Image Reproduction Assignment

Alexander Kurmazov, 19BS-01

April 2021

1 Idea

In my implementation, there are three essential entities: *source image*, *goal image*, and *reproduced image* (i.e., the best chromosome of an iteration). First, user gives the program the *source image* as an input. It is then resized to some specific resolution (512x512 by default) and converted to the gray scale. Next, image filtering is applied to the resized gray scale *source image* so that to perform noise reduction. Finally, Canny edge detection is applied to the image, yielding the *goal image*. Further on, the program tries to converge to the *goal image*.

(a) Source image (b) Goal image

Figure 1: Showing the difference between source and goal images

2 Chromosome representation

I have derived the following structure: a chromosome is a set of genes (i.e., lines with different characteristics). Each chromosome is a solution to the given problem (i.e., an approximation to the goal image that has some fitness score). A gene, in turn, is a set of the following parameters:

- *position* - coordinates of where line starts and ends
- *length* - a value that is the distance between two line coordinates
- *angle* - a floating point value in radians
- *visibility* - True/False whether line is visible or not

In code, the **position** parameter is represented as two coordinates: (x1, y1), (x2, y2) - of where a line starts and ends. Since a gene already has parameters **length** and **angle**, the second coordinate, (x2, y2), is indeed redundant, but it is still stored so that not to recalculate where line ends every time it needs to be drawn. The ending coordinates are calculated as follows:

$$x_2 = x_1 + \cos(angle) \cdot length$$

$$y_2 = y_1 + \sin(angle) \cdot length$$

Each of the proposed parameters mutates on every iteration of the algorithm with some small chance. More details on this will follow in **Mutation** section.

3 Population size

I was experimenting with this constant a lot. During the process, I noticed that the smaller population size is, the sooner the chromosomes of the population will become too similar to each other. On the other hand, setting this constant to big values leads to slow iteration time. Experimentally, I have come to the conclusion that setting population size to value in between of [6, 12] gives a good trade-off between quality of the reproduced image and elapsed time.

4 Selection

The selection technique in my solution is following: the chromosomes are sorted according to their fitness scores and the N best of them are chosen. The value N is equal to the population size constant.

5 Fitness function

Initially, each one of the chromosomes of a population is produced into an image (i.e., the *reproduced image*). Then, for each of the images, I compute the fitness score of an image by getting its structural similarity with the *goal image* (i.e., SSIM of **scikit-image** Python package).

Another thing to mention is that the **fitness function** is the most time-consuming part of the code, so I had a desire to improve the performance of it somehow. The good decision was to try using multiprocessing, which has improved the algorithm performance at least by factor of 3 (in terms of the time elapsed on an iteration).

6 Crossover

On every iteration of the algorithm, for every pair of chromosomes in a population, the crossover is performed by swapping chromosomes' genes by some randomly generated single crossover point that is anywhere between 0 and the constant size of genes in a chromosome.

7 Mutation

On every iteration of the algorithm, for every gene in the chromosome of a population, mutation is performed. Different parameters have different probabilities of being mutated on an iteration. These probabilities are listed below:

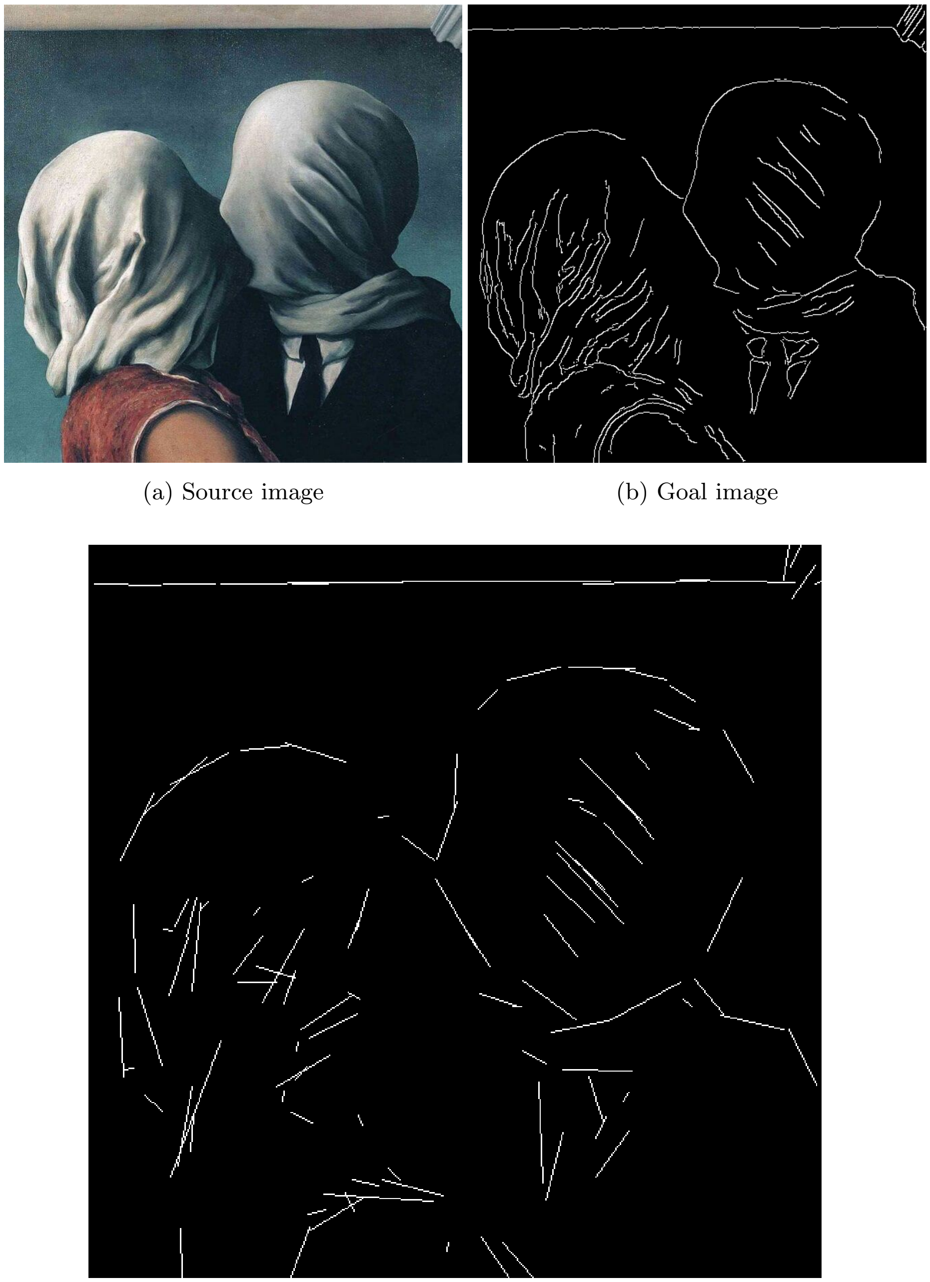
- *position* - 0.5 % for each of the (x, y) pair of being increased or decreased by a random value in the range from 1 to 5
- *length* - 0.5 % of being increased or decreased by an exponentially distributed random value with the mean of 5.0
- *angle* - 0.5 % of being increased or decreased by a uniformly distributed random value in the range from 0 to $0.2 \cdot \pi$
- *visibility* - 0.005 % of being changed to the opposite boolean value

It can be noticed that only *visibility* parameter differs from the others by the factor of 0.01. This decision is based on the desire of getting rid of the unwanted lines on the *reproduced* image such way, that the image would not look too blank. If I set this parameter to greater values, then the algorithm would exploit this by trying to get better fitness score by removing most of the lines, instead of putting them in place accordingly to the *goal image*.

8 Examples of reproduced images

Next will follow four examples. All of them very reproduced on different amount of iterations.

8.1 René Magritte, The Lovers



(a) Source image

(b) Goal image

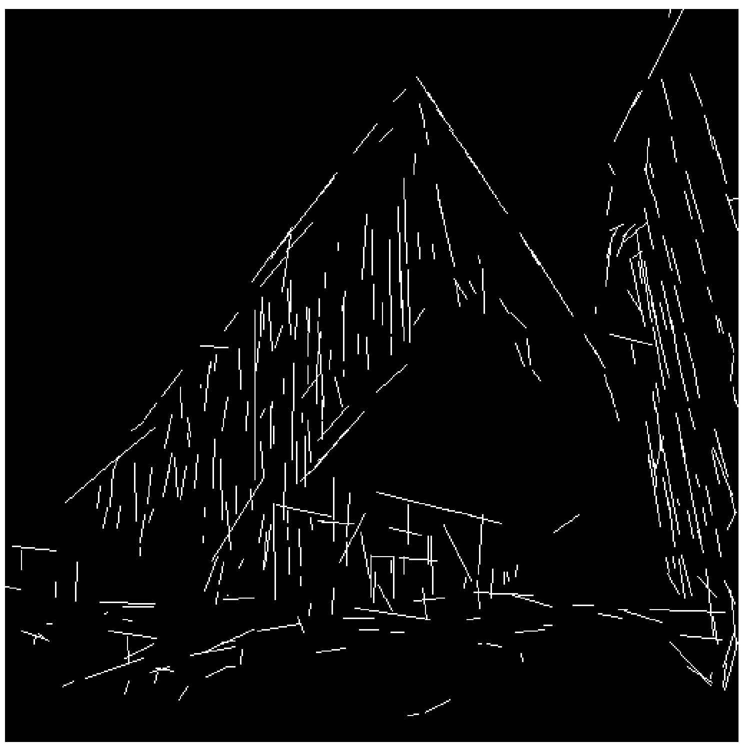
(c) Reproduced image

Figure 2: 32000 iterations. SSIM: 82%

8.2 Innopolis University

(a) Source image

(b) Goal image



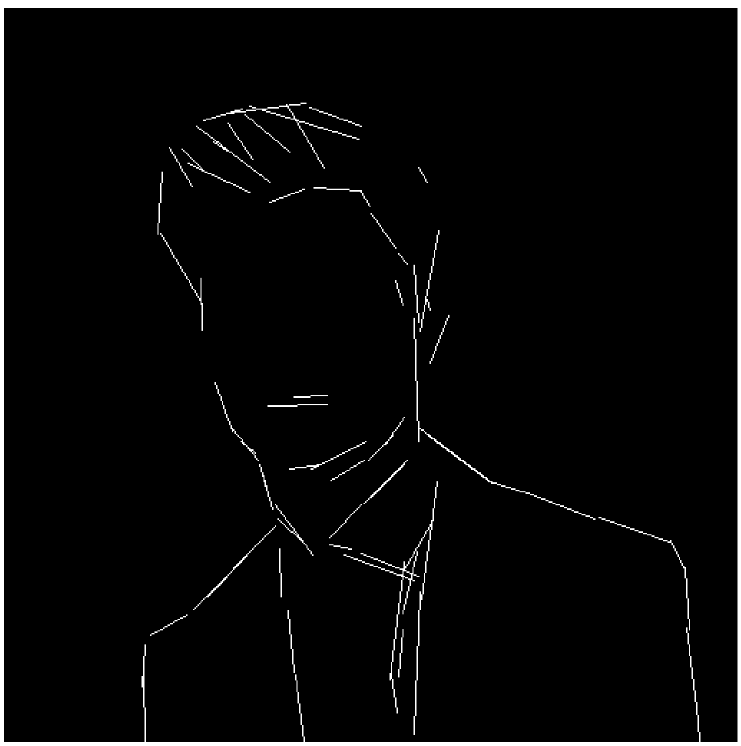
(c) Reproduced image

Figure 3: 5500 iterations. SSIM: 68%

8.3 Myself

(a) Source image

(b) Goal image

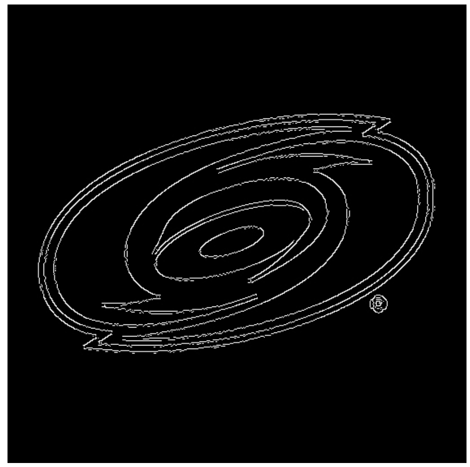


(c) Reproduced image

Figure 4: 50000 iterations. SSIM: 91%

8.4 Carolina Hurricanes logo

(a) Source image



(b) Goal image

(c) Reproduced image

Figure 5: 20000 iterations. SSIM: 86%

9 Thoughts on what art is

In my opinion, art can be anything that causes anyone to feel something. By all means, I think that every person can have his/her own emotions on a certain product of art, even though they might mismatch with what the author wanted people to feel.

I consider the results of my algorithm's work art because, from my point of view, the algorithm itself is only a tool, no more significant than a brush in the hands of a painter. The real artist, in turn, is the one who exploited the facilities of this tool in order to create something that has some idea behind. For instance, one of the images I tried to reproduce with my script was "The Lovers" by René Magritte. For an unknown reason, this particular piece of art gives me a light feeling of anxiety when I see it, and I wanted to emphasize this feeling with my algorithm by setting some set of parameters that could help to achieve it. The result can be seen above, in **figure 4(c)**.

All in all, it is necessary to point out that there are no true definition of art, that is not vague in any sense and gives a clear explanation on what it really is. So it is up to a person to decide what he or she considers to be an art.