Citation

Trist, K., Ciesielski, V. and Barile, P. (2011) ‘An artist’s experience in using an evolutionary algorithm to produce an animated artwork’, *Int. J. Arts and Technology,* Vol. 4, No. 2, pp. 155-167.

Problem Description

This paper explores the construction of an artwork using a software system named ‘The Shroud’ for evolutionary art based on genetic programming. The Shroud software either starts with an empty canvas or a canvas with a large number of lines as inputs, and evolves to output an image that is close enough to the target image.

The problem is unique because, unlike most evolutionary algorithms, the goal is not obtaining an image that is similar to the target image as fast and competent as possible. Instead, we want to combine all generations of images into an animation suitable for public display. Therefore, speed and efficiency need to give way to a constant rate of image appearance and a meaningful, recognizable subject emerging at an appropriate time. To have creative control over the animations, artists can determine the specific values for different parameters such as the maximum number of strokes that can be drawn, number of generations, fitness target…

In the problem, and individual is a program that draws an image on a canvas. For each individual, the authors have chosen to use a sequence of pencil lines to generate a pencil sketch of a target image. We can control where each line starts, in which direction it goes, and the its darkness with the function: *Draw(x, y, angle, grey)*. If the line extends beyond the boundaries of the image, we truncate the lines. If two pixels overlay, we take the average value of the darkness.

The fitness of an individual is calculated by summing up the pixel differences between the evolved image and the target image:

$$\sum_{i=1}^{width}\sum_{j=1}^{height}|evolved\_image(i,j) - target\_image(i,j)|$$

At the beginning of our evolutionary search, we generate an initial population of random programs, each containing a random number of *Draw* commands with random parameters. After calculating the fitness of each individual as described earlier, we apply certain selection, mutation and crossover methods (explained under parameters) to obtain a new generation of programs. Repeat the process until we encounter a stopping condition.

Parameters

- Selection – Elitism: Always copy the best individual from the current generation to the next generation unchanged to ensure the best individual in the next generation is at least as good as in the previous generation.
- Mutation and crossover – give a higher probability of being selected to fitter individuals.
- Number of individuals – the optimal number of individuals in a population is four.
- When to stop? – The evolutionary process is terminated when a pre-specified number of generations have been completed or a pre-specified level of similarity to the target image is reached.

- Write Nth best – only render every Nth best individual. As seen from Figure 1 and Figure 2, we can go through an enormous number of generations to obtain a desirable image. This parameter ensures that our animation does not have to include too many intermediate images.
- Fitness write threshold – only write a frame if the fitness has improved this amount since the last frame; another parameter used to avoid including less meaningful images in the animation.