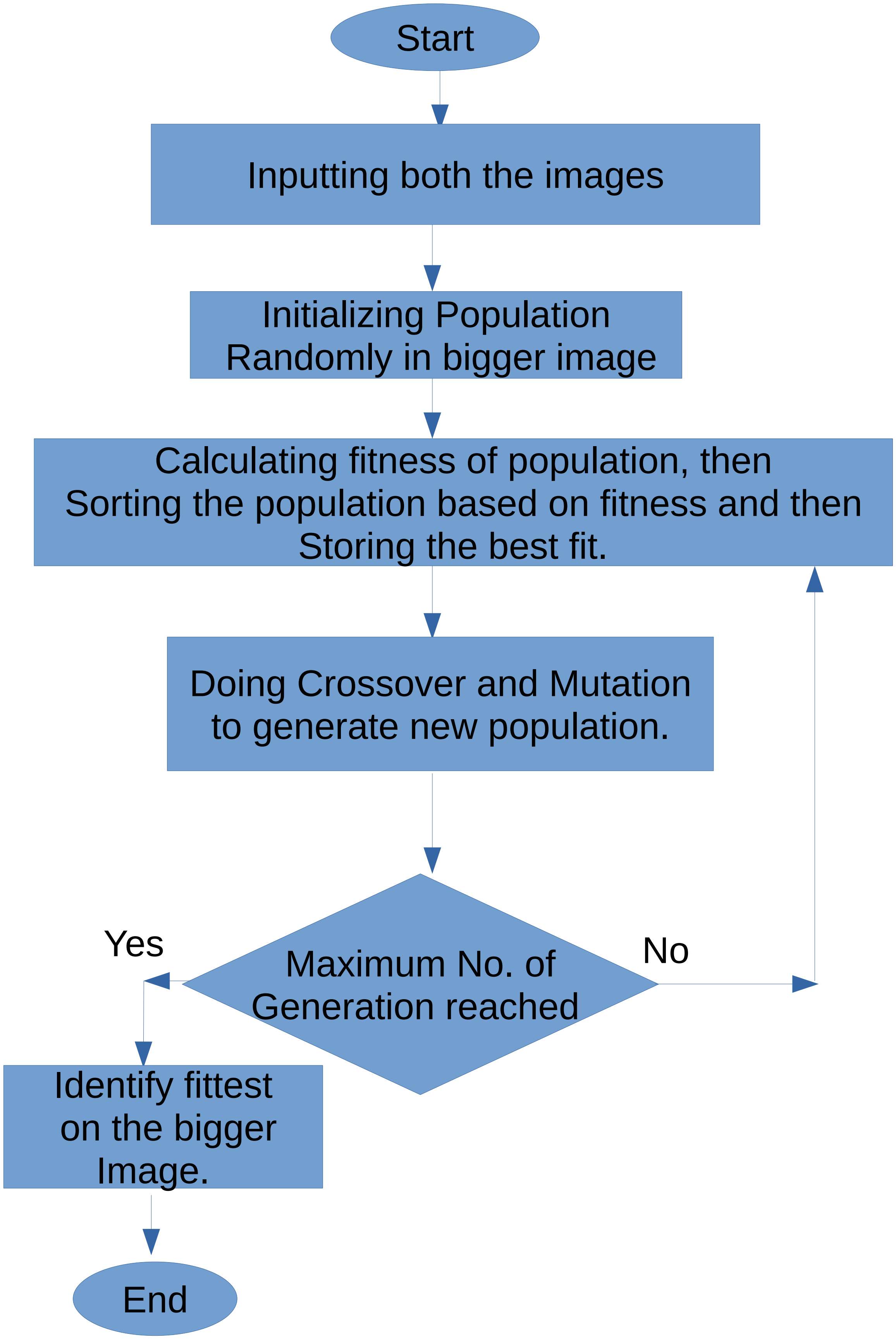


Figure B:

• Description

We have added descriptive comments in our code that can be analyzed in the **code.py** file to have a better understanding of our code. The following flowchart concisely represents the overall flow of our code:


Start
Inputting both the images
Initializing Population Randomly in bigger image
Calculating fitness of population, then Sorting the population based on fitness and then Storing the best fit.
Doing Crossover and Mutation to generate new population.
Maximum No. of Generation reached
Yes
No
Identify fittest on the bigger Image.
End

The above is the flowchart of our code to solve image recognition problem.

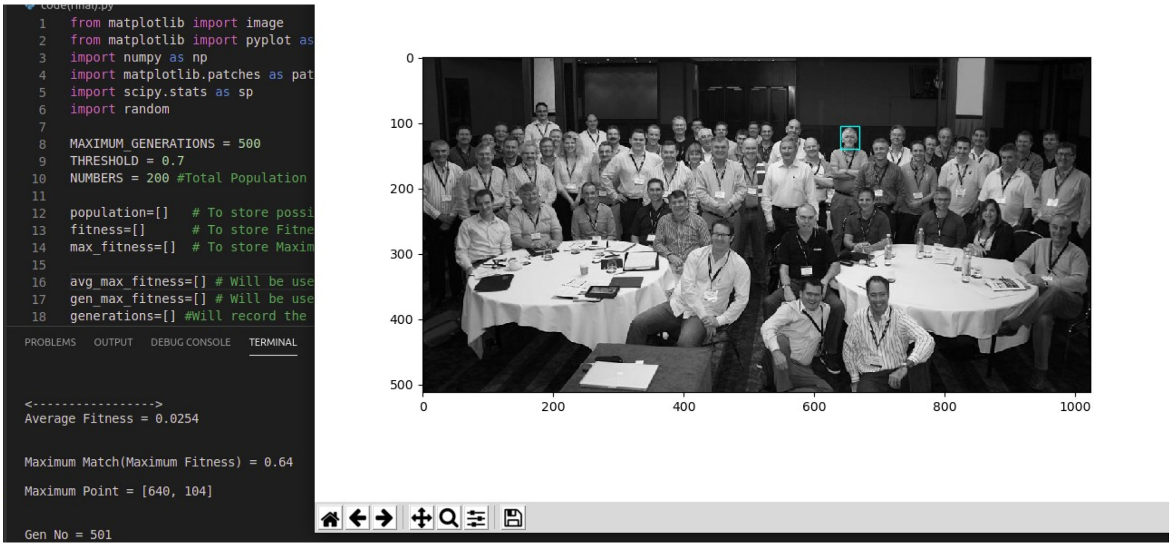
- **Results/Experiments/Observations:**

Note: Zoom to have a better look at the screen shots especially terminal and different variables of our code visible in the screen shots.

1. Observation-1:

We increase the chances of finding Baby G **early** by increasing the size of our Population. The following are the results of our code when we kept the number of Generations (**500**) and threshold(**0.7**) constant and the gradually increased the population size (200, 300, 400, 500):

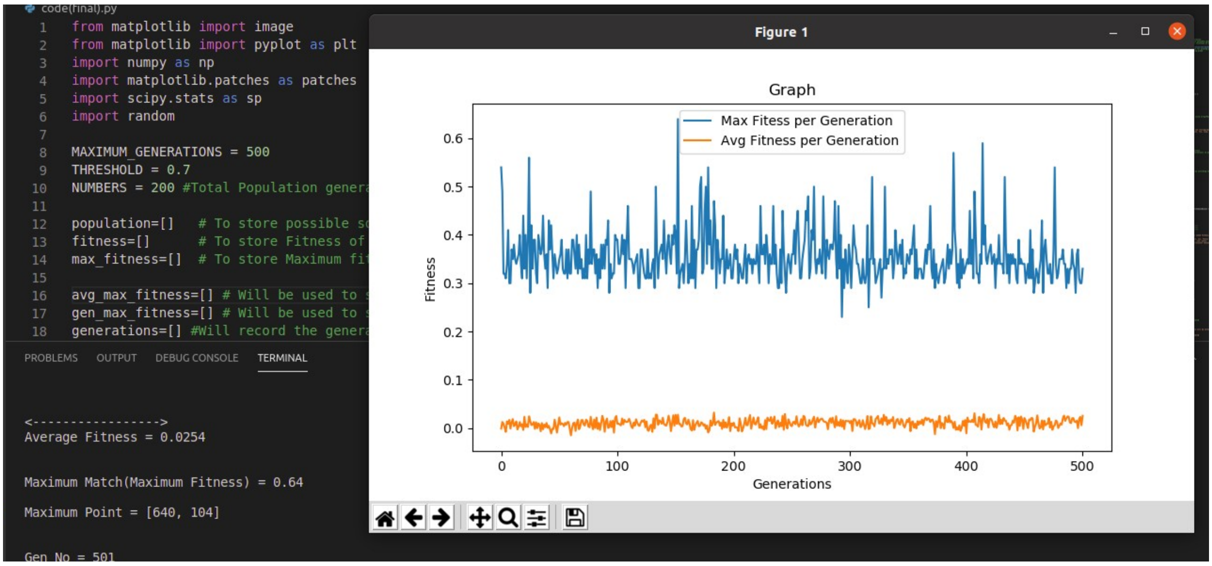
1. With population size **200**, It took all the **500** generation and we found Baba G with **0.64** fitness value.



code(final).py

```
from matplotlib import image
from matplotlib import pyplot as
import numpy as np
import matplotlib.patches as pat
import scipy.stats as sp
import random

MAXIMUM_GENERATIONS = 500
THRESHOLD = 0.7
NUMBERS = 200 #Total Population

population=[]   # To store possi
fitness=[]      # To store Fitne
max_fitness=[]  # To store Maxim

avg_max_fitness=[] # Will be use
gen_max_fitness=[] # Will be use
generations=[] #Will record the
```

```
<----------------->
Average Fitness = 0.0254


Maximum Match(Maximum Fitness) = 0.64

Maximum Point = [640, 104]


Gen No = 501
```

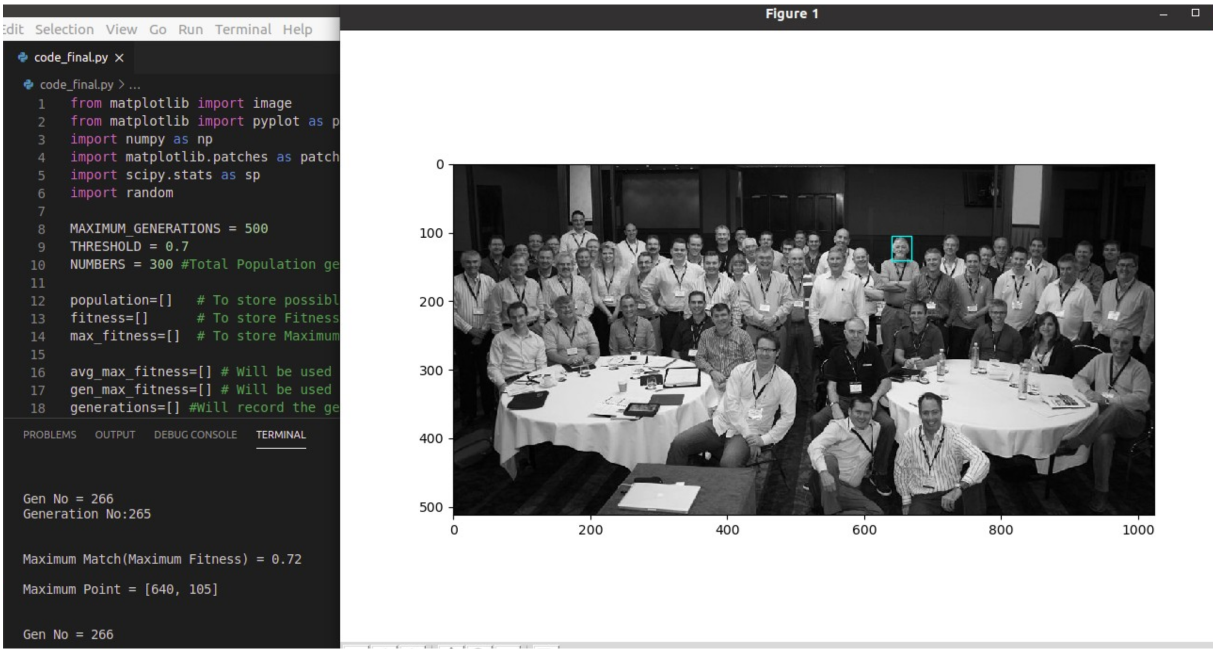


code(final).py

```
from matplotlib import image
from matplotlib import pyplot as plt
import numpy as np
import matplotlib.patches as patches
import scipy.stats as sp
import random

MAXIMUM_GENERATIONS = 500
THRESHOLD = 0.7
NUMBERS = 200 #Total Population genera

population=[]   # To store possible so
fitness=[]      # To store Fitness of
max_fitness=[]  # To store Maximum fit

avg_max_fitness=[] # Will be used to s
gen_max_fitness=[] # Will be used to s
generations=[] #Will record the genera
```

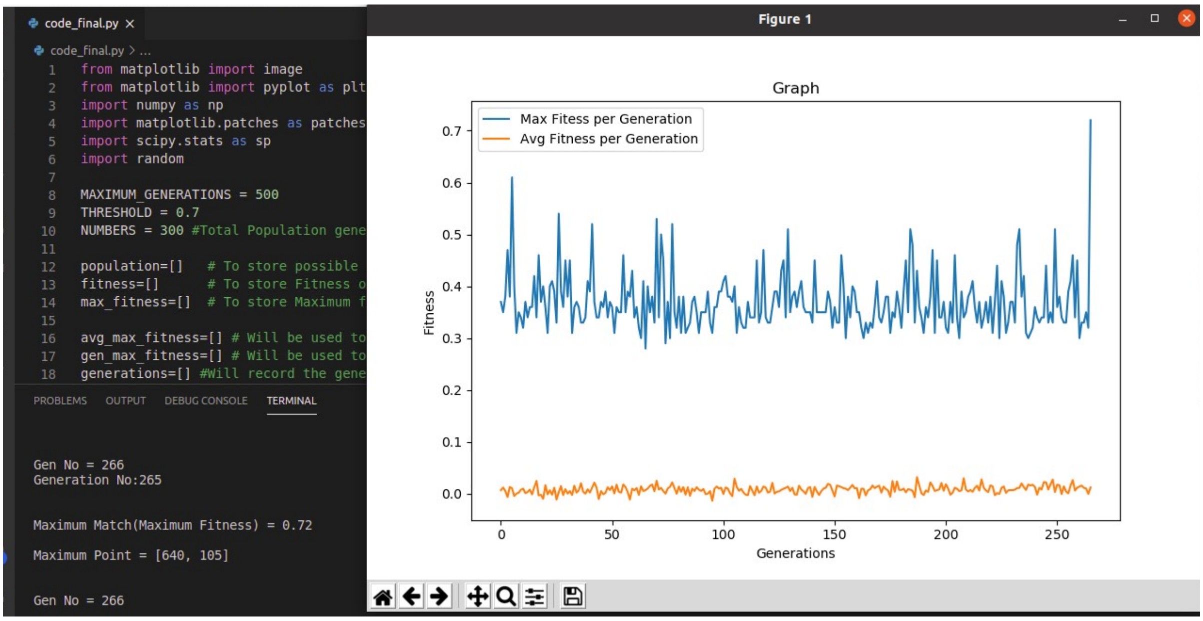
```
<----------------->
Average Fitness = 0.0254


Maximum Match(Maximum Fitness) = 0.64

Maximum Point = [640, 104]


Gen No = 501
```

2. With population size **300**, It took **265** generations and we found Baba G with **0.72** fitness value.


Edit Selection View Go Run Terminal Help
code_final.py ×
code_final.py > ...
Figure 1
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
0
100
200
300
400
500
0
200
400
600
800
1000

```
from matplotlib import image
from matplotlib import pyplot as p
import numpy as np
import matplotlib.patches as patch
import scipy.stats as sp
import random

MAXIMUM_GENERATIONS = 500
THRESHOLD = 0.7
NUMBERS = 300 #Total Population ge

population=[]   # To store possibl
fitness=[]      # To store Fitness
max_fitness=[]  # To store Maximum

avg_max_fitness=[] # Will be used
gen_max_fitness=[] # Will be used
generations=[] #Will record the ge
```

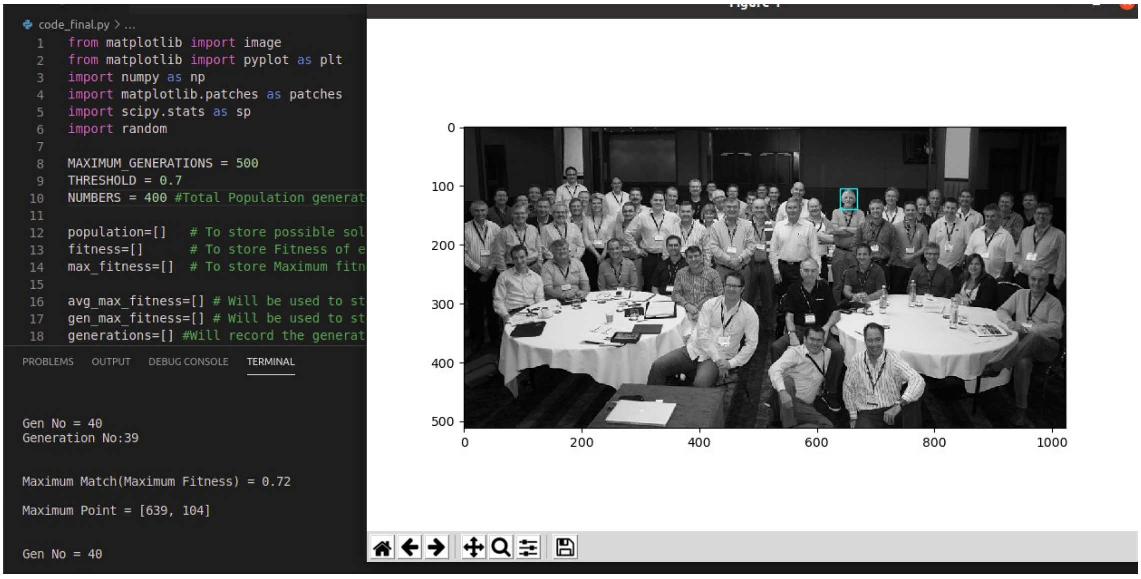
```
Gen No = 266
Generation No:265


Maximum Match(Maximum Fitness) = 0.72

Maximum Point = [640, 105]


Gen No = 266
```



```
from matplotlib import image
from matplotlib import pyplot as plt
import numpy as np
import matplotlib.patches as patches
import scipy.stats as sp
import random

MAXIMUM_GENERATIONS = 500
THRESHOLD = 0.7
NUMBERS = 300 #Total Population gene

population=[]   # To store possible
fitness=[]      # To store Fitness o
max_fitness=[]  # To store Maximum f

avg_max_fitness=[] # Will be used to
gen_max_fitness=[] # Will be used to
generations=[] #Will record the gene
```

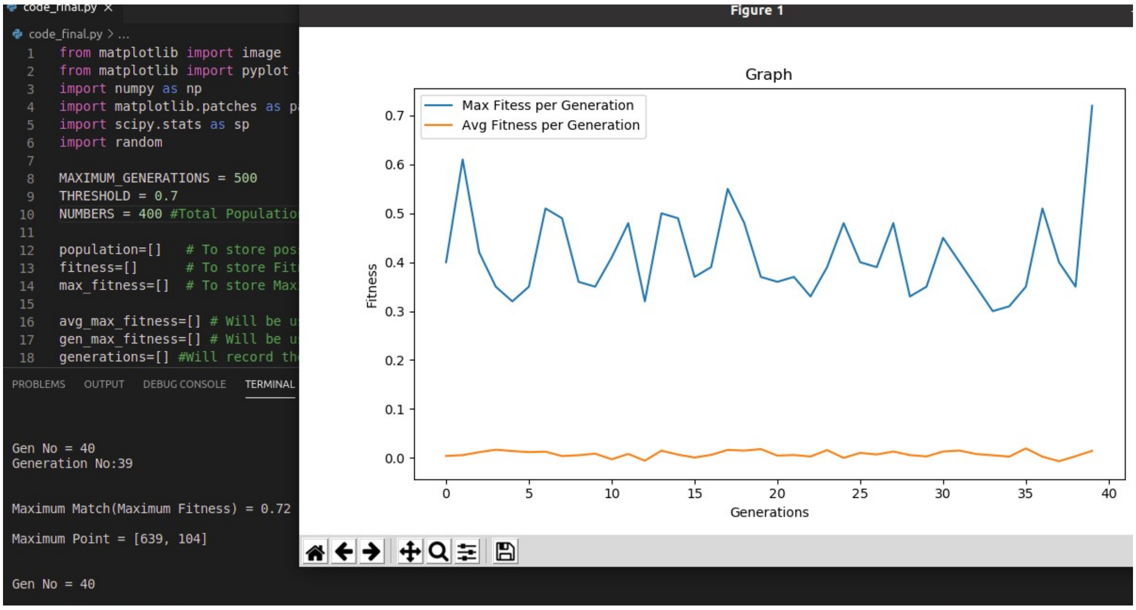
```
Gen No = 266
Generation No:265


Maximum Match(Maximum Fitness) = 0.72

Maximum Point = [640, 105]


Gen No = 266
```

3. With population size **400**, It just took **40** generations and we found Baba G with **0.72** fitness value.



```
from matplotlib import image
from matplotlib import pyplot as plt
import numpy as np
import matplotlib.patches as patches
import scipy.stats as sp
import random

MAXIMUM_GENERATIONS = 500
THRESHOLD = 0.7
NUMBERS = 400 #Total Population generat

population=[]   # To store possible sol
fitness=[]      # To store Fitness of e
max_fitness=[]  # To store Maximum fitn

avg_max_fitness=[] # Will be used to st
gen_max_fitness=[] # Will be used to st
generations=[] #Will record the generat
```

```
Gen No = 40
Generation No:39


Maximum Match(Maximum Fitness) = 0.72

Maximum Point = [639, 104]


Gen No = 40
```



```
from matplotlib import image
from matplotlib import pyplot
import numpy as np
import matplotlib.patches as p
import scipy.stats as sp
import random

MAXIMUM_GENERATIONS = 500
THRESHOLD = 0.7
NUMBERS = 400 #Total Populatio

population=[]   # To store pos
fitness=[]      # To store Fit
max_fitness=[]  # To store Max

avg_max_fitness=[] # Will be u
gen_max_fitness=[] # Will be u
generations=[] #Will record th
```

```
Gen No = 40
Generation No:39


Maximum Match(Maximum Fitness) = 0.72

Maximum Point = [639, 104]


Gen No = 40
```

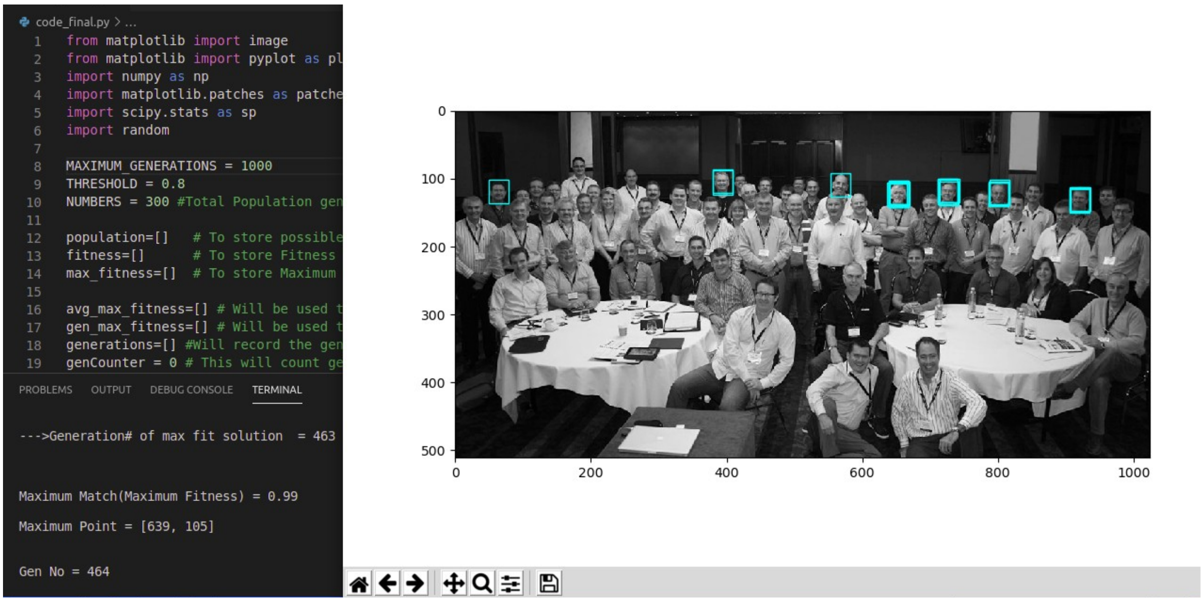
It is not necessary that the Baba G will be found earlier and earlier by increasing population size, there is slight randomness to it because in some experiments we have also observed that increasing population size didn’t make Baba G appear early but those were rare and this is the general trend that we have observed.

2. **Observation-2** (Build-in corelation function vs our self made corelation function)**:**

In the later part of our project, We switched to a built-in correlation function of Python because we found it more reliable than our(Our self-made correlation function can be found in the comments of the function named Correlation() in code.py file.). When we were using our function self-made correlation function, then our graphs were very chaotic, and our code was showing very random behavior. We were only finding baba G at 2 points randomly. The problem with our function was that we were comparing two images pixel by pixel, and then we were returning values between 0 – 1. The closer to 1, the better because 1 meant a complete match. We realized later on that with a slight change of location, pixels change values, and it is not possible to find Baba G the way we were doing it. This is the reason We preferred the built-in function over ours.

3. **Observation-3** (We are selecting from Multiple Solutions):

It is an obvious observation that we get multiple solutions and we are then selecting the best out of them. We had many solutions above the fitness value 0.5 fitness which are shown below:



```
from matplotlib import image
from matplotlib import pyplot as pl
import numpy as np
import matplotlib.patches as patche
import scipy.stats as sp
import random

MAXIMUM_GENERATIONS = 1000
THRESHOLD = 0.8
NUMBERS = 300 #Total Population gen

population=[]   # To store possible
fitness=[]      # To store Fitness
max_fitness=[]  # To store Maximum

avg_max_fitness=[] # Will be used t
gen_max_fitness=[] # Will be used t
generations=[] #Will record the gen
genCounter = 0 # This will count ge
```

```
--->Generation# of max fit solution  = 463


Maximum Match(Maximum Fitness) = 0.99

Maximum Point = [639, 105]


Gen No = 464
```

The highest fitness is at the point close to Baba G in the above figure with fitness **0.99,** as you can observe through the terminal shown in the above screen shot. So, We are selecting the best of the best.

4. **Observation-4** (Experimenting with changing crossover function):

We have done some experiments by changing our function named **crossover()** as well. You can observe our experiments in the comments of that function in the file named code.py. We have chosen to select the point of crossover randomly because that implementation was showing us more accurate and reliable results than others.

5. **Observation-5** (Minimum Population Size):

By fixing the threshold at 0.7 and maximum generation at 500, And by changing our population size, We found that at a minimum population of around 100 is finding Baba G correctly but with a relatively low fitness value of around 0.6 and it that's why it some times gives a wrong output. With a minimum population size of 200, We are having better maximum fitness values of around 0.7 for Baba G and it is showing better and more consistent results.

6. Observation-6 (Population Size vs no of Generations):

By increasing population size, the number of generations decreases(means Baba comes early if we use a threshold of around 0.7) in most cases, and vice versa.

7. Observation-7 (Increasing Population Size and Time Consumption):

Our Algorithm is consuming more time by increasing Population Size while fixing number of Generations and fitness threshold because it had to do more computations per generation.