AI Assignment № 2

Danat Ayazbayev, BS-18-03 April 26, 2020

What is representation of algorithm?

Step 1

My algorithm makes initial population by creating 10 images. Every image consists from single colour. Here are the colours that I chose:
Brown 🟫 RGB (165,42,42)
Red 🟥 RGB (255,0,0)
Green 🟩 RGB (0,128,0)
Purple 🟪 RGB (128,0,128)
Pink ◻ RGB (255,192,203)
Black ⬛ RGB (0,0,0)
Yellow 🟨 RGB (255,255,0)
Orange 🟧 RGB (255,165,0)
White [As this document's blank page colour] RGB (255,255,255)
Blue 🟦 RGB (0,0,255)

Step 2

It runs an infinite loop, where following operations are executed:

Step 2.1

Fitness function for current population is calculated.

Step 2.2

N best images being selected. They will be parents for the next population.

Step 2.3

Crossover is performed.

Step 2.4

Mutation is performed.

Step 2.5

After **Step 2.4** algorithm produced 10 new images, they are considered to be new generation and replace old generation, which is also 10 images.
Then algorithm goes back to **Step 2.1**.

Which selection mechanism is being used?

Let's consider in details **Step 2.2** from previous section.
Before selecting parents, algorithm sorts current population members by their decreasing order of fitness scores. Then it chooses first N parents, that means best suitable N parents for producing new population. By the way, I chose $N = 2$.

Before aforementioned approach, I tried another selection mechanism, but decided to change it due to some disadvantages, which I will describe later. First of all, let's consider the idea behind it. It was probabilistic selection of parents. In other words, after algorithm calculated fitness scores of current population members, it assigned every member a probability being a parent for the next generation based on the fitness score that member had. Probability is calculated by following way:

1st step

Let sum be the sum of all fitness scores of population members. sum is calculated.

2nd step

Let p_i be the probability for i-th population member of being parent for next generation, and f_i be fitness score of i-th member. Algorithm calculates p_i by the formula $p_i = \frac{f_i}{sum}$.

The major drawback in such algorithm is that fitness score does not grow, and even often degrades because after 5-10 generations, probabilities of all members for being a parent for the next generation will differ only by 5-10%. Thus, usually not the best parents will be selected for next generation.

Which image manipulation techniques was applied?

To start, algorithm was written in $python$ $3.8.0$ with pip $20.0.2$ version. For image manipulations, only the $Image$ class from the $pillow$ library was used.
Before every other operations, reference image name is hard-coded into the code. After that, with the help of library, algorithm resizes reference image to 512x512 pixel image. Only then, initial population is created.

Describing another manipulation, before algorithm creates initial population, array of aforementioned colours' named is also hard-coded into the code. Then, again with the help of library, initial population images are created by taking as an arguments the colour names and path. By the way, during evolution, images are accumulated. For example, 0-th generation from $image0.png$ till $image9.png$, 1-st generation from $image10.png$ till $image19.png$, x-th generation from $image\{10x\}.png$ till $image\{10x+9\}.png$.
And another one is when after crossover children are being mutated, algorithm gets children's 2-dimensional array of pixels and change them according to the mutation function.

What is the fitness function?

Before my current fitness function, I tested the one which is simpler. Let's describe the formula for it.
$R1_{ij}$=RGB Red value of pixel at position i, j of reference image.
$G1_{ij}$=RGB Green value of pixel at position i, j of reference image.
$B1_{ij}$=RGB Blue value of pixel at position i, j of reference image.
$R2_{ij}$=RGB Red value of pixel at position i, j of population member image.
$G2_{ij}$=RGB Green value of pixel at position i, j of population member image.
$B2_{ij}$=RGB Blue value of pixel at position i, j of of population member image.
D_{ij}="Pixel difference" value between reference image's and population member image's pixels at position i, j.
D_{ij}=$\sqrt{(R1_{ij}-R2_{ij})^2+(G1_{ij}-G2_{ij})^2+(B1_{ij}-B2_{ij})^2}$
H=image height in pixels
W=image width in pixels
F=Total fitness score, the bigger score - the more suitable the population member image is.

$$F=-\sum_{i=0}^{H-1}\sum_{j=0}^{W-1}D_{ij}$$

Minus is because the smaller the value of sum itself without sign, the more the population member image is suitable. With minus situation is vice versa.

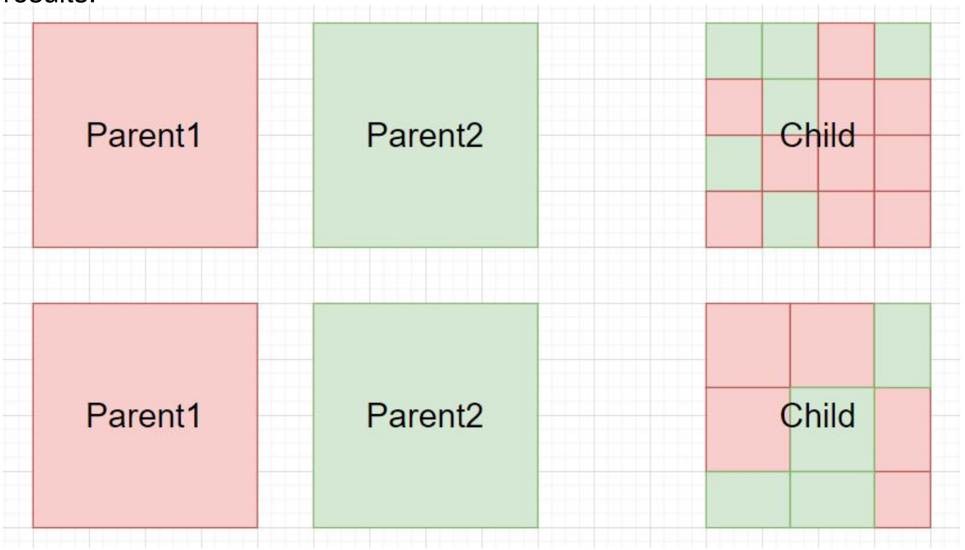
Now let's talk about the actual fitness function that I used in the algorithm. Actually, I modified aforementioned fitness function. I used normal distribution with x as D_{ij} and with mean as 0, and found more suitable to take variance (σ) as 30. I used it because I think it is more fair that overall, very little differences between pixels make population member image more suitable and similar to reference image, but huge differences is very bad. And I think that exponential scaling is better here. Now, formula is as follows:

$$F = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{D_{ij}}{\sigma})^2}$$

Again, the more is fitness score - the more suitable is population member image.

Crossover function

As mentioned before, after algorithm chose parents according to fitness scores, it performs crossover between that parents and produce 10 new child images. With each child, first of all, it chooses variable $square\_size$ randomly, such that $min\_square\_size \leq square\_size \leq max\_square\_size$. Range boundaries are hard-coded values 3 and 30, correspondingly. Then it takes child's back to back square blocks of pixels, then puts random parent's corresponding pixels into child's pixels. To make it more clear, let's take some examples of possible crossover results:



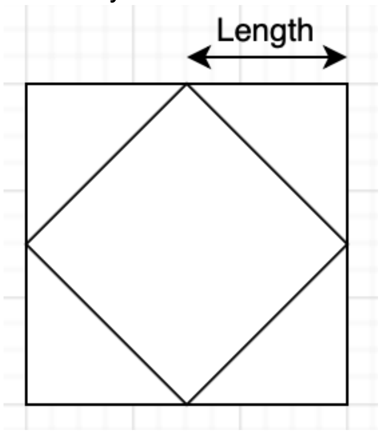
As it can be seen from the picture, when image height or width value in pixels is not divisible by $square\_size$, squares are cut near the image border.

Mutation function

After crossover is finished, algorithm performs mutation on children. I configured it such that it does mutation only on 8 children from 10. This decision was made because mutations have big probability to decrease fitness scores of images. When 2 children's pixels remain almost the same as parents' pixels, fitness will definitely not degrade, because in the worst case, when all 8 other children performed bad mutation, population will be reproduced by aforementioned 2 not mutated children.

Let's describe the process of mutation. Firstly, algorithm chooses to draw square or rhombus randomly. Square ■ has sides parallel to the image frame. Rhombus ◆ is just a square which

is rotated by 45 degrees.
Now let's talk about how figure size is chosen. Size for square is the square side length divided by 2, and for the rhombus is the distance from the center to the sharp corner. I hard-coded minimum and maximum possible figure sizes as 3 and 30. And algorithm chooses figure size randomly.



The only thing left to describe is into what colour figure is painted. In the code there are another hard-coded information - the array of disturbances which consists of 10 numbers. Before drawing the figure onto the child image, algorithm chooses random pixel from that child image and takes its colour. To say in more details - algorithm takes its **RGB** values. Then algorithm creates new colour based on chosen pixel's colour, by changing chosen colour's each **RGB** value. Here aforementioned array of disturbances helps, let's say for i-th child we have $disturbance_i$. For each **RGB** value of i-th child a random number from $-disturbance_i$ to $disturbance_i$ is added. Then a figure is painted with a new colour and put at random place in the child image. Position is determined by randomly chosen coordinates of figure's center. Center is chosen such that figure fits into the image frame. After last child image is mutated, old population is replaced by a new one.
I chose all disturbances to be 16 except for children images which are never mutated. I consider this value ideal because if I will take less, population will evolve even slower than now, and if I will take more, population will also evolve slower, because there are big chance that colour chosen for the figure will be far away from that which would be suitable.

What is art for me?

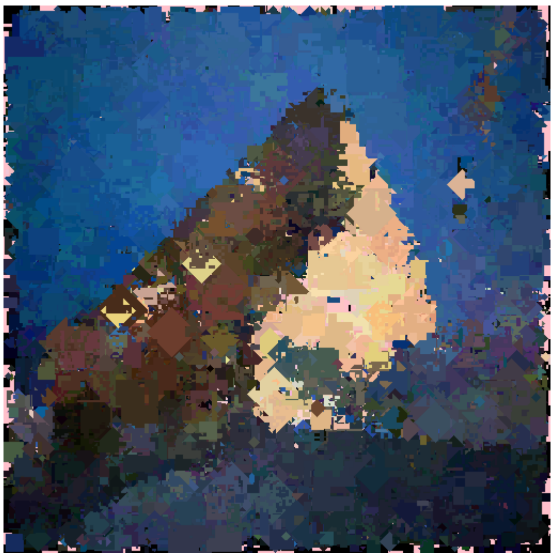
I think that art is created when creator puts his/her soul into his/her work. This often happens when he/she did not think of the idea on which the work is based before, and very interested in finishing it and seeing the results. Creator wants to be excited at the end, so he/she performs carefully and do everything "from the heart".
I think that art is relative notion. For example, professional painter may not consider one of his/her work as art, because he/she did very little mistake, which is visible only to his/her eyes. But if some random person will look to his/her work, that person most probably will be very excited and call the work "art".

Artistic aspect of my output images

Let’s refer to the previous section. I think that output images of my algorithm are works of art because for me it is so. I did not think about evolutionary algorithms before, and I became interested in the assignment. I spent a lot of time on implementing the idea which may seem primitive to others, but I am satisfied with the results of my work.

Examples of images

Innopolis University



Soviet tank T-34-85