Introduction to Artificial Intelligence
Assignment 2

Done by:
Lukyanchikova Elena, BS17-02,
e.lukyanchikova@innopolis.ru

project: github.com/elukyanchikova/picture_generation_gen_alg

1. Algorithm used

To generate images as were stated in the assignment task a Genetic algorithm was used. It works as follows: having an ideal picture algorithm tries to generate from scratch(totally black picture) a new one that will be like the ideal. To check this alikeness, if the newly generated image is better than the previous version of it, the fitness function that is hardcoded in source code file is used. “Better” means that the new picture is much similar to the image that was taken as ideal.

The new image is generated with circles. Thus, at every iteration (“generation”) 100 sets (in the current implementation - this number could be changed in source code and definitely will affect the output results) of (x,y) coordinates of the circle center, radius, and RGB are generated, than they are applied to the new picture. After this fitness function is calculated and, if it shows better fitness of the new picture, it is accepted as a new one and iteration continue. This method is called Mutation.

2. Method for using input image

The input image is used as an “ideal picture”. This means that the algorithm generates a new picture that is as similar as possible to the original image and calculates a fitness function value for it. If the newly generated version is more similar to the ideal picture than the previous generated, then the new image is chosen.

3. Fitness function

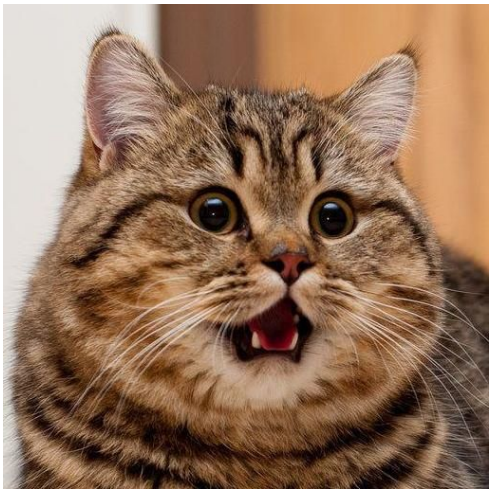
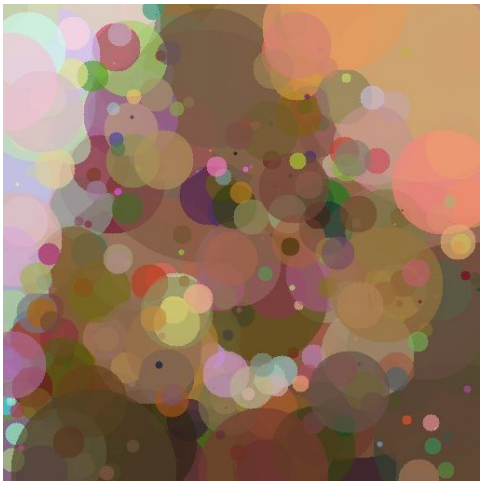
For calculating fitness of the image a least square method is used: introduced function is $(originalImage - newImage)^2$ that is calculated for each of the corresponding pixels in originalImage and newImage arrays and summed up to get the final value.

4. Input/Output results

1. First attempt

i.This cat was generated with 1000 iteration (it took about 40 minutes to finish). Fitness increased by 8 times :

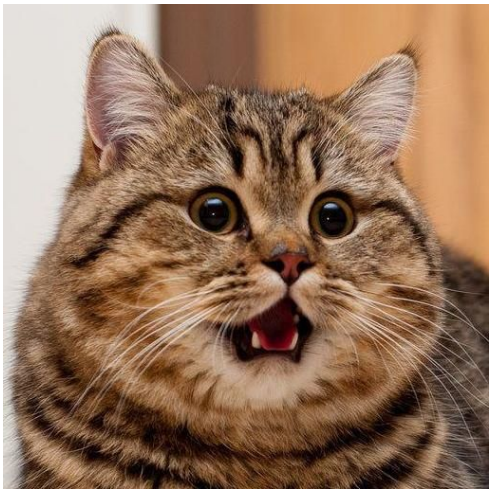
$$\frac{fitF_0}{fitF_{1000}} = \frac{19768587379.0}{2467874840.0} = 8$$

Original image 1000 generations

ii. When the number of iterations was increased - 3000, this cat appeared. Related to the origin black picture fitness increased by 9.8 times

$$\frac{fitF_0}{fitF_{1000}} = \frac{19768587379.0}{2016773986.0} = 9.8$$



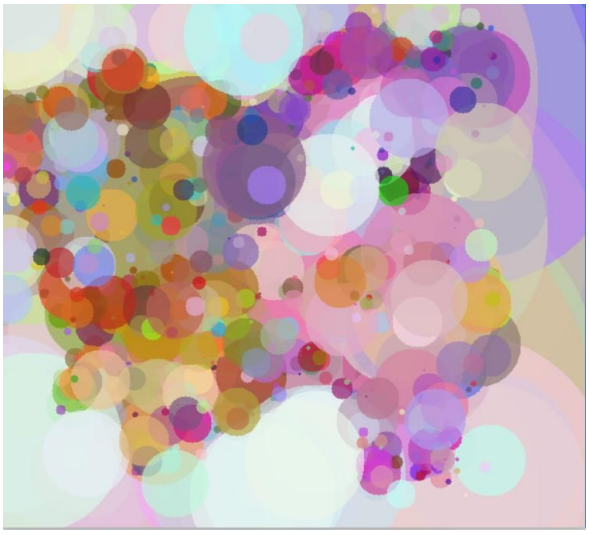

Original image 1000 generations

2. Second picture

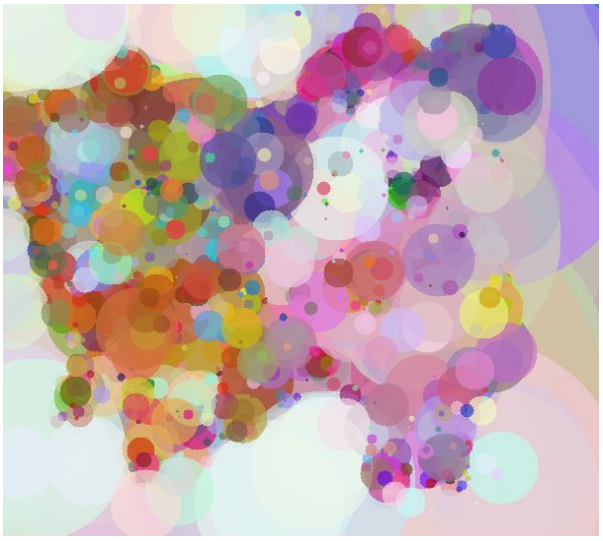
i. This one happens to be the most colorful of those I tested, perhaps, this could mean that the implemented algorithm works better for cartoons, drawn pictures, not photos.

Origin image

800 generations



1500 generations



3000 generations

Fitness increase for each of the stage (related to fitness of first, black picture):

$$fitInc_{800} = \frac{fitF_0}{fitF_{800}} = \frac{7061073357.0}{1035549515.0} = 6.8$$

$$fitInc_{1500} = \frac{fitF_0}{fitF_{1500}} = \frac{7061073357.0}{947609172.0} = 7.4$$

$$fitInc_{3000} = \frac{fitF_0}{fitF_{3000}} = \frac{7061073357.0}{855563792.0} = 8.2$$

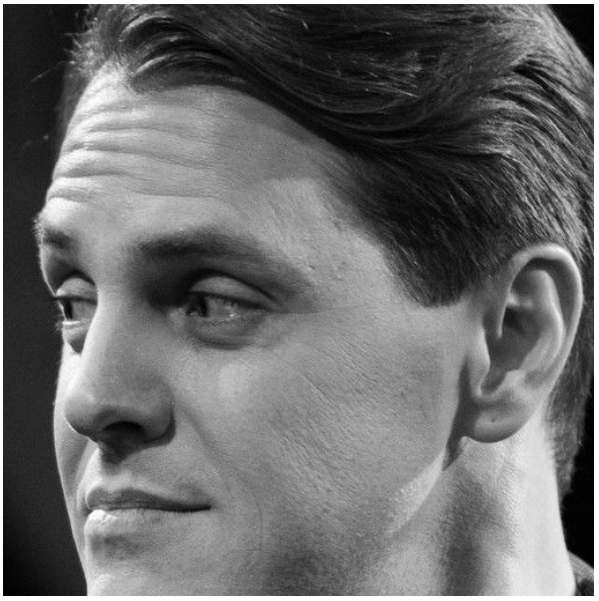
ii. In comparison to the first image of a cat this picture has worse relative fitness increase (9,8 and 8,2 correspondingly after 3000 iterations). However, it can be noticed, that:

	Initial fitness order	Absolute fitness order(3000 iterations)	Relative fitness increase
Pic.1(cat)	2×10^{10}	2×10^{9}	10
Pic.2	7×10^{9}	8×10^{8}	10

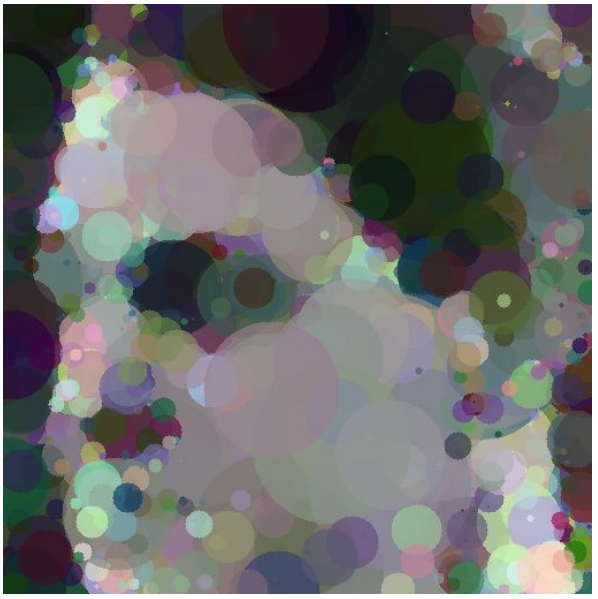
The order of Relative fitness increase is the same for both pictures, so, the difference in absolute values can be explained by different initial fitness function values: first generated picture of pic.1 fitted to the ideal 1 worse than first generated picture of pic.2 to its ideal.

3. Third picture - human photo

i. For the third test, a photo of a real human was chosen.



Orirginal image



3000 generations

The result of 17 times Relative fitness increase is surprising:

$$fitInc_{3000} = \frac{fitF_0}{fitF_{3000}} = \frac{11141651278.0}{641056733.0} = 17.3$$

ii. This gives an interesting idea: if the picture is black-white, perhaps, it is reasonable to choose only one channel of RGB and mutate on it, setting each color

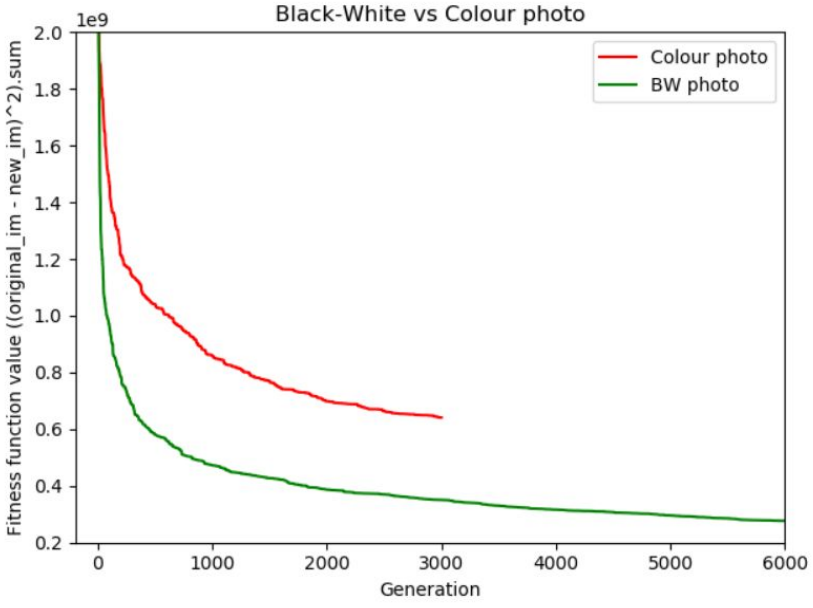
of a pixel to a generated value. The expected result is the increase in speed of algorithm converges.

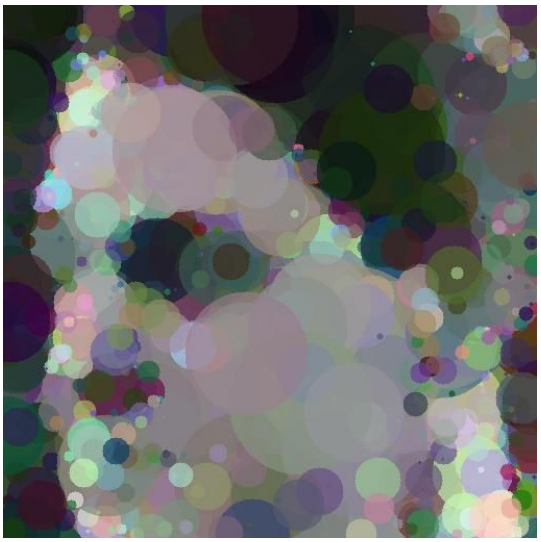
iii. The algorithm was modified (second source code file) and the result is as follows:

1000 generations 2000 generations 3000 generations

4000 generations 5000 generations 6000 generations

The assumption happen to be correct - generated image quality increases almost by 2 times if the mutation is done over one of RGB channel.



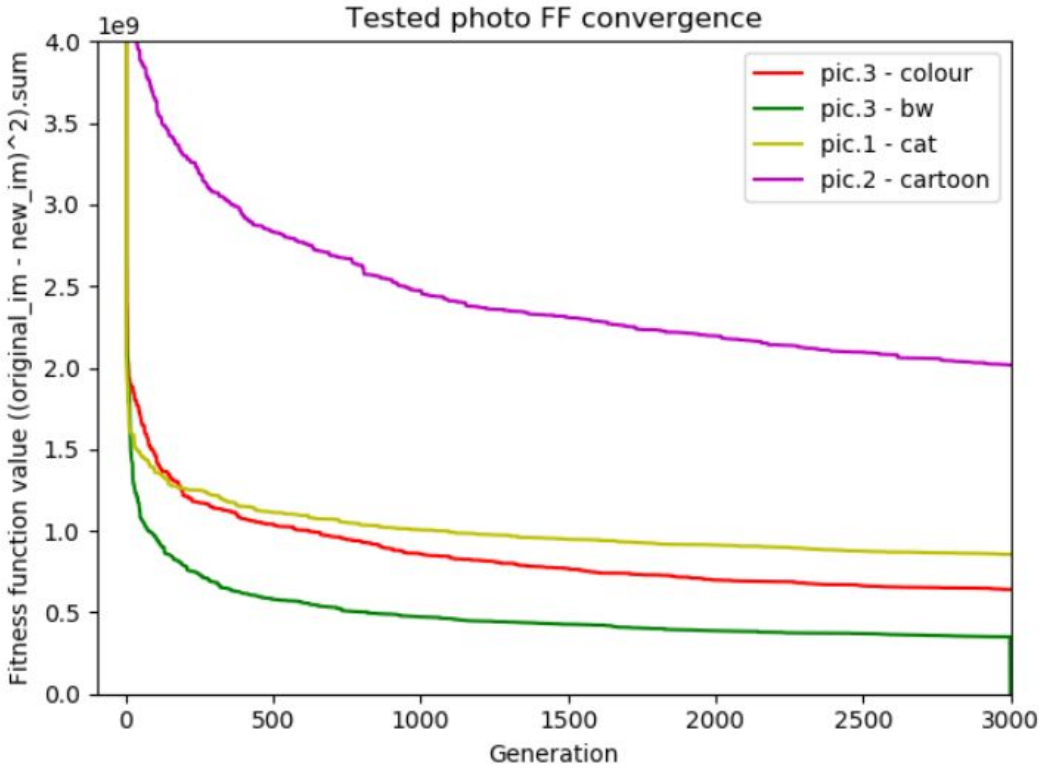


Original alg - 3000 generations

Modified alg - 3000 generations

$$fitInc_{col} = \frac{fitF_0}{fitF_{3000}} = \frac{11141651278.0}{641056733.0} = 17.3 \qquad fitInc_{bw} = \frac{fitF_0}{fitF_{3000}} = \frac{10297169499.0}{350367447.0} = 29.4$$

4. Grapg of fitness function convergence for each of the picture



5. Why the results are artistic

“In the most general sense, art is a mastery whose product gives aesthetic pleasure.” © Wikipedia

The implemented algorithm creates a picture from scratch but looking for some ideal - this is pretty similar to the way the real artists paint their works: inspired by some object, an artistic image appears with a help of imagination. Then the artist tries to bring this fantasy in the real world, on canvas, comparing the drawn image with one that is lives in her/his imagination. Often, when the drawn picture on some step differs from the ideal one in the way that the previous variant of a picture is much similar than the one after modification, the artist clears these “bad changes”, edit them. Thus, the whole process of the artwork is pretty alike the algorithm workflow.

In addition, the generated picture make the implementer of the algorithm (me) happy when I look at them, especially at those that we generated from the photo of a real person - it is so beautiful:)