Brock University

Department of Computer Science

Genetic Programming for Non-Photorealistic Rendering

Maryam Baniasadi
Technical Report # CS-13-08
June 2013

Brock University
Department of Computer Science
St. Catharines, Ontario
Canada L2S 3A1
www.cosc.brocku.ca

Genetic Programming for Non-Photorealistic Rendering

Maryam Baniasadi

A thesis submitted to the
School of Graduate Studies
in partial fulfillment of the
requirements for the degree of
Masters of Science

Faculty of Computer Science,
Brock University, St. Catharines, Ontario

Abstract

This thesis focuses on developing an evolutionary art system using genetic programming. The main goal is to produce new forms of evolutionary art that filter existing images into new non-photorealistic (NPR) styles, by obtaining images that look like traditional media such as watercolor or pencil, as well as brand new effects. The approach permits GP to generate creative forms of NPR results. The GP language is extended with different techniques and methods inspired from NPR research such as colour mixing expressions, image processing filters and painting algorithm. Colour mixing is a major new contribution, as it enables many familiar and innovative NPR effects to arise. Another major innovation is that many GP functions process the canvas (rendered image), while is dynamically changing. Automatic fitness scoring uses aesthetic evaluation models and statistical analysis, and multi-objective fitness evaluation is used. Results showed a variety of NPR effects, as well as new, creative possibilities.

Acknowledgements

I would like to thank the following individuals who supported me in all aspects in this work:

- Brian Ross, for all his exciting ideas, support, patience and understanding in all aspects of my work. And also for giving me this chance to have a break and welcome me back to finish this work.
- My parents and my sister, Tayebeh, Ali and Mojdeh, for how supportive they are despite the long distance that separates us.
- Steve Bergen, for sharing his work with me and helping me when I need him.
- Cale Fairchild, for all his patience and help which helped this work to be finished in time.
- Brock University and the Computer Science Department for all their resources and facilities.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Evolutionary Computation (EC) is a branch of computational intelligence which is based on the biological evolution mechanism [23, 33, 66]. All the techniques in this area use the idea of having populations of individuals which are generated using evolutionary reproduction such as crossover and mutation. Some of the famous EC techniques are: genetic algorithm, evolutionary programming, evolutionary strategy and genetic programming.

Evolutionary art and design is one of the research areas in EC [43,64,75]. Using EC techniques, the aim is to have creativity in making new artworks, images and designs. Many researchers have been working in the area of evolutionary art to discover creative and intelligent systems capable of generating interesting art works. Evolutionary art and design can be used to improve an existing design or find some new solutions for a specific problem.

A field of research in computer graphics is non-photorealistic rendering (NPR) [11,30,77]. The idea of NPR is to have systems which are capable of generating imagery which looks like traditional human generated art done in natural media such as watercolor, pen-and-ink and pencil.

In this thesis research, the main attempt is to create an evolutionary art system with NPR characteristics, which means to have results look like traditional media such as watercolor, oil and pencil, as well as brand new effects. Genetic programming is used in this research. There are a few evolutionary art papers that use genetic programming to create NPR effects such as [5,14,31,55]. However they did not go to great depths in this regard.

For this thesis the GP language is extended with the ideas and techniques inspired from the NPR literature [11, 24, 30, 50, 69, 77]. A goal for the implementation is to not limit the creativity aspect of GP. If pre-existing NPR

algorithms are used, there may be few new "discoveries" possible. One idea to encourage more creativity is to have operators and data used by different NPR algorithms. These techniques can be categorized to different areas such as: canvas/paper, brush/pencil, colour, filter and other ideas. For each of these categories there are a number of different methods and techniques which can be used as a building block functions in the GP language.

One thing to be investigated is how to render the canvas, in other words, how to choose the locations as the starting points to apply the brush strokes. Neufeld [14] processed the canvas using top-left to bottom-up method applying the painting expression for every pixel on the canvas. For this research other ideas are used. One idea is to use a "mask" to find the next unrendered pixel to start from in top-left to bottom-up method. Another idea is to use a "mask" in a random selection method.

In this thesis two categories of fitness evaluation are considered, namely aesthetic models and statistical analyses. There is some research to find a way of scoring images according to how aesthetically pleasing they are [19, 35, 41, 58]. The first attempt for this thesis research is to apply DFN (deviation from normality) as the aesthetic fitness function [38] which measure correspondence of an image to a bell curve gradient. Another analysis, the colour histogram test [45], calculates the quadric histogram distance between the colours of a source and target image. With the hope of finding more creative and some new brand results by GP, some statistical analysis and feature tests are used in this thesis. For example the value of mean, standard deviation of the result images can be used as fitness scores. Other tests such as entropy, skewness and kurtosis are considered.

The results from this research are evaluated in two ways: performance analysis and a human survey. First the fitness values from different experiments are analyzed and compared with each other. Since fitness performance does not guarantee if the result images are aesthetically pleasing to a human, a human survey is also undertaken, similar to ones in [60, 63]. The survey is used to see if viewers find the GP evolved images to have NPR-like characteristics.

1.1 Goal

In this thesis the main goal is to have an artistic evolutionary system using genetic programming to generate non-photorealistic rendering (NPR).

There are three main contributions introduced to reach the main goal in this research:

(1) Extend GP Language
(2) Using different canvas processing methods
(3) Using different fitness evaluation methods

This research is motivated by, and extended, Neufeld *et al.* [14]. The main common concepts with his work are: (i) using image processing filters such as moment and sobel (ii) using the painting algorithm from [50] (iii) using DFN, mean, standard deviation and CHISTQ as fitness scores. However, in this thesis, the GP language is extended by adding different colour mixing functions and some NPR techniques such as Kuwahara and mean shift segmentation. Neufeld has two versions of painting functions (paint and general paint); here, there are eight types of painting functions which make different types of results. Here also there are more fitness functions implemented, namely entropy, skewness and kurtosis. Another difference between Neufeld's and this research is to have other methods to process the canvas. Neufeld processes every pixel of the image. Here, a mask buffer is used to paint uncoloured pixels. Another method randomly selects some pixels within the image to be painted (as the center of the brush stroke). A mask buffer is used in this method too.

1.2 Thesis Structure

The structure of this thesis is as follows. Chapter 2 contains background information about genetic programming and multi-objective optimization. Chapter 3 reviews the literature in the area of NPR, evolutionary art and different methods of image evaluation. Chapter 4 describes the implemented system model in details. Chapter 5 explains the methods used as fitness evaluation of GP. Chapter 6 analyses the performance of the results. Chapter 7 contains different experiments to show the effects of using different parameters. Chapter 8 shows some miscellaneous experiments. Chapter 9 shows the human survey results. Chapter 10 describes the results and compares them to other research. Chapter 11 summarizes the work done for this thesis, and describes possible future work.

Chapter 2

Background

2.1 Genetic Programming

Genetic Programming (GP) is an evolutionary algorithm (EA) technique which tries to find good solutions for a problem by evolving a population of computer programs for that specific problem [66, 67]. It uses the idea of evolution in nature which finds the fittest individuals to breed and produce new offspring for the next generation.

2.1.1 Representation

The GP population is comprised of computer programs. GP can be determined as a branch of genetic algorithm (GA). One of the benefits of GP over GA is to use variable length structures to generate each individual. Here each individual or computer program is represented by a tree structure, which is a hierarchical composition of functions and terminals. The terminal nodes are leafs in the tree structure, and they can be constant or non-argument functions. The rest of the tree is made of function nodes which can be arithmetic functions, mathematical functions, logical functions or any specific functions for the problem domain. The GP can be constrained by the type it uses for the terminals and the function's arguments by applying strongly-typed GP [51]. For example the function set can contains of some arithmetic functions such as X+Y which has two arguments of type "double", and some logical function such as If_Then_Else using "boolean" type for the condition argument.

2.1.2 Genetic Programming Algorithm

Genetic programming algorithm is shown in Figure 2.1. First, random computer programs are generated using the function and terminal sets as the initialized generation. Each computer program is evaluated and given a fitness score which shows how well it works according to the problem. Then with a selection method some good scoring individuals are chosen for reproduction, namely crossover and mutation, to generate some new offspring for the next generation. This process continues until termination criteria is met.

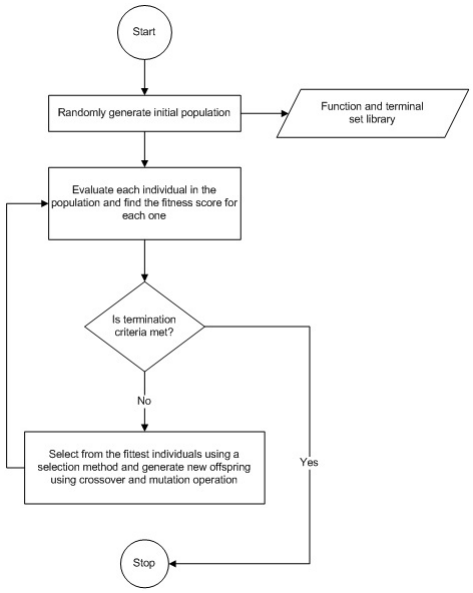


Figure 2.1: Genetic Programming Algorithm

2.1.3 Crossover and Mutation

Subtree crossover generates new offspring by randomly selecting two nodes and replaces the subtree rooted at the first parent crossover point by the selected subtree of the second parent. A copy of each parent is used for this operation with the purpose of reusing the original parents. If the size of the offspring is greater than the maximum depth parameter, the operation will

be reapplied again on the same parents. Figure 2.2 shows an example of crossover.

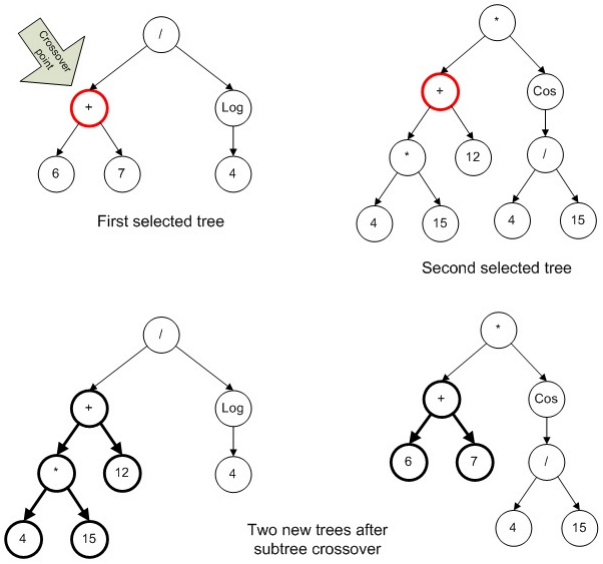


Figure 2.2: An example of subtree crossover operation

Subtree mutation which can be considered as a crossover operation between a randomly selected individual and a randomly generated tree. Here the type of selected node in the selected tree for mutation should be the same as the rooted node of the randomly generated one. See Figure 2.3

2.1.4 Selection

There are different probabilistic selection methods to select individuals for reproduction. Here tournament selection is used as GP selection method. In this method, k individuals (k is the tournament size) from the population are randomly selected and compared based on their fitness scores and then the best individual is chosen to do a crossover or reproduction operation. Tournament selection just considers which individual is better than another but not how much better. Moreover, since for crossover operation two individuals are needed, two tournament selections are made.

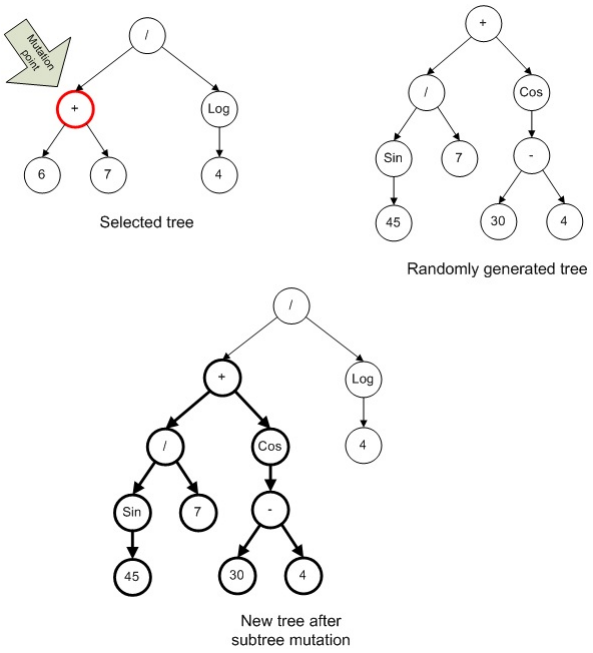


Figure 2.3: An example of subtree mutation operation

2.1.5 Evaluation

After generating the individuals of each generation (also after initialization phase) there is an evaluation step. Each individual is rendered on some input values or specific values for that problem and then a fitness function is run. It evaluates the individual according to the results it made, and a numerical value is calculated and assigned to that individual as its fitness score. This score shows how well each individual or program can solve the defined problem. The individuals with better scores have a higher chance to be selected for reproduction.

2.2 Multi-Objective Optimization

Although there are some problems that have a single goal to solve, many problems in the real world have more than one characteristic to be solved, this leads research into the area of multi-objective optimization (MOO) [29, 42]. Here the goal is to find optimal solutions respecting to all the dimensions

(objectives) of the problem.

2.2.1 Weighted Sum

One common method to analyze a multi-objective problem is to apply a weight to each score of each objective and combine them, which is known as *weighted sum*.

$$FitnessScore = f1 * W1 + f2 * W2 + ... + fn * Wn \tag{2.1}$$

This computes one score to be assigned to the individual, and so the problem can be considered as a single-objective problem. By changing the weight of each dimension, the importance of each dimension can be changed.

2.2.2 Pareto Ranking

There are many problems which have objectives belong to different concepts, and there is no natural way to combine them together. Pareto Ranking [20,42] keeps all the scores of the individual independent from each other, and compare individuals by the notion of dominance. One individual dominates the other if it is as good as the others in all the dimensions, and is better in at least one dimension. It can be formulated as follow:

X *dominates* Y:

$$\forall i : X_i >= Y_i \wedge \exists i : X_i > Y_i \tag{2.2}$$

Here the individuals which cannot be dominated by any other individuals in the population get rank 1. Then all these rank 1's individuals are removed from the population and the remaining individuals are ranked again to find the rank 2. This process continues until all the individuals in the population are assigned by a rank. The rank assigned to each individual is considered its fitness score. Therefore at the end of GP run there are a number of rank 1 solutions for that problem which can be considered as potential solutions. Moreover, this method has some disadvantages such as being inefficient for high dimensions problems since it is difficult to find distinct domination between the individuals of the population and almost every individuals get rank 1. Figures 2.4 shows an example of Pareto scoring for a minimization problem. Here individuals 1,2 and 3 get rank 1's because of their scores in Objective 2, Objective 1 and Objective 4 respectively. Since individual 4 and 5 can not dominate each other both get rank 2's.

Individual	Raw score of each object				Pareto
	Object1	Object2	Object3	Object4	
1	2.34	0.73	124	30	1
2	0	1.32	340	100	1
3	3.56	3.45	1200	1	1
4	4.32	1.03	350	45	2
5	5.27	0.89	350	15	2

Figure 2.4: Pareto scoring example

2.2.3 Rank Sum

Rank Sum is a method which can be used for multi-objective problems with many dimensions [28]. Here the scores of all dimensions for an individual are calculated and stored separately as $(f_1, ..., f_k)$ and each fitness score is ranked comparing the individual with all other ones in the population to obtain to $(r_1, ..., r_k)$ for each individual. Using a weighted sum function

$$fitnessscore = \sum_{i=1}^{k} r_i * W_i \tag{2.3}$$

a fitness score is assigned to each individual.

Normalized rank sum is similar to the rank sum in obtaining $(r_1, ..., r_k)$ for each individual. However these ranks are then normalized by dividing each ranks by the maximum value of that rank $(r_1/r_{1max}, ..., r_k/r_{kmax})$. Then these new normalized ranks are used in a weighted sum function to calculate the fitness score for the individuals.

Figures 2.5 shows examples of scoring for rank sum and normalized rank sum. Here individuals are ranked for each objective (base on the raw scores in Figure 2.4). Then the sum of the ranks are calculated by adding up the ranks of objects for each individual. To calculate the normalized rank, each object rank is divided by the max rank of that objective then the sum of all normalized objectives for each individual is calculated.

Individual	Ranks each object				Sum Ranks	Normalized ranks
	Object1	Object2	Object3	Object4		
1	2	1	1	3	7	1.45
2	1	4	2	5	12	2.5
3	3	5	4	1	13	2.8
4	4	3	3	4	14	2.95
5	5	2	3	2	12	2.55
Max rank:	5	5	4	5		

Figure 2.5: Rank sum and normalized rank sum scoring

Chapter 3

Literature Review

3.1 Non Photorealistic Rendering

Non Photorealistic Rendering (NPR) is a field in computer graphic research which is focusing on generating images that look like traditional human generated art done in natural media, such as watercolor, pen, ink and pencil [11, 30, 77]. While "photorealism" techniques are judged by how their results are closed to a real photograph, in NPR how effectively the images "communicate the essence of a scene" [11] is important. NPR research can be divided to three main categories [11]: artistic media simulation, user-assisted image creation, and automatic image creation. The purpose of artistic media simulation is to represent physical properties of an artistic medium such as pencil, pen-and-ink, watercolor, etc. The second category tries to have systems with the ability of guiding the user to create artistic images. The third category of research creates artistic images automatically. Some representative NPR research is as follows.

In Haeberli [57] abstract impressionistic images are created by implementing an interactive system help the user to have varies of representations from a single source image. Point-sampling the source image at a set number of locations by the user, the brush strokes are rendered on those positions. Each brush stroke has five properties which are location, colour, size, direction and shape.

In Colton *et al.* [69] an NPR system namely Painting Fool is combined with a machine vision system which is used to analyze a short video of a person expressing an emotion and extract some specific information for the

Painting Fool system. Mapping this data to a specific artistic style, this system renders a portrait painting which is showing that emotion. The NPR system has two main steps, namely segmenting the entire image and rendering the shapes in each segment using different painting styles.

Hertzmann [3] has a method to generate hand-painted images using series of spline brush strokes. This system uses a numbers of layers. The sketch in the first layer is drawn by a large brush then smaller brushes are used for upper layer with a method to find the region that the small brushes need to be applied (the blurred of the source image). Using different brush sizes and having a long curved brush strokes which is aligned to normal gradients of the source image, this system generates pleasing results.

Huang *et al.* [24] describes a paint brush style simulation which uses brush strokes with multiple colours and different textures assigned to each stroke. The idea is that in real hand drawings, strokes have various colours and textures, and to implement it here each bristle of a stroke has its own colour. A simple easily implemented method to have lighting effects is proposed. This paper focuses on the strategy of rendering the strokes. There are two phases: calculating the properties of the stroke; and finding out the stroke path with an orientation field.

More NPR techniques can be found in pen and ink [22], watercolor [32, 36, 78], pencil [76], pastel and oil [9, 52, 80].

3.2 Evolutionary Art

There are number of researchers and artists attracted to the field of evolutionary design for generating art. Richard Dawkins was the pioneer in this area [64]. Karl Sims [43] and William Latham and Stephen Todd [75] were the first that combine evolutionary techniques and computer graphics to create artistic works of great complexity. However a human is required to guide these systems by scoring the results of evolution. Peter Bentley [26–28] and Adrian Thompson [7, 8] showed that evolution was capable of innovation without human guidance. The evolutionary art systems need to have the ability to be creative. Obviously that is different from finding an optimal solution since in the art area having an optimal program does not necessarily mean it will have visual artistic values to a human judging it.

3.2.1 Evolutionary Art and NPR

Barile *et al.* [55] propose an automatically evolutionary system using genetic programming, generates non-photorealistic results such as "Shroud of Turin" effect, "decal" effect and "starburst" effect. Three visual canvasses are used which for two of them the user can define the applied brushes. The GP trees are made by "draw" nodes which contain specific local information of target image such as the HSV colour space, the gradient map and the importance map to place colour on the canvas. To evaluate and find the best results of each generation, the target image is compared pixel by pixel to each generated images.

In Izadi and Ciesielski [5], the GP is used to create NPR images using two types of strokes, namely "empty triangle" and "tiled triangle". To evaluate the results of each generation, a fitness function is implemented to compare the target image with generated canvas pixel by pixel and to do so the average normalized differences between the three channels of the target and result image is computed. The "draw-triangle" function has eight terminals, which are the inputted information to draw a triangle on the canvas: the lines length, the angles on lines and the three channel colours of RGB.

Kang *et al.* [25] use evolutionary algorithm with variable-length chromosome representation to generate NPR having optimized rendering quality by using multi-level brush strokes. There is an equation defined as fitness function to minimize the distance between generated images and the input image using the minimum number of brush storks. This formula considers the optimal result to be the one with minimum use of brush strokes and minimum difference with input image.

In Neufeld *et al.* [14] artistic image filters are evolved with GP [45]. The image is filtered using a dynamic buffer which is initialized with the source image as a canvas. An image filter is a complex expression implemented as a GP tree, and composed of variety range of image processing functions, such as paint stroke operator and terminals. The filter language is designed so that components can be combined together in complex and unexpected ways. The images are evaluated using the multi-objective approach from [40]. Also, the filter language uses some techniques from [50, 65].

In Machado *et al.* [59] GP is used to evolve image colouring filters that colour grayscale images by taking the lightness channel of the training images and computing the hue channel. This is done by comparing computed results to desired training examples. The best evolved programs can be generally

used for other gray scale images. There is a fitness function to evaluate the individuals in which the colour distance between the desired output and the target one is used to assign the fitness score.

In Colton and Torres [70], the possibility of automatically evolving an image filter to approximate a target filter (for instance the filters of Photoshop) is investigated. Here a tree-based representation of image filters is used and fitness score is assigned to each individual of the population by referring to an image produced by the target filter. Given an image and a target filtered version of that image, this system constructs a filter such that the filtered image approximates the target filtered image. The bitmap RGB values of the images are used to evolve good solutions. Various numbers of filters are also used in this investigation such as charcoal, dark strokes, diffuse glow, film grain, glowing edges, neon glow, patchwork and stamp filters.

In Yip [15], a genetic algorithm is used to evolve new aesthetic filtered images using a target pre-filtered image as inspiration. Here some simple filter functions are used to create new filtering chromosomes. There are also different measurement methods such as "mean and standard deviation of its histogram" and "entropy" to find a fitness score for each individual.

3.2.2 GP and Procedural Textures

Sims [44] is one of the first work of using procedural texture to evolve aesthetic images. There are also other interactively generating procedural texture in [6, 34, 48, 74].

GP is used to implement an unsupervised evolutionary art application namely Gentropy to produce 2D procedural textures [45]. The idea is to merge mathematical functions and image manipulation functions to make an expression. One or more target texture images is given to the system as the desired texture features so that the result images can have the same characteristics as the targets. Colour histogram matching, wavelet analyses, and a specific histogram matching for comparing the smoothness or contrast are applied to evaluate the result images.

Ross *et al.* [41] uses an unsupervised evolutionary art system to create aesthetic images. Here procedural texture images are evolved with genetic programming using multi-objective fitness evaluation. Ralph's model of aesthetics [38] is the main feature in this investigation (described in Section 3.3.1).

Xu *et al.* [62] introduced an interactive GP system called IMAGENE to evolve novel and artistic images from a set of source images. To generate

a colour image, this system evolves three individual for each RGB colour channels. This system lets the user to select the function and terminal sets and also set some of the GP parameters value.

In [19], GP is used to automatically evolve procedural texture images. This study focuses on comparing the results of multiple experiments applying four different aesthetic measures as fitness function.

3.3 Image Evaluation

3.3.1 Aesthetic Measure Functions

Machado and Cardoso [58] present a theory which illustrates that aesthetic values depend on biological and cultural issues, namely on visual image processing. Aesthetic visual value depends on two factors: (1) Processing Complexity (PC) in which the lower value is preferable; (2) Image Complexity (IC) in which the higher value is preferable. That is, a complex image is not necessarily difficult to process. Thus, images that are simultaneously visually complex and easy to process are the ones that have higher aesthetic value. The following formula expresses this theory:

$$M(I) = \frac{IC(I)}{PC(I)} \tag{3.1}$$

This formula can be used as a fitness score in the evolutionary system like genetic programming.

In Ralph [38], hundreds of samples of fine art are investigated and it is illustrated that many art works correspond to normal or bell curve distribution of bell curve gradient. The bell curve model suggests that a viewer is most attracted to changes in an image like the edges between different colours and obviously the regions with constant colours are of less interest. Furthermore, larger changes are more noticeable than smaller ones. There are several steps to compute the value of distribution from normality (DFN) for an image. The DFN value can be used as a fitness score in GP system knowing that an image with a precis normal distribution will have DFN equal to zero. An image with bad fit of normal distribution will get higher DFN value.

In Spehar *et al.* [12], the relation between the aesthetic pleasing aspects of fractals and the way they can be generated is studied. Three categories of

fractal pattern are comprehensively studied, namely natural fractals, mathematical fractals and human fractals. It is found that fractal images with a fractal dimension around 1.35 are more interesting. That is, images with a higher fractal dimension are considered complex, and images with a lower dimension are considered uninteresting. Using the mentioned value there is a formula that can be used as a fitness function in evolutionary system. That is, for an image I with fractal dimension of d the following formula is used for aesthetic measure:

$$M(I) = max(0, 1 - |1.35 - d(I)|) \tag{3.2}$$

With this formula images with fractal dimension between 1.1 and 1.6 will have positive value.

3.3.2 Statistical Analysis

Hertzmann *et al.* [4] present a machine learning system which uses analogy to create NPR filtered images. Given the unfiltered and filtered source image and also the unfiltered target image this system generates the filtered target image using two matching algorithms, namely BestApproximateMatch and BestCoherenceMatch. The idea is to compare the statistics pertaining of each pixel q in the target pair against statistic for every pixel p in the source pair to fine the best match. To implement BestApproximateMatch algorithm the approximate-nearest-neighbor search (ANN) and tree-structured vector quantization (TSVQ) are used.

Graham *et al.* [35] reviews the statistical regularities in artwork which include pairwise spatial statistics, sparseness, luminance, colour and composition level statistics. The relation of these statistical regularities with human visual perception is studied. It summarizes the different attempts of using statistics to model aesthetic principals. *Stylometry* which is related to quantify art style in literature is studied from statistical point of view.

In Cover *et al.* [46], Shannon's Entropy is described as a method of measuring uncertainty with random variable:

$$H(X) = -\sum_{x \in X} p(x) Logp(x) \tag{3.3}$$

where X is a finite set of random variables and p(x) is the distribution of x in X. Rigau *et al.* [39] used Shannon Entropy to measure the *order* and

Kolmogorov Complexity for complexity in an image for *Birkhoff's Aesthetic Measure* [16] which is the ratio between order and complexity in an image.

There are other statistical measures such as skewness and kurtosis which are used in evaluating aesthetic values of an image [13,71,72]. Skewness shows if the tile of the probability density of each left or right side is longer or it distributes on both side of the mean equivalently. Kurtosis shows the "peakedness" of probability distribution. Hughes *et al.* [47] presents a new method for quantification of artistic style by improving a sparse coding model. Sparse coding model is trained to maximize the kurtosis of the distribution of responses of an arbitrarily selected image from a specific image space.

3.3.3 Interactive Evaluation

Many evolutionary art systems use human response as a fitness score to find the best solutions of each generation especially in case that there is no explicit evaluation function to obtain the desired results. For this kind of problem, the interactive genetic algorithm (IGA) can be applied to use the human preference as a fitness function [2, 9, 10, 17, 53].

Chapter 4

System Architecture

This chapter explains the details of the implemented system for this research. This includes describing the GP, all the implemented options and parameters, the GP language and the fitness functions.

4.1 System Algorithm

The ECJ [73] GP is used to implement the automated artistic evolutionary system. Before evolution is started, all the defined parameters are set by the user through an implemented GUI. The parameter file for GP evolution is created. Then a pre-processing step initializes and processes the source image. In the initialization part the source image and the target colour image (if it is different from source image) are read. This system is designed to work in "RGB" or "HSV" colour mode. For both of the colour modes the general ranges of the channels are converted to float values between 0 to 1 in this phase and store in specific arrays. The pre-processing step includes calculating the "moment" and "sobel edge" for the source image and also creating the colour histogram of the chosen target colour image. The target colour image can be the source image or any other image supplied by the user and is used to compare with the colour histogram of each individual in the population.

Next, GP initializes the first population of trees using the selected functions and defined terminals. During the evaluation phase, each individual paints the canvas, which can be initialized by the source image or by a specific colour such as white. Based on the fitness functions that the user selected

for the run, the canvas is evaluated and a fitness scores are assigned to the individual. At the end of each generation all the individuals are ranked using "normalized rank sum". This new assigned rank is used in tournament selection to select individuals for crossover and mutation operations. Figure 4.1 shows the architecture of the system.

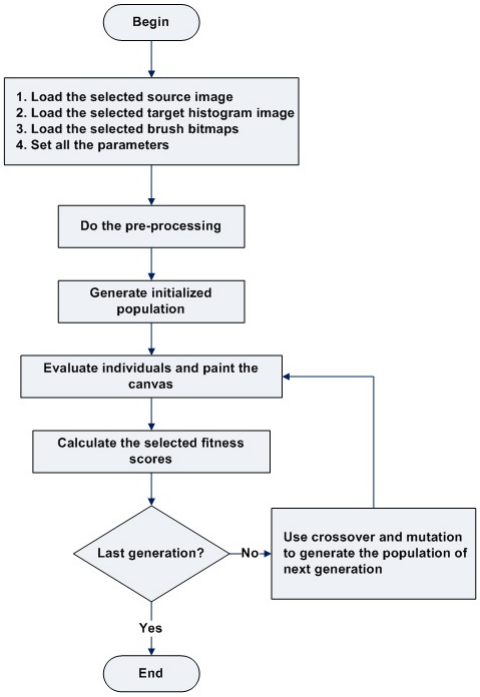


Figure 4.1: The architecture of the system

Since for both "RGB" and "HSV" colour mode the value of channels are converted to [0,1], all of the functions are designed to work in this range and there is no need for having separate functions for each mode.

4.2 System Parameters

This system is designed with the idea of having a flexible system which gives the users the opportunity of having variety of different results by choosing the parameters in the way they want. These parameters can be divided into the GP parameters and other parameters.

4.2.1 GP Parameters

GP parameters can be set by users. Below is the list of these parameters and a short description about them:

- number of generations:

  This is the number generations for a GP run.

- populations size:

  This is the number of individuals in each generation.

- tree initializer:

  There are two main ways to generate tree expressions for the initialized generation, namely grow and full methods. However there is another methods which is used in this research, namely ramp half-and-half. This method combines the grow and full methods.

- initialize tree max and min depth:

  These parameters restrict the size of initialized trees.

- crossover and mutation percentage:

  This parameters show how many times GP uses crossover or mutation for genetic operation.

- max tree depth:

  Each of the genetic operation has the max depth parameter to define the maximum size of tree they would make.

- probability of selection terminals/nonterminals:

  These parameters define the probability the crossover and mutation select terminal or nonterminal nodes to do the genetic operation.

- size of tournament selection:

  This parameter defines how many individuals are randomly selected by tournament selection method.

4.2.2 NPR parameters

There are two basic ways to process the tree expression. The first way is to process the whole image and apply the tree expression to each unpainted pixel. The second way is to randomly select some pixels on the canvas and run the trees expression on them. Other methods are defined in Section 4.3.

There are many brush bitmaps available for the user to pick to be used in the painting functions. Figure 4.2 shows six examples of these bitmaps. All of them are created by myself for this thesis. Using different brush bitmaps gives the system the ability of making different painting effects on the canvas. There are eight pre-defined brush sizes. The maximum brush size is the real brush bitmap size and the other 7 sizes are 0.8, 0.7, 0.6, 0.5, 0.4, 0.3 and 0.2 scales of the original brush bitmap size. The number of brush sizes for each experiment can be set by the users. These brush bitmaps are used in painting function which is described in section 4.4. In these functions if the current brush bitmap value is zero, the canvas is not painted.

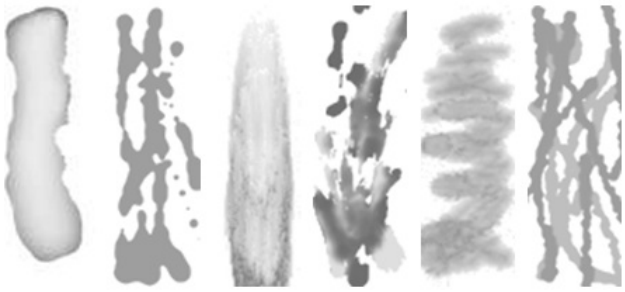


Figure 4.2: The brush bitmaps

There are some filtered brush bitmaps and paper textures which can be selected to be used as terminals in the tree expression. See Figure 4.3 and Figure 4.4.

This system can work in two different colour modes namely, RGB (Red, Green, Blue) and HSV (Hue, Saturation, Value). This mode is selected by the user.

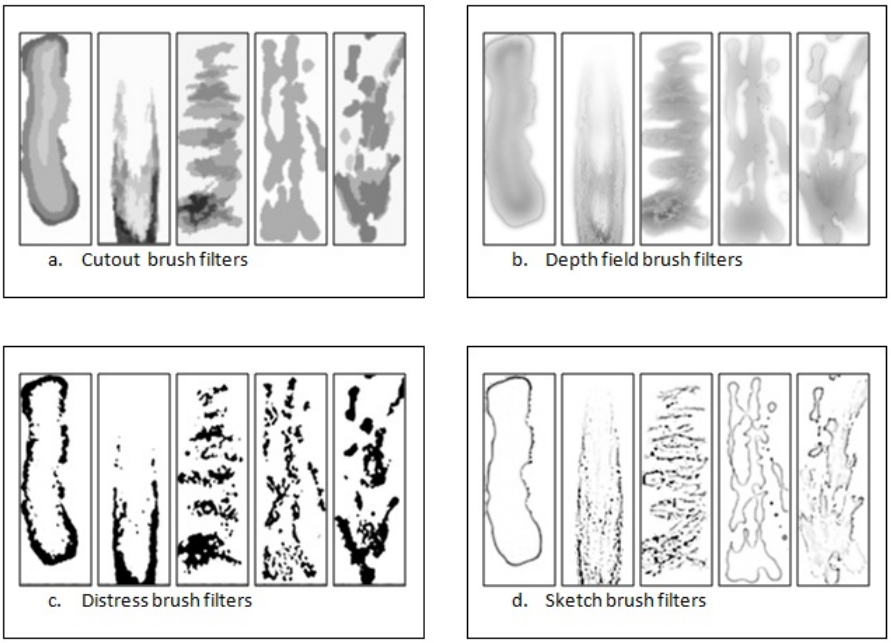


Figure 4.3: Brush Filters

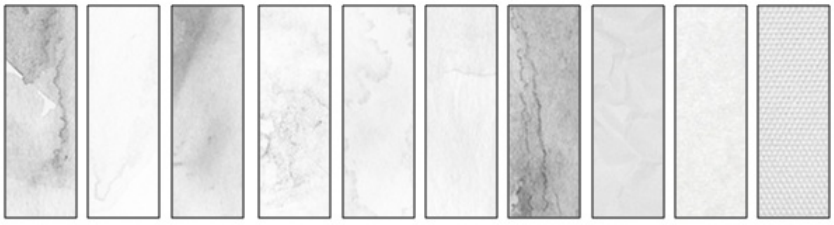


Figure 4.4: The paper textures

4.3 Pixel Selection and Painting

In the evaluation phase the source image is processed based on the selected method. There are two processing methods. The first one (Algorithm 1) is to randomly select a number of pixel locations. This number is defined by the user. There is a mask buffer to save the pixel location, which is then painted. This buffer is initialized by zero and when a pixel is painted the value of same location in this mask is changed to one (which means it is painted). Whenever a pixel location is selected the status of the mask buffer is checked to see if it is painted before. In this way the previous painted pixels do not get painted again. The second method (Algorithm 2) is to process the whole image using a mask buffer (the same as the first method) to paint all of the unpainted pixels.

```
while n ≤ NomberOfSelection do
    Randomly find a "X" and "Y";
    if Mask[X][Y] == unpainted then
        Set values for the terminals;
        Evaluate the tree expression;
        n++;
    end
end
```

Algorithm 1: The algorithm for randomly selecting pixels

```
for all Y in image height do
    for all X in image width do
        Set "X" and "Y" by current pixel location;
        if Mask[X][Y] == unpainted then
            Set values for the terminals;
            Evaluate the tree expression;
        end
    end
end
```

Algorithm 2: The algorithm to process image pixel by pixel

In both methods, the selected pixel location is considered as the center of the brush stroke to put on the canvas (for the paint functions). During the evaluation of the tree expression the canvas is painted and all the painted pixels are set to one in the buffer mask. In the next step the painted canvas gets a fitness score evaluation by the selected fitness functions.

4.4 GP language

4.4.1 Paint Functions

One of the main goal of this research is to have different NPR functions implemented as building blocks for the GP system use. The system is flexible and lets the user have some decision making in selecting the functions. Tables 4.1 show names, a short description and arguments type for each of them.

All of the paint functions are shown the same basic structure. However they are different in some ways. Paint0 is the basic painting function which is implemented based on [14,50]. The painting method is shown in Algorithm 3. The "rotation buffer" is used to solve the "hole" problem that happens with rotating the brush by θ angle. In the rotation formula a new x and y is calculated using current X, Y and θ. Since these new values for x and y should be an integer value, the results from the calculation should be rounded up to the nearest integer. This conversion makes some pixel locations be missed, and therefore unpainted. Therefore there are some holes in each painted brush on the canvas. To solve this problem the new (x,y) locations are calculated with more than one formula (Formula 4.1) to find the missing locations. Figures 4.5a. and 4.5b. show an example of rotated brush using one formula to find the locations and using all equations in Formula 4.1.

However, with this single change another problem occurs, which is painting one location more than one time and losing the brush bitmap texture in the result. Therefore a "rotation buffer" is used. This buffer is like mask buffer and initialized by zero. Then in the painting algorithm each painted location is set to one in the rotation buffer. So if a rotated location is calculated again with any of the Formula 4.1, it will be ignored. See Figure 4.5c.

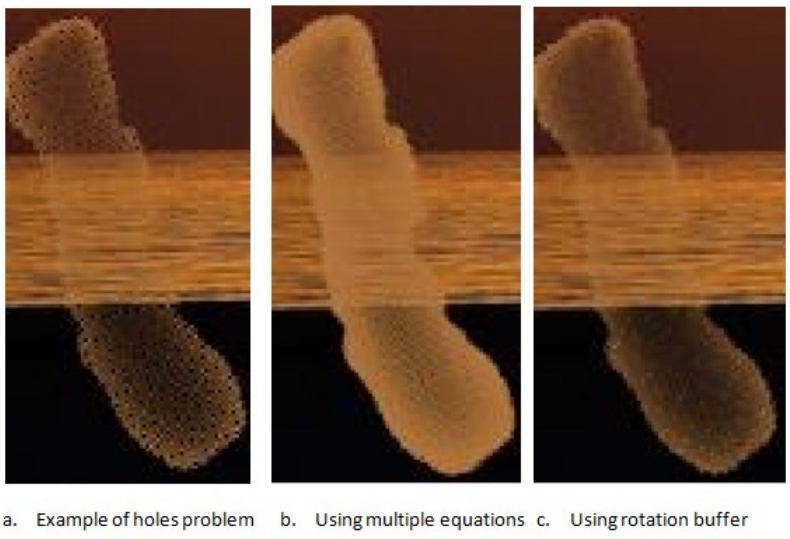


Figure 4.5: An example of rotating a brush

Input: X, Y, θ, Brush Length, Brush Width, Brush bitmap
Resize the brush bitmap based on the "L" and "W" of brush;
Set center of the brush to (X,Y);
for *Brush length and width* **do**
    Find the pixel location using "X", "Y" and θ;
    if *RotationBuffer[X][Y] != Rotated* **then**
        Mix the colour of the canvas and source image using the value of the bitmap on that location;
        Change the colour of the canvas with the new calculated value;
        Set the pixel to "1" in the mask buffer;
        Set the pixel to "1" in the rotation buffer;
    end
end

Algorithm 3: The basic algorithm painting functions

$$\begin{aligned}
new_x1 &= \cos(theta) * X - \sin(theta) * Y \\
new_y1 &= \sin(theta) * X + \cos(theta) * Y \\
new_x2 &= \cos(theta) * X - \sin(theta) * Y + 0.5 \\
new_y2 &= \sin(theta) * X + \cos(theta) * Y + 0.5 \\
new_x3 &= \cos(theta) * X - \sin(theta) * Y - 0.5 \\
new_y3 &= \sin(theta) * X + \cos(theta) * Y - 0.5
\end{aligned} \tag{4.1}$$

To mix the colour of canvas and source, "normal blend" method is used which is also known as the "alpha blend" method (Formula 4.2). The intensity is set by the current bitmap location intensity.

$$\begin{aligned}
R &= source_r * (1 - Intensity) + canvas_r * Intensity \\
G &= source_g * (1 - Intensity) + canvas_g * Intensity \\
B &= source_b * (1 - Intensity) + canvas_b * Intensity
\end{aligned} \tag{4.2}$$

With the hope of introducing more creativity, the "paint2", "paint_RGB" and "paint4" functions were implemented. Paint2 does not use the current colour of the canvas and source image. Instead, it has two subtree arguments and the value of them are used in the mixing colour phase. In the basic painting algorithm the "normal blend" method is used to mix the colour of source image and canvas, using the value of brush bitmap. However one idea to have more creativity is to let GP find a new equation to do the mixing colours instead of using the general methods. That is, paint4 has a colour mixing expression as its last argument. The result of evaluating this tree is a colour to be painted on the canvas. Paint_RGB is similar to paint4 but different in having a separate colour mixing expression for each red, green and blue channel.

Table 4.1: NPR painting functions

Name	Description	Arguments
Paint0	This function has two children as "brush size" and "brush bitmap" to paint a brush stroke on the canvas using the current data.	(F, F)
Paint1	This function has three children as "brush size", "brush bitmap" and "angle of the brush". The other part is same as Paint0.	(F,F,F)
Paint2	This function has five children as "brush size", "brush bitmap", "angle of the brush", "colour1" and "colour2" which are the colours to be mixed and painted on the canvas.	(F,F,F,V,V)
Paint4	This function has four children. The first three are the same as Paint1 function. The last child is a tree expression to find the final colour to be painted on the canvas. This tree type is called "mixTree" in this system.	(F,F,F,MV)
Paint_ Kuwahara	This function has two children as "length" and "width". It uses the paint algorithm and "kuwahara" technique to paint on the canvas.	(F,F)
Paint_Dodge	This function has three children like Paint1. It uses the paint algorithm and "Dodge" method to mix the colours.	(F,F,F)
Paint_Burn	This function has three children like Paint1. It uses the paint algorithm and "Burn" method to mix the colours.	(F,F,F)
Paint_RGB	This function has six children. The first three are the same as Paint1. The other three children are different tree expressions to find the red,green and blue colour to be painted on the canvas.	(F,F,F,MV, MV,MV)
Paint_HSV	This function is similar to Paint_RGB but can be selected just in "HSV" mode.	(F,F,F,MV, MV,MV)

4.4.2 Other GP Functons and Terminals

For this research strongly-typed GP is used by defining three types of float, vector and mixTree. For each of these types a function and terminal set is defined and as it was mentioned the user can select the functions and some terminals for GP run. The float type is used in arithmetic functions and computer vision functions (see Table 4.2). All of the NPR functions work with vector type. Here the vector type is used to work with colour values. The vector type functions are listed in Tables 4.1, 4.3 and 4.4. The mixTree type is used to specify the functions and terminals which are used to make the mixing tree expression for paint functions such as paint4. The mixTree type functions work with vector and float types which are defined by "mixTree_vector" and "mixTree_Float" types to be used just by this tree expression (see Tables 4.5 and 4.6).

Table 4.2: The list of float functions

Name	Description	Arguments
Add	Add and return the value of its two children.	(F,F)
Sub	Subtract and return the value of its two children.	(F,F)
Mul	Multiply and return the value of its two children.	(F,F)
Div	Divide and return the value of its two children.	(F,F)
Min	Return the minimum value of its two children.	(F,F)
Max	Return the maximum value of its two children.	(F,F)
Abs	Return the abstract value of its child.	F
Round	Round and return the value of its child.	(F,F)
Avg	Return the average value of its two children.	(F,F)
Log	Return the Log value of its child.	F
Exp	Return the Exp value of its child.	F
Sin	Return the Sin value of its child.	F
Cos	Return the Cos value of its child.	F

Table 4.3: NPR functions

Name	Description	Arguments
Luminance	This function has one child as a colour and calculate the luminosity of it using the following equation: $luminance = 0.299f * red + 0.587f * green + 0.114f * blue$	V
Brighten	This function makes the given colour lighter using an intensity value which is the second child of this function.	(V,F)
Darken	This function makes the given colour darker using an intensity value which is the second child of this function.	(V,F)
Dodge	This function applies the "dodge" method to the two given children colours using an intensity value which is the third child of this function.	(V,V,F)
Burn	This function applies the "burn" method to the two given children colours using an intensity value which is the third child of this function.	(V,V,F)
Normal Blend	This function applies the "normal blend" method to the two given children colours using an intensity value which is the third child of this function.	(V,V,F)
Difference Blend	This function applies the "difference blend" method to the two given children colours using an intensity value which is the third child of this function.	(V,V,F)
Overlay Blend	This function applies the "overlay blend" method to the two given children colours.	(V,V)
Kuwahara	This function has three children as "length", "width" and "brush bitmap". It changes the canvas using these arguments in "kuwahara" technique.	(F,F,F)

Table 4.4: NPR functions

Name	Description	Arguments
Mean Shift Segmentation	Its arguments are "length", "width" and "brush bitmap". This function changes the colours on the canvas using "mean shift segmentation" technique.	(F,F,F)
Median	This function has a child as "size" which specifies the size of square to apply the algorithm. It changes the canvas.	F

The following functions were modified from algorithms obtained from the following sources:

- Kuwahara [61]
- Mean Shift Segementation [54]
- Median [37]
- Brighten, Darken, Dodge, Burn, Difference Blend, Overlay Blend [21]

Table 4.5: MixTree functions

Name	Description	Arguments
mixAdd	Add two first vector colour children with the intensity value from the third child.	(V,V,F)
mixSub	Subtract two first vector colour children with the intensity value from the third child.	(V,V,F)
mixMul	Multiply two first vector colour children with the intensity value from the third child.	(V,V,F)
mixDiv	Divide two first vector colour children with the intensity value from the third child.	(V,V,F)
mixAbs	Return the abstract value of its child.	F
mixAvg	Calculate the average of two first vector colour children with the intensity value from the third child.	(V,V,F)
mixCBrn	Mix two first vector colour children with the intensity value from the third child using burn equation.	(V,V,F)

Table 4.6: MixTree functions

Name	Description	Arguments
mixCDog	Mix two first vector colour children with the intensity value from the third child using Dodge equation.	(V,V,F)
mixNBld	Mix two first vector colour children with the intensity value from the third child using normal blend method.	(V,V,F)
mixDBld	Mix two first vector colour children with the intensity value from the third child using difference blend method.	(V,V,F)
mixOBld	Mix two first vector colour children using overlay blend method.	(V,V)
mixBrt	Brighten first vector colour child with the intensity value from second child.	(V,F)
mixDrk	Darken first vector colour child with the intensity value from second child.	(V,F)
mixLog	Calculate log value of its child.	V
mixExp	Calculate Exp value of its child.	V
mixSin	Calculate Sin value of its child.	V
mixCos	Calculate Cos value of its child.	V

The arithmetic functions are shown in Table 4.2. These functions are standard in GP literature [66]. There are also different logic and converting functions for float and vector types (Table 4.8). Terminals are also defined for each these types. The list of float and vector type terminals are in Table 4.7. For mixTree type the canvas_c, source_c and brush bitmap are available. All the floating image processor terminals (5*5 - 13*13) work on canvas. Since the canvas is changing during rendering, these terminals cannot be pre-computed on canvas.

To make a colour lighter or darker, equation 4.3 can be applied to each of the channels(red, green, blue).

$$\begin{aligned} Brighten_R &= ((red * (1 - Intensity)/255 + Intensity) * 255) \\ Darken_R &= ((red * (1 - Intensity)/255) * 255) \end{aligned} \tag{4.3}$$

To have the ability of blending colours three methods are used. The normal blend equation is shown in 4.2. To calculate the new colour using

"difference blend" and "overlay blend" methods the following equations can be applied to each channel of the selected colour mode. Figures 4.6a and 4.6b show the results of using these equations. These results are made using the brush bitmap which is used for Figure 4.5c. And for all of them, r1 is set by the colour of brush center and r2 is each pixel's colour on the canvas.

$$\begin{aligned} Difference_R &= ((r1 - r2) * Intensity) + (r2 * (1 - Intensity))) \\ if(r1 > 128) \quad Overlay_R &= 255 - (255 - r1) * (255 - r2)/128 \\ else \quad Overlay_R &= r1 * r2/128 \end{aligned} \tag{4.4}$$

To do the "dodge" and "burn" the following equations are used:

$$\begin{aligned} red &= r1 * (1 + 254 * r2/255) \\ Dodge_R &= (r1 * (1 - Intensity) + red * Intensity) \\ \\ red &= 255 - (255 - r1) * (1 + 254 * (255/r2)/255) \\ Burn_R &= (r1 * (1 - Intensity) + red * Intensity) \end{aligned} \tag{4.5}$$

Figures 4.6c and 4.6d show the results of using dodge and burn equations to mix the colours.

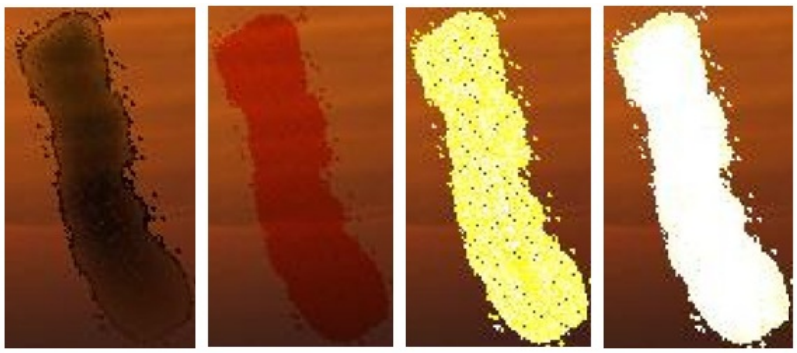


Figure 4.6: An example of different colour blending methods

Table 4.7: Float and Vector Terminals

Name	Description	Type
ERC	Ephemeral Random Number	F
X	Current X location	F
Y	Current Y location	F
L	Current length of brush	F
W	Current width of brush	F
THETA	Current angle of rotation	F
Bitmap	Current bitmap intensity	F
LUMINANCE	Current luminance of source image	F
MeanK K=5,7,9,11,13	Return the mean of K*K square pixels with the center of current (x,y).	F
MedianK K=5,7,9,11,13	Return the median of K*K square pixels with the center of current (x,y).	F
StdK K=5,7,9,11,13	Return the standard deviation of K*K square pixels with the center of current (x,y).	F
MinK K=5,7,9,11,13	Return the minimum value within a 5*5 square pixels with the center of current (x,y).	F
MaxK K=5,7,9,11,13	Return the maximum value within a 5*5 square pixels with the center of current (x,y).	F
SkewK K=5,7,9,11,13	Return the skewness of K*K square pixels with the center of current (x,y).	F
KurtosisK K=5,7,9,11,13	Return the kurtosis of K*K square pixels with the center of current (x,y).	F
SOBEL	Current sobel of source image	V
MOMENT	Current moment of source image	V
CANVAS_C	Current canvas colour	V
SOURCE_C	Current source colour	V
AvgK K=5,7,9,11,13	Calculate the average of K*K square pixels for each channel with the center of current (x,y).	V

Table 4.8: Logic and converter functions

Name	Description	Type
FtoV	Convert its three float child to a vector type	(F,F,F)
VtoF_R	Read and return the red value of its vector type child	V
VtoF_G	Read and return the green value of its vector type child	V
VtoF_B	Read and return the blue value of its vector type child	V
IfLT_F	If the first child is less than the second one, It will first process the third child and then the forth one. Otherwise It will do the revers. It returns the float result value.	(F,F,V,V)
IfLT_V	If the first child is less than the second one, It will first process the third child and then the forth one. Otherwise It will do the revers. It returns the vector result value.	(F,F,V,V)
IfGT_R	If the red value of first child is greater than the red value of second one, It will first process the third child and then the forth one. Otherwise It will do the revers. It returns the vector result value.	(V,V,V,V)
IfGT_G	If the green value of first child is greater than the green value of second one, It will first process the third child and then the forth one. Otherwise It will do the revers. It returns the vector result value.	(V,V,V,V)
IfGT_B	If the blue value of first child is greater than the blue value of second one, It will first process the third child and then the forth one. Otherwise It will do the revers. It returns the vector result value.	(V,V,V,V)

4.5 Fitness Functions

As said before, this is an automatic evolutionary system. Therefore some different evaluation methods are implemented for this research. They can be classified as either aesthetic measure functions or statistical analyses. For aesthetic measures, the deviation from normality is used. For statistical analyses, there are many measures such as mean, standard deviation, and others. More detailed descriptions about them follow.

4.5.1 Aesthetic evaluation

DFN: To calculate the DFN value there are some steps to go through as follows. These steps are from [38].

Step 1: In the first step the colour gradient of an image is found by calculating the red, green and blue values of each pixel (i,j) in the image:

$$|\nabla r_{ij}|^2 = \frac{(r_{i,j} - r_{i+1,j+1})^2 + (r_{i+1,j} - r_{i,j+1})^2}{d^2} \tag{4.6}$$

where $r_{i,j}$ is the red value at pixel (i,j). Using the same formula the value of the green and blue channels are computed. The d value is a scaling factor which is calculated through the following formula:

$$d = 0.001 * \sqrt{(image \quad width)^2 + (image \quad height)^2}$$

Having the colour gradient of each pixel the stimulus of that pixel (i,j) is computed as:

$$S_{ij} = \sqrt{|\nabla r_{ij}|^2 + |\nabla g_{ij}|^2 + |\nabla b_{ij}|^2} \tag{4.7}$$

And finally the response to that pixel colour is computed as:

$$R_{ij} = Log(S_{ij}/S_0) \tag{4.8}$$

where S_0 is the threshold of detection, and is equal to 2. $R_{i,j}$ is set to IGNORE (which is equal to -1) if $S_{i,j} = 0$ or if $Log(S_{ij}/S_0) < 0$. Applying the above formulas, the response for all of the pixels (the whole image) can be found. In the following steps the value of $R_{i,j} = -1$ is ignored in the calculation.

Step 2: The weighted mean and standard deviation are calculated using $R_{i,j}$:

$$\mu = \frac{\sum_{i,j} R_{i,j}^2}{\sum_{i,j} R_{i,j}} \qquad \sigma^2 = \frac{\sum_{i,j} R_{i,j}(R_{i,j} - \mu)^2}{\sum_{i,j} R_{i,j}} \tag{4.9}$$

Having the value of $\mu, \sigma^2 and R_{i,j}$ for each pixel, a histogram is made using a bin width of $\sigma/100$ and weight of $R_{i,j}$ to set the value of corresponding bin in the histogram.

Step 3: The DFN or derivation from normality is calculated in this step by computing the closeness of fit between the response's actual distribution and the hypothesized bell distribution:

$$DFN = 1000 * \sum p_i Log(\frac{p_i}{q_i}) \tag{4.10}$$

where p_i is the observed probability in the *ith* bin of the histogram, and q_i is the expected probability assuming a normal distribution with the computed mean and standard deviation above. When $p_i = 0$, that bin is ignored. A DFN value of zero means that a perfect normal distribution exists, while higher DFN values indicate poorer fits to the normal distribution.

CHISTQ: Colour Histogram Quadratic distance is one of the fitness scores for this research. This means to compare and find the distance between the colour histogram of the generated images by a target image. This method is used in QBIC system such as VisualSEEk [68]. In this method the quadratic distance between two histograms (Histogram_A and and Histogram_B) is calculated using following equation:

$$dist(i,j) = |histogramA_i - histogramB_i| * a(i,j) * |histogramA_j - histogramB_j| \tag{4.11}$$

where i and j are two indexes into the histograms and a(i,j) is the colour similarity within index i and j. To calculate the colour similarity matrix the formula from [79] is used:

$$a(i,j) \equiv 1 - d(c_i, c_j)/d_{max} \tag{4.12}$$

where c_i and c_j are *ith* and *jth* colours and $d(c_i, c_j)$ is MTM (Mathematical Transform to Munsell [49]) distance and d_{max} is the maximum distance between any two colours. Since this calculation is done in RGB colour space in this system, the d_{max} is set to $\sqrt{3}$.

To obtain the overall measure of histogram differences, the distance values (dist(i,j)) are added up for all of the possible combinations of i and j considering the histogram size. It should be mentioned that this computation is somewhat time consuming.

Entropy: To calculate this fitness function the Shannon's Entropy formula is used (see section 3.3.2). Two methods are used to make a histogram and find the probability of value x occurring in it.

(i) Response Entropy: First one is to use the method of creating histogram for DFN fitness function which mean to use the calculated response data.

(ii) RGB/HSV Entropy: In the second way $|\nabla r_{ij}|^2$ is calculated just using the channels value:

$$|\nabla r_{ij}|^2 = (r_{i,j} - r_{i+1,j+1})^2 + (r_{i+1,j} - r_{i,j+1})^2$$

Similar to other parameters, this is a user option.

4.5.2 Statistical evaluation

Mean and Standard deviation: These two values are calculated by 4.9 equations.

Skewness and Kurtosis: Similar to entropy, there there are two ways to calculate the value of skewness and kurtosis:

(i) Response Skewness and Kurtosis: applied to DFN histogram

(ii) RGB/HSV Skewness and Kurtosis: applied to canvas image

In both calculations following formulas are used:

$$Skewness = \frac{\sum(x-\mu)^3}{n * \sigma^3} \qquad Kurtosis = \frac{\sum(x-\mu)^4}{n * \sigma^4} - 3 \tag{4.13}$$

Chapter 5

Performance Analyses

This chapter contains two categories of experiments that statistically analyze CHISTQ and DFN fitness performance. The goal is to show how these fitness criteria can be affected by changing some parameters.

5.1 Colour Histogram Experiments

The goal of the colour histogram experiments is to show that colour histogram performance is affected by selection of various parameters. In other words, these experiments investigate how applying a colour target model to a target image can be affected by parameter decisions. Experiment 1 is the reference (base) experiment and the other three ones are variants to be compared with it.

5.1.1 Experiment Setup

There are four experiments for colour matching test. Table 5.1 shows the GP parameters for all of these experiments:

Table 5.1: GP parameters

Parameter	Value
Generations	15
Population Size	500
Tournament Size	3
Crossover Rate	90%
Maximum Crossover Depth	17
Mutation Rage	10%
Maximum Mutation Depth	17
Prob. of Terminals in Cross/Mut	10%
Initialization Method	Half-and-Half
Tree Grow Max/Min	5 / 5
Tree Full Max/Min	5 / 7

Table 5.2: Other parameters

Parameter	Value
Functions	Sub, Sin, Cos, Min5, Max5, Avg5, Mean5, Std5, Median5, Skew5, Kurtosis5
	Paint4, mixAdd, mixSub, mixMul, mixDiv, mixAbs, mixAvg, mixNBld, mixDBld, mixOBld, mixBrt, mixSin, mixCos,
	FtoV, VtoF_R, VtoF_G, VtoF_B, IfLT_F, IfLT_V, IfGT_R, IfGT_G, IfGT_B
Terminals	X, Y, W, L, THETA, CANVAS_C, SOURCE_C, ERC
# brush bitmaps	5
# brush sizes	4
Processing Model	Process the whole image to find the next unpainted pixel

The function and terminal sets are the same for all of these experiments (see Table 5.2). All the brush bitmaps, brush sizes and number of brushes are the same as well. For all of these experiments, pixel selection scheme in whole image to find the next unpainted pixel.

Figure 5.1: Colour target image for Experiment 1, Experiment 2 and Experiment 4

Figure 5.2: Source Image for all experiments

The formatting of experiments are:

- Experiment 1: DFN, MEAN, STD and CHISTQ

  Four fitness scores are used: DFN (target value is 0.0), mean (target value is 3.03), standard deviation (target value is 0.75) and CHISTQ (target value is 0.0). Normalized sum ranks is used with weight equal to 1. Figure 5.1 is used as the colour target image for colour histogram matching. Figure 5.2 shows the source image.

- Experiment 2: Weighted CHISTQ

  Same as experiment 1 above, except a weight of 3.0 is set for CHISTQ.

- Experiment 3: Target colour Image

  This experiment is like to Experiment 1, except a different in target colour image for CHISTQ test. Here the source image (Figure 5.2) is used as the target colour image.

- Experiment 4: DFN, Mean and CHISTQ

  Here the standard deviation is removed from fitness scores.

5.1.2 Results

For each of these experiments 10 runs are executed. The average CHISTQ scores of each generation for all runs are calculated. Figure 5.3 shows these results. According to this chart in all the experiments CHISTQ score is improved during evolution. Changing the parameters in Experiment 2, Experiment 3 and Experiment 4 result in better CHISTQ scores.

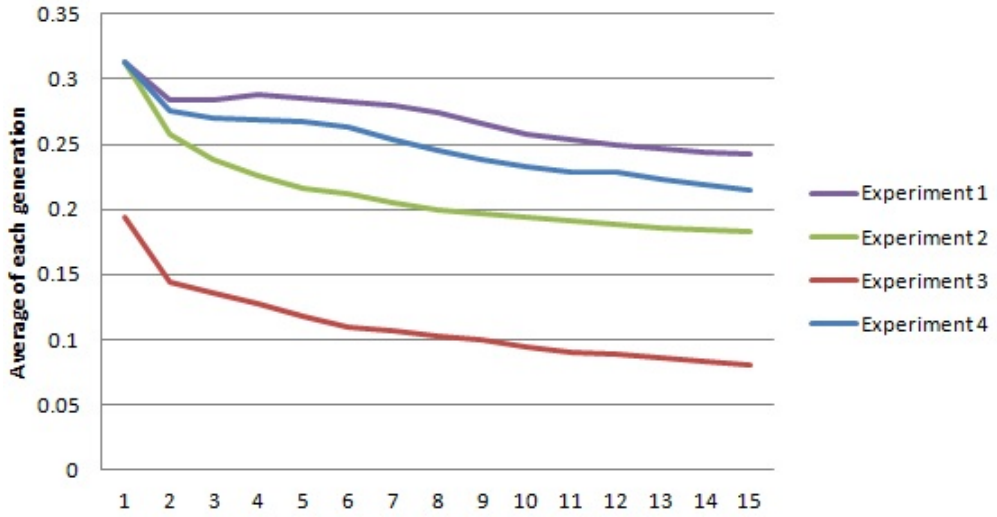


Figure 5.3: Compare average population score of CHISTQ during the generations

	Experiment 1	Experiment 2	Experiment 3
Experiment 2	←		
Experiment 3	←	←	
Experiment 4	-	↑	↑

Table 5.3: Mann-Whitney test scores of CHISTQ experiments with $\alpha = 0.05$. Each arrow points to the superior experiment and a dash means no statistically significant difference

In Experiment 3, the source image is used as target colour image and the results show it has the best effect in having lower values in colour matching scores. This seems reasonable since the target is the same as the source (canvas image). The change in Experiment 2 by putting a weight of 3 for CHISTQ also causes to have lower scores compared to not having a weight. This is because the CHISTQ has more bias in the overall fitness score. And finally, by reducing the number of fitness objectives for Experiment 4, the colour matching score has a better chance to improve. The average of CHISTQ scores for Experiment 4 shows this improvement compared to Experiment 1.

To statistically analyze and compare the experiments, the Mann-Whitney test is used. This non-parametric test is selected because the data being analyzed do not necessarily have normal distribution. In this test, if the absolute value of the z_score is greater than the absolute value of z_critical we can say the difference is statistically significant. The average CHISTQ scores of last generation of ten runs are used to compare the experiments and the α is equal to 0.05 for all these tests. Figure 5.3 show the results of this test. Experiment 2 and Experiment 3 have statistically significant difference with Experiment 1. This means by setting a weight and also using the source image as target colour image the improvement is also statistically significant. However removing the standard deviation from fitness scoring makes an improvement in CHISTQ scores, it is not statistically significant according to Mann-Whitney test.

From 10 runs of each experiment, 3 result images are hand picked and illustrated in Figure 5.4. These images are best of the last generation.

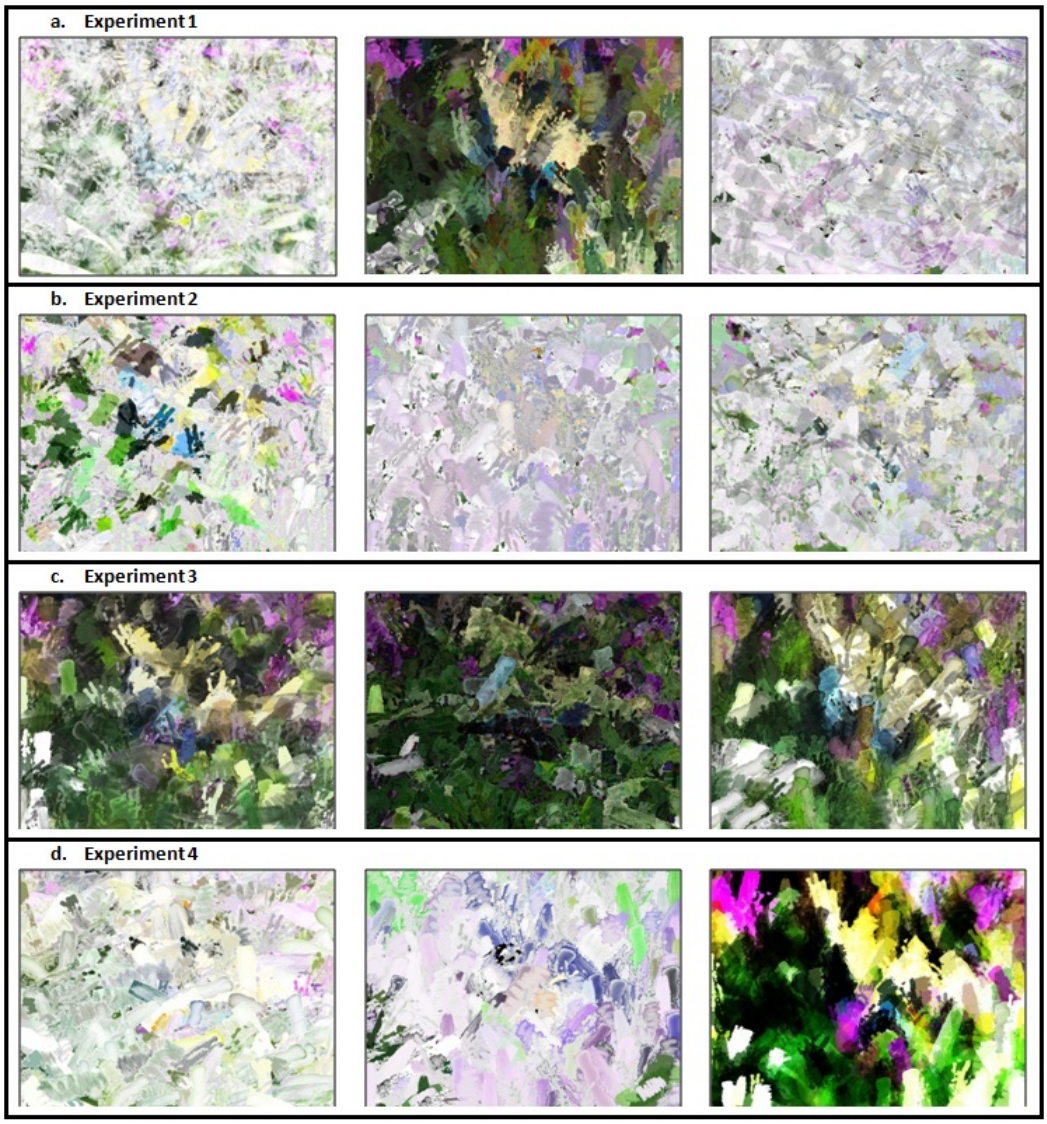


Figure 5.4: Hand picked of best of last generation within ten runs of CHISTQ experiments

5.2 DFN Experiments

The goal of this experiment is to show how changing some parameters can affect DFN scores during evolution. The parameters to examine are: (i) the effect of different brush bitmap styles, such as dark or harsh brushes, light brushes and normal ones; (ii) the effect of setting weights for DFN scores; (iii) the effect of using different target colour images on DFN scores.

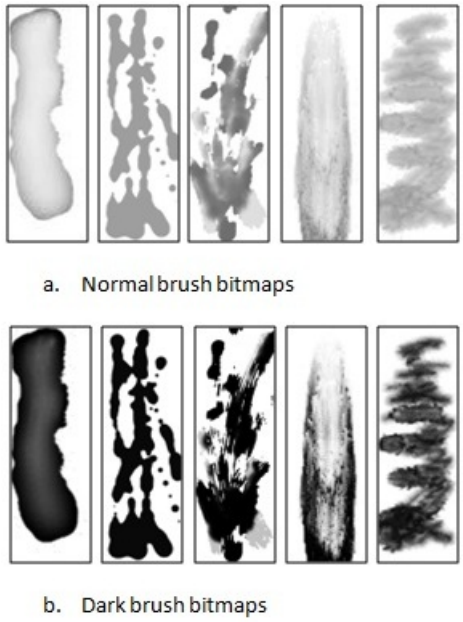


Figure 5.5: Dark and Normal Brush Bitmaps

5.2.1 Experiment Setup

The experiments are:

- Experiment 1: Different Brush styles (Normal - Dark - Bright)

  Here experiments use different brush bitmap styles, such as normal, dark and bright bitmap brushes. Figure 5.5 shows the normal and dark bitmaps used. There are also three groups of bright bitmaps for this experiment (see Figure 5.6). The target colour image is the same as Experiment 1 in CHISTQ tests (Figure 5.1).

- Experiment 2: Weighted DFN

  This experiment sets a weight of three for DFN score, using Bright 3 brush bitmaps (Figure 5.6).

- Experiment 3: Changing Target colour Image

  Different target colour images are used to see the effect of them on DFN value (See Figure 5.7).

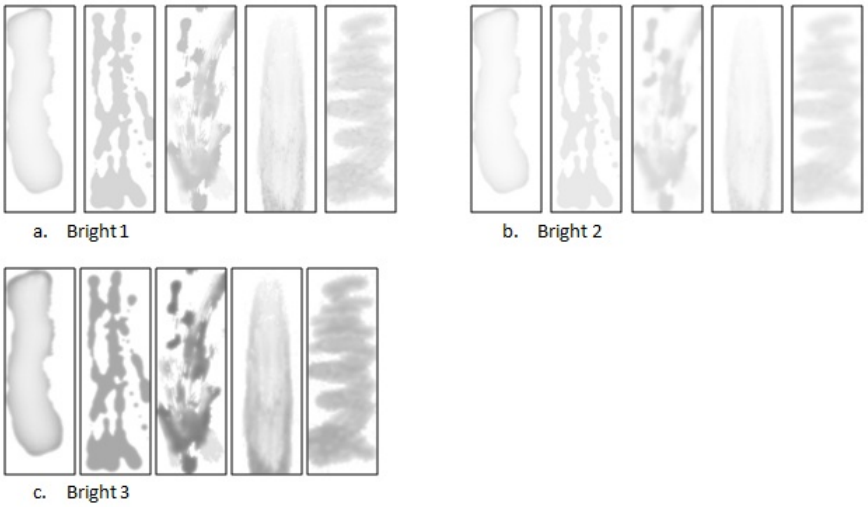


Figure 5.6: Different Bright Brush Bitmaps

In these experiments the same GP parameters are used as in Table 5.1 and Table 5.2. For each of these experiments 10 runs are executed.

5.2.2 Results: Experiment1 - Brush styles

In Figure 5.8, the average of DFN values during evolution for all 10 runs are shown. Bright brush bitmaps caused had larger value for DFN values in the early generations. However, GP improves this score during evolution to values closer to that of normal brush bitmaps. On the other hand, with dark brush bitmaps, there is an improvement during the runs, but it doesn't reach the bright and normal brush scores. Using Mann-Whitney test with $\alpha = 0.05$ shows the difference between the dark and normal is statistically

Figure 5.7: Different Target colour Images

significant. Also the difference between the dark and all bright tests are statistically significant(see Table 5.4).

Some result images using different brush bitmap style are shown in Figure 5.9. For each brush style, within the 10 runs 3 images are hand-selected from the best of last generation. Chapter 7 focuses on the effect of using different brush styles in more detail.

In conclusion using different bright brush bitmaps (see Figure 5.6) and comparing them with normal brush bitmaps show no big difference in DFN scores (see Figure 5.8). The Mann-Whitney tests also show there is no statistically significant difference in the scores.

	Normal	Bright 1	Bright 2	Bright 3
Bright 1	-			
Bright 2	-	-		
Bright 3	-	-	-	
Dark	↑	↑	↑	↑

Table 5.4: Mann-Whitney test scores of DFN for different brush styles with $\alpha = 0.05$. Each arrow points to the superior experiment and a dash means no statistically significant difference

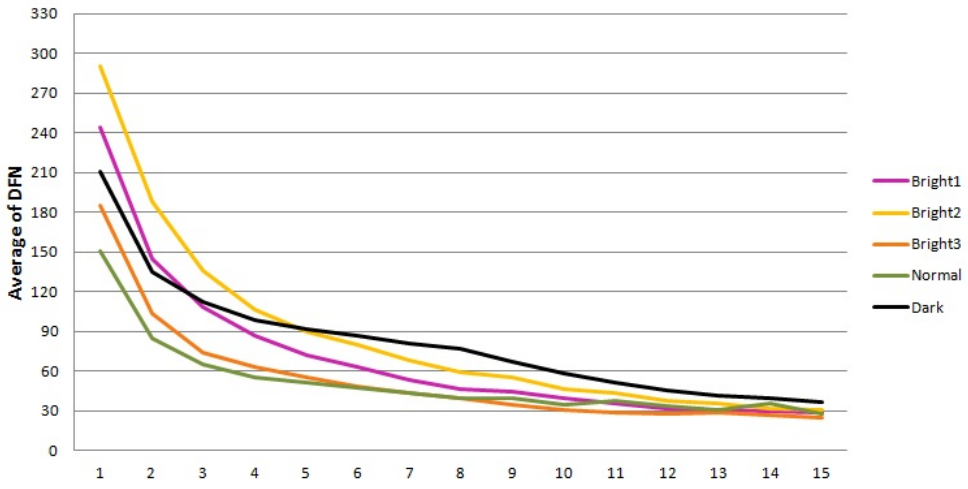


Figure 5.8: Compare average population score of DFN during the generations using normal, dark and different bright bitmaps

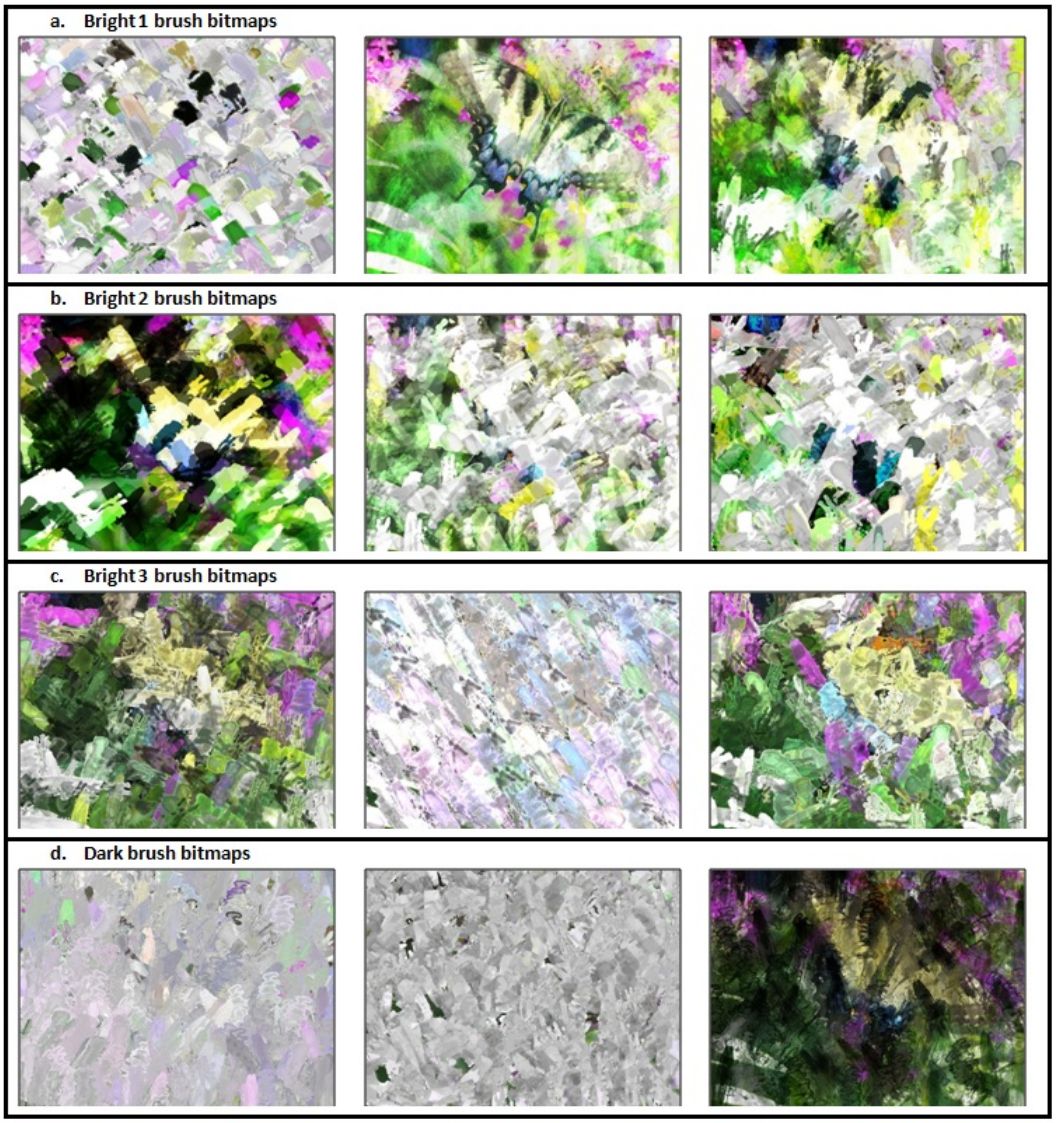


Figure 5.9: Hand picked of best of last generation within ten runs of brush styles experiments

	Normal	Bright 3
Bright 3	-	
Weighted Bright 3	←	←

Table 5.5: Mann-Whitney test scores of DFN for using weight of three with $\alpha = 0.05$. Each arrow points to the superior experiment and a dash means no statistically significant difference

5.2.3 Results: Experiment2 - Weighted DFN

In the second experiment the setting weight for DFN scores is studied on one of the bright brush bitmaps (Bright 3), and the results are compared with experiments from just using the Bright 3 brush bitmaps, and also using normal ones. For both of these comparisons the Mann-Whitney test with $\alpha = 0.05$ shows the improvement (or difference) is statistically significant (see Table 5.5). Figure 5.10 shows three best result images of last generation within ten run for weighted DFN experiment.

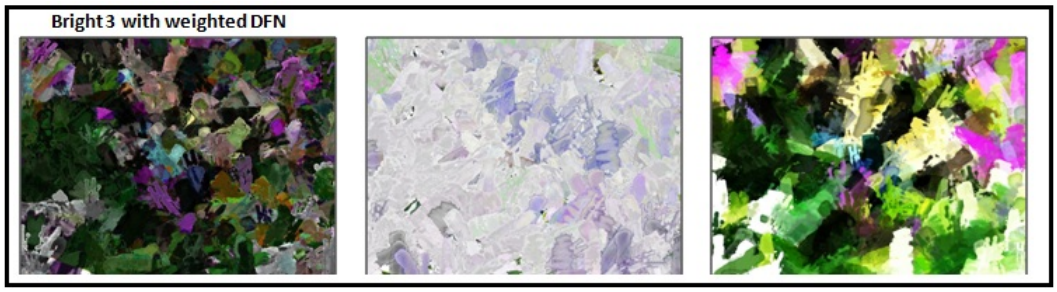


Figure 5.10: Hand picked of best of last generation within ten runs of Bright 3 with weighted DFN

5.2.4 Results: Experiment3 - Target colour image

Figure 5.7 shows different target colour images used to investigate the effect of them on DFN score. Figure 5.11 shows the average of DFN score during the generations for ten runs. DFN values are improved for all the experiments. However Target 4 has the poorest DFN score at the end of the run. Mann-Whitney with $\alpha = 0.05$ is used to compare the results of each experiments one by one using the DFN value of the last generation for 10 runs. According to this test the difference is statistically significant for some comparisons. The results are shown in Table 5.6. According to this test, Target 1 and Target 2 are the best target colour images to have lower DFN score. And there is no statistically significant different between using Target 1, Target 2 and Target 5 in DFN improvement. This is the case in comparing the results of using source image, Target 3, Target 4 and Target 5.

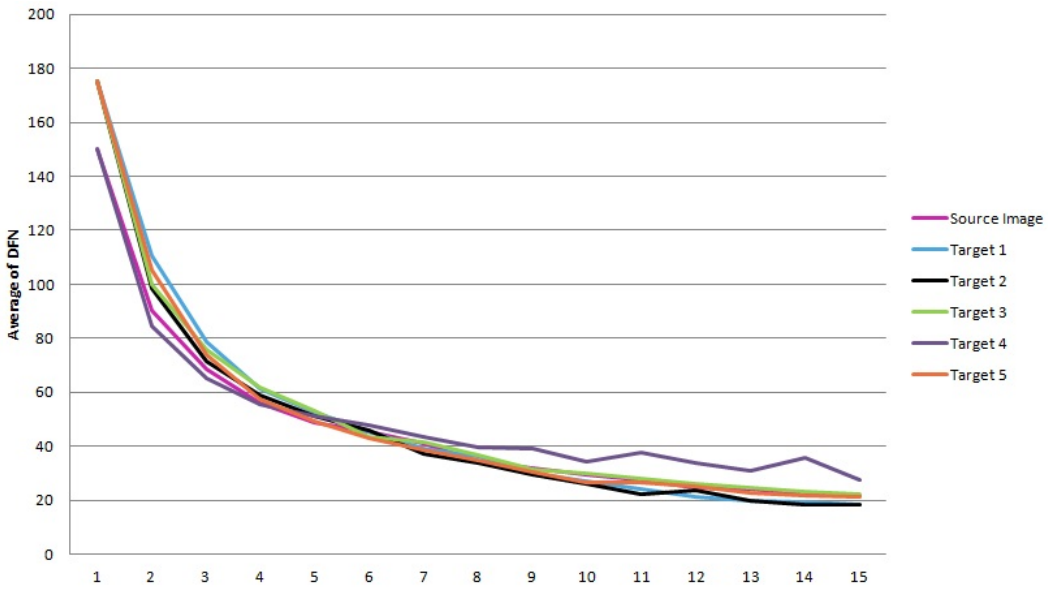


Figure 5.11: Compare average population score of DFN during the generations using different target colour images

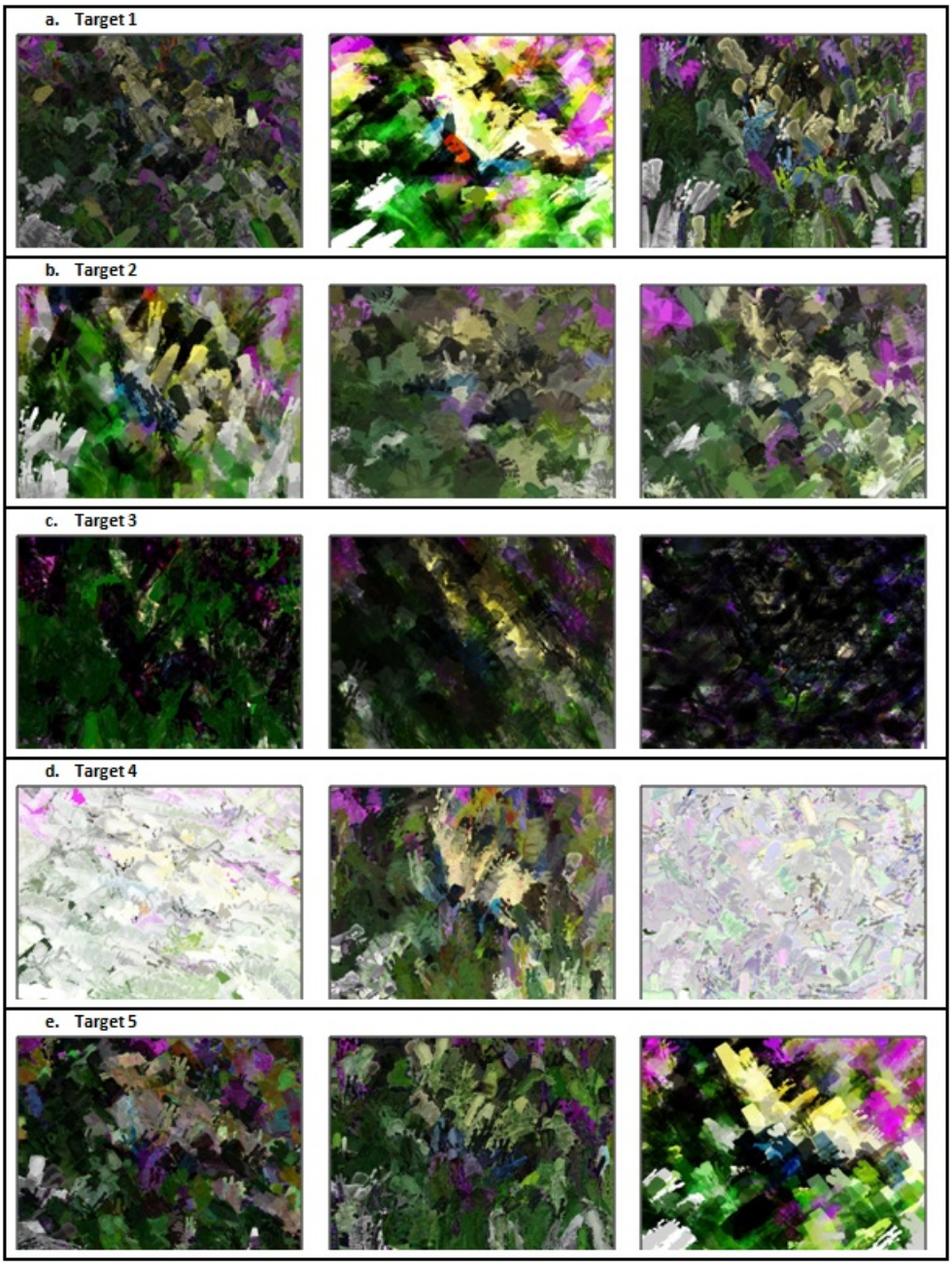


Figure 5.12: Hand picked of best of last generation within ten runs of different target colour image experiments

	Source Image	Target 1	Target 2	Target 3	Target 4
Target 1	←				
Target 2	←	-			
Target 3	-	↑	↑		
Target 4	-	↑	↑	-	
Target 5	-	-	-	-	-

Table 5.6: Mann-Whitney test scores of DFN for using different target colour images with $\alpha = 0.05$. Each arrow points to the superior experiment and a dash means no statistically significant difference

To show the effect of using different target colour images some result images are hand-selected from the best of the last generation within the ten runs (see Figure 5.12). The effect of using different target colour images on created images through the generations is studied in more detail in Chapter 7.

5.3 Conclusion

CHISTQ and DFN performance are studied in this chapter. It is shown that using the source image as target colour image, setting weight for CHISTQ and reducing fitness objects gives more chance to an improved CHISTQ score. By using the source image and setting weight this difference is statistically significant.

Effects on DFN scores were also studied. Dark brush bitmaps are shown to create poor DFN scores compare to normal and bright brush bitmaps. Similar to CHISTQ, setting a weight for DFN causes statistically significant improvement in this score. Different target colour images can also affect DFN score. However, the results are not conclusive.

Chapter 6

Parameter Variation

This chapter focuses on the flexibility of the system to have different style of results by changing some parameters. Each section studies one parameter, for example, the effect of changing brush size, target colour image, brush bitmap, GP language and so forth. For this study three runs are executed for each case and all illustrated images are hand picked from the best of last generation within three runs.

This chapter is not as rigorous as Chapter 6. Here the goal is to show the general effects of different parameter choices. Due to the random nature of evolution these effects are not guaranteed. Rather, the examples show generally seen effects of the parameters.

6.1 Brush Size: Big Vs Small

Here the effect of using different brush sizes on result images is studied. The goal is to show how the result images are changed by fixing all the parameters of this system and using big or small brush sizes. Different variation of experiments are executed. The fixed parameters for these experiments are shown in Table 7.2. However, the GP language, target colour image, brush bitmap style are changed to have six different comparisons. Source Image 1 (see Figure 6.1 a) is used as the canvas image for all these experiment except for sixth and seventh ones. For these last two, Source Image 2 and Source Image 4 are used sequentially (see Figure 6.1 b and d). In each sample, one best image of the last generation within the three runs is hand picked.

Table 6.1: This is fixed parameters list

Parameter	Value
# brush bitmaps	5
# brush sizes	4
Processing Model	Process the whole image to find the next unpainted pixel
Fitness Function	DFN, Mean, Standard Deviation, CHISTQ
Generations	15
Population Size	500
Tournament Size	3
Crossover Rate	90%
Maximum Crossover Depth	17
Mutation Rage	10%
Maximum Mutation Depth	17
Prob. of Terminals in Cross/Mut	10%
Initialization Method	Half-and-Half
Tree Grow Max/Min	5 / 5
Tree Full Max/Min	5 / 7

Table 6.2: First Function Set

Functions	Sub, Sin, Cos
	Paint4, mixAdd, mixSub, mixMul, mixDiv, mixAbs, mixAvg, mixNBld, mixDBld, mixOBld, mixBrt, mixSin, mixCos
	FtoV, VtoF_R, VtoF_G, VtoF_B, IfLT_F, IfLT_V, IfGT_R, IfGT_G, IfGT_B
Terminals	X, Y, W, L, THETA, CANVAS_C, SOURCE_C, ERC
	Min5, Max5, Avg5, Kurtosis5, Mean5, Std5, Median5, Skew5

Figure 6.1: The Source Images

Two different function sets (see Table 7.1 and Table 6.3) and different target colour images (Figure 6.2) are used to have these six experiments.

Table 6.3: Second Function Set

Functions	Add, Sub, Mul, Div, Sin, Cos
	Paint4, Paint_RGB, Mean Shift Segmentation, mixAdd, mixSub, mixMul, mixDiv, mixAbs, mixAvg, mixNBld, mixDBld, mixOBld, mixBrt, mixSin, mixCos
	FtoV, VtoF_R, VtoF_G, VtoF_B, IfLT_F, IfLT_V, IfGT_R, IfGT_G, IfGT_B
Terminals	X, Y, W, L, THETA, CANVAS_C, SOURCE_C, ERC
	Min5, Max5, Avg5, Kurtosis5, Mean5, Std5, Median5, Skew5

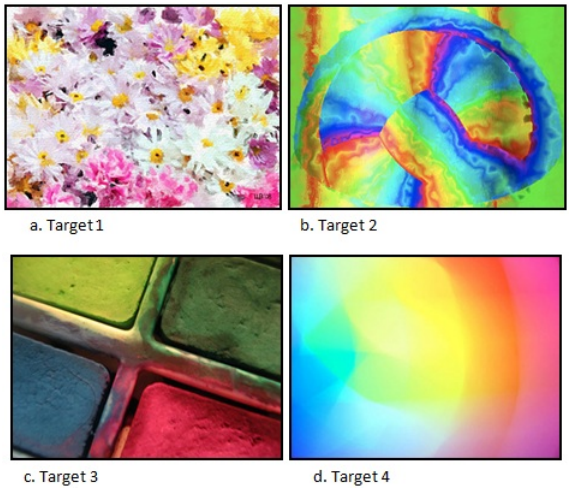


Figure 6.2: Target colour images used for different experiments

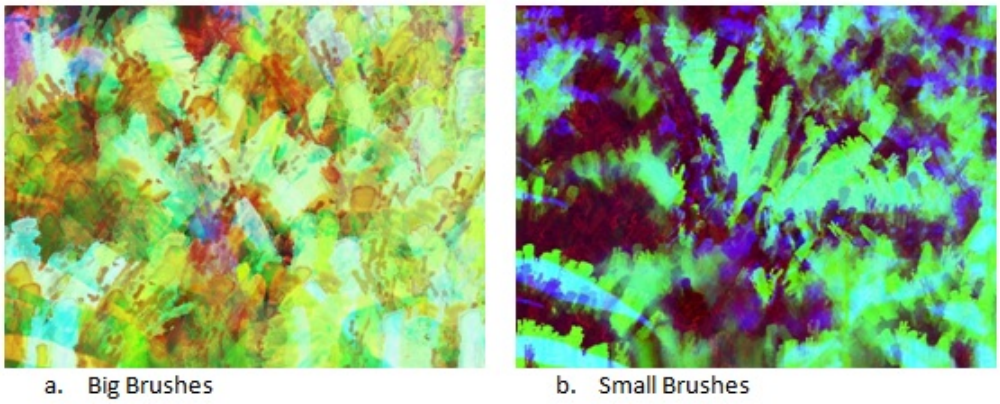
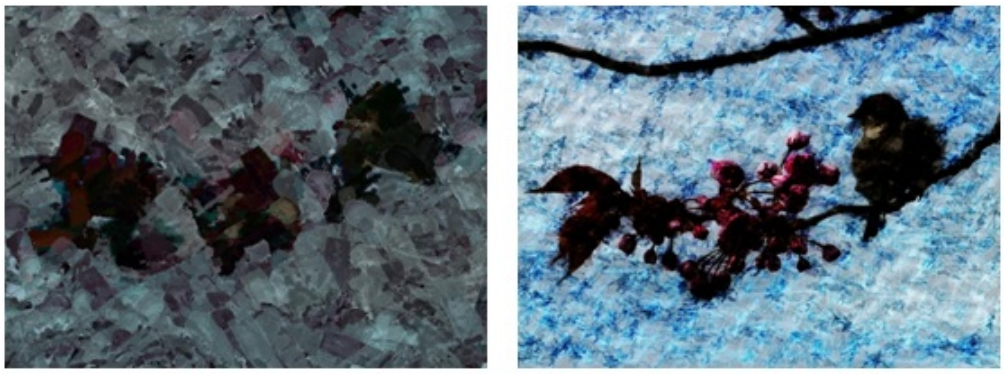


Figure 6.3: First sample of comparing big and small brushes. Table 6.3 is used as function set. Target 2 in Figure 6.2 is used as target colour image.

a. Big Brushes b. Small Brushes

Figure 6.4: Second sample of comparing big and small brushes. Table 7.1 is used as function set. Target 1 in Figure 6.2 is used as target colour image. The bright 1 brush bitmaps are used.

Figures 6.3, 6.4, 6.5 and 6.6 show that using small brushes has the potential for details from the source image to be recognizable in the result. However this is not the case for Figures 6.7 and 6.8. It is obvious that by changing the brush sizes the GP generates different styles of images.



a. Big Brushes b. Small Brushes

Figure 6.5: Third sample of comparing big and small brushes. Table 7.1 is used as function set. Target 2 in Figure 6.2 is used as target colour image.

Figure 6.6: Fourth sample of comparing big and small brushes. Table 7.1 is used as function set. Target 3 in Figure 6.2 is used as target colour image.

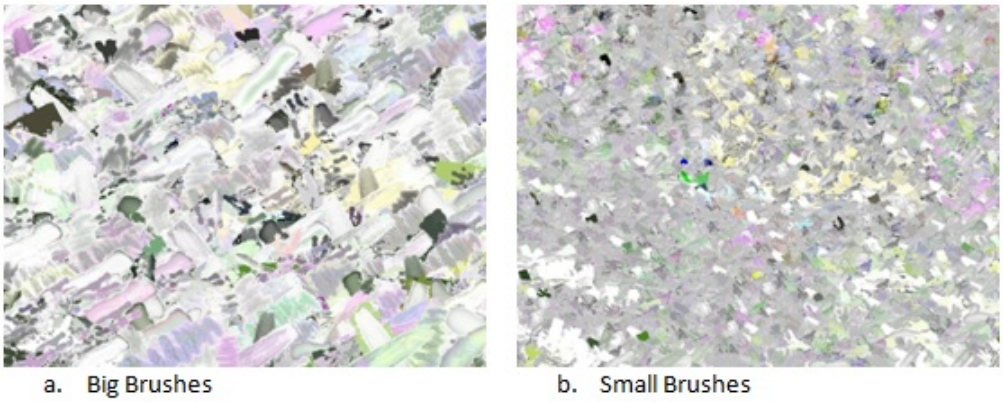


Figure 6.7: Fifth sample of comparing big and small brushes. Table 7.1 is used as function set. Target 1 in Figure 6.2 is used as target colour image.

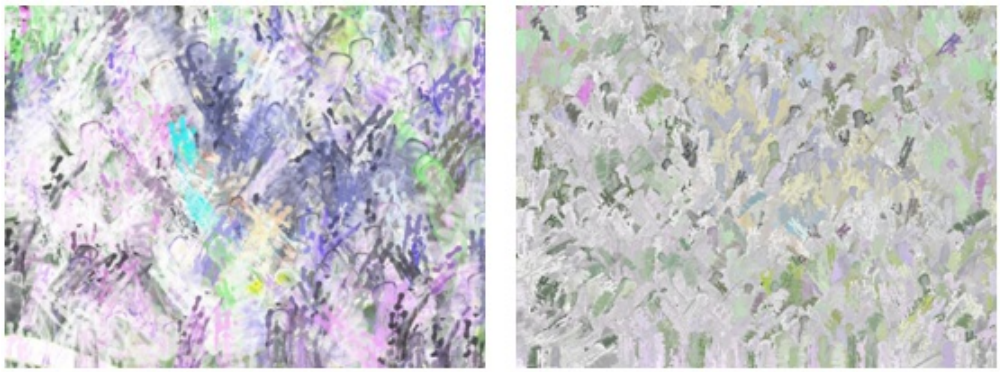


Figure 6.8: Sixth sample of comparing big and small brushes. Table 7.1 is used as function set. Target 1 in Figure 6.2 is used as target colour image. The dark brush bitmaps are used.

The fitness values for each of the sample images and the target values are shown in Table 6.4. The values of mean and standard deviation for all the samples are almost met the target values. However this is not case for DFN and CHISTQ. Comparing the third and fourth samples (Figures 6.5 and 6.6) with the other four ones (Figures 6.3, 6.4, 6.7 and 6.8) shows the effect of using different source images. Using the butterfly source image has less recognizable details in the results, even by using small brush sizes. However the similar characteristics of bird and red flower against the blue background is more recognizable. More details about the effect of source image is discussed in Section 6.2

Note that, the fitness objectives are not explicitly measuring source image details. Thus, the source image details are lost in some cases (Figure 6.3 and others). The effect of having such a measure will be shown with using luminosity matching test in Section 6.10.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Sample 1	Big Brush	15.62	3.01	0.75	0.18
	Small Brush	10.35	3.06	0.74	0.26
Sample 2	Big Brush	15.48	3.02	0.86	0.21
	Small Brush	9.77	2.99	0.73	0.18
Sample 3	Big Brush	14.68	2.97	0.81	0.44
	Small Brush	32.24	3.22	0.75	0.37
Sample 4	Big Brush	15.40	3.00	0.74	0.23
	Small Brush	24.50	3.04	0.75	0.26
Sample 5	Big Brush	21.45	3.02	0.74	0.19
	Small Brush	13.12	3.08	0.74	0.25
Sample 6	Big Brush	14.66	2.96	0.73	0.16
	Small Brush	23.83	3.02	0.77	0.23

Table 6.4: Fitness scores of using big and small brushes for six samples and the target values

6.2 Canvas Images

The goal of this section is to show the effect of using different source images. Having many different experiments (Section 6.1) with Source Image 1 (Figure 6.1) shows that most of the details of the source image were not recognizable in result images. The big difference of this image compared to the other three source images used here is that it does not have a clear subject in the image. The background colour of Source Image 2, Source Image 3 and Source Image 4 is different from the foreground objects in the images. However, the butterfly in Source Image 1 may not be recognizable if it is viewed from a distance.

To show this effect on result images, two experiments are executed on these source image with similar parameters (Table 7.2 and Tabel 7.1). The target colour image is also the same (Target 2 and Target 4 in Figure 6.2). Three runs are executed for each image and one image from the best of last generation is hand-selected (see Figures 6.9 and 6.10.

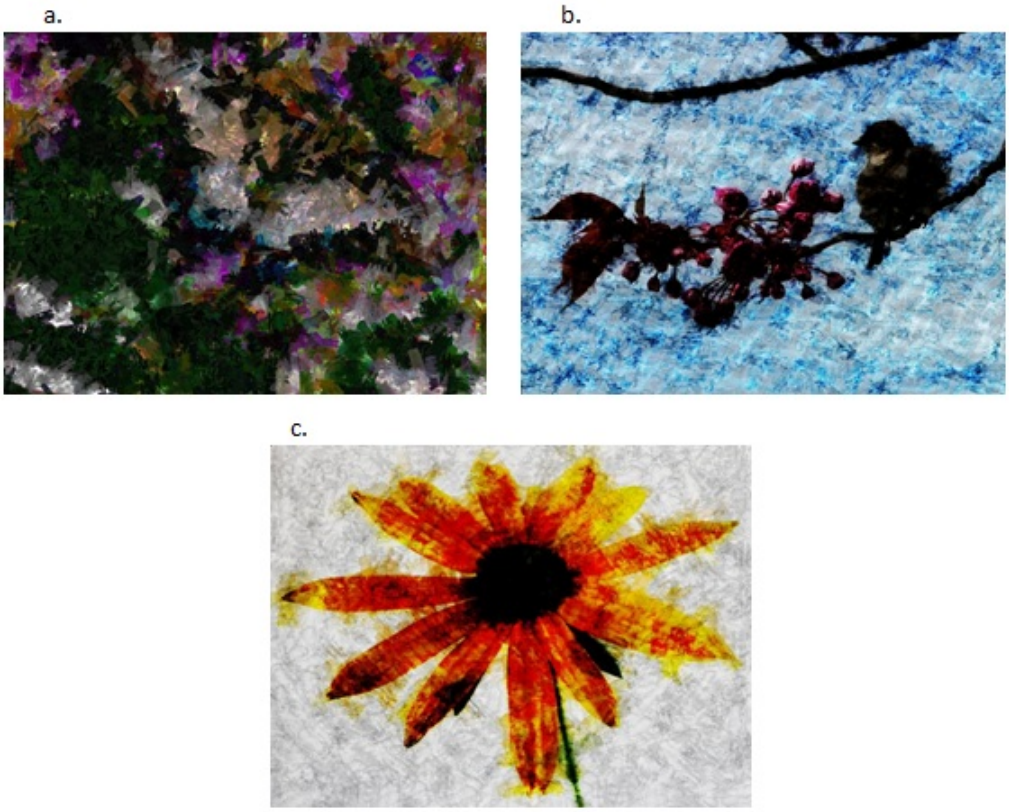


Figure 6.9: First sample of comparing canvas images. Table 7.1 is used as function set. Target 2 in Figure 6.2 is used as target colour image.

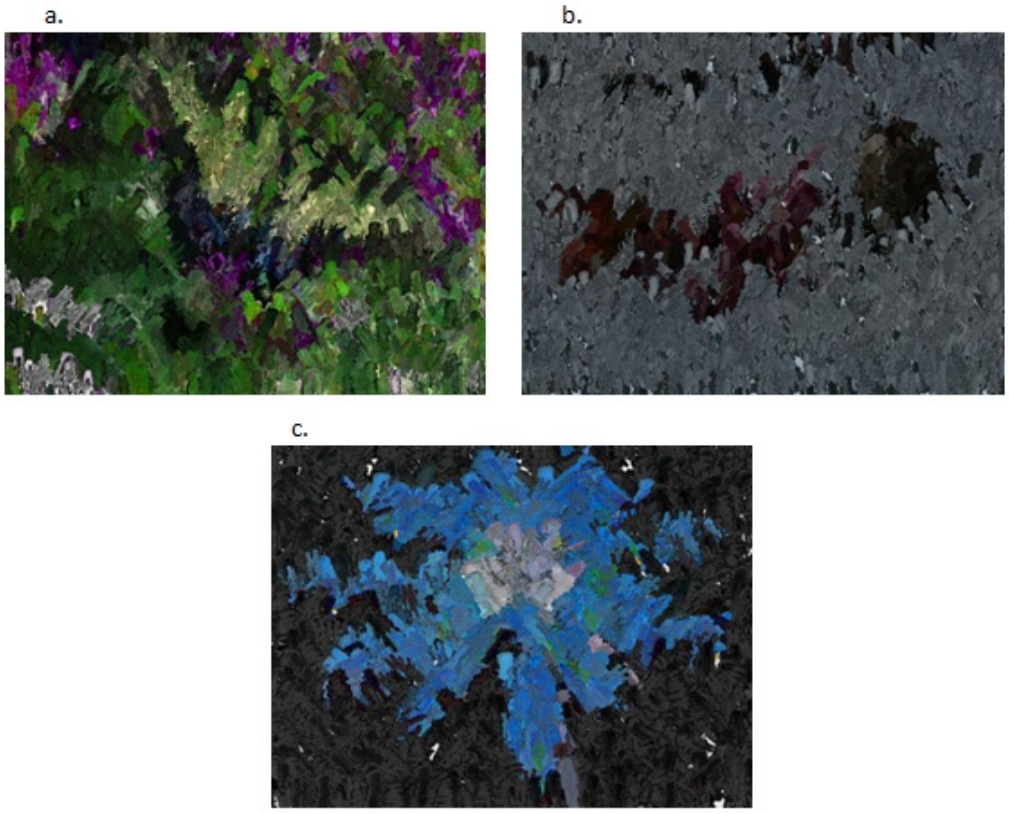


Figure 6.10: Second sample of comparing canvas images. Table 7.1 is used as function set. Target 4 in Figure 6.2 is used as target colour image.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Sample 1	a	10.11	2.99	0.82	0.36
	b	32.24	3.22	0.75	0.37
	c	11.02	2.97	0.74	0.36
Sample 2	a	9.36	3.04	0.84	0.36
	b	30.35	3.03	0.82	0.39
	c	17.44	3.16	0.78	0.34

Table 6.5: Fitness scores of using different canvas images for three samples and the target values

The fitness scores for each of these result images are shown in Figure 6.5. Images b in both samples (Figures 6.9 and 6.10) have the higher DFN value which can be because of its noisier texture that look like noisy compared to the other images.

6.3 Target Colour Image

Here the effect of using different target colour images is studied. The distance between the target colour image and each generated image is calculated using the CHISTQ test. Figure 6.11 shows the target colour images used here. There are six groups of experiment for this study. For experiment in each group 3 runs are executed. The result image from the best of the last generation is hand-selected within the 3 runs and illustrated in this section. The fixed parameters for these experiments are shown in Table 7.2.

Table 6.6: Third Function Set

Functions	Add, Sub, Mul, Div, Sin, Cos
	Paint4, Kuwahara, Mean Shift Segmentation, Median, mixAdd, mixSub, mixMul, mixDiv, mixAbs, mixAvg, mixNBld, mixDBld, mixOBld, mixBrt, mixSin, mixCos,
	FtoV, VtoF_R, VtoF_G, VtoF_B, IfLT_F, IfLT_V, IfGT_R, IfGT_G, IfGT_B
Terminals	X, Y, W, L, THETA, CANVAS_C, SOURCE_C, ERC
	Min5, Max5, Avg5, Kurtosis5, Mean5, Std5, Median5, Skew5

Figure 6.11: Target colour images used for target colour image experiments

In first, second and third samples (Figure 6.12 - Figure 6.13) the Source Image 1 in Figure 6.1 is used and Table 7.1, Table 6.3 and Table 6.6 are used as function set in first, second and third group of experiments sequentially. All these functions sets are different in using paint and NPR functions as in first function set just Paint 4 is used, in second function set Pain 4, Paint_RGB and Mean Shift Segmentation are used. And in third function set Pain 4, Kuwahara, Mean Shift Segmentation and Median are used. Target 3 has more light colours of white and pink and causes having the same effect in result images (see Figure 6.12 c. , Figure 6.13 b. and Figure 6.14 b.). Target 2 causes having more green and blue in the results. Similarly, based on the colour range and gradient of target colour image, the results have different number of colour and texture.

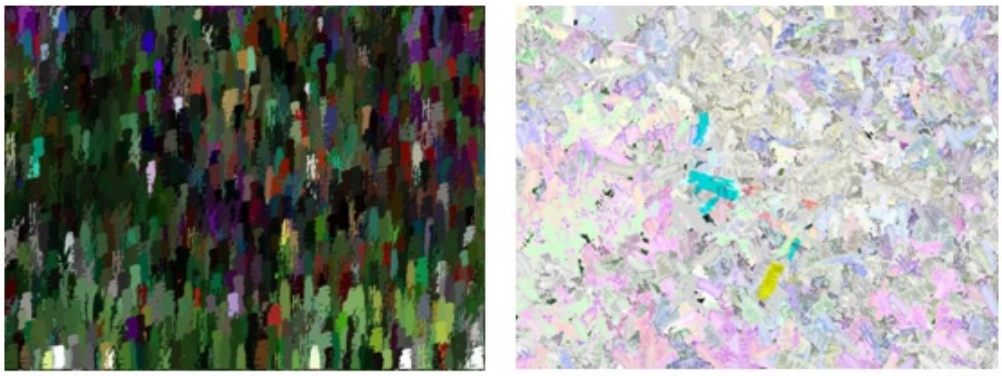


Figure 6.12: First target colour image results. Table 6.6 is used as function set. Target 2 and Target 3 in Figure 6.11 are used.

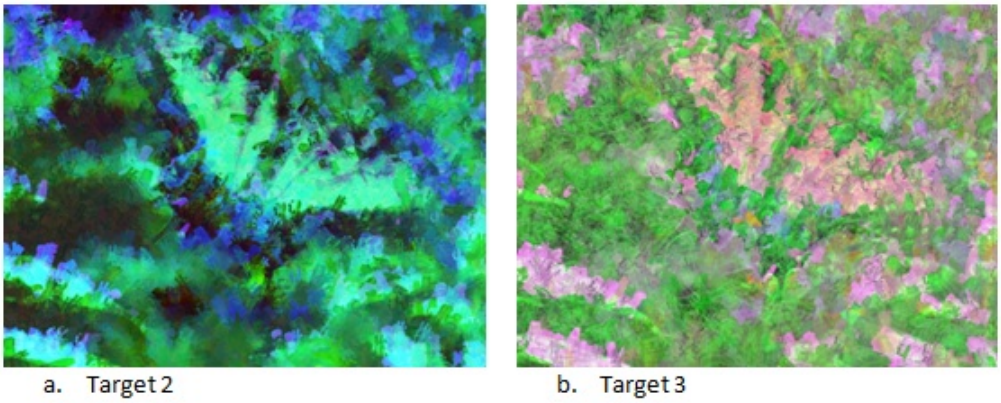


Figure 6.13: Second target colour image results. Table 6.3 is used as function set. Target 2 and Target 3 in Figure 6.11 are used.

In Figure 6.15 to Figure 6.17 the source images are Source Image 3, Source Image 2 and Source Image 4 in Figure 6.1 sequentially. In sample four and five there is an experiment using the source image as target colour image.

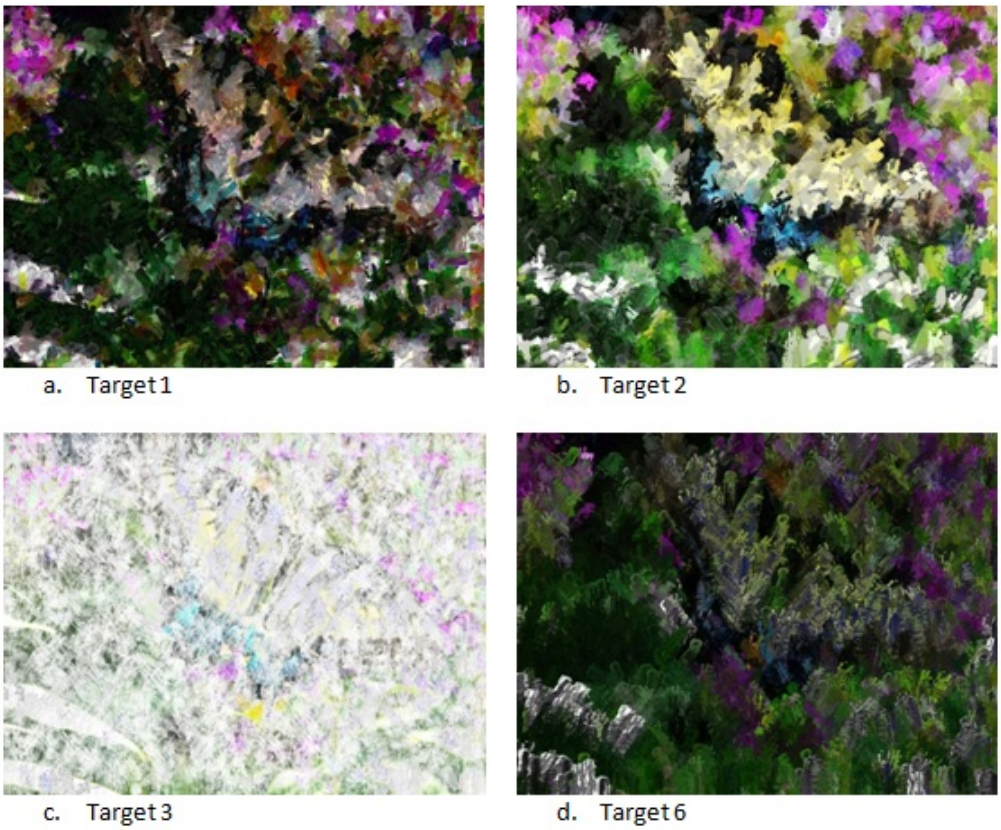


Figure 6.14: Third target colour image results. Table 7.1 is used as function set. Target 1, Target 3, Target 3 and Target 6 in Figure 6.11 are used.

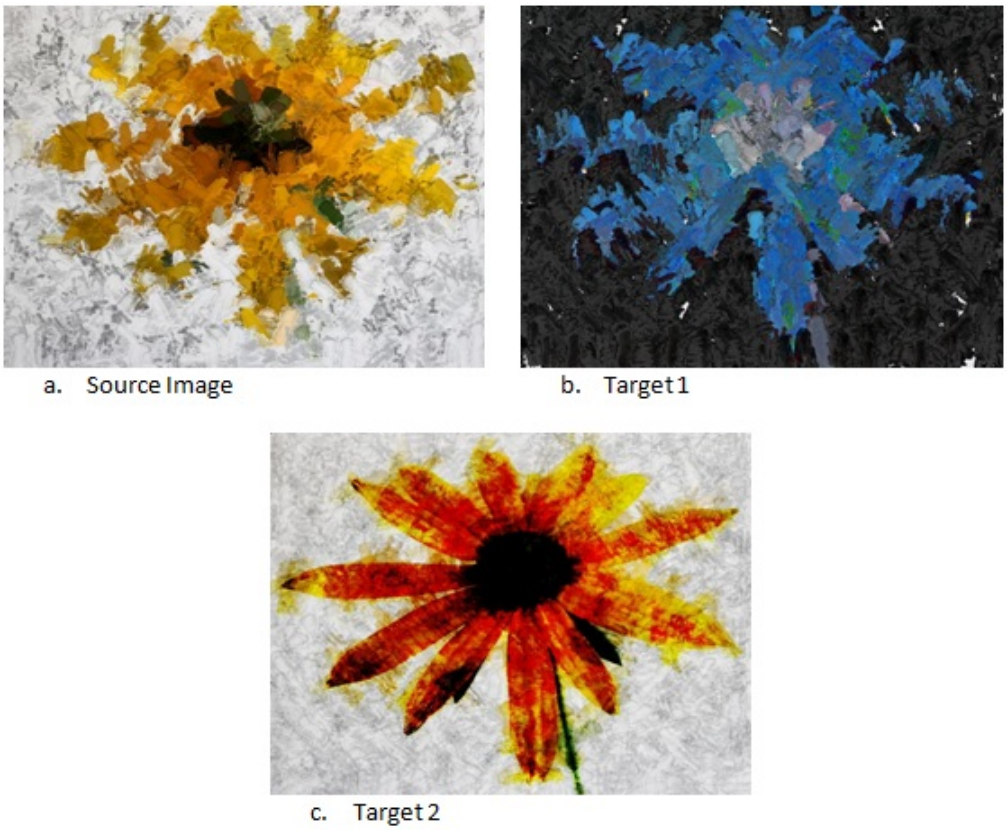


Figure 6.15: Fourth target colour image results. Table 7.1 is used as function set. Source image, Target 1 and Target 2 in Figure 6.11 are used.

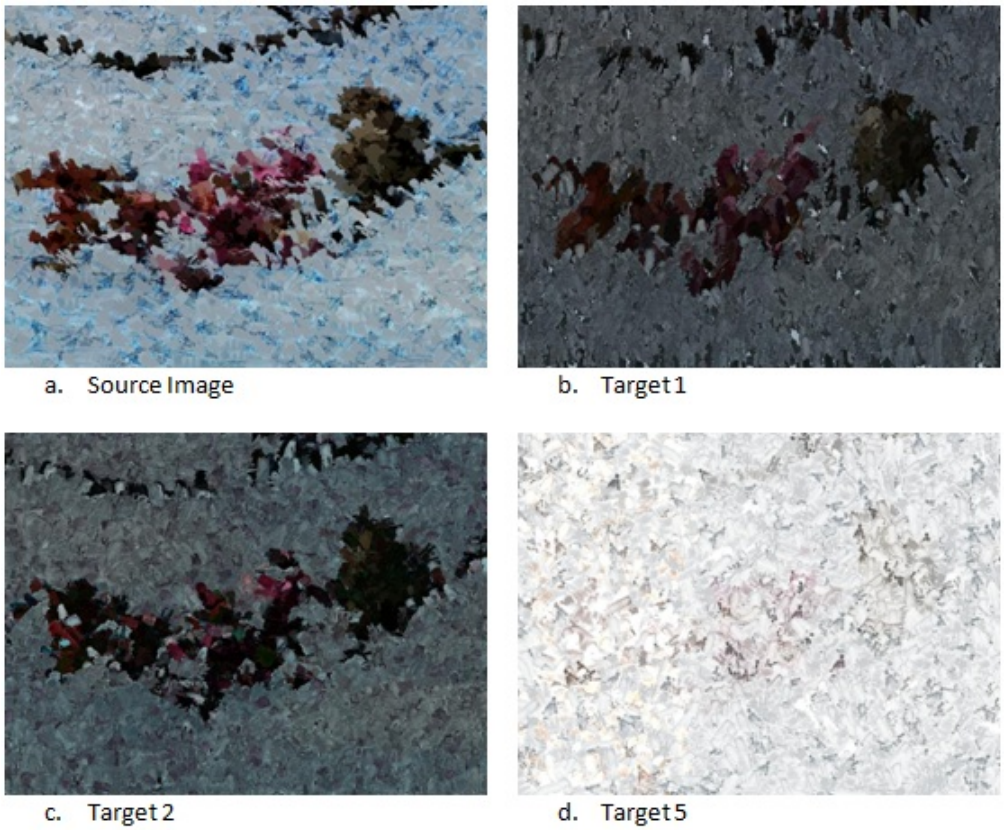


Figure 6.16: Fifth target colour image results. Table 7.1 is used as function set. Source image, Target 1, Target 2 and Target 5 in Figure 6.11 are used.

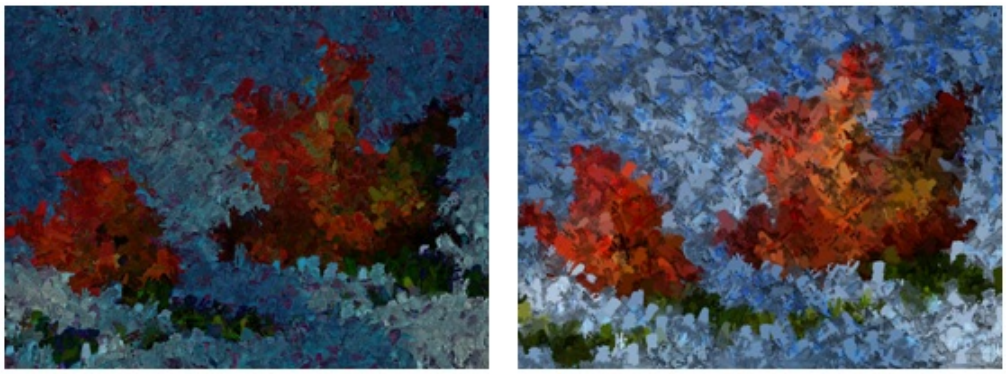


Figure 6.17: Sixth target colour image results. Table 7.1 is used as function set. Target 1 and Target 4 in Figure 6.11 are used.

For all the illustrated samples the fitness scores are shown in Table 6.7. Using Target 3 (Figure 6.11) cause to have better CHISTQ value in sample 1 and sample 3 as the colour range of result images are more near to this target colour image however, it makes poorer DFN value. Using the source image as target colour image also makes lower value for CHISTQ (see sample 4) however, in sample 5 this value is better for Target 5. Target 5 has just two colours (black and white) and more close to its result image (Figure 6.16 c) compare to the result of using source image as target colour image. In sample 6, using Target 4 makes better CHISTQ. Target 4 has green, blue, red and yellow (which is near to light green) and Source Image 4 (Figure 6.1 d) mostly has these 4 colours.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Sample 1	Target 2	37.59	3.45	0.76	0.37
	Target 3	38.25	3.19	0.76	0.15
Sample 2	Target 2	13.40	3.01	0.76	0.29
	Target 3	10.18	3.01	0.73	0.29
Sample 3	Target 1	9.25	3.05	0.83	0.33
	Target 2	10.49	3.07	0.85	0.32
	Target 3	20.59	3.04	0.71	0.19
	Target 6	8.58	3.03	0.81	0.29
Sample 4	Source Image	18.21	3.04	0.87	0.12
	Target 1	17.44	3.16	0.78	0.34
	Target 2	11.02	2.97	0.74	0.36
Sample 5	Source Image	22.27	3.04	0.79	0.16
	Target 1	30.35	3.03	0.82	0.39
	Target 2	18.77	3.03	0.79	0.44
	Target 5	33.06	3.02	0.80	0.12
Sample 6	Target 1	12.06	2.99	0.73	0.35
	Target 4	24.50	3.04	0.75	0.26

Table 6.7: Fitness scores of using different target colour images for six samples and the target values

6.4 Brush Bitmap Intensity

Using different brush bitmap intensities can also affect the results. Table 7.2 and Table 7.1 are the parameters used for the experiments of this section. The experiments are executed in 3 runs and the best of the last generation is hand picked within these runs for each experiment. For these experiments two groups of bitmap brushes are used (figure 6.18). First group has normal, dark and two bright brush bitmaps. In second group normal and bright brush bitmaps are tested.

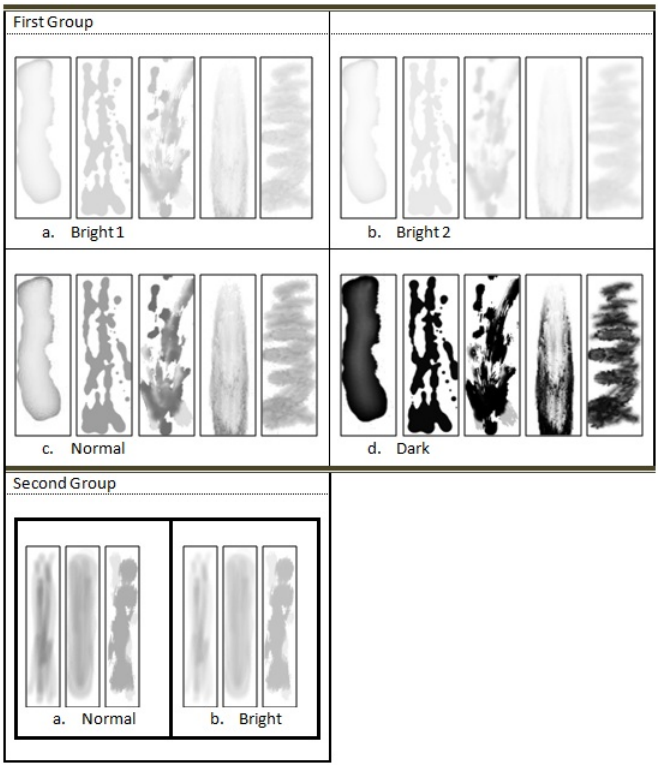


Figure 6.18: Different brush bitmap intensities categorized into two groups.

There are three samples in this section which is illustrated in Figure 6.19, Figure 6.20 and Figure 6.21. Using dark brush bitmaps results in dark, unclear images. However, bright brush bitmaps work vice-versa. Table 6.8 shows the fitness score for each object of these illustrated samples.

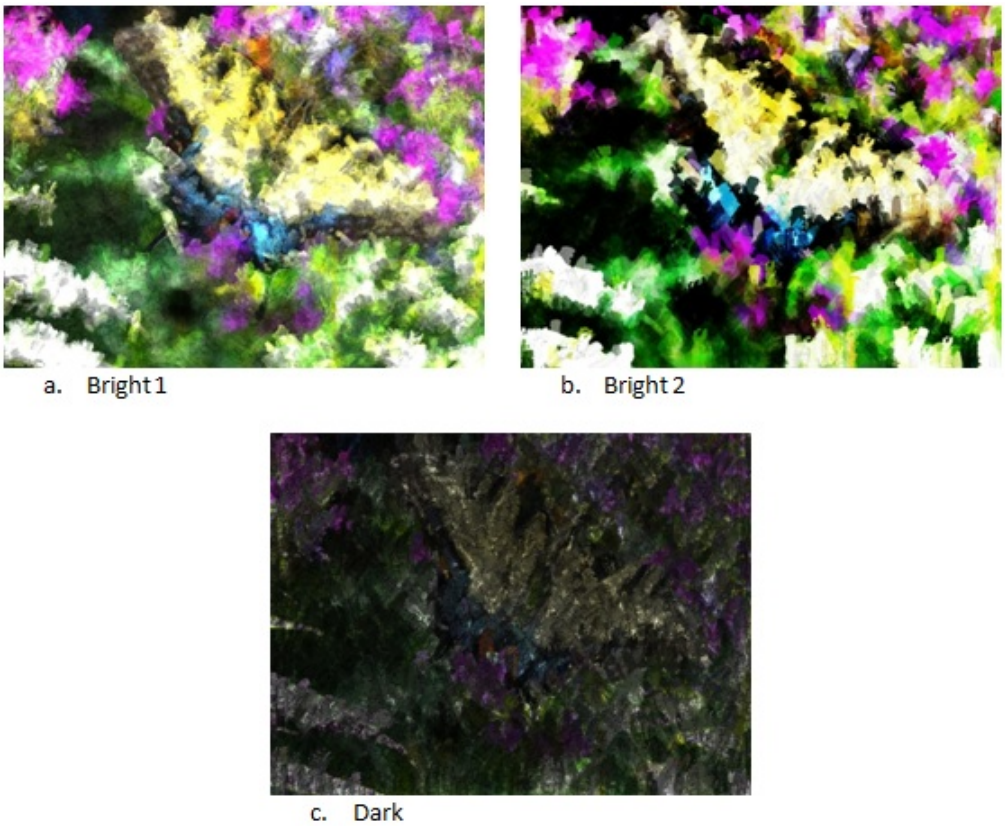


Figure 6.19: First sample of comparing brush intensity. First group of brushes in Figure 6.18 is used. Table 7.1 is used as function set. Source Image 1 in Figure 6.1 is used.

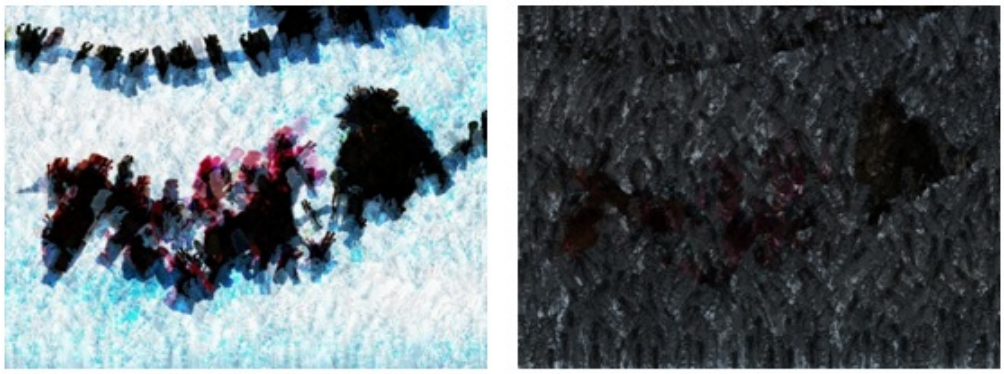


Figure 6.20: Second sample of comparing brush intensity. First group of brushes in Figure 6.18 is used. Table 7.1 is used as function set. Source Image 2 in Figure 6.1 is used.

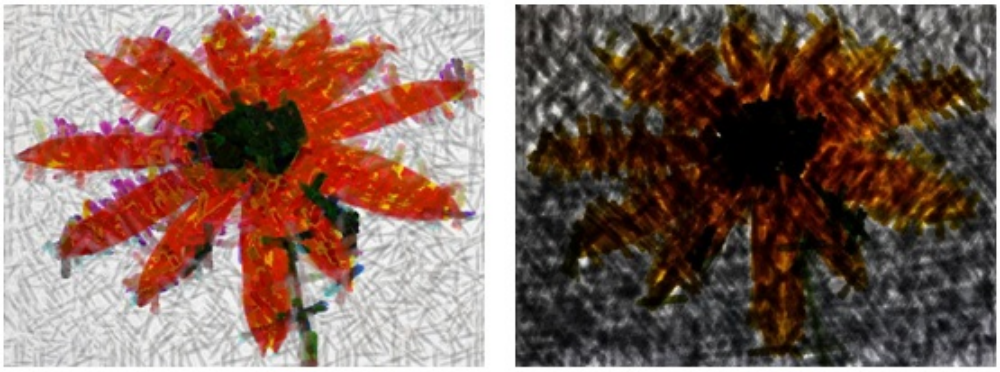


Figure 6.21: Third sample of comparing brush intensity. Second group of brushes in Figure 6.18 is used. Table 7.1 is used as function set. Source Image 3 in Figure 6.1 is used.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Sample 1	Bright 1	14.97	3.07	0.77	0.31
	Bright 2	10.48	3.04	0.92	0.28
	Dark	9.22	2.99	0.77	0.36
Sample 2	Bright 2	11.57	3.05	0.84	0.33
	Dark	22.88	3.03	0.75	0.42
Sample 3	Normal	26.85	3.02	0.77	0.39
	Bright	15.33	3.04	0.69	0.35

Table 6.8: Fitness scores of using different brush intensities for three samples and the target values

6.5 Brush Bitmap Style

Applying different brush bitmap styles also makes different types of images. Here for each experiment just one brush bitmap is used to see its effect. The GP language and other parameters are shown in Table 7.1 and Table 7.2. Six brush bitmaps are studied here which are divided into two groups and applied on two different source images. Each of these six brush bitmaps are used in six separate experiments as the only applied brush bitmaps. Three runs are executed for each experiment and the best of the last generation is hand selected within three runs. The first brush bitmap group and their results are shown in Figure 6.22 and Figure 6.24 and the second group in Figure 6.23 and Figure 6.25.

Table 6.9 shows the fitness score of each illustrated results. Image a in first group has the poorest DFN and CHISTQ scores however the mean values are almost the same for all three images. In second group, Image a has the largest value for DFN however the value of mean, standard deviation and CHISTQ are near each other of all three experiments.

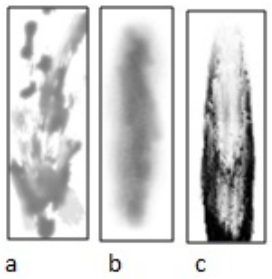


Figure 6.22: First group of brush bitmap styles

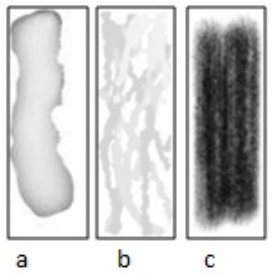


Figure 6.23: Second group of brush bitmap styles

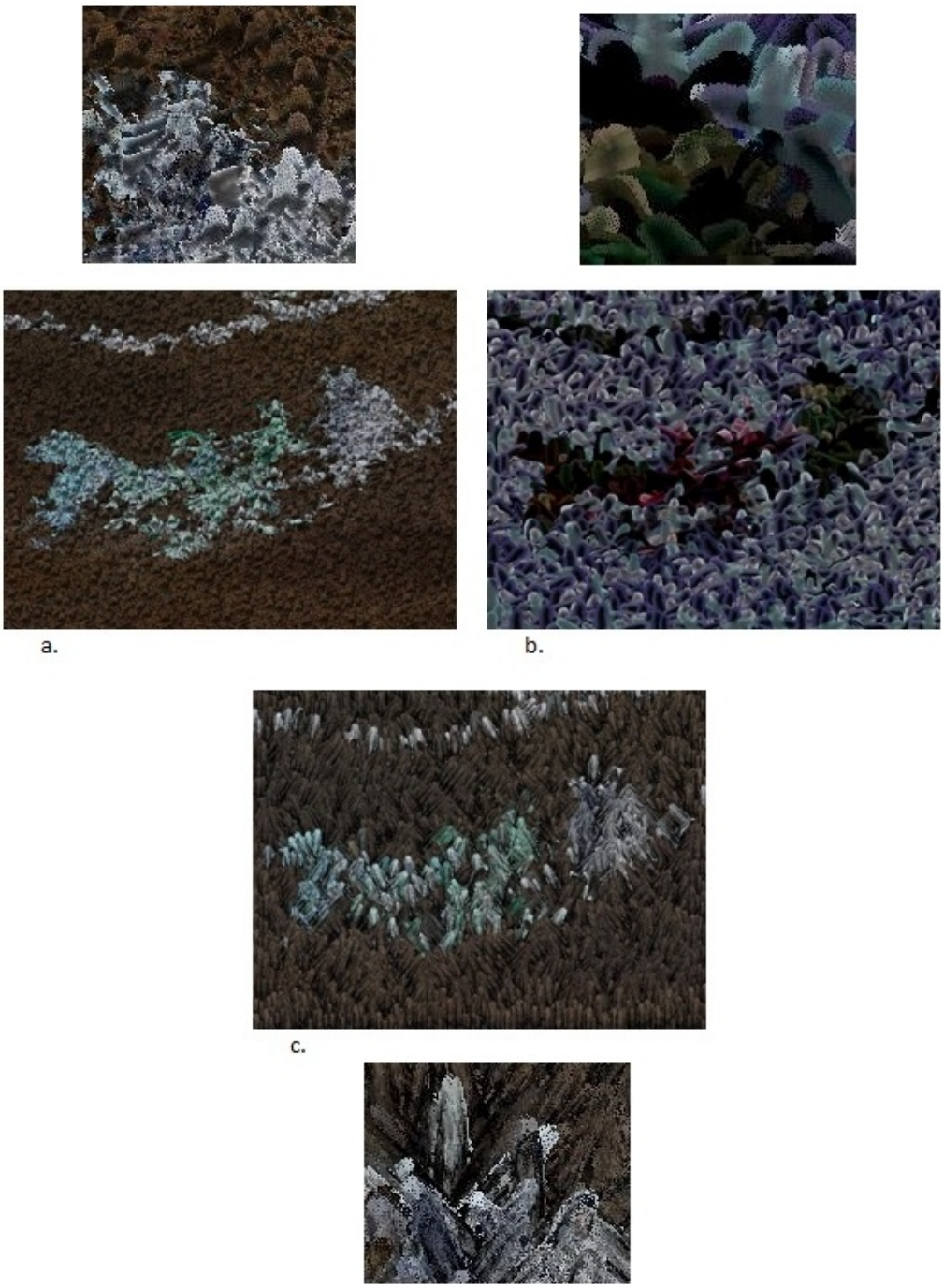


Figure 6.24: Results of first group of brush bitmap styles with details.

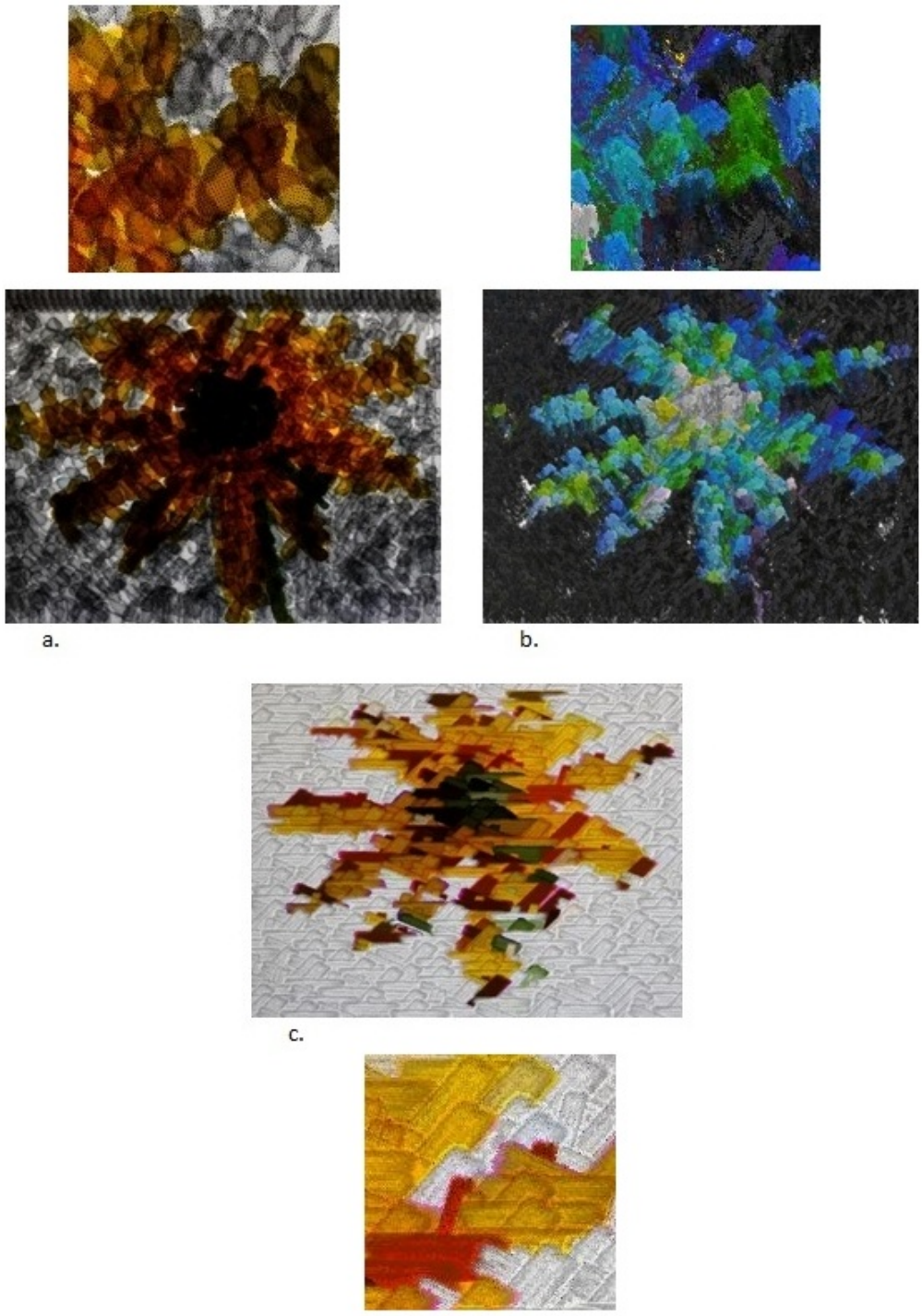


Figure 6.25: Results of second group of brush bitmap styles with details.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
First Group	a	17.62	3.34	0.73	0.73
	b	14.94	3.02	0.82	0.41
	c	13.79	3.13	0.76	0.41
Second Group	a	22.15	3.12	0.75	0.39
	b	11.70	3.03	0.75	0.34
	c	11.63	3.05	0.77	0.39

Table 6.9: Fitness scores of using different brush bitmap styles for six samples and the target values

6.6 Pixel Selection

As described in Chapter 4, there are two pixel selection methods: randomly select a number of unpainted pixels, and find the next unpainted pixel by processing the whole image. Each of these methods creates different effects on resulting images. There are four groups of tests in this section and for each of the experiments there are 3 executed runs. Table 7.2 and Table 7.1 are the parameters for these tests. For the random selection method, the number of selected pixels is 1500 for all experiments. By changing this number, the level of details in the resulting images is changed. To paint the canvas, the unpainted pixels are rendered white.

The best of the last generation is hand selected within 3 runs for each of these samples (see Figures 6.26). Using the random selection method, the results have less detail of the source image but the objects are still recognizable specially in images a-b and g-h in Figures 6.26.

The fitness scores for each illustrated results are shown in Table 6.10. Processing the whole image for the next unpainted pixel has more chance to have better CHISTQ score since there is no white areas in the results.

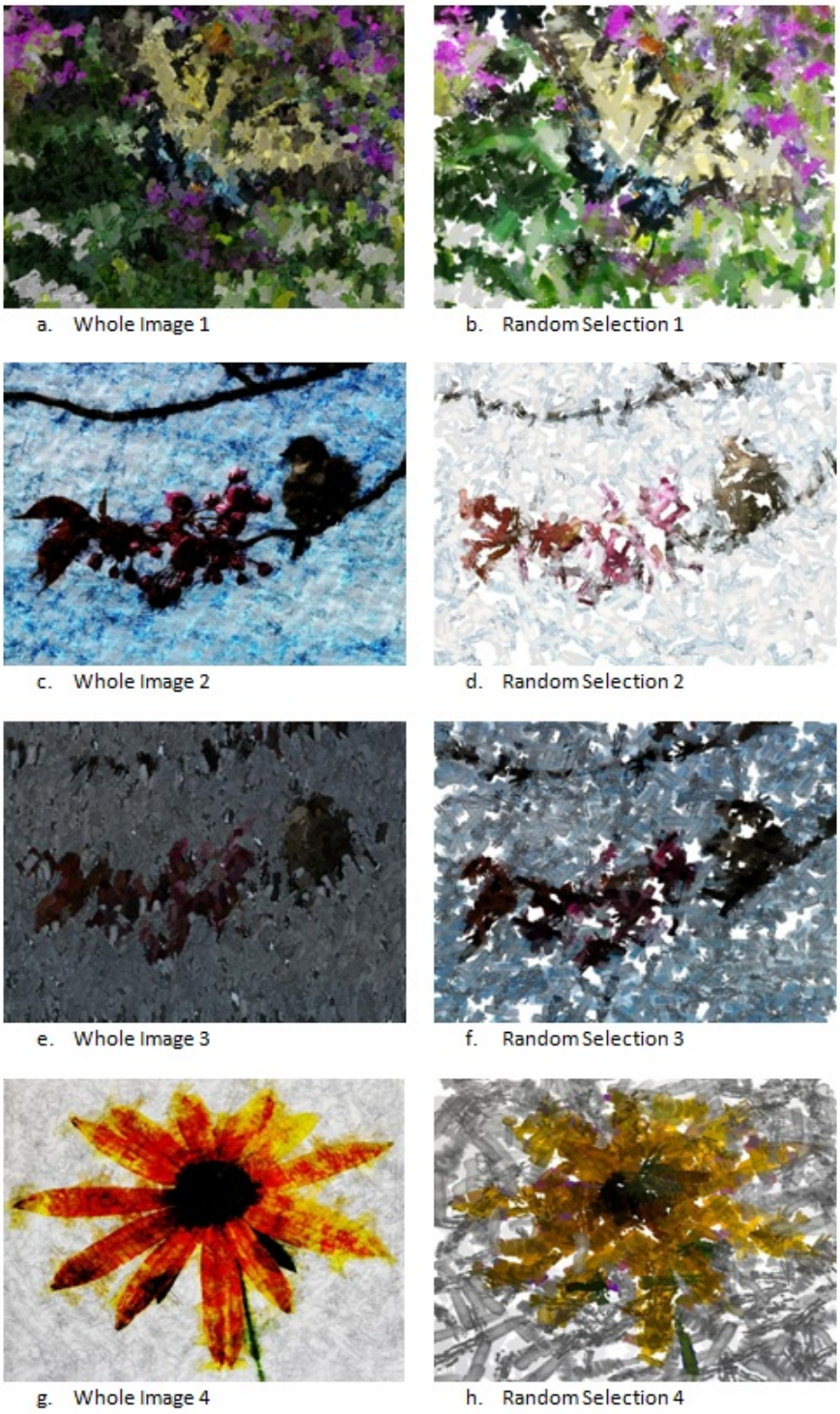


Figure 6.26: Comparing two pixel selection methods.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Sample 1	Random Selection	10.50	3.04	0.84	0.36
	Next Unpainted	10.51	3.16	0.89	0.38
Sample 2	Random Selection	32.24	3.22	0.75	0.37
	Next Unpainted	17.45	3.10	0.85	0.42
Sample 3	Random Selection	30.35	3.03	0.82	0.39
	Next Unpainted	14.11	3.08	0.85	0.41
Sample 4	Random Selection	11.02	2.97	0.74	0.36
	Next Unpainted	17.08	3.18	0.76	0.40

Table 6.10: Fitness scores of using different pixel selection methods for four samples and the target values

6.7 White Canvas

One of the parameters that causes different resulting images is to use a white canvas instead of the pre-painted canvas with the source image. This effect is studied in this section. Five experiments are executed within 3 runs using Table 7.2 and Table 7.1 parameter values. The best of last generation within 3 runs are hand picked and illustrated in Figures 6.27, 6.28, 6.29, 6.30 and 6.31. The difference between Figure 6.27 and Figure 6.28 is to use big and small brush sizes. This causes then to have a similar range of colours in first two samples (The same as Figure 6.29 and Figure 6.30).

The fitness scores for each of illustrated results and the target values are shown in Table 6.11. Using white canvas cause better DFN score for sample 1 and sample 2 however, this is not the case for sample 3 to sample 5. The mean values are almost met in all cases.

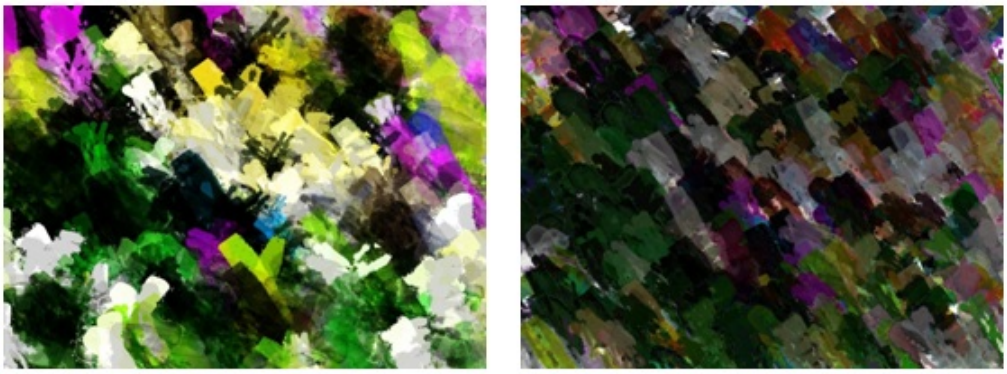


Figure 6.27: First sample of comparing white and source image canvas. Source Image 1 in Figure 6.1 is used. The target colour image is Target 2 in Figure 6.2. Big brush sizes are also used.

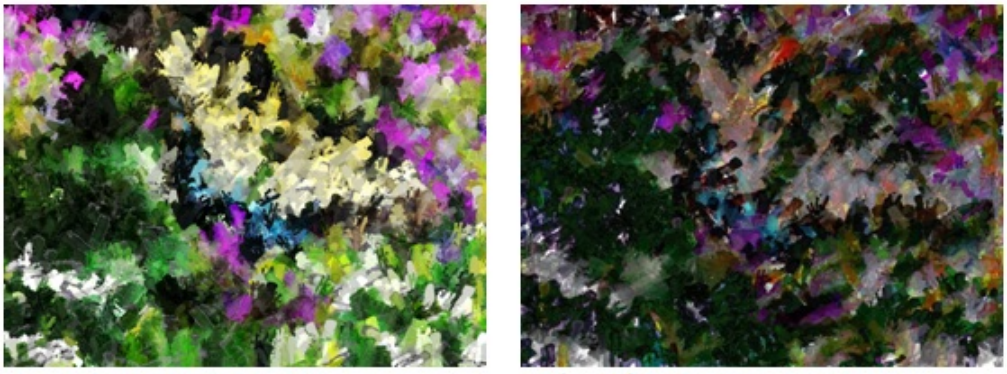


Figure 6.28: Second sample of comparing white and source image canvas. Source Image 1 in Figure 6.1 is used. The target colour image is Target 2 in Figure 6.2. Small brush sizes are also used.

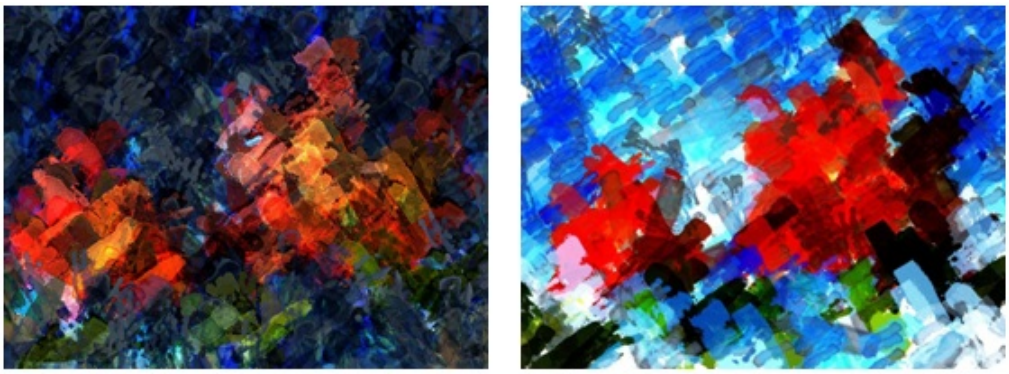


Figure 6.29: Third sample of comparing white and source image canvas. Source Image 4 in Figure 6.1 is used. The target colour image is Target 4 in Figure 6.2. Big brush sizes are also used.

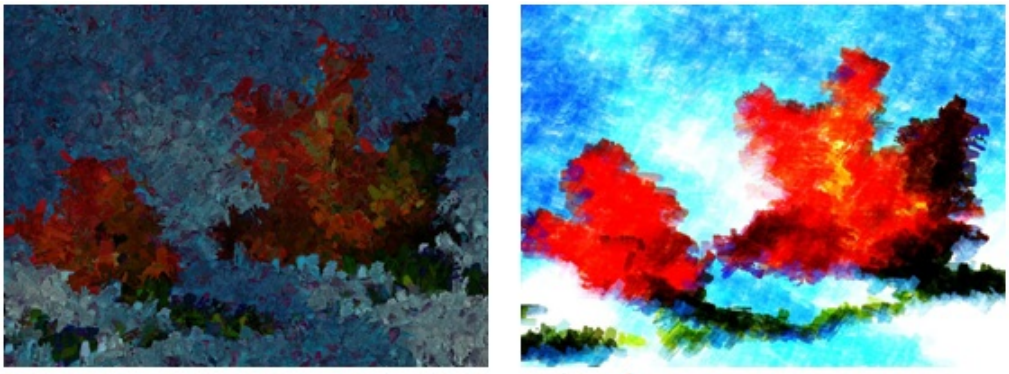


Figure 6.30: Fourth sample of comparing white and source image canvas. Source Image 4 in Figure 6.1 is used. The target colour image is Target 4 in Figure 6.2. Small brush sizes are also used.

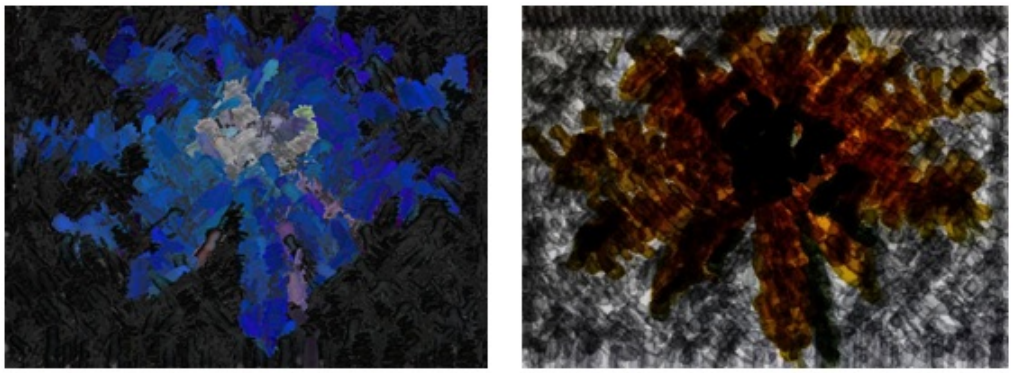


Figure 6.31: Fifth sample of comparing white and source image canvas. Source Image 3 in Figure 6.1 is used. The target colour image is Target 4 in Figure 6.2. Small brush sizes are also used.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Sample 1	Source image Canvas	12.62	3.02	0.90	0.31
	White Canvas	8.14	2.99	0.84	0.36
Sample 2	Source image Canvas	10.49	3.07	0.85	0.32
	White Canvas	8.33	3.03	0.77	0.37
Sample 3	Source image Canvas	12.82	3.09	0.75	0.32
	White Canvas	17.15	3.05	0.79	0.23
Sample 4	Source image Canvas	11.54	3.03	0.73	0.35
	White Canvas	20.20	3.02	0.81	0.23
Sample 5	Source image Canvas	27.31	3.04	0.75	0.34
	White Canvas	32.23	3.08	0.75	0.35

Table 6.11: Fitness scores of using white and source image canvas for five samples and the target values

6.8 GP Fitness test

Different combinations of GP fitness objectives can be used to have different types of result images. In this section five experiments are studied. The parameters and function set can be found in Table 7.2 and Table 7.1. For each experiment 3 runs are executed and the best of the last generation images are illustrated here (See Figures 6.32 and 6.33). The CHISTQ test is used for all experiments. DFN is not used in first and second experiments (Figure 6.32) and the resulting images look more like a filtered photograph however, using luminance direct match in images a, makes it near to a watercolor or mixed media painting. Using Entropy makes brush effects in the second result of images b. CHISTQ and DFN are used as two objectives for images c, d and e (Figure 6.33). However they are different in the third score which is Entropy in image c, Kutosis in image d and Mean-Standard deviation in image e.

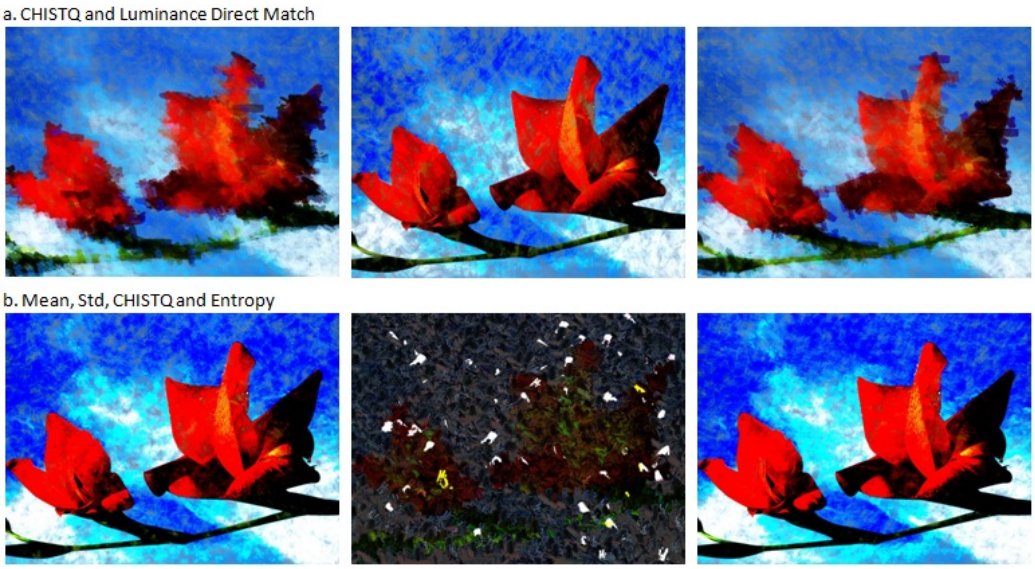


Figure 6.32: Results from different fitness functions. Source Image 4 in Figure 6.1 is used.

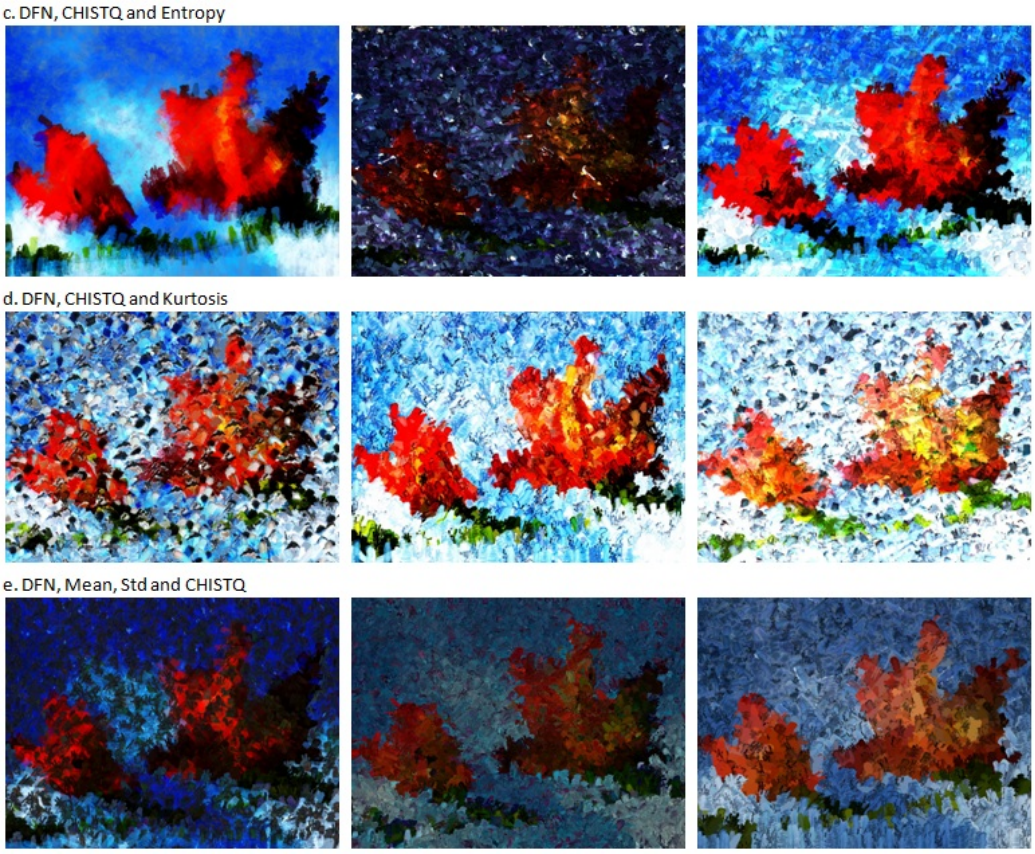


Figure 6.33: Results from different fitness functions. Source Image 4 in Figure 6.1 is used.

		DFN	Mean	Std	CHISTQ	Luminance Direct Match	Entropy	Kurtosis
Target Values		0.00	3.03	0.75	0.00	0.00	0.00	0.00
CHISTQ and Luminance Direct Match	1	75.09	2.22	0.82	0.38	1732871.00	5.44	2.03
	2	71.45	2.52	0.73	0.34	1627622.00	5.17	2.15
	3	79.66	2.20	0.83	0.37	1395196.00	5.43	2.07
Mean, Std, CHISTQ and Entropy	1	75.50	2.63	0.79	0.36	2426278.00	5.18	2.29
	2	54.52	2.94	0.78	0.45	5428876.00	5.35	1.73
	3	72.47	2.64	0.81	0.36	2631614.00	5.22	2.30
DFN, CHISTQ and Entropy	1	60.29	2.32	0.87	0.38	2274997.00	5.48	1.93
	2	31.72	2.95	0.91	0.40	5365914.00	5.56	2.09
	3	17.82	3.03	0.82	0.36	2925622.00	5.63	1.71
DFN, CHISTQ and Kurtosis	1	21.92	3.44	0.85	0.24	3322646.00	5.86	1.18
	2	26.29	3.21	0.91	0.28	4393638.00	5.84	1.20
	3	31.45	3.40	0.87	0.28	5633809.00	5.89	0.84
DFN, Mean, Std and CHISTQ	1	39.76	2.63	0.75	0.40	4864646.00	5.58	1.79
	2	17.20	2.88	0.73	0.39	4310322.00	5.75	1.33
	3	19.83	2.86	0.74	0.18	2627777.00	5.72	1.34

Table 6.12: Fitness scores of using different fitness scores for five samples and the target values. The yellow cells show the objects which are not included in the experiment.

The fitness values for each objective of illustrated images are shown in Table 6.12. The first two groups, which do not use DFN as a fitness objective, look like photographs have the poorest DFN values, except for the second image of the second group which has paint effects. The first group has the best score for luminance direct match values. The Entropy values are almost similar in all cases.

6.9 GP Language

Different GP languages can make different effects on the result images. There can be many way of combining defined functions and terminals in this system. Here four sample function sets are used (see Table 6.13). The main difference between these languages is in using different combinations of painting and NPR functions. Three runs are executed using each of these languages and the other parameters are fixed for all (See Table 7.2).

Table 6.13: Function Set 2

Function1	Paint4
Function2	Paint_RGB, Median
Function3	Paint1, Median, Brighten, Darken, Normal Blend, Overlay Blend, Difference Blend
Function4	Mean Shift Segmentation, Kuwahara, Medina, Brighten, Darken, Burn, Dodge, Normal Blend, Difference Blend, Overlay Blend
Common Functions	Add, Sub, Mul, Div, Sin, Cos
	mixAdd, mixSub, mixMul, mixDiv, mixAbs, mixAvg, mixNBld, mixDBld, mixOBld, mixBrt, mixSin, mixCos
	FtoV, VtoF_R, VtoF_G, VtoF_B, IfLT_F, IfLT_V, IfGT_R, IfGT_G, IfGT_B
Common Terminals	X, Y, W, L, THETA, CANVAS_C, SOURCE_C, ERC
	Min5, Max5, Avg5, Kurtosis5, Mean5, Std5, Median5, Skew5

The best of the last generation images of three runs are illustrated in Figure 6.34 and Figure 6.35. Based on the results, Paint_RGB makes more colourful images compared to Paint4 function. In Paint_RGB separate values are generated for each red, green and blue colour however in Paint4 one value is generated for all three and this is the main reason of having more gray images in rows a in Figures 6.34 and 6.35. Function set 3 makes weak paint. Function Set 4 makes the results look like filtered images rather than a painting.

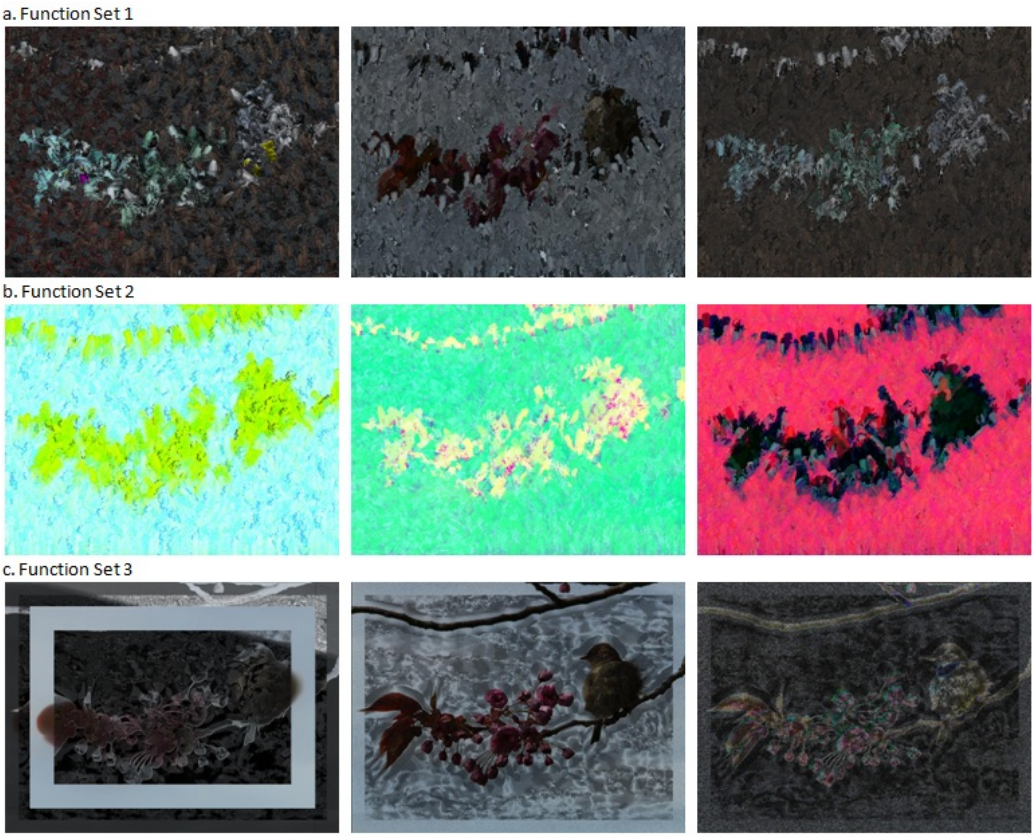


Figure 6.34: First sample of using different function sets. Source Image 2 in Figure 6.1 is used. The target colour image is Target 4 in Figure 6.2.

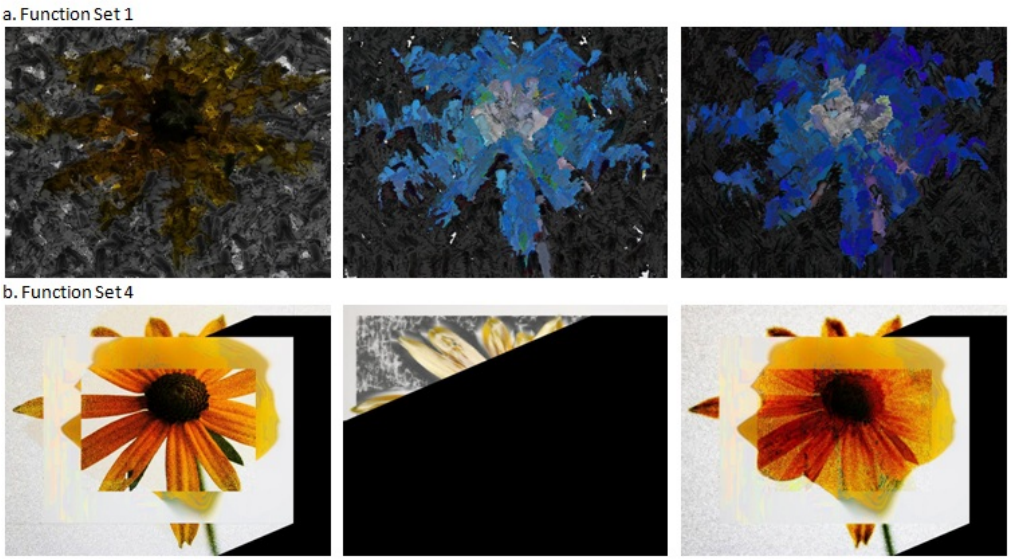


Figure 6.35: Second sample of using different function sets. Source Image 3 in Figure 6.1 is used. The target colour image is Target 4 in Figure 6.2.

Table 6.14 shows the fitness score for each illustrated results and the target values. The value of Mean, standard deviation and CHISTQ are almost similar for all illustrated results however, the DFN values are different. Function set 4 makes poorer DFN compared to other three function sets.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Function Set 1	1	13.46	3.06	0.78	0.37
	2	30.35	3.03	0.82	0.39
	3	12.45	2.98	0.78	0.38
Function Set 2	1	20.54	3.05	0.75	0.31
	2	24.71	3.10	0.75	0.32
	3	12.67	3.03	0.71	0.37
Function Set 3	1	48.71	3.03	0.78	0.37
	2	14.99	2.99	0.76	0.41
	3	16.92	3.03	0.74	0.37
Function Set 1	1	20.87	3.06	0.79	0.36
	2	17.44	3.16	0.78	0.34
	3	27.31	3.04	0.75	0.34
Function Set 4	1	43.68	3.03	0.75	0.34
	2	44.17	3.07	0.77	0.35
	3	37.26	2.99	0.74	0.34

Table 6.14: Fitness scores of using different functions sets and the target values

6.10 Using Luminance Direct Match

With the hope of having more details of the source image appear in resulting images, the luminance direct match is used as a fitness score. For this measurement the luminosity of the source image is compared with the luminosity of the generated images. The distance between these two is used as a fitness objective, and a zero score means the result image luminosity matches that of the source image. Four experiments are executed within three runs.

One result image from the best of the last generation within three runs is hand picked for each experiment (see Figure 6.36). These results show the clear effect of luminance direct match in having more details of the source images in the resulting images.

Table 6.15 shows the fitness values for each illustrated image. The luminance direct match value is always poorer when it is not used as an fitness object during evolution. CHISTQ has better score when luminance direct match is used and the reason is obvious by looking the images again. The colour ranges of all the results with this fitness objective are closer to the colour ranges of the source images. The only exception is in Figure 6.36d. which has different colours from its source image. DFN values also tend to be lower when it is used.

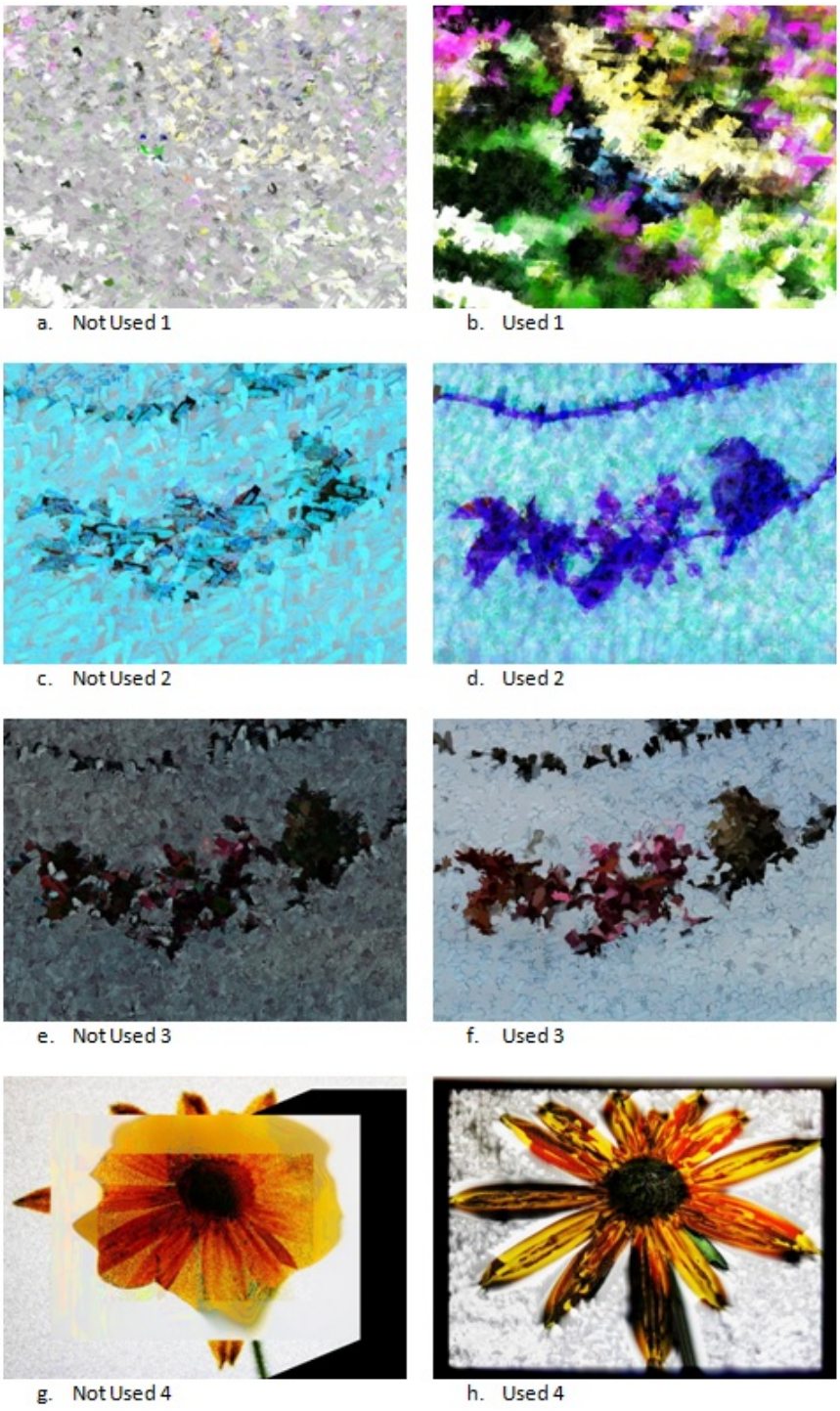


Figure 6.36: Result of using and not using luminance direct match.

		DFN	Mean	Std	CHISTQ	Luminance Direct Match
Target Values		0.00	3.03	0.75	0.00	0.00
Sample 1	Not Used	17.87	3.03	0.74	0.33	6388665.00
	Used	18.27	2.95	0.84	0.11	3629332.00
Sample 2	Not Used	35.32	2.80	0.77	0.56	2634940.00
	Used	22.34	2.69	0.65	0.57	1893570.00
Sample 3	Not Used	41.55	3.08	0.75	0.43	5842332.00
	Used	30.77	2.89	0.85	0.12	2074175.00
Sample 4	Not Used	164.23	2.35	0.94	0.07	3013884.00
	Used	47.88	3.01	0.77	0.11	2976867.00

Table 6.15: Fitness scores of using and not using luminance direct match and the target values

6.11 HSV Mode

Changing the colour mode is another way to have different resulting images. This system has two colour modes namely, RGB and HSV. All previous sections studied the effects of different parameters in RGB mode. Here the colour mode is set to HSV to show the effect of this colour mode. The GP language and parameters of Tables 7.1 and 7.2 are used for all the tests.

For each experiment 3 runs are executed and one image from the best of the last generation within the three runs are hand picked (see Figure 6.37). HSV mode makes more green and blue colours in the resulting images.

The fitness scores of illustrated images is shown in Table 6.16. Using HSV mode causes poorer DFN values. The CHISTQ values are almost the same for both colour modes.

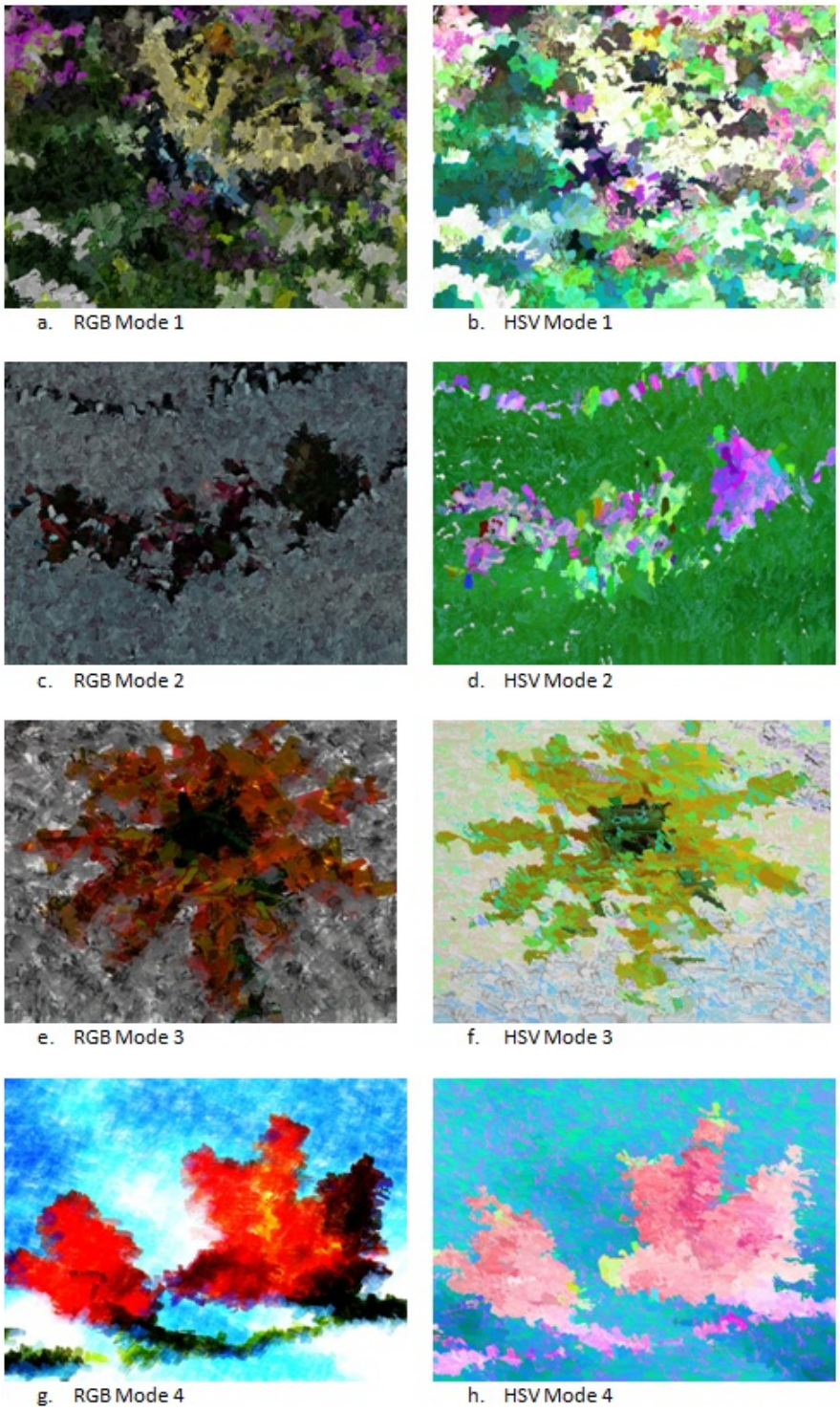


Figure 6.37: Comparing results of using RGB and HSV modes.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Sample 1	RGB Mode	10.50	3.04	0.84	0.36
	HSV Mode	39.64	3.16	0.92	0.36
Sample 2	RGB Mode	18.77	3.03	0.79	0.44
	HSV Mode	20.07	3.02	0.81	0.49
Sample 3	RGB Mode	17.44	3.04	0.76	0.41
	HSV Mode	34.21	3.16	0.85	0.41
Sample 4	RGB Mode	20.20	3.02	0.81	0.23
	HSV Mode	20.82	2.94	0.79	0.38

Table 6.16: Fitness scores for RGB and HSV colour modes.

6.12 Number of Generations

On this chapter up to now, the number of generations for all the executed experiments is 15. The resulting images can be different earlier in evolution, because they are less developed and unrefined. To see the effect of this GP parameter, the high rank images of generation 5 are compared with high ranks of generation 15.

Three runs are executed for each experiment and the result of each run for these three experiments are shown in Figures 6.38, 6.39 and 6.40. In most of the illustrated results more details of the source images can be seen in generation 15 like in third image of Figure 6.38, first and third images of Figure 6.39 and all images of Figure 6.40.

Table 6.17 shows the fitness scores of illustrated images. The DFN values of generation 15 are always better than generation 5 for all the samples. This shows the improvement of DFN during evolution. In sample 2 the CHISTQ also improves in generation 15 however, this score has similar values in generation 5 and 15 for the other two samples. In the first sample CHISTQ has poorer values in generation 15. This can be happen in a multi-objective evaluation as not all objectives improve during a run.

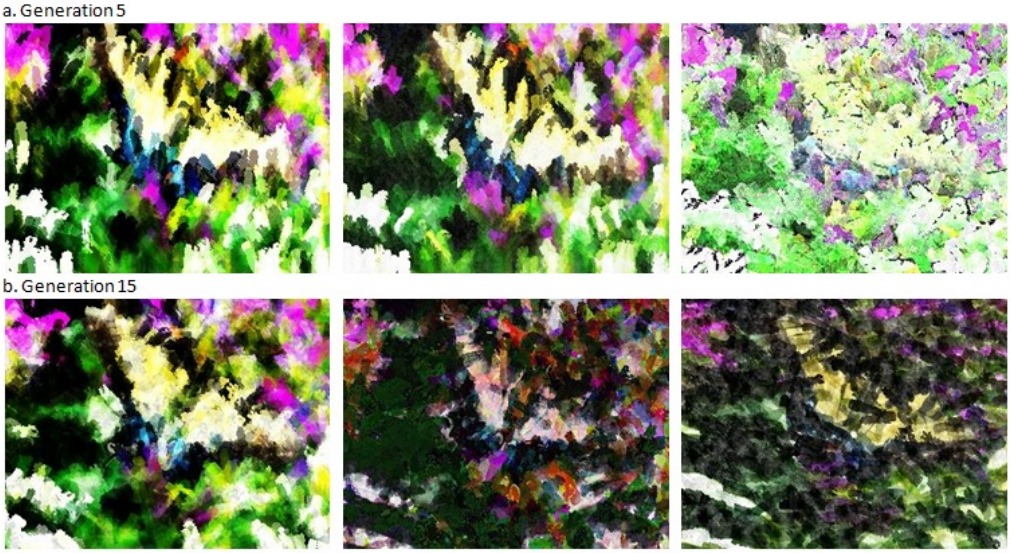


Figure 6.38: First comparison for number of generations . Source Image 1 in Figure 6.1 is used. The target colour image is Target 4 in Figure 6.2.

Figure 6.39: Second comparison for number of generations . Source Image 2 in Figure 6.1 is used. The target colour image is the source image

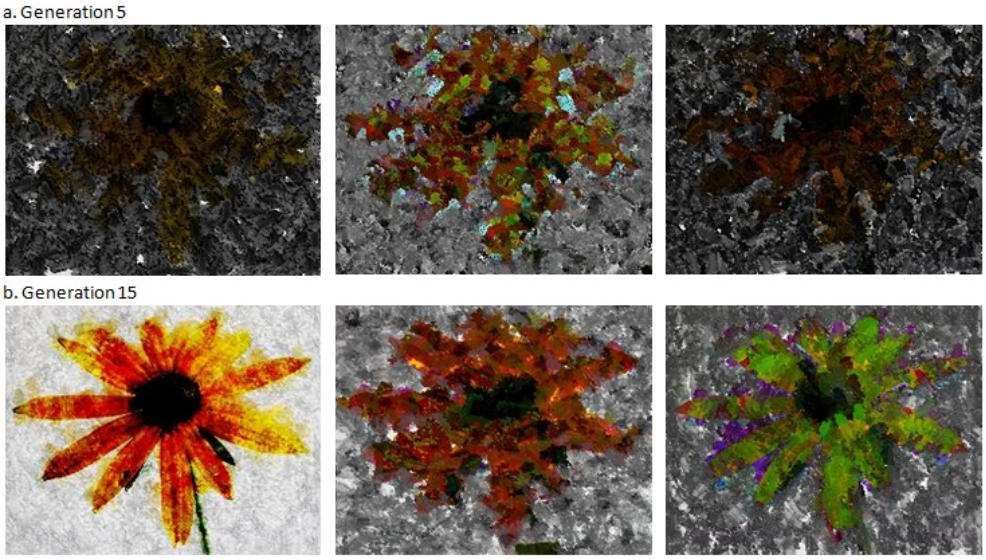


Figure 6.40: Third comparison for number of generations . Source Image 3 in Figure 6.1 is used. The target colour image is Target 2 in Figure 6.2.

		DFN	Mean	Std	CHISTQ
Target Values		0.00	3.03	0.75	0.00
Sample 1 Generation 5	1	13.94	3.15	0.99	0.26
	2	11.45	3.00	0.94	0.28
	3	17.45	3.57	0.78	0.30
Sample 1 Generation 15	1	10.48	3.04	0.92	0.28
	2	10.09	3.04	0.79	0.32
	3	8.92	3.02	0.79	0.34
Sample 2 Generation 5	1	23.65	2.96	0.83	0.25
	2	29.32	2.99	0.74	0.30
	3	29.36	3.15	0.81	0.12
Sample 2 Generation 15	1	22.27	3.04	0.79	0.16
	2	22.16	3.02	0.75	0.25
	3	21.65	3.03	0.88	0.07
Sample 3 Generation 5	1	30.23	3.04	0.79	0.41
	2	18.13	3.24	0.77	0.40
	3	25.17	3.06	0.86	0.40
Sample 3 Generation 15	1	11.02	2.97	0.74	0.36
	2	17.44	3.04	0.76	0.41
	3	19.12	3.06	0.78	0.42

Table 6.17: Fitness scores of generation 5 and generation 15.

6.13 Conclusion

This chapter studied the effect of using different parameters and how they change the nature of the results. Some parameters make more changes than the others. For example different brush sizes (Figure 6.4), canvas images (Figure 6.9), brush bitmap styles (Figure 6.23), GP language (Figure 6.35) or using luminance direct match (Figure 6.36) make very big changes in the resulting images. Some parameters make less changes such as using white canvas (Figuer 6.29) and different fitness tests (Figure 6.33). However, the effect of randomness in the evolutionary system should be considered as well.

Chapter 7

Miscellaneous Examples

The goal of this chapter is show a selection of results that highlight the variety of effects possible with the system. All the illustrated experiments are hand picked within the five best of the last generation of three runs.

Table 7.1: The GP languages

Common Function	FtoV, VtoF_R, VtoF_G, VtoF_B, IfLT_F, IfLT_V, IfGT_R, IfGT_G, IfGT_B
Common Terminals	X, Y, W, L, THETA, CANVAS_C, SOURCE_C, ERC Min5, Max5, Avg5, Kurtosis5, Mean5, Std5, Median5, Skew5
Function1	Add, Sub, Mul, Div, Sin, Cos
	Paint4, Paint_RGB, Mean Shift Segmentation, Median, mixAdd, mixSub, mixMul, mixDiv, mixAbs, mixAvg,
Terminal1	Min9, Max9, Avg9, Kurtosis9, Mean9, Std9, Median9, Skew9
Function2	Sub, Sin, Cos
	Paint_RGB, mixAdd, mixSub, mixMul, mixDiv, mixAbs, mixAvg, mixNBld, mixDBld, mixOBld, mixBrt, mixSin, mix-Cos
Function3	Sub, Sin, Cos
	Paint4, mixAdd, mixSub, mixMul, mixDiv, mixAbs, mixAvg, mixNBld, mixDBld, mixOBld, mixBrt, mixSin, mixCos
Terminal3	Min13, Max13, Avg13, Kurtosis13, Mean13, Std13,Skew13, Median13

Table 7.2: This is fixed parameters list

Parameter	Value
# brush bitmaps	5
# brush sizes	4
Processing Model	Process the whole image to find the next unpainted pixel
Fitness Function	DFN, Mean, Standard Deviation, CHISTQ
Generations	15
Population Size	500
Tournament Size	3
Crossover Rate	90%
Maximum Crossover Depth	17
Mutation Rage	10%
Maximum Mutation Depth	17
Prob. of Terminals in Cross/Mut	10%
Initialization Method	Half-and-Half
Tree Grow Max/Min	5 / 5
Tree Full Max/Min	5 / 7

Figure 7.1: target Colour Images

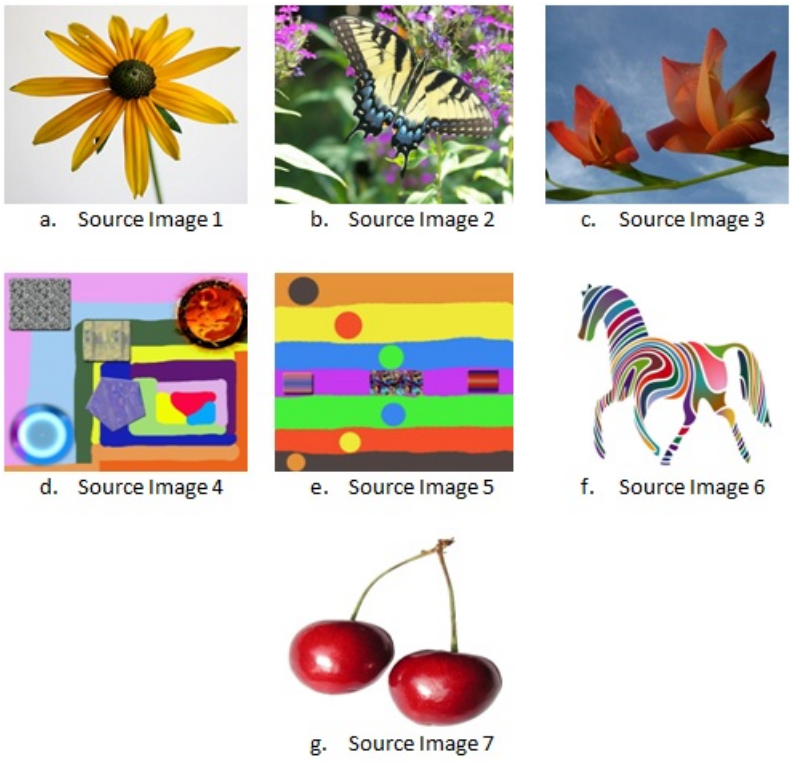


Figure 7.2: Source Images

Experiment 1: Here Source Image 1 in Figure 7.2 and Target Colour Image 1 in Figure 7.1 are used as source image and target colour image. Two samples are shown here. Figure 7.3 is hand picked from the five best of the last generation and Figure 7.4 is the best of the whole run. These images are selected within three runs. The GP language contain common functions, common terminals, function 1 and terminal 1 in Table 7.1. The other parameters are shown in Table 7.2.

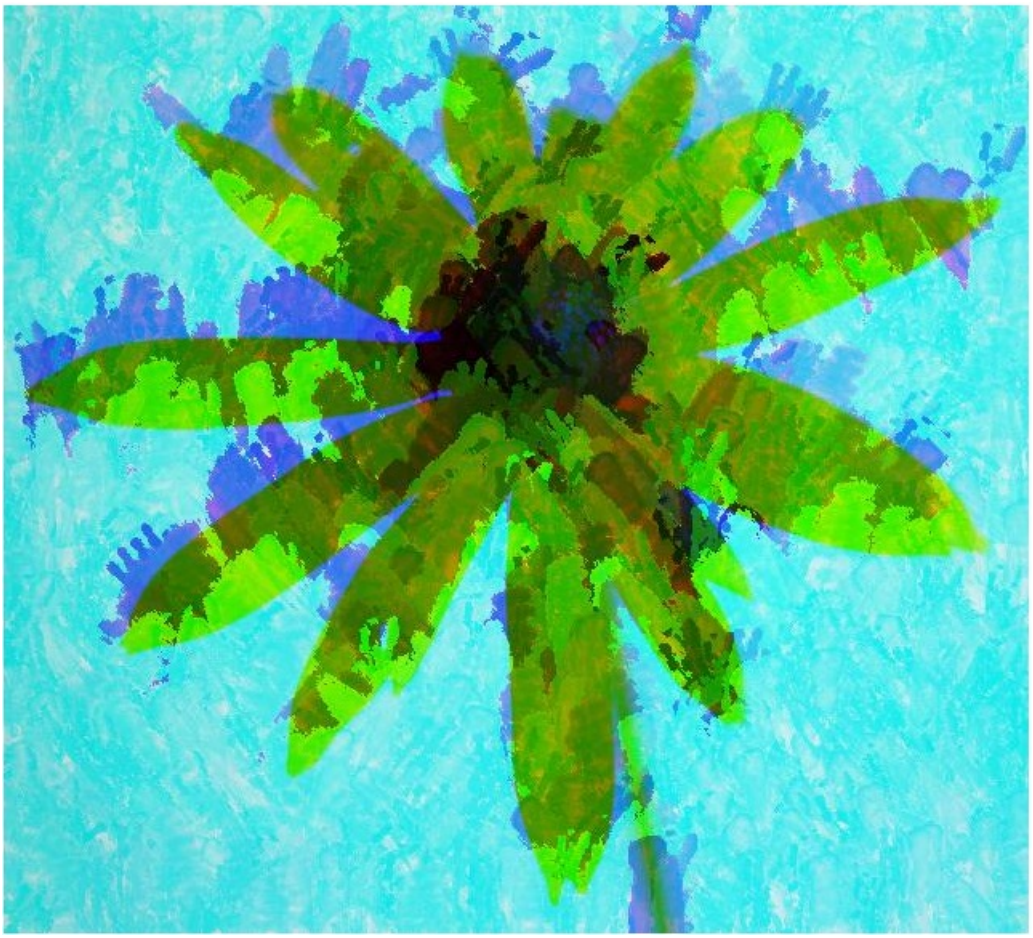


Figure 7.3: Experiment 1: one of the best images of last generation. This image has the watercolor effect.

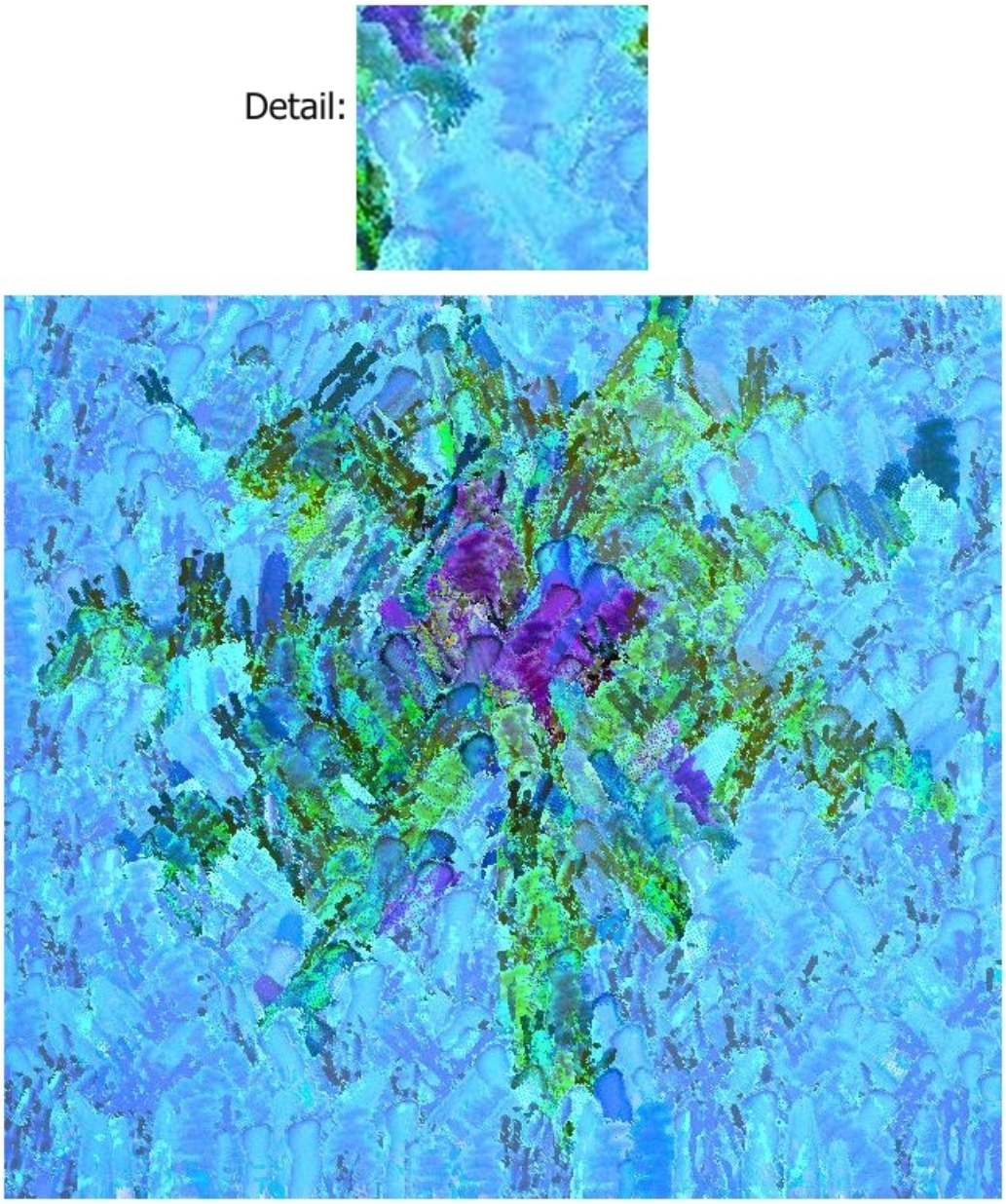


Figure 7.4: Experiment 1: the best scoring image of the whole run. This image is near to acrylic painting.

Experiment 2: The source image is the same as experiment 1 here but different target colour image (Target Colour Image 2 in Figure 7.1) and GP language. Figure 7.5 shows a sample result. The GP language is common parts, function 2 and terminal 2 in Table 7.1.

Figure 7.5: Experiment 2: one of the best images of last generation.

Experiment 3: One of my digital images (Source Image 4 in Figure 7.2) is used as the source hand-drawn image for this experiment. Target Colour Image 1 (Figure 7.1) is used for this experiment. The GP language is the same as experiment 2 in all the functions and terminals except here Paint4 is used instead of Paint_RGB. Two sample results are illustrated here. The first image (Figure 7.6) is selected from the best of the whole run and Figure 7.7 is from the best of the last generation.

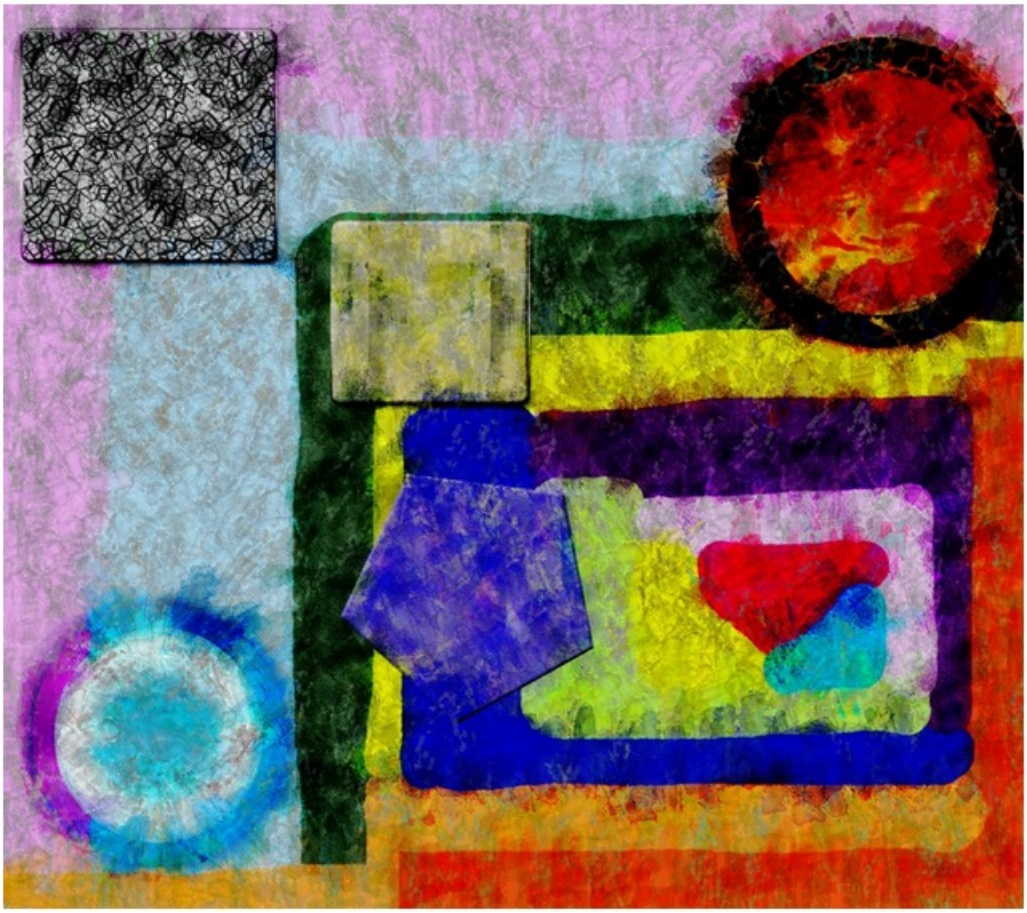


Figure 7.6: Experiment 3: the best scoring image of the whole run. This image has mixd media effect.

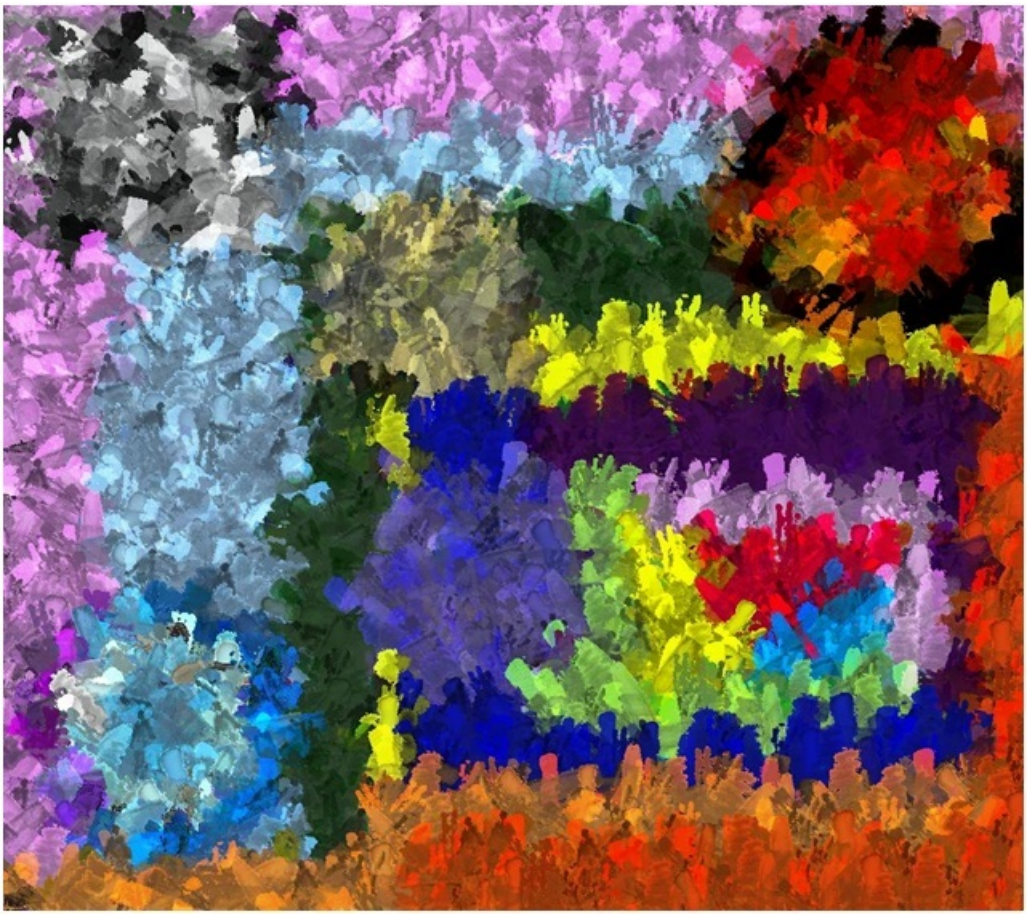


Figure 7.7: Experiment 3: one of the best images of last generation. This image looks like oil painting.

Experiment 4: Here the source image is from my designed painting (Source Image 5 in Figure 7.2). The GP language and the target colour images is similar with experiment 3. Figure 7.8 is hand picked from five best images of the last generation.

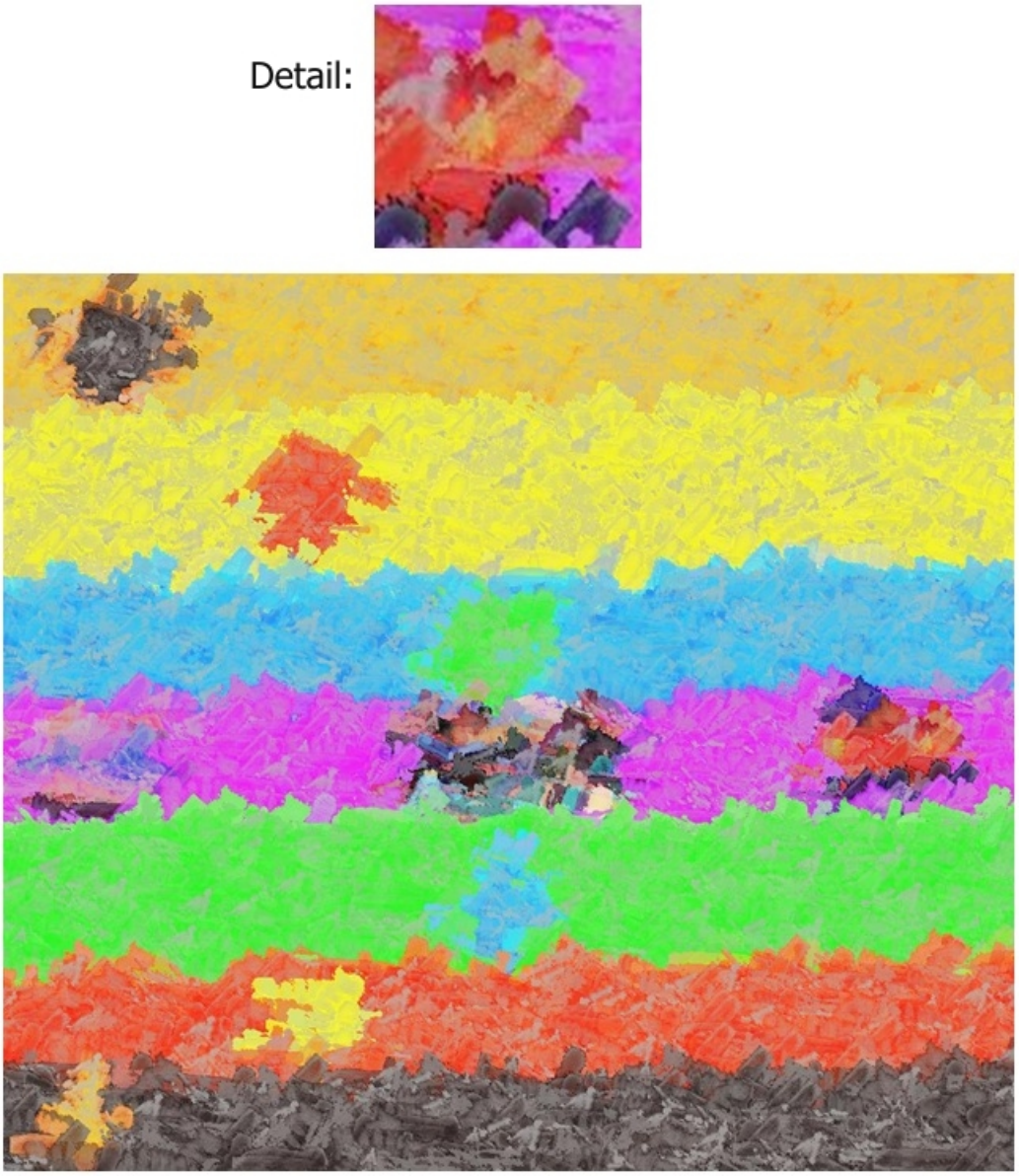


Figure 7.8: Experiment 4: one of the best images of last generation.

Experiment 5: A photograph is used as the source image for this experiment (Source Image 2 in Figure 7.2). White canvas is used here and one of the fitness objectives is luminance direct match. The GP language contains the function 3, terminal 3 and common parts except the 5*5 image processing terminals in Table 7.1. Also all the parameters are the same as Table 7.2 but the population size is 300. Figure 7.9 is one resulting image.

Figure 7.9: Experiment 5: one of the best images of last generation.

Experiment 6: Here another photograph is used as the source image (Source Image 3 in Figure 7.2). The GP language is similar to experiment 3. Three runs are executed for this experiment and two image from the five best of the last generation are selected and shown in Figures 7.10 and 7.11.

Figure 7.10: Experiment 6: one of the best images of last generation. This image has watercolor effect.

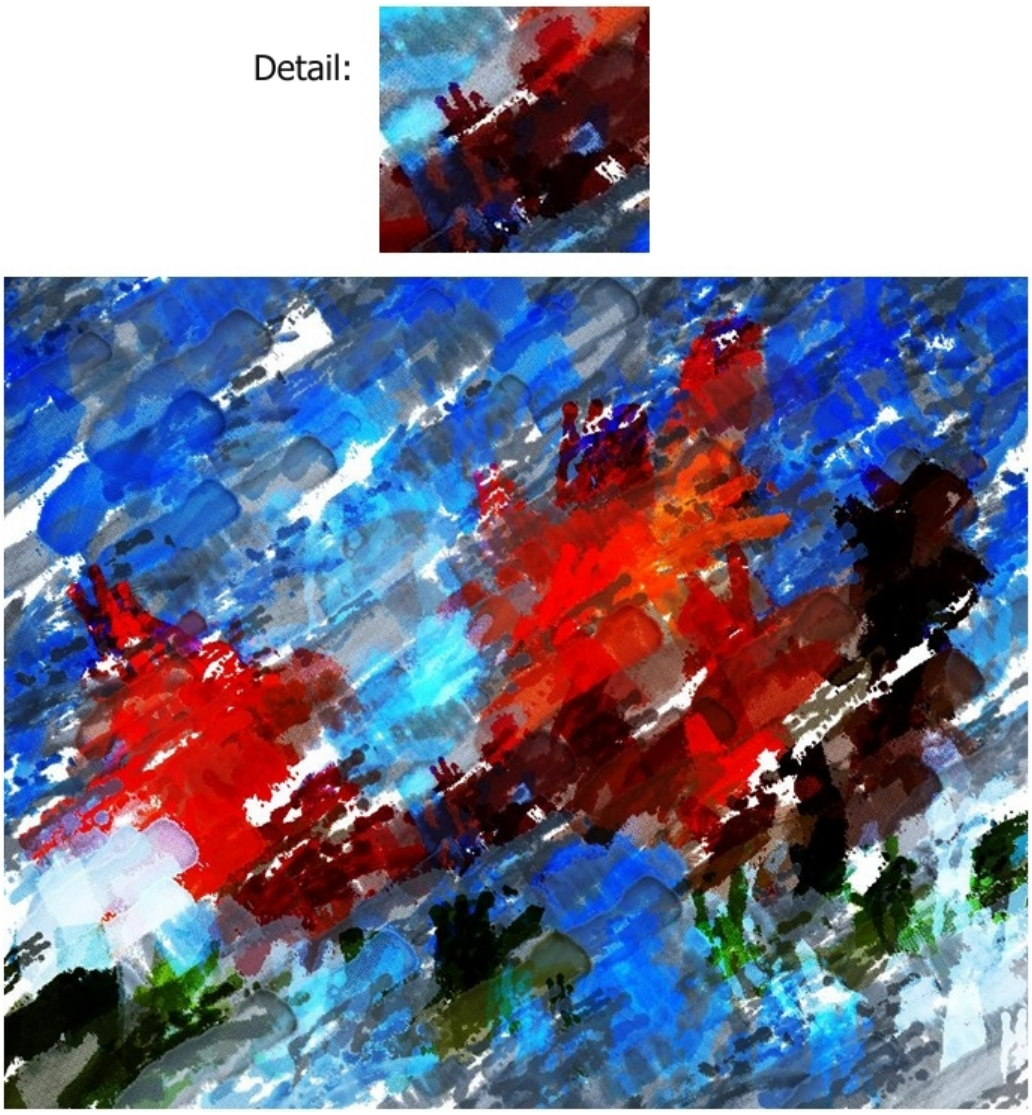


Figure 7.11: Experiment 5: one of the best images of last generation.

Experiment 7: A painting source image is used for this experiment (Source Image 7 in Figure 7.2). White canvas is used and luminance direct match is one of the fitness function. The GP language is similar to experiment 5 except here Paint_RGB is used instead of Paint4. Three runs are executed and one of the resulting images from five best of the last generation is illustrated in Figure 7.12.

Figure 7.12: Experiment 7: one of the best images of last generation.

Experiment 8: Here Source Image 9 in Figure 7.2 is used which is its own target colour image. Luminance direct match is used as one of the fitness objective. Here 1000 pixels are randomly selected to be painted and also white canvas is used. The GP language is similar to experiment 7. One of the best images of the last generation is shown in Figures 7.13 and 7.14.

Figure 7.13: Experiment 8: one of the best images of last generation.

Figure 7.14: Experiment 5: one of the best images of last generation.

Chapter 8

Human Survey

Different evaluation functions are implemented for the automatic evolutionary system. These functions contains aesthetic and statistical measurements such as DFN, CHISTQ, mean and standard deviation. The goal of using aesthetic measures is to have resulting images which are beautiful and have aesthetic value to a human viewer. These theories are new and relatively untested, however, and they are not a guarantee of aesthetic quality. Therefore many evolutionary art systems use interactive evolution to have the human select the preferred images for next generation.

In this chapter the results of a human survey are presented. The goal is to show that the good scoring resulting images (the ones which are selected as the best ones with this GP system) have NPR-like characteristics according to human viewers. The survey is similar to those in [60, 63]. The survey contains twenty questions. In each question there are two images to be compared and selected. The question for each pair is

> The following images are the result of a computer-based image processing system. For each pair of images, please select the image that you feel looks most like a painting that a human artist might have created. Note that you should not base your decision on whether you prefer one image more than the other. Rather, choose the image that you feel looks more like an image that a person might have painted.

For each pair of images one is selected from the generation 0 and the other is the best of the last generation. To select an image from generation 0, the results are sorted base on their rank value and the middle score is selected.

This process is carried out on randomly selected experiments and a pool with many middle scoring images from generation 0 is made. Random selection of the best of the last generation within different experiments a pool with the best results is also done. Then one image from generation 0 and one from last generation are selected randomly to make a pair for each question.

An online survey was made using SurveyMonkey website [1]. There are some limitation for the basic account in this website which caused me to have 4 surveys with 5 questions in each, for the 20 questions in total. Participants might not complete all of questions in one survey or might not completed all 4 surveys. There is no login username and password to prevent participant from filling out the surveys more than once.

The generation 0 and last generation images are randomly placed on right or left of the page and labeled with a and b letters. The fitness scores were not shown in the test and participant are asked to select a or b for each question (see Figure 8.1). Students from Brock Computer Science department were invited to participate in this survey via email. The survey was appeared by the Brock University Research Ethics Board (REB).

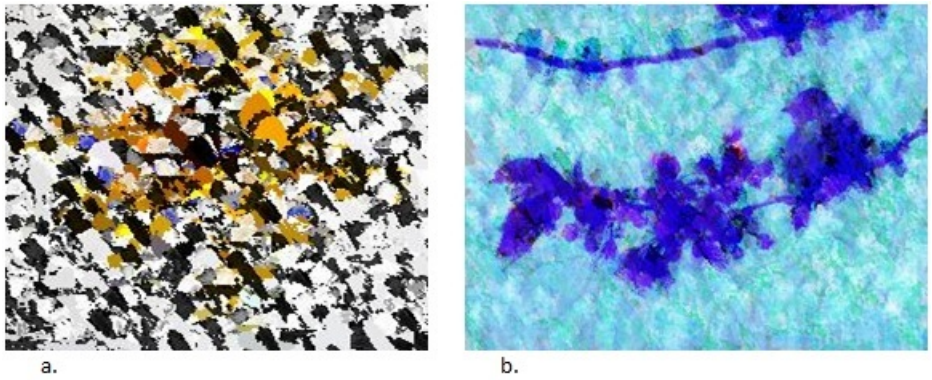


Figure 8.1: A question in the survey

A total of 37 completed surveys were collected. Uncompleted surveys are ignored in the analyses. The number of correct selection for all twenty questions are illustrated in Figure 8.2. A “correct” answer is when the user selects the best of the last generation image from the pair. The sign test(a non parametric test) is used to analyze the result of this survey. This test is used to determine if there is no difference in the medians of the continuous

distributions of two random variables X and Y [18]. According to the results, only one question has more wrong answers than correct ones (question number 7 with 11 correct answer within 37 answers). The number of correct and wrong selection for question 12 is so near with 19 correct answers and 18 wrong answers. Using the sign test shows that the participants selected the best of the last generation images with the better fitness score, using $\alpha = 0.05$.

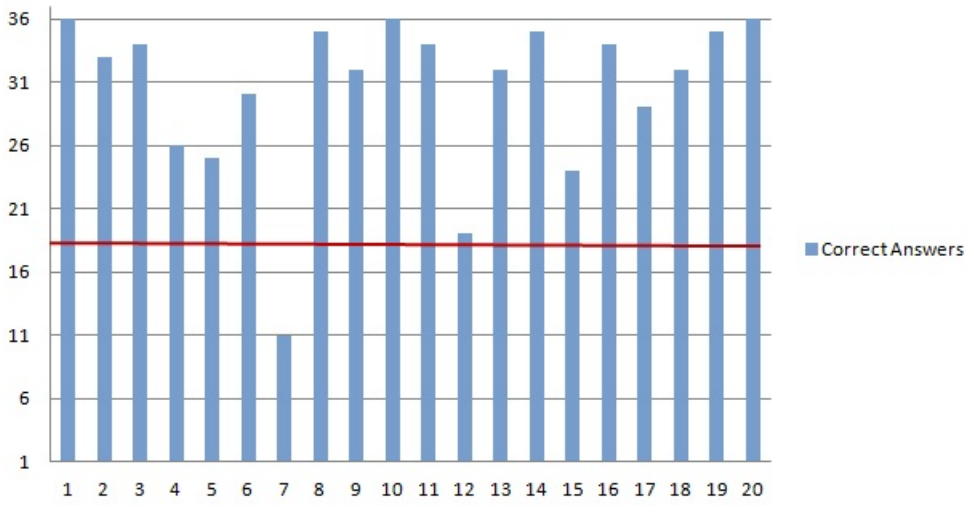


Figure 8.2: The diagram of correct answers for twenty questions of the human survey. Horizontal line shows the mid-point at 18.

The results give statistical significant evidence that the tests used in GP create images with NPR qualities to the human eyes. However, using a stronger survey which corrects some of the earlier mentioned weaknesses, would strengthen the results.

Chapter 9

Comparison to Related Work

The goal of this chapter is to compare this research with other related work. All of these papers were reviewed in the literature review. Here they are compared with the system in more detail.

9.1 Neufeld *et al.*

This thesis is very similar with Neufeld *et al.* [14] in many areas: (i) Using automatic fitness functions with aesthetic evaluation such as DFN and CHISTQ. (ii) Using multi-objective evaluation. (iii) Using a similar GP language such as using paint algorithm from [50]. (iv) Using pre-filter functions such as moment and sobel. (v) Strongly typed GP is applied using float and vector types.

However, there are some differences between these two systems:

- Neufeld *et al.* uses Pareto ranking as multi-objective evaluation method and here normalized rank sum is used. With Pareto, an outlier individual can be selected, having a good score in one objective and very bad scores in other objectives. However, this does not happen with normalized rank sum, because outliers are not common. To keep diversity, there is a penalty method for the similar individuals in the population.

- Here different versions of paint algorithm are implemented with the hope of having more creativity.

- Using the idea of mix tree as a child for the paint function is a major

new idea instead of using a fixed colour mixing algorithm which makes more variety of NPR effects.

- Having more NPR functions such as Kuwahara, mean shift segmentation, brighten and darken.
- Using more fitness evaluations such as entropy, kurtosis and luminance direct match. The user is capable of selecting any fitness function from the list of implemented functions and also can put any weight of each of the selected objectives.
- Neufeld *et al.* processes the whole image and applies the tree expression to all pixels. Here there are two methods to process the canvas. The first is to randomly select a number of pixels and the second one is processing the whole image to find the next unpainted pixel. Applying these two methods save the processing time and also get the chance of have more unexpected and interesting results.
- The users can use any number and styles of brush bitmaps, while in Neufeld *et al.* there are five fixed brush bitmaps.

9.2 Barile *et al.*

Barile *et al.* [55] has an automatic GP system using computer graphic techniques to have NPR results. They have image pre-processing phase in which the gradient and importance maps are calculated. There, the result image is compared with the target image pixel by pixel to evaluate each individual. My system is stronger in this point as it uses many different evaluations including some aesthetic evaluation such as Ralph's bell curve model. In their GP language two types are defined namely, prog and draw. The prog node just accepts draw nodes and each draw node is a brush stroke on the canvas. They have a very small language compare to my system which has many different paint and NPR functions.

Here the users can select any of the functions and terminals to make different types of effects however in Barile *et al.* the goal is to generate three types of effects, namely Shroud of Turin, Decal and Starburst. They used three canvases and for two of them the user can defined the brushes, which is similar to my system as any number of brush bitmaps can be set for a

run. There, HSV mode is used but this system works in both HSV and RGB modes based on user preference.

9.3 Izadi and Ciesielski

Izadi and Ciesielski [5]'s GP system uses triangular strokes. They used and compared two types of strokes namely empty triangle and filled triangle. This is one of the main difference with my system since any styles of brush strokes can be used by importing the brush bitmaps as parameters of the system. Their system's fitness calculation is also different. For each RGB channel of each pixel the difference between the target and the generated image is added up and normalized and then the average of normalized value of three channels is used as a fitness score. In my system aesthetic and statistical evaluation are used as different objects in an multi-objective evaluation. They have a much smaller rendering language compare to this system as three functions are defined in their system. Two of these functions just accept a draw function as their children and the third function is for drawing a triangle on the canvas. Here different types of languages are implemented to generate more creative and unexpected styles of images. Using different types of target colour images is another creative advantage of my system.

9.4 Kang *et al.*

Using a genetic algorithm, Kang *et al.* [25] define a problem to have multi-level brush strokes on the white canvas in a way to have rendered images look as close as possible to the input image with minimum number of brush strokes. The length and orientation of the brush strokes are calculated from the input image. The width of the brush is depend of which level it is painted. The level 0 has the smallest scale brushes and the size is increased by going to the next levels. This system works in RGB colour mode. It is obvious that this system is very different from my system. Although they have impressive results, the style of the resulting images does not vary.

9.5 Yip

In Yip [15], genetic algorithm is used to automatically evolve image filters which use a combination of some simple filters, such as threshold, convolution and median. It uses different measurements to evaluate the generated image filters such as texture measurement, standard deviation, mean of the grey level histogram and entropy. He applies his measurement on a target filtered image, and the scores are used to find the similarity of generated images with it. In my system DFN, mean, standard deviation, CHISTQ, entropy and other measurements are available for use in evolution. Giving this option to the user to select different objectives as fitness functions causes to have different effects in the resulting images. There, the resulting images are more look like filtered images than NPR effects.

9.6 Collomosse

Collomosse [56]'s genetic algorithm uses painterly rendering algorithms to create a oil paint style image from an input photograph. The fitness function is base on salience measure from [31] which calculates the rarity, visibility of image and classifies the image. There is a pre-processing phase to find salience map, intensity gradient image and classification probability of the pixels. Interactive evaluation is used. The resulting images are impressive. However, there is no creativity in generating new styles and effects which is seen in this thesis.

9.7 Hertzmann *et al.*

There are lots of NPR research from computer graphics [11,30,77]. Generally, each algorithm normally results in one effect, with some user parameters. The advantage of using these algorithms is to have really aesthetic effects in the resulting images however, there are some disadvantages in using them like there is no diversity and variability in the results. One example of these is Hertzmann's system.

Hertzmann *et al.* [4] image analogies is a matching learning system which uses an image and the filtered version of that image as a training data to find a filter solution and apply that to a new photograph. The results are significant as the filtered version of the second image looks remarkably like

the first filtered image. The goal of such system is to simulate an existing effect on other images and therefore there is no creativity and exploration in this system. Although not all the resulting images of my system is similar to real hand painted art work, my goal is to have unexpected, new styles and effects.

9.8 Summary

There are few things which make my system very innovative, for example, a large set of fitness functions and powerful GP language,and especially colour mixing and canvas processing. My system creates a wide variety of NPR effects, generally not seen in the other evolutionary art systems above.

Chapter 10

Conclusion and Future Work

10.1 Conclusion

This thesis showed a system that generates non-photorealistic rendering effects using an automatic genetic programming system. The goal of this research was to have a creative and flexible system for evolutionary art. Therefore this system has many parameters and options which can be set according to the users preference. For example the number of brushes, the brush bitmap style and intensity, different aesthetic and statistical fitness functions, and different NPR and painting functions.

Performance analyses were done to show the effect of parameters defined. The effect of using different target colour image, setting weight for CHISTQ or DFN score, changing brush bitmaps intensity and other parameters were studied. Moreover, the potential of this system to generate a variety of creative results was shown using different parameters . This study showed that changing some parameters has effect on different styles of resulting images such as brush sizes, brush bitmap styles, using luminance direct match, using different target colour image and changing GP language.

A human survey was hosted online to see if human viewers also select the resulting images which are the best ones in this GP system according to their scores. Using the sign test showed the participants chose the best scoring images with a 95% certainty.

In conclusion, this research had many exciting and interesting results showing mixed media, watercolor and other painting effects. There were also many aesthetic results with very unique colourful effects. Applying random

selection method on a white canvas made many beautiful abstract images.

There were also some limitations with this system. It can be difficult to get preconceived effects like Hertzmann *et al.* [4]. It is not easy to get special effects when you want it because of the random and probabilistic manner of evolutionary system and complexity of GP tree. More detailed image analyzer and filters might create more results similar to human paintings. For example, having a clear subject in an image matters in this system. This system was very slow when using big images big GP trees, and large filter areas (Some runs took 40 hours to finish).

10.2 Future Work

There are many other ideas which can be added to this research as the future work. These are:

- Extend GP language:

  Add new filters, NPR algorithms and painting methods. Use new colour mixing algorithms. Add more high lever computer vision features. Moreover, texture generating functions such as Perlin noise or fractals can be added to create some new effects in resulting images.

- GP structure:

  Apply multi-pass processing in GP. For instance, there can be two trees generated with separate languages so that the result of processing the first tree will be the input data for second tree and the result of second tree is the final image to be evaluated

- New aesthetic fitness models:

  Using other aesthetic measures can make different effects and help to have more natural results. For example models mention in Graham *et al.* [35] should be considered.

- Implement on GPU:

  This may solve the execution speed problem when using big images or large filter areas.

- Interactive evaluation:

This would make the system a useful tool for an artist. Both automatic and interactive evaluation could be used.

Bibliography

[1] Surveymonkey. http://www.surveymonkey.com/.

[2] Bousseau. A, Kaplan. M, Thollot. J, and Sillion. X. F. Interactive watercolor rendering with temporal coherence and abstraction. In *NPAR '06 Proceedings 4th International Symposium on Non-Photorealistic Animation and Rendering*, pages 141–149, 2006.

[3] Hertzmann. A. Painterly rendering with curved brush strokes of multiple sizes. In *SIGGRAPH '98 Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, pages 453–460, New York, 1998. ACM.

[4] Hertzmann. A, Jacobs. C. E, Oliver. N, Curless. B, and Salesin. D. H. Image analogies. In *SIGGRAPH '01 Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pages 327–340, New York, 2001. ACM.

[5] Izadi. A and Ciesielski. V. Evolved strokes in non photorealistic rendering. In Ardil. C, editor, *Proceedings of the World Academy of Science Engineering and Technology*, pages 1–4. Springer, 2010.

[6] Rowbottom. A. Evolutionary art and form. In Bentley. P. J, editor, *Evolutionary Design by Computers*, San Francisco, CA, 1999. Morgan Kaufman Publishers Inc.

[7] Thompson. A. Evolving fault tolerant systems. *Genetic Algorithms in Engineering Systems: Innovations and Applications, IEEE Conference*, (414):524–529, 1995.

[8] Thompson. A and Layzell. P. Analysis of unconventional evolved electronics. *Communications of the ACM*, 42(4):71–79, 1999.

[9] Baxter. B, Scheib. V, Lin. M. C, and Manocha. D. Dab: Interactive haptic painting with 3d virtual brushes. In *SIGGRAPH '01 Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pages 461–468. ACM Press, 2001.

[10] Cho. S. B. Towards creative evolutionary systems with interactive genetic algorithm. *Applied Intelligence*, 16(2):129–138, 2002.

[11] Gooch. B and Gooch. A. *Non-Photorealistic Rendering*. A K Peters, 2001.

[12] Spehar. B, Clifford. C. W. G, Newell. B. R, and Taylor. R. P. Universal aesthetic of fractals. *Computers and Graphics*, 27(5):813820, 2003.

[13] Li. C. Aesthetic visual quality assessment of paintings. *IJSTSP: IEEE Journal of Selected Topics in Signal Processing*, 3(2):236–253, 2009.

[14] Neufeld. C, Ross. B. J, and Ralph. W, editors. *The Evolution of Artistic Filters*, Vancouver, 2006. Evolutionary Art Competition.

[15] Yip. C. Evolving image filters. Master's thesis, Imperial College of Science, Technology and Medicine, 2004. Imperial College of Science, Technology, and Medicine.

[16] Birkhoff. G. D. *Aesthetic Measure*. Harvard University Press, Cambridge, MA, USA, 1933.

[17] Hart. A. D. Toward greater artistic control for interactive evolution of images and animation. *Applications of Evolutionary Computing*, 4448:527–536, 2007.

[18] Wackerly. D, Mendenhall. W, and Scheaffer. R. L. *Mathematical Statistics with Applications*. Duxbury Advanced Series, sixth edition, 2002.

[19] Heijer. E den and Eiben. A. E. Comparing aesthetic measures for evolutionary art. In *EvoApplications (2)*, pages 311–320, 2010.

[20] Goldberg. D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1989.

[21] Jansson. E. Brush painting algorithms. Technical report, datorteknik, 2004.

[22] Winkenbach. G and Salesin. D.H. Computer-generated pen-and-ink illustration. In *SIGGRAPH '94 Proceedings of Annual Conference Series on Computer Graphics Proceedings*, New Yourk, ACM Press, 1994.

[23] Holland. J. H. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.

[24] Huang. H, Fu. T, and Li. C. Anisotropic brush for painterly rendering. In *Proceedings of Computer Graphics International. Computer Graphics Society*, 2010.

[25] Kang. H, Chakraborty. U, Chui. C, and He. W. Multi-scale stroke-based rendering by evolutionary algorithm. In *Proceedings International Workshop on Frontiers of Evolutionary Algorithms*, pages 546–549. JCIS, 2005.

[26] Bentley. P. J. *Generic Evolutionary Design of Solid Objects Using a Genetic Algorithm.* PhD thesis, Division of Computing and Control Systems, Department of Engineering, University of Huddersfield, 1996.

[27] Bentley. P. J. Aspects of evolutionary design by computers. In *Proceedings of the 3rd On-line World Conference on Soft Computing in Engineering Design and Manufacturing*, 1998.

[28] Bentley. Peter J. and Wakefield. Jonathan P. Finding acceptable solutions in the pareto-optimal range using multiobjective genetic algorithms. In *Soft Computing in Engineering Design and Manufacturing*, pages 231–240. Springer-Verlag, 1998.

[29] Branke. J, Deb. K, Miettinen. K, and Slowiski. R. (Eds.). *Multiobjective Optimization: Interactive and Evolutionary Approaches.* Spriger, 2008.

[30] Collomosse. J and Kyprianidi. J. E. *Artistic Stylization of Images and Video. Tutorial at Eurographics.* Retrieved from http://kahlan.eps.surrey.ac.uk/EG2011/, 2011.

[31] Collomosse. J and Hall. P. Genetic paint: A search for salient paintings. In *Applications on Evolutionary Computing*, pages 437–447, 2005.

[32] Curtis. C. J, Anderson. S. E, Seims. J. E, Fleischer. K. W, and Salesin. D. H. Computer-generated watercolor. In *SIGGRAPH '97 Proceedings of Annual Cofnerence Series on Computer Graphics Proceedings*, Addision-Wesley, 1997.

[33] Fogel. L. J, Owens. A. J, and Walsh. M. J. *Artificial intelligence through simulated evolution*. Wiley, 1966.

[34] Graf. J and Banzhaf. W. Interactive evolution of images. *Proceedings International Conference on Evolutionary Programming*, pages 53–65, 1995.

[35] Graham. D. J and Redies. C. Statistical regularities in art: Relations with visual coding and perception. *Vision Research*, 50(16):1503–1509, 2010.

[36] Hays. J and Essa. I. Image and video based painterly animation. In *NPAR '04 3rd International Symposium on Non-Photorealistic Animation and Rendering*, pages 113–120. ACM New York, NY, USA, 2004.

[37] Huxtable. J. JH LABS. Java Image Processing. Retrieved from http://www.jhlabs.com, 2004. Last accessed April 16, 2013.

[38] Ralph. W. J. Painting the bell curve: The occurrence of the normal distribution in fine art. Unpublished, 2006.

[39] Rigau. J, Feixas. M, and Sbert. M. Conceptualizing birkhoff's aesthetic measure using shannon entropy and kolmogorov complexity. In *Computational Aesthetics*, pages 105–112, 2007.

[40] Ross. B. J and Zhu. H. Procedural texture evolution using multi-objective optimization. *New Generation Computing*, 22(3):271–293, 2004.

[41] Ross. B. J, Ralph. W, and Zong. H. Evolutionary image synthesis using a model of aesthetics. In *IEEE Congress on Evolutionary Computation*, pages 1087–1094. CEC, 2006.

[42] Deb. K. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 2009.

[43] Sims. K. Artificial evolution for computer graphics. *ACM Computer Graphics*, 25(4):319–328, 1991.

[44] Sims. K. evolution of equations for procedural models. *The Visual Computer 9*, 9:466476, 1993.

[45] Wiens. A. L and Ross. B. J. Gentropy: evolving 2d textures. *Computers and Graphics Journal*, 26(1):75–88, 2006.

[46] Cover T. M and Thomas J. A. *Elements of Information theory*. Wiley Series in Telecommunications, 1991.

[47] Hughes. J. M, Graham. D. J, and Rockmore. D. N. Quantification of artistic style through sparse coding analysis in the drawings of pieter bruegel the elder. *Proceedings Natunal Academic Science*, 107(4):12791283, 2010.

[48] Lewis. M. Aesthetic evolutionary design with data flow networks. *Proceedings Generative Art 2000*, 2000.

[49] Miyahara. M and Yoshida. Y. Mathematical transform of (r,g,b) color data to munsell (h,v,c) color data. In *SPIE Proceedings Visual Communication and Image Processing*, volume 1001, page 650657, 1988.

[50] Shiraishi. M and Yamaguchi. Y. An algorithm for automatic painterly rendering based on local source image approximation. In *NPAR '00 Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering*, pages 53–58. ACM, New York, 2000.

[51] David J. Montana. Strongly typed genetic programming. *Evolutionary Computation*, 3:199–230, 1994.

[52] Siu-Hang C. N, Baxter. W, Wei. Li-Yi, and Govindaraju. N. K. Detail-preserving paint modeling for 3d brushes. In *NPAR '10: Non-Photorealistic Animation and Rendering*, pages 27–34, 2010.

[53] Xie. N, Laga. H, Saito. S, and Nakajima. M. Ir2s: interactive real photo to sumi-e. In *NPAR '10: Non-Photorealistic Animation and Rendering*, pages 63–71, 2010.

[54] NIH. ImageJ. Image Processing and Analysis in Java. Retrieved from http://rsb.info.nih.gov, 2004. Last accessed April 16, 2013.

[55] Barile. P, Ciesielski. V, and Trist. K. Non-photorealistic rendering using genetic programming. In *SEAL '08 Proceedings of the 7th International Conference on Simulated Evolution And Learning*, pages 299–308, Melbourne, 2008. Springer.

[56] Collomosse. J. P. Evolutionary search for the artistic rendering of photographs. In *The Art of Artificial Evolution*, pages 39–62, 2008.

[57] Haeberli. P. Paint by numbers: Abstract image representation. *Computer Graphics*, 24(4):207–214, 1990.

[58] Machado. P and Cardoso. A. Computing aesthetics. In de Oliveira. F.M., editor, *SBIA 1998. LNCS (LNAI)*, pages 219–228, Heidelberg, 1998. Springer.

[59] Machado. P, Dias. A, Duarte. N, and A Cardoso, editors. *Giving Colour to Images*. Proceedings AISB 2002 Symposium on AI and Creativity in the Arts, 2002.

[60] Walsh. P and Gade. P. The use of an aesthetic measure for the evolution of fractal landscapes. In *IEEE Congress on Evolutionary Computation*, pages 1613–1619, 2011.

[61] Kolas. Q. Image processing with gluas. Retrieved from http://codecave.org/image-processing, 2005. Last accessed April 16, 2013.

[62] Xu. Q, D'Souza. D. J, and Ciesielski. V. Evolving images for entertainment. In *IE*, page 26, 2007.

[63] Bergen. S. R. automatic structure generation using genetic programming and fractal geometry. Master's thesis, Department of Computer Science, Brock Univeristy, 2011.

[64] Dawkins. R. *The Blind Watchmark*. Penguin Books Ltd, 1986.

[65] Gonzalez. R and Woods. R. *Digital Image Processing*. Prentice Hall, 2 edn edition, 2000.

[66] Koza. J. R. *Genetic Programming: On the programming of computers by means of natural selection*. MIT Press, 1992.

[67] Poli. R, Langdon. W. R, and McPhee. N. F. *A Field Guide to Genetic Programming*. http://lulu.com and freely available at http://www.gp-field-guide.org.uk, 2008. With contributions by Koza. J. R.

[68] Smith. J. R and Chang. S. F. Visual seek: a fully automated content-based image query system. In *MULTIMEDIA '96 Proceedings of the 4th ACM International Conference on Multimedia*, pages 87–98, New York, 1996. ACM.

[69] Colton. S, Valstar. M. F, and Pantic. M. Emotionally aware automated portrait painting. In *Proceedings of the 3rd International Conference on Digital Interactive Media in Entertainment and Arts*, pages 304–311, 2008.

[70] Colton. S and Torres. P. Evolving approximate image filters. In *Proceedings of the EvoWorkshops*, pages 467–477. LNCS, 2009.

[71] Fan. S, Wang. R, Zhang. Y, and Guo. K. Classifying computer generated graphics and natural images based on image contour information. *Journal of Information and Computational Science*, 9(10):28772895, 2012.

[72] Khan. S. S and Vogel. D. Evaluating visual aesthetics in photographic portraiture. In *EXPRESSIVE: 8th International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging (CAe), Annecy*, 2012.

[73] Luke S. Ecj - a java-based evolutionary computation research system. Retrieved from http://cs.gmu.edu/ eclab/projects/ecj/. Last accessed April 16, 2013.

[74] Rooke S. The genetic-evolutionary art process of Steven Rooke. Retrieved from http://srooke.com/process.html, 1993. Last accessed April 16, 2013.

[75] Todd. S and Latham. W. *Evolutionary Art and Computers*. Academic Press, 1992.

[76] Yamamoto. S, Mao. X, and Imamiya. A. Colored pencil filter with custom colors. In *PG '04 Proceedings of 12th Pacific Conference on Computer Graphics and Applications*, pages 329–338. IEEE Computer Society Washington, DC, USA, 2004.

[77] Strothotte. T and Schlechtweg. S. *Non-Photorealistic Computer Graphic modeling rendering and animation*. Morgan Kaufmann, 2002.

[78] Van Laerhoven. T and Van Reeth. F. Brush up your painting skills: Realistic brush design for interactive painting applications. *The Visual Computer: International Journal of Computer Graphics*, 23(9):763–771, 2007.

[79] Niblack. W, Barber. R, Equitz. W, Flickner. M. D, Glasman. E. H, Petkovic. D, Yanker. P, Faloutsos. C, and Taubin. G. Qbic project: querying images by content, using color, texture, and shape. In *Storage and Retrieval for Image and Video Databases*, volume 1908, pages 173–187, 1993.

[80] Van Haever. W, Van Laerhoven. T, Di Fiore. F, and Van Reeth. F. From dust till drawn: A real-time bidirectional pastel simulation. *The Visual Computer*, 23(9-11):925–934, 2007.

Appendix A

Scores and Resulting Images of Experiments

Name	Value
U Statistic	87
Mean	50
Standard Deviation	13.22876
Z score	2.796937
Z Critical for 0.05	-1.64485

a. Experiment 1 and 2

Name	Value
U Statistic	100
Mean	50
Standard Deviation	13.22876
Z score	3.779645
Z Critical for 0.05	-1.64485

b. Experiment 1 and 3

Name	Value
U Statistic	66
Mean	50
Standard Deviation	13.22876
Z score	1.209486
Z Critical for 0.05	-1.64485

c. Experiment 1 and 4

Name	Value
U Statistic	100
Mean	50
Standard Deviation	13.22876
Z score	3.779645
Z Critical for 0.05	-1.64485

d. Experiment 2 and 3

Name	Value
U Statistic	13
Mean	50
Standard Deviation	13.22876
Z score	-2.79694
Z Critical for 0.05	-1.64485

e. Experiment 2 and 4

Name	Value
U Statistic	0
Mean	50
Standard Deviation	13.22876
Z score	-3.77964
Z Critical for 0.05	-1.64485

f. Experiment 3 and 4

Table A.1: Mann-Whitney test scores of CHISTQ for four experiments

Name	Value
U Statistic	47
Mean	50
Standard Deviation	13.22876
Z score	-0.22678
Z Critical for 0.05	-1.64485

a. Normal and Bright 1

Name	Value
U Statistic	40
Mean	50
Standard Deviation	13.22876
Z score	-0.75593
Z Critical for 0.05	-1.64485

b. Normal and Bright 2

Name	Value
U Statistic	51
Mean	50
Standard Deviation	13.22876
Z score	0.075593
Z Critical for 0.05	-1.64485

c. Normal and Bright 3

Name	Value
U Statistic	28
Mean	50
Standard Deviation	13.22876
Z score	-1.66304
Z Critical for 0.05	-1.64485

d. Normal and Dark

Name	Value
U Statistic	54
Mean	50
Standard Deviation	13.22876
Z score	0.302372
Z Critical for 0.05	-1.64485

e. Bright 1 and Bright 2

Name	Value
U Statistic	36
Mean	50
Standard Deviation	13.2288
Z score	-1.0583
Z Critical for 0.05	-1.6449

f. Bright 1 and Bright 3

Name	Value
U Statistic	79
Mean	50
Standard Deviation	13.22876
Z score	2.192194
Z Critical for 0.05	-1.64485

g. Bright 1 and Dark

Name	Value
U Statistic	36
Mean	50
Standard Deviation	13.22876
Z score	-1.0583
Z Critical for 0.05	-1.64485

h. Bright 2 and 3

Name	Value
U Statistic	74
Mean	50
Standard Deviation	13.22876
Z score	1.814229
Z Critical for 0.05	-1.64485

i. Bright 2 and Dark

Name	Value
U Statistic	85
Mean	50
Standard Deviation	13.22876
Z score	2.645751
Z Critical for 0.05	-1.64485

j. Bright 3 and Dark

Table A.2: Mann-Whitney test scores of DFN for different brush intensities

Name	Value
U Statistic	82
Mean	50
Standard Deviation	13.22876
Z score	2.418973
Z Critical for 0.05	-1.64485

a. Normal and Weighted Bright 3

Name	Value
U Statistic	81
Mean	50
Standard Deviation	13.22876
Z score	2.34338
Z Critical for 0.05	-1.64485

b. Bright 3 and Weighted Bright 3

Table A.3: Mann-Whitney test scores of DFN for weighted DFN experiment

Name	Value
U Statistic	74
Mean	50
Standard Deviation	13.22876
Z score	1.814229
Z Critical for 0.05	-1.64485

a. Source Image and Target 1

Name	Value
U Statistic	73
Mean	50
Standard Deviation	13.22876
Z score	1.738637
Z Critical for 0.05	-1.64485

b. Source Image and Target 2

Name	Value
U Statistic	41
Mean	50
Standard Deviation	13.22876
Z score	-0.68034
Z Critical for 0.05	-1.64485

c. Source Image and Target 3

Name	Value
U Statistic	29
Mean	50
Standard Deviation	13.22876
Z score	-1.58745
Z Critical for 0.05	-1.64485

d. Source Image and Target 4

Name	Value
U Statistic	58
Mean	50
Standard Deviation	13.22876
Z score	0.604743
Z Critical for 0.05	-1.64485

e. Source Image and Target 5

Name	Value
U Statistic	58
Mean	50
Standard Deviation	13.22876
Z score	0.604743
Z Critical for 0.05	-1.64485

f. Target 1 and Target 2

Name	Value
U Statistic	22
Mean	50
Standard Deviation	13.22876
Z score	-2.1166
Z Critical for 0.05	-1.64485

g. Target 1 and Target 3

Name	Value
U Statistic	75
Mean	50
Standard Deviation	13.22876
Z score	1.889822
Z Critical for 0.05	-1.64485

h. Target 1 and Target 4

Name	Value
U Statistic	38
Mean	50
Standard Deviation	13.22876
Z score	-0.90711
Z Critical for 0.05	-1.64485

i. Target 1 and Target 5

Name	Value
U Statistic	18
Mean	50
Standard Deviation	13.22876
Z score	-2.41897
Z Critical for 0.05	-1.64485

j. Target 2 and Target 3

Name	Value
U Statistic	80
Mean	50
Standard Deviation	13.22876
Z score	2.267787
Z Critical for 0.05	-1.64485

k. Target 2 and Target 4

Name	Value
U Statistic	41
Mean	50
Standard Deviation	13.22876
Z score	-0.68034
Z Critical for 0.05	-1.64485

l. Target 2 and Target 5

Name	Value
U Statistic	67
Mean	50
Standard Deviation	13.22876
Z score	1.285079
Z Critical for 0.05	-1.64485

m. Target 3 and Target 4

Name	Value
U Statistic	62
Mean	50
Standard Deviation	13.22876
Z score	0.907115
Z Critical for 0.05	-1.64485

n. Target 3 and Target 5

Name	Value
U Statistic	70
Mean	50
Standard Deviation	13.22876
Z score	1.511858
Z Critical for 0.05	-1.64485

o. Target 4 and Target 5

Table A.4: Mann-Whitney test scores of DFN for different Target Color images

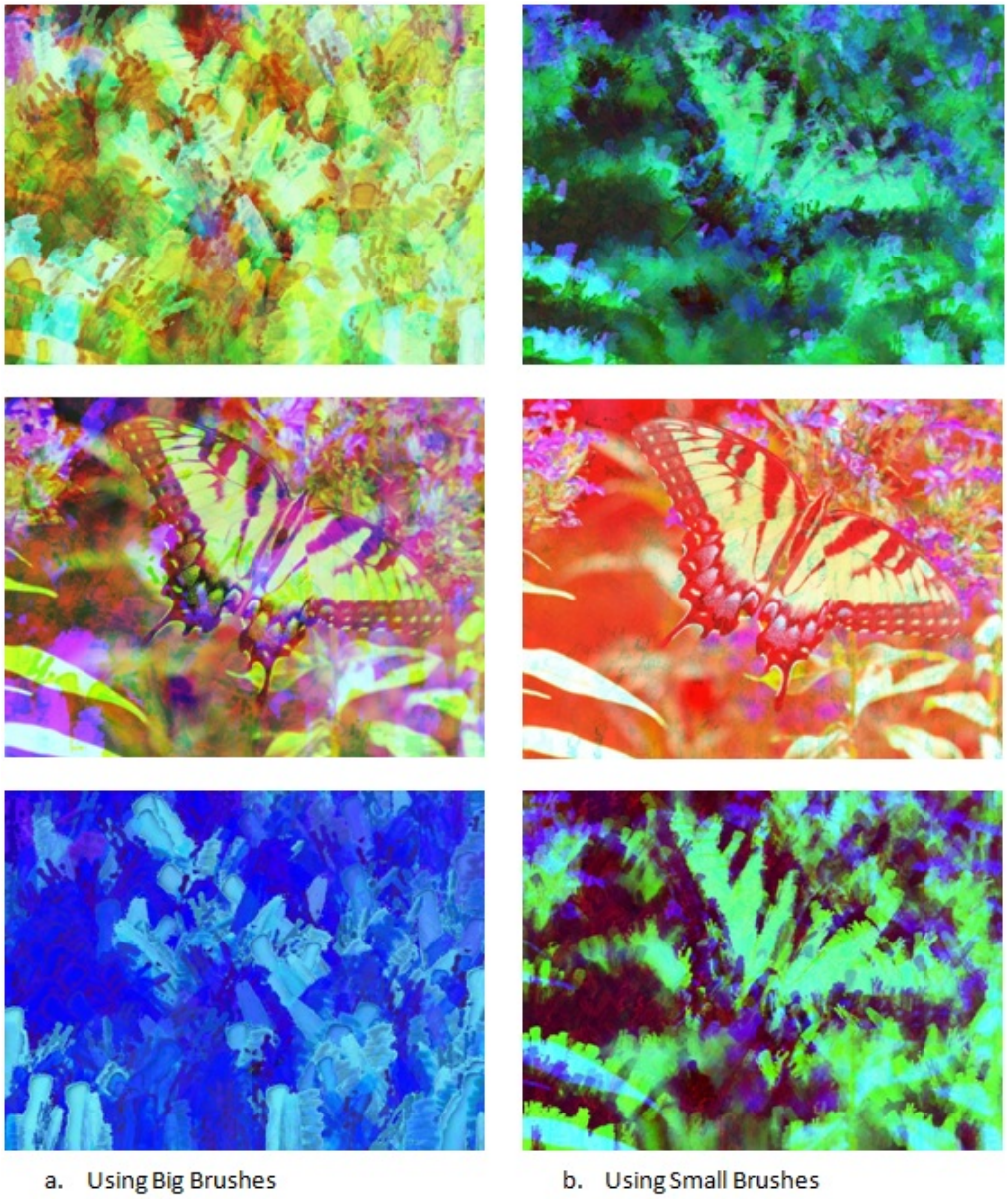


Figure A.1: First sample: The best of the last generation of three runs to compare big and small brushes.

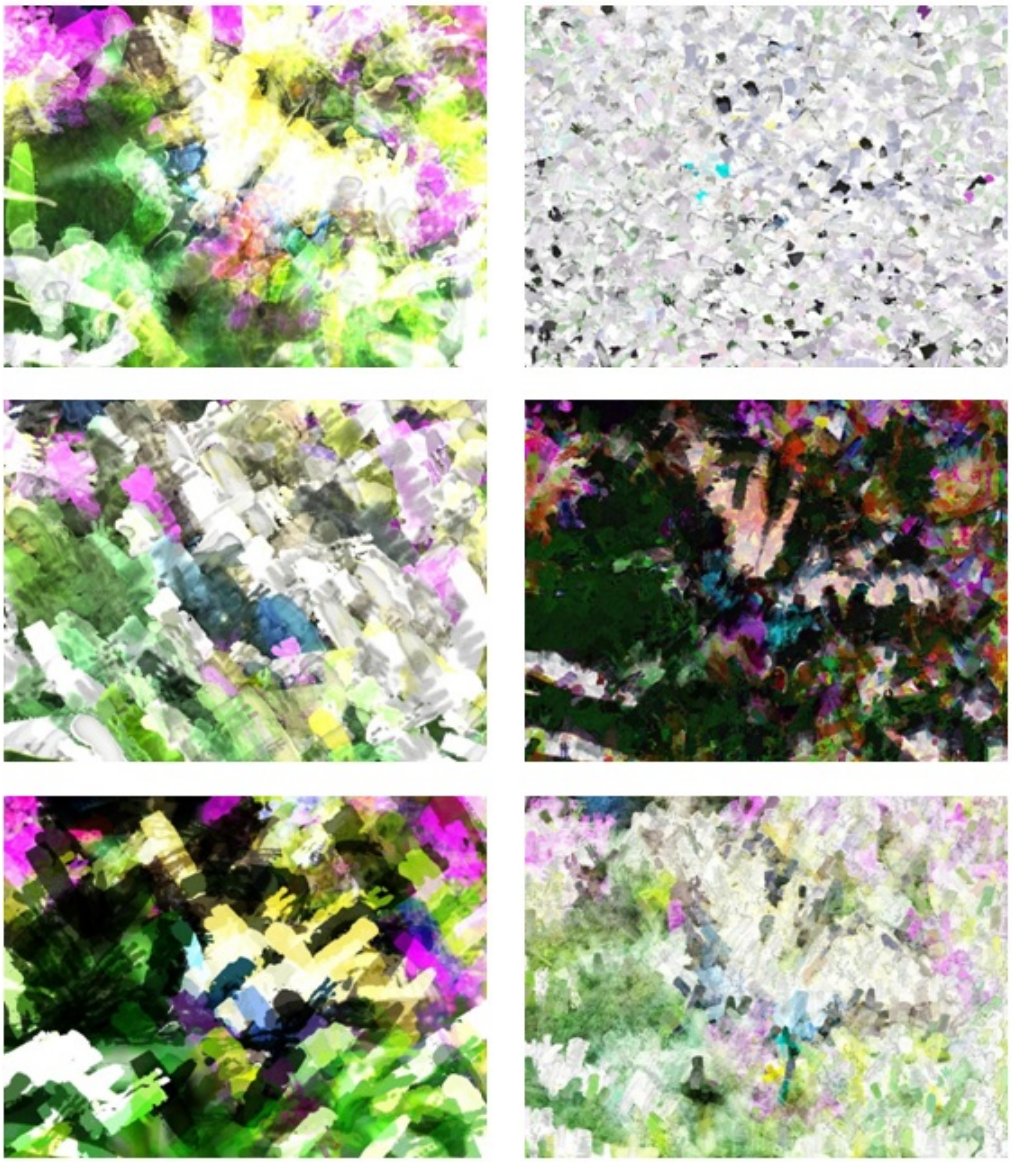


Figure A.2: Second sample: The best of the last generation of three runs to compare big and small brushes.

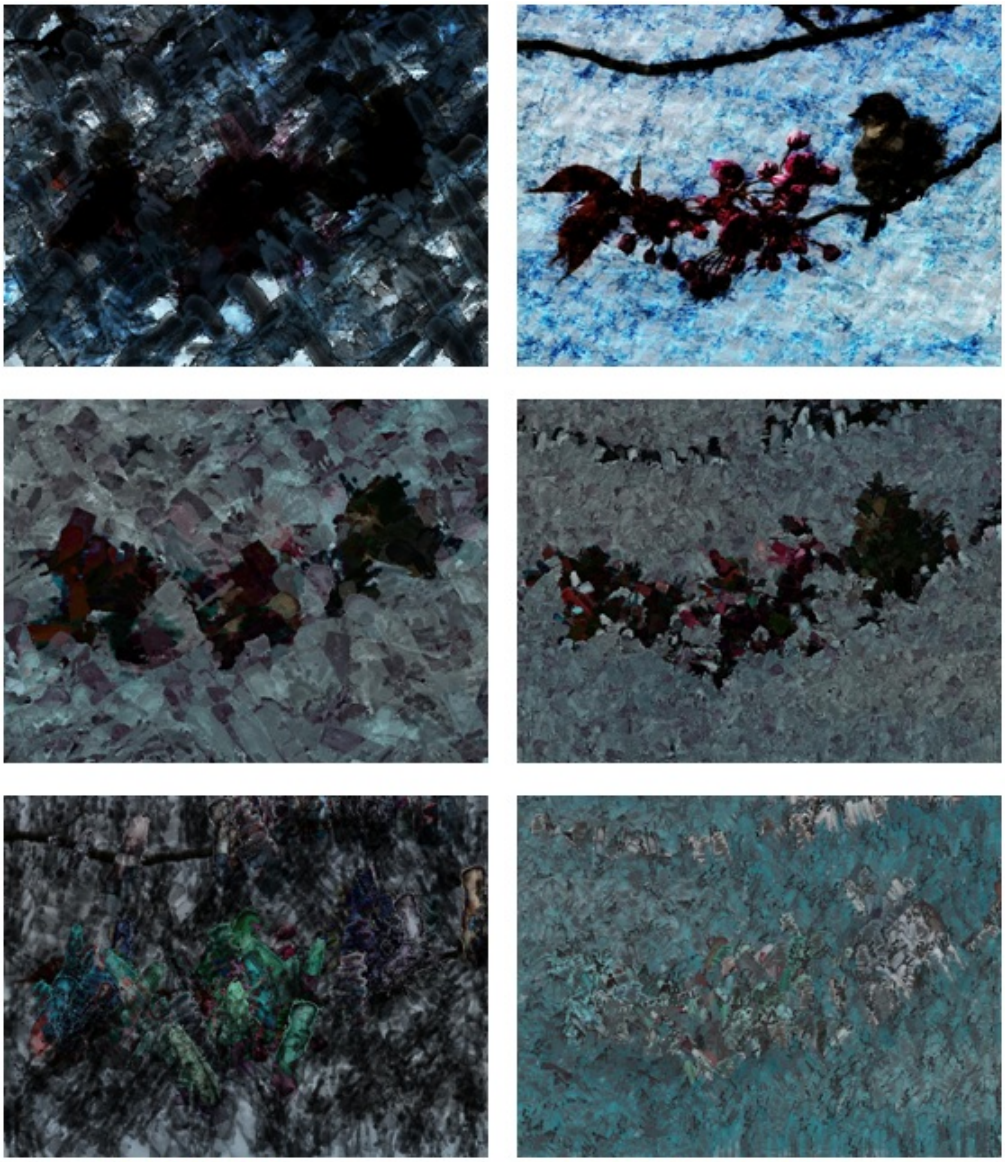


Figure A.3: Third sample: The best of the last generation of three runs to compare big and small brushes.

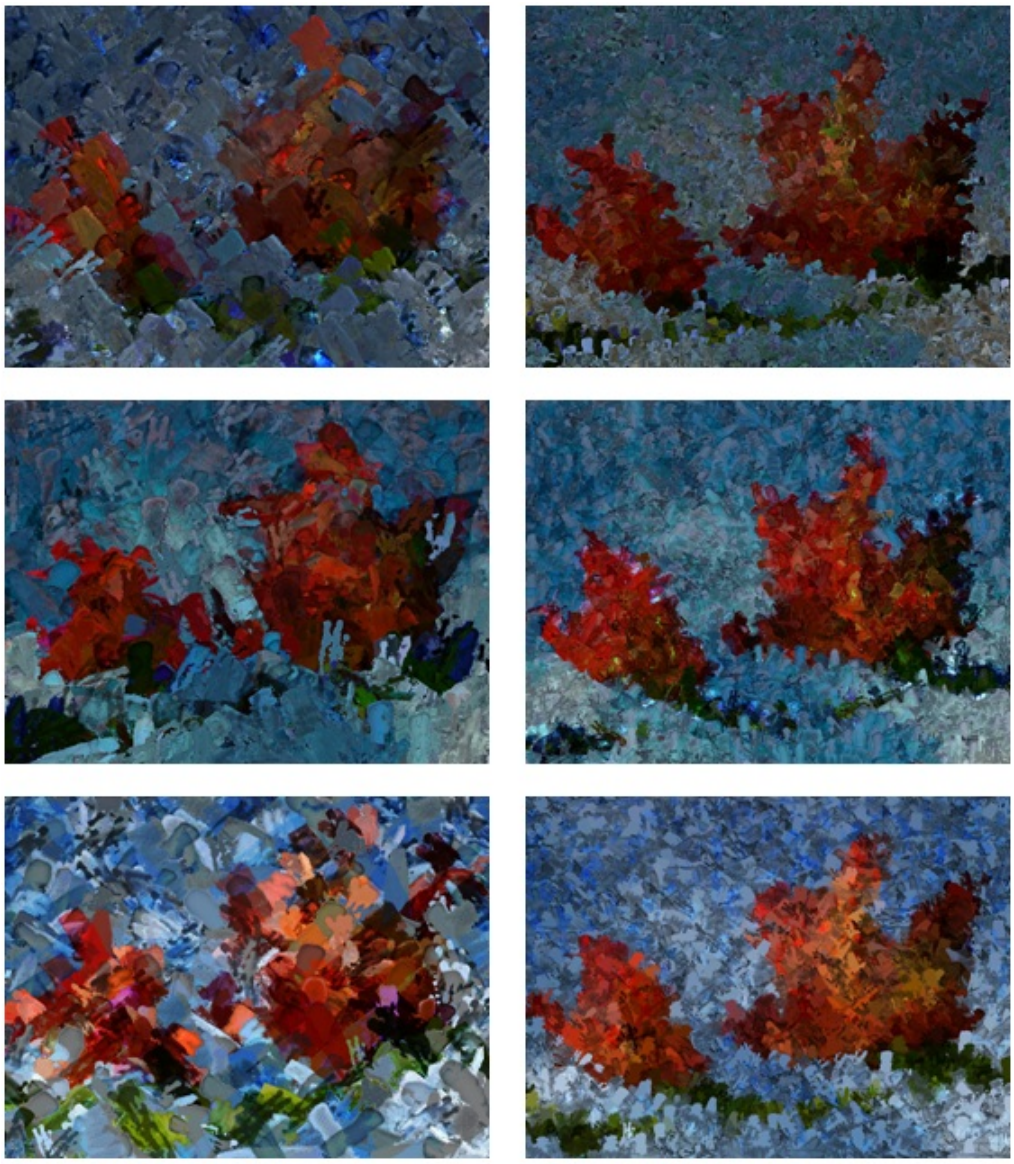


Figure A.4: Fourth sample: The best of the last generation of three runs to compare big and small brushes.

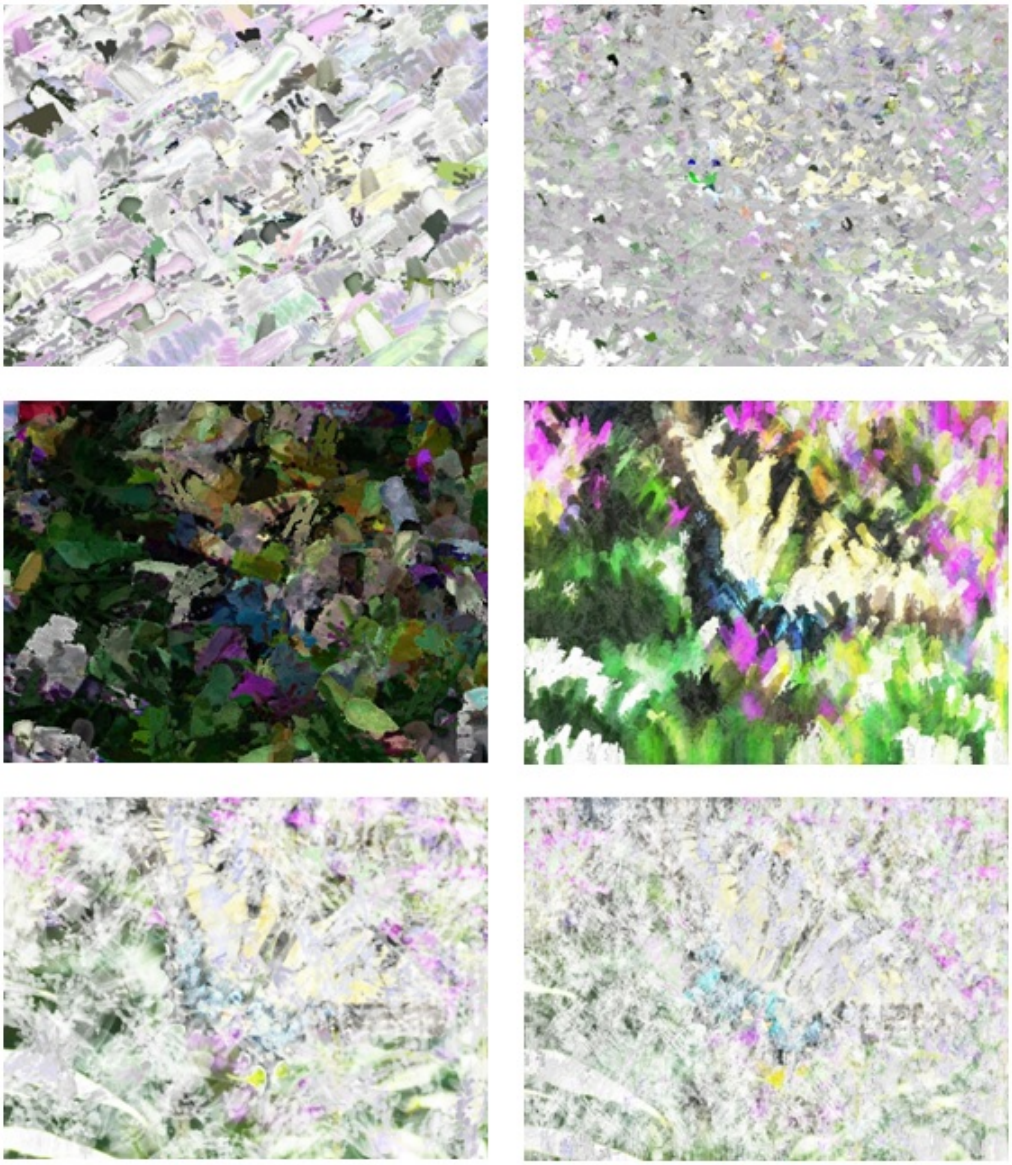


Figure A.5: Fifth sample: The best of the last generation of three runs to compare big and small brushes.

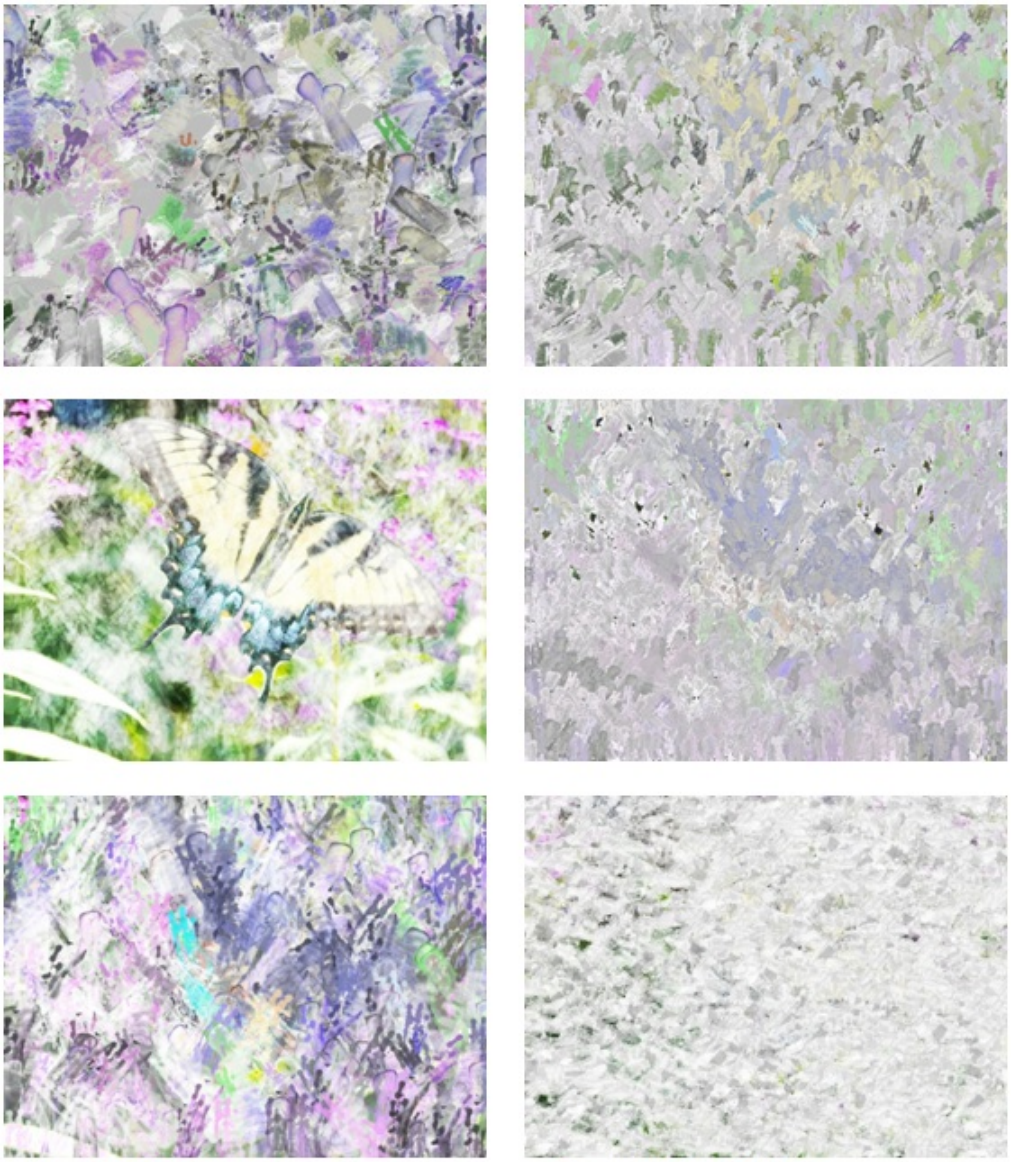


Figure A.6: Sixth sample: The best of the last generation of three runs to compare big and small brushes.

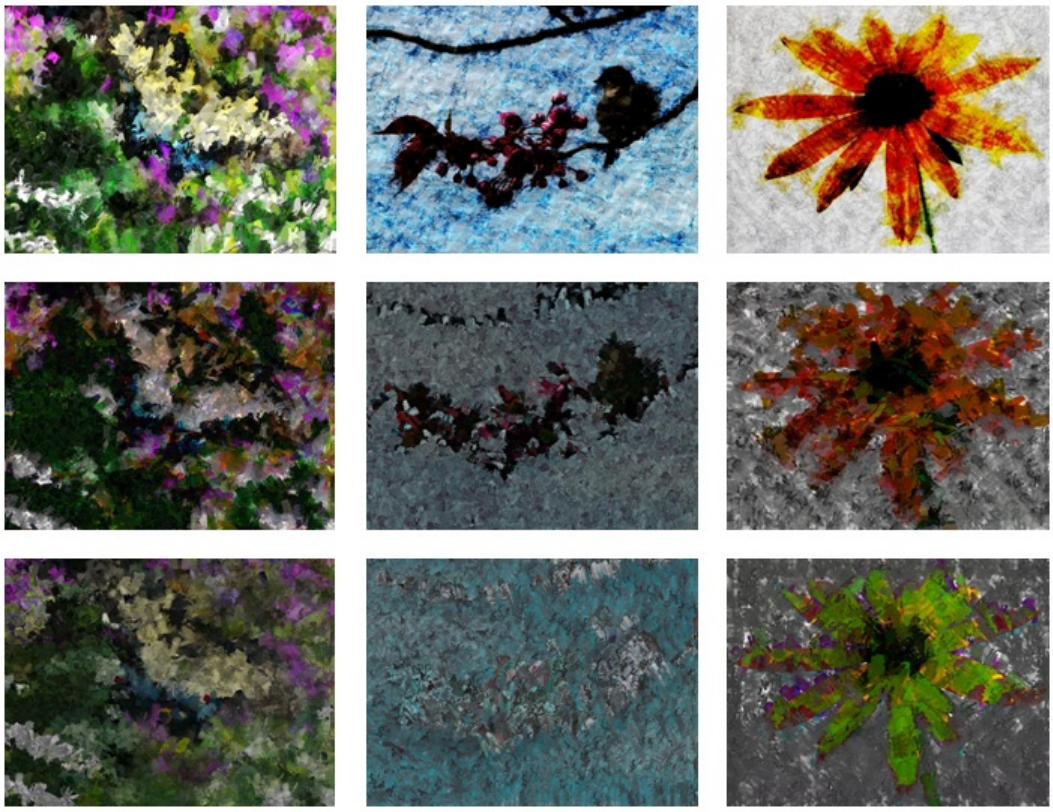


Figure A.7: Fist sample: The best of the last generation of three runs to compare using different canvas images.

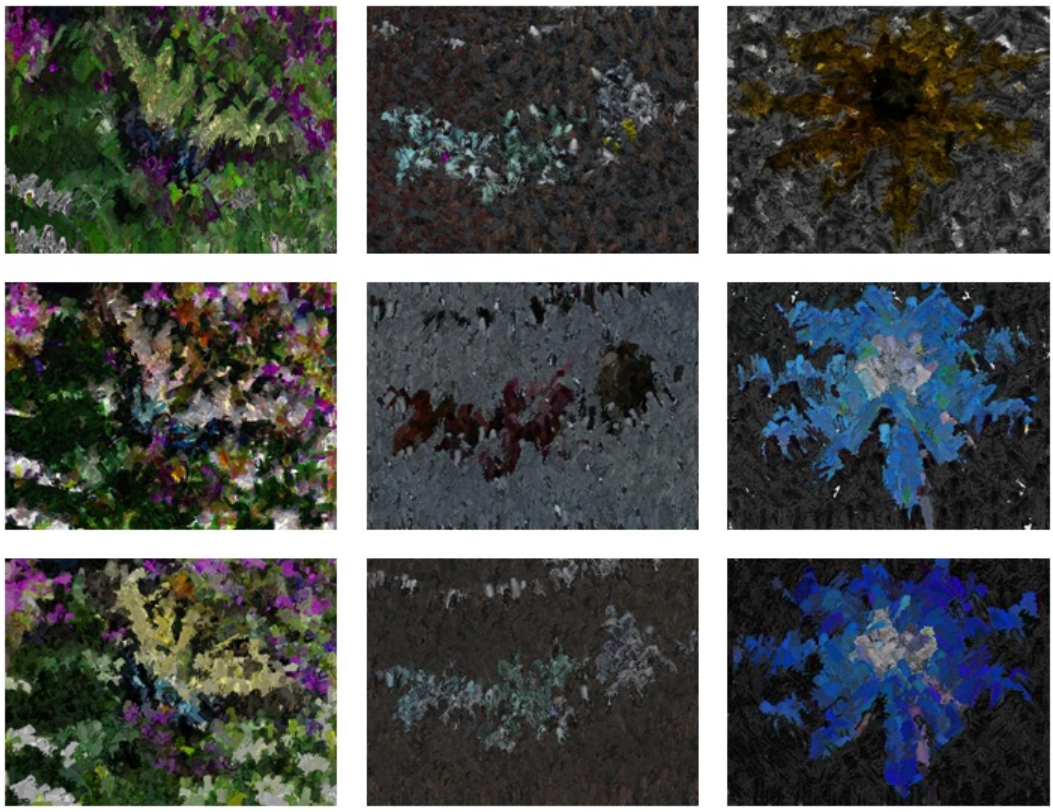


Figure A.8: Second sample: The best of the last generation of three runs to compare using different canvas images.

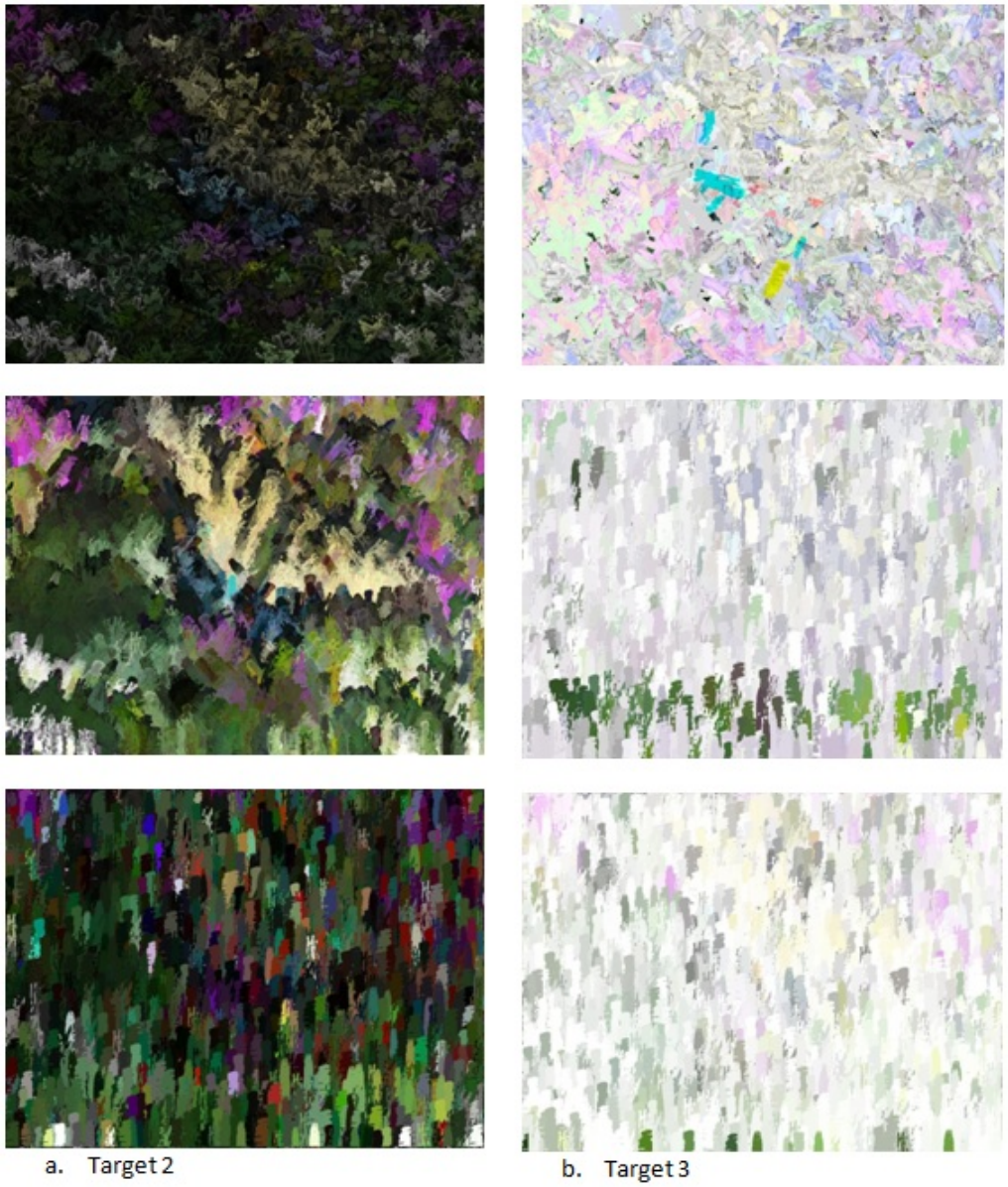


Figure A.9: First sample: The best of the last generation of three runs using different target color image.

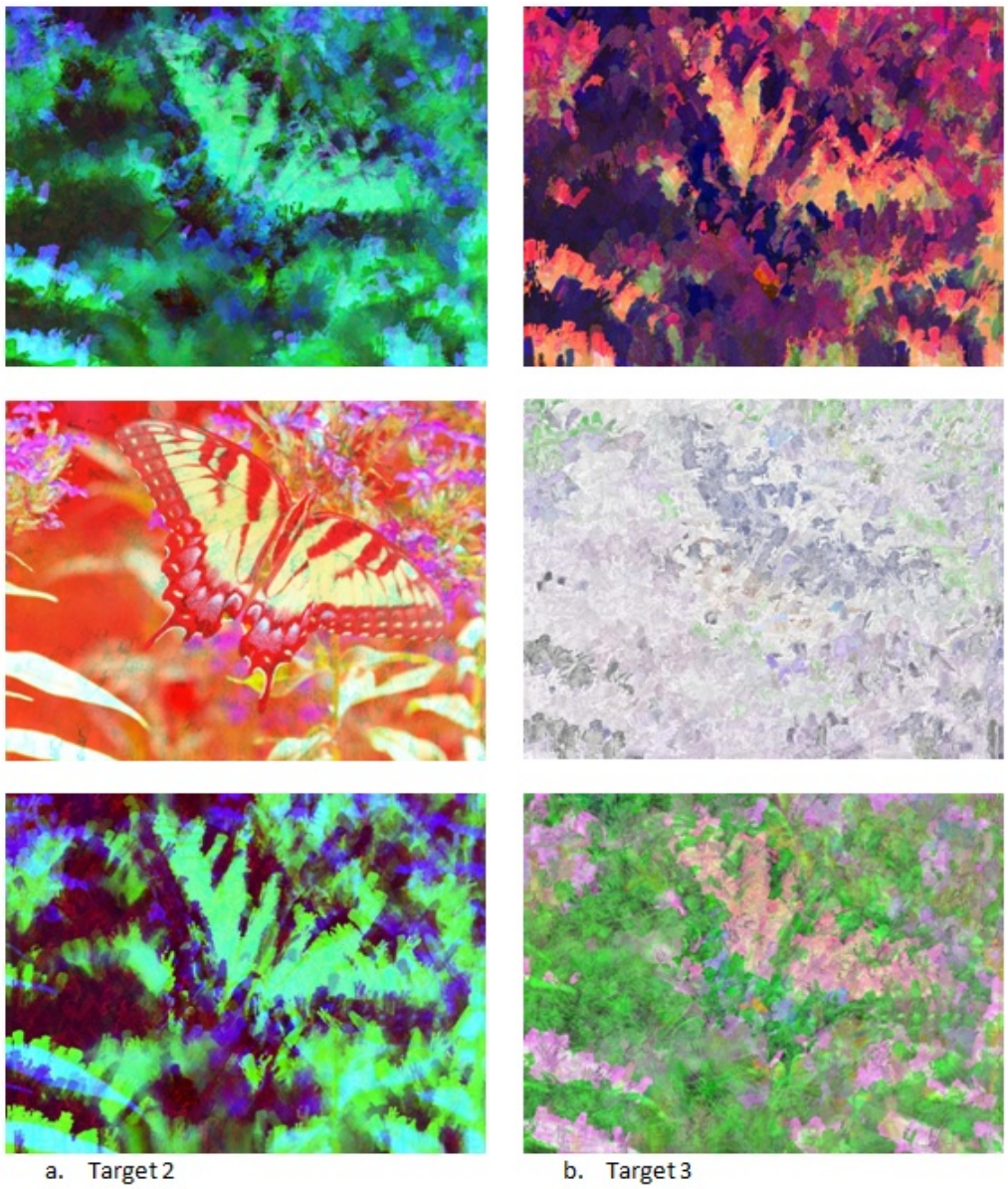


Figure A.10: Second sample: The best of the last generation of three runs using different target color image.

Figure A.11: Third sample: The best of the last generation of three runs using different target color image.

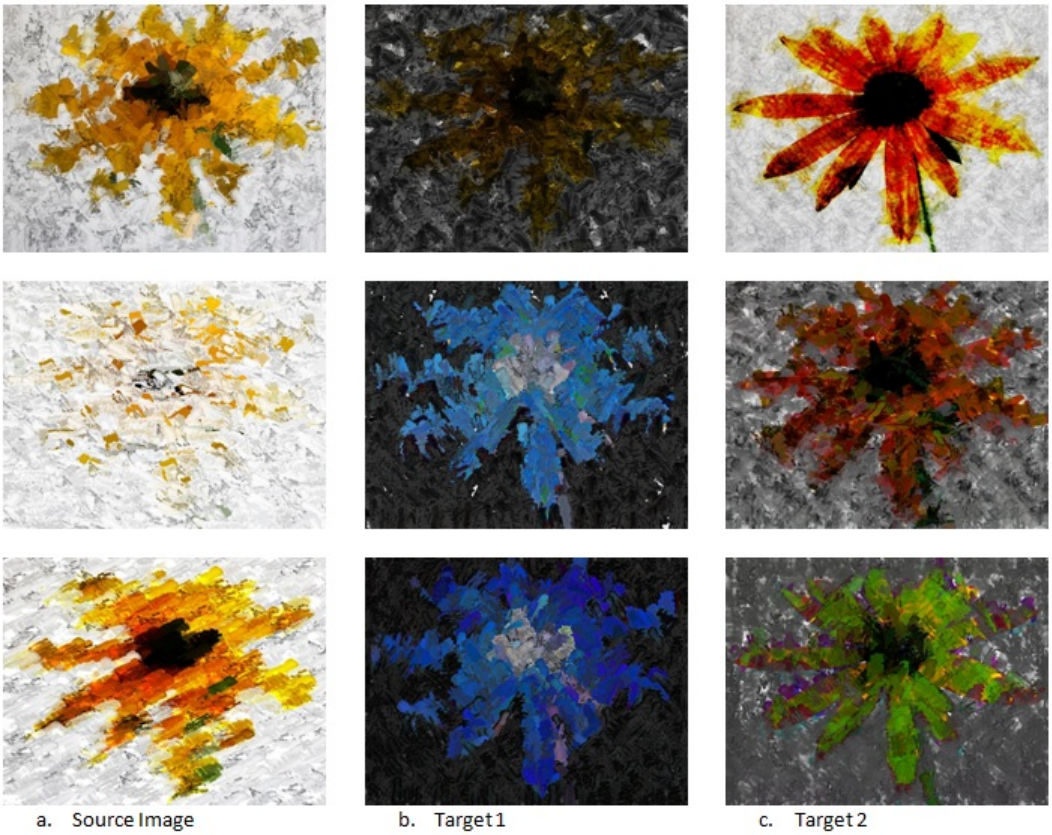


Figure A.12: Fourth sample: The best of the last generation of three runs using different target color image.

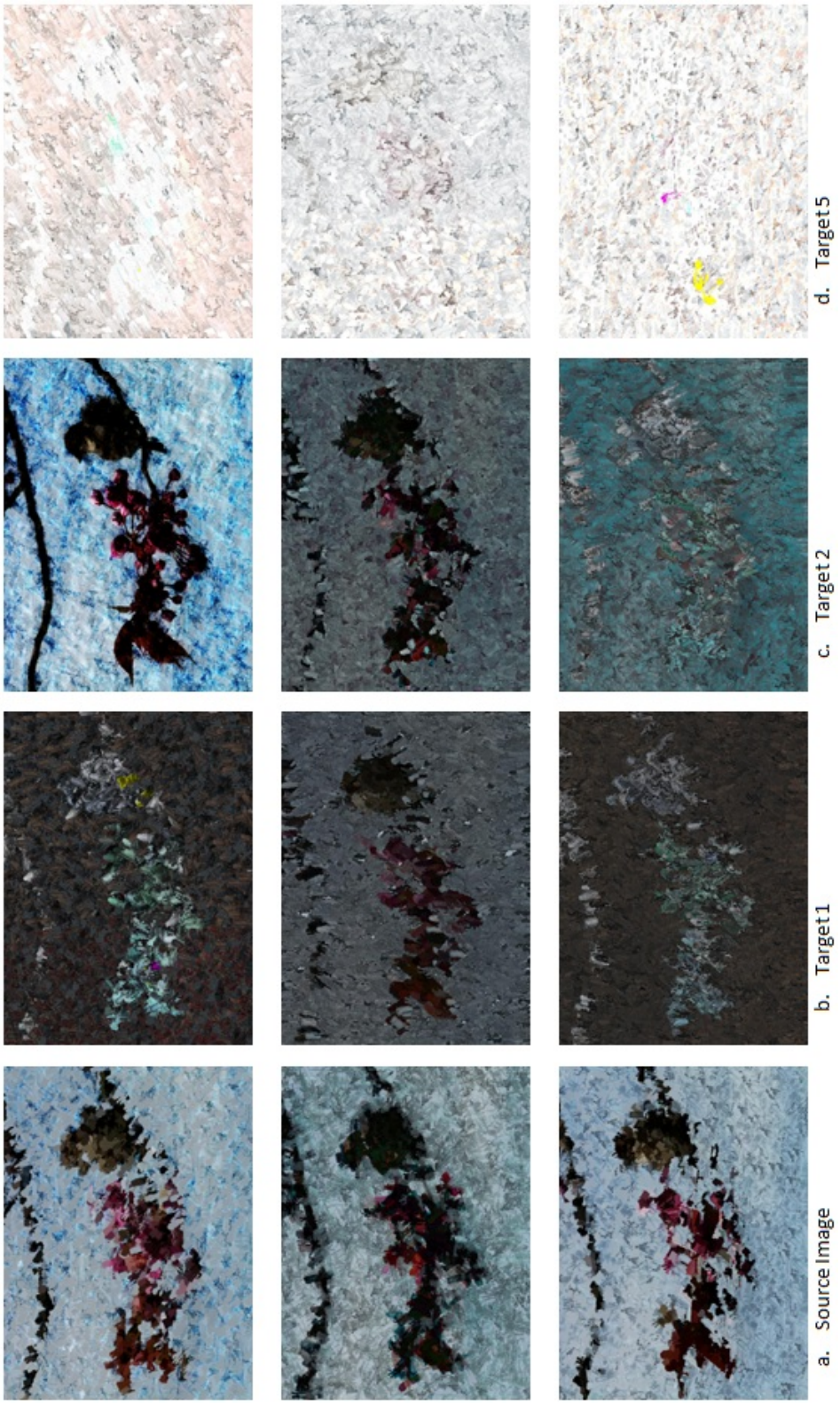


Figure A.13: Fifth sample: The best of the last generation of three runs using different target color image.

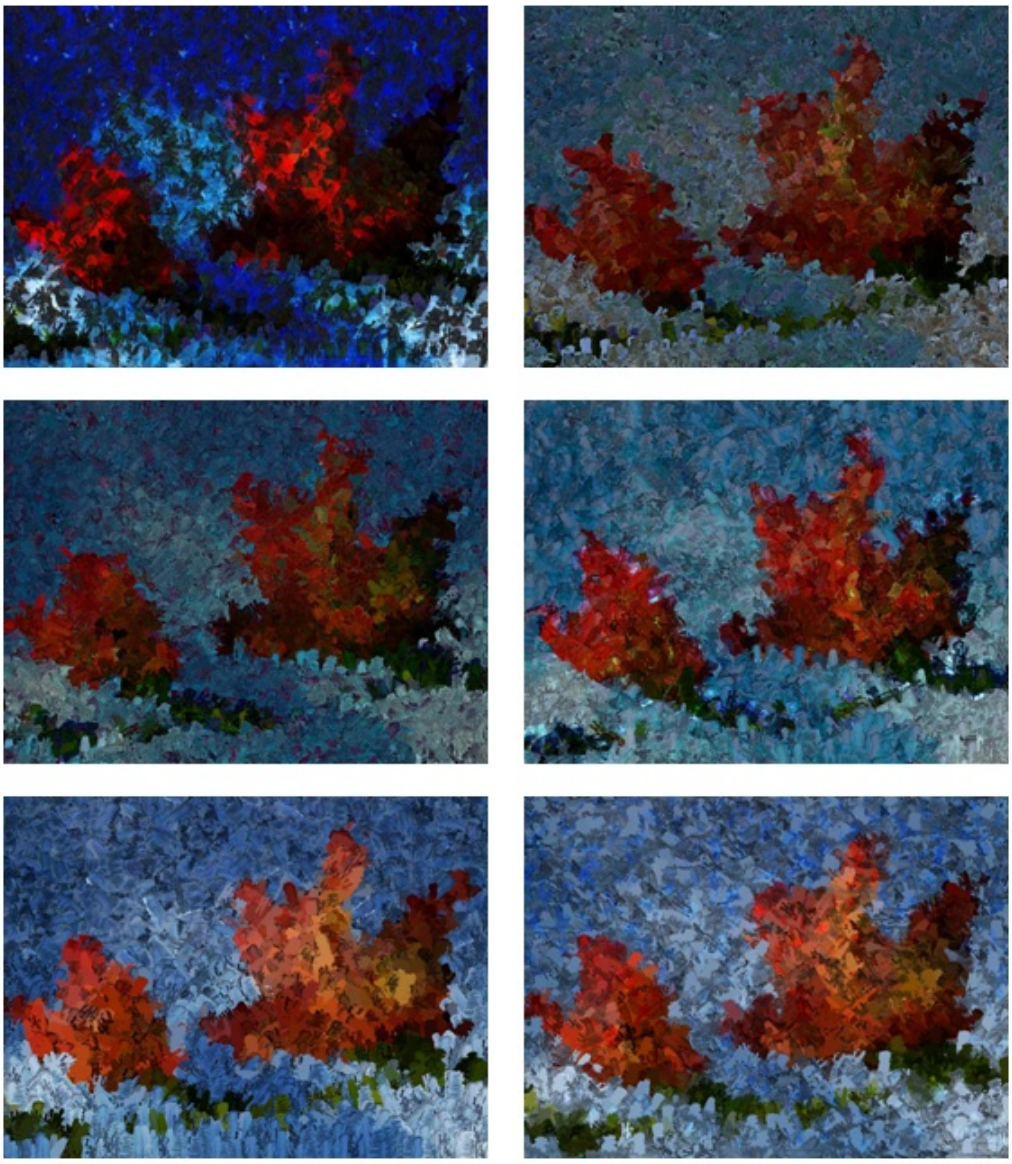


Figure A.14: Sixth sample: The best of the last generation of three runs using different target color image.

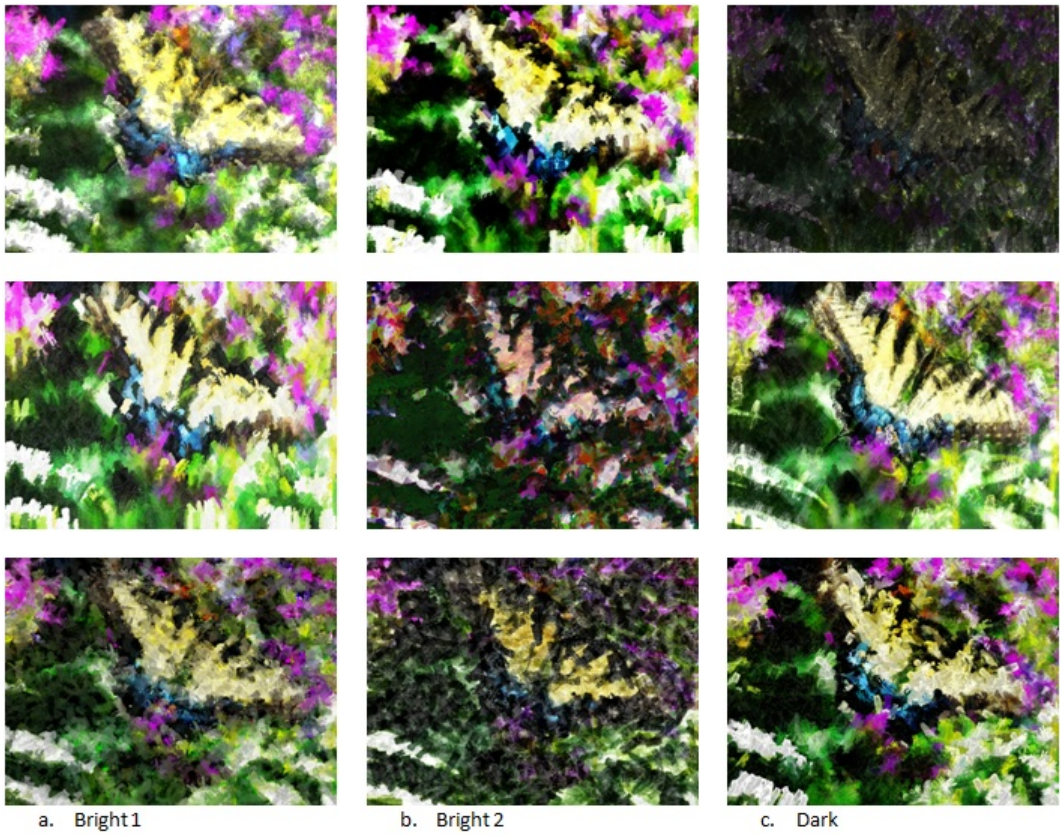


Figure A.15: First sample: The best of the last generation of three runs using different brush bitmap intensities.

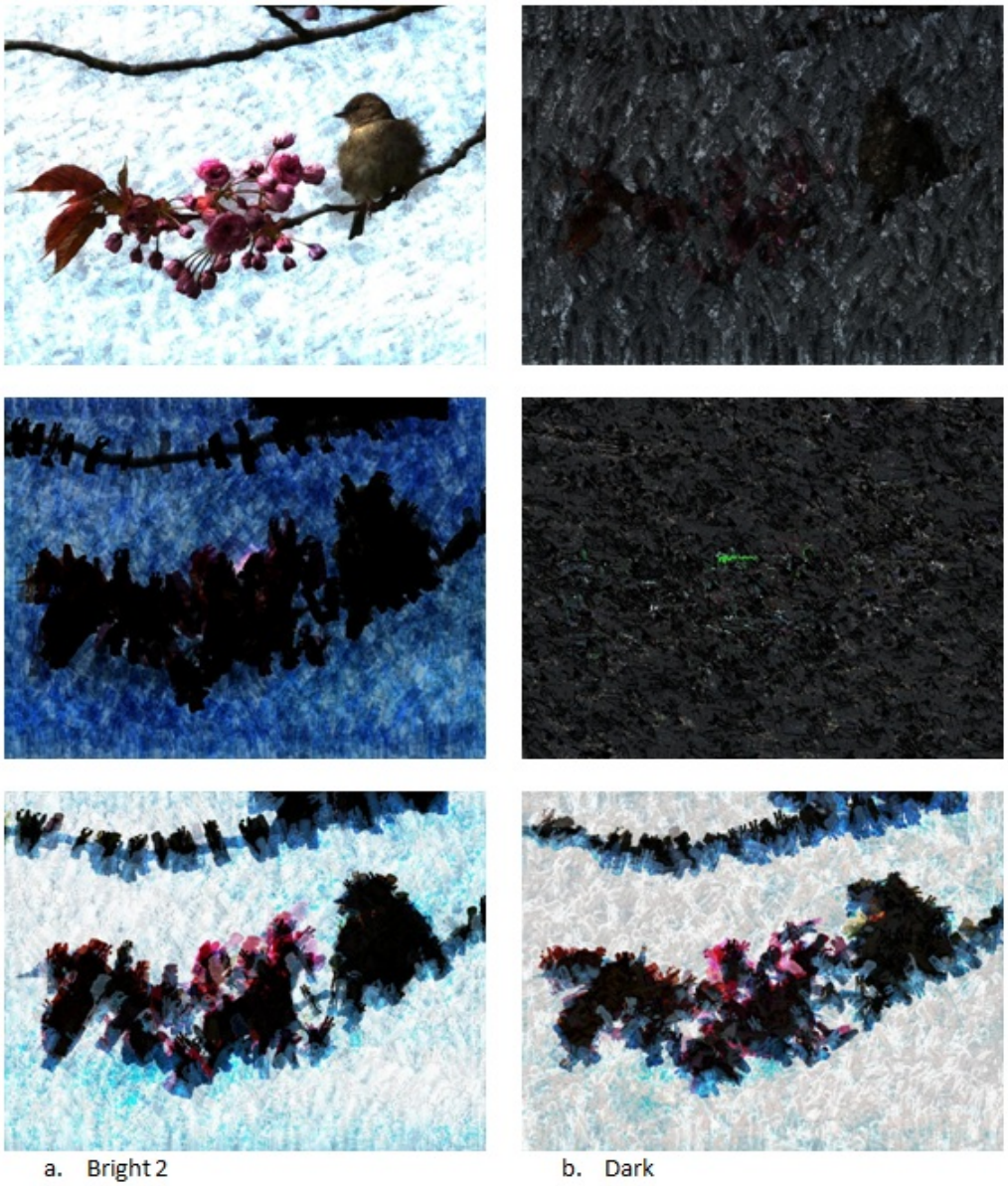


Figure A.16: Second sample: The best of the last generation of three runs using different brush bitmap intensities.

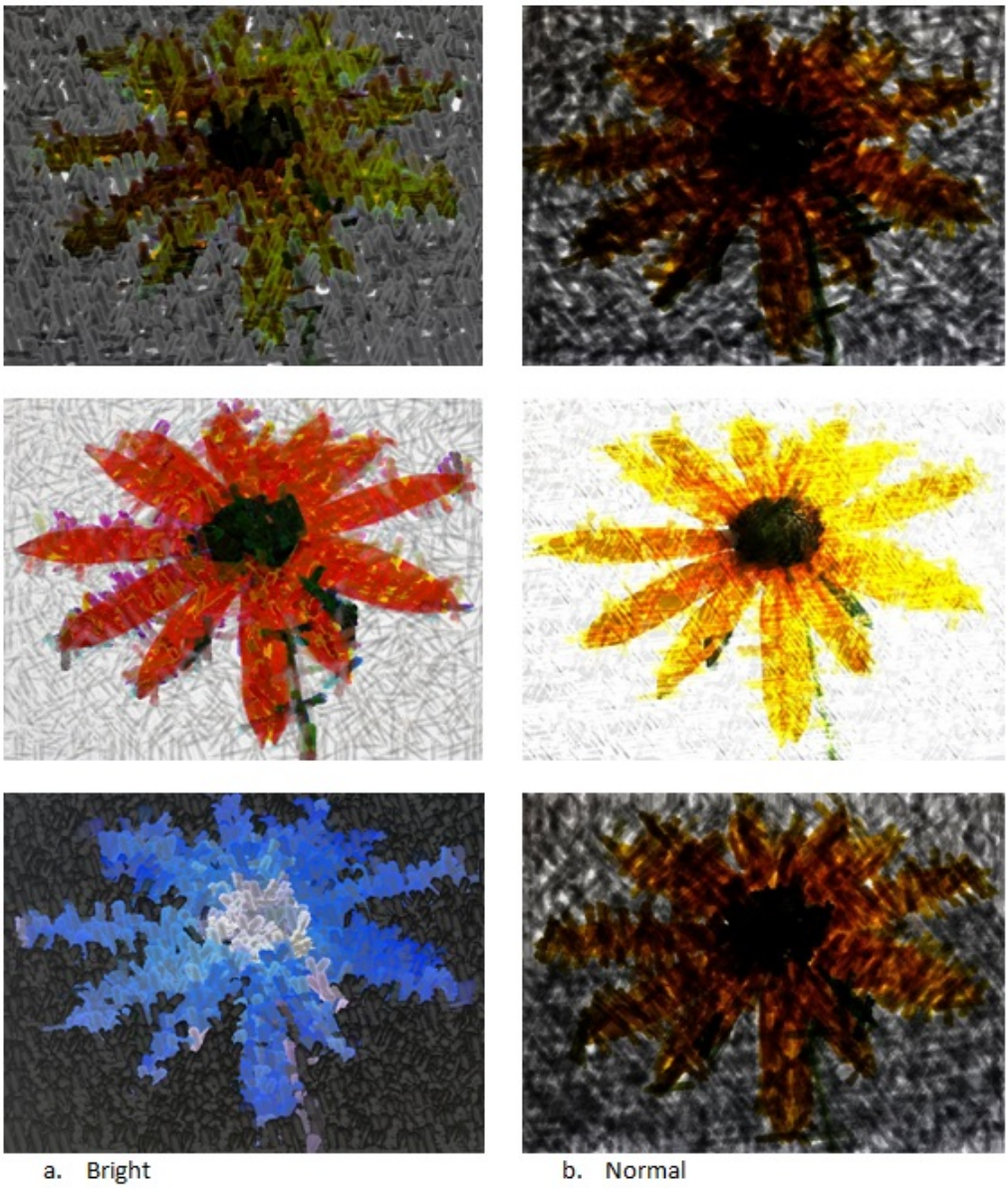


Figure A.17: Third sample: The best of the last generation of three runs using different brush bitmap intensities.

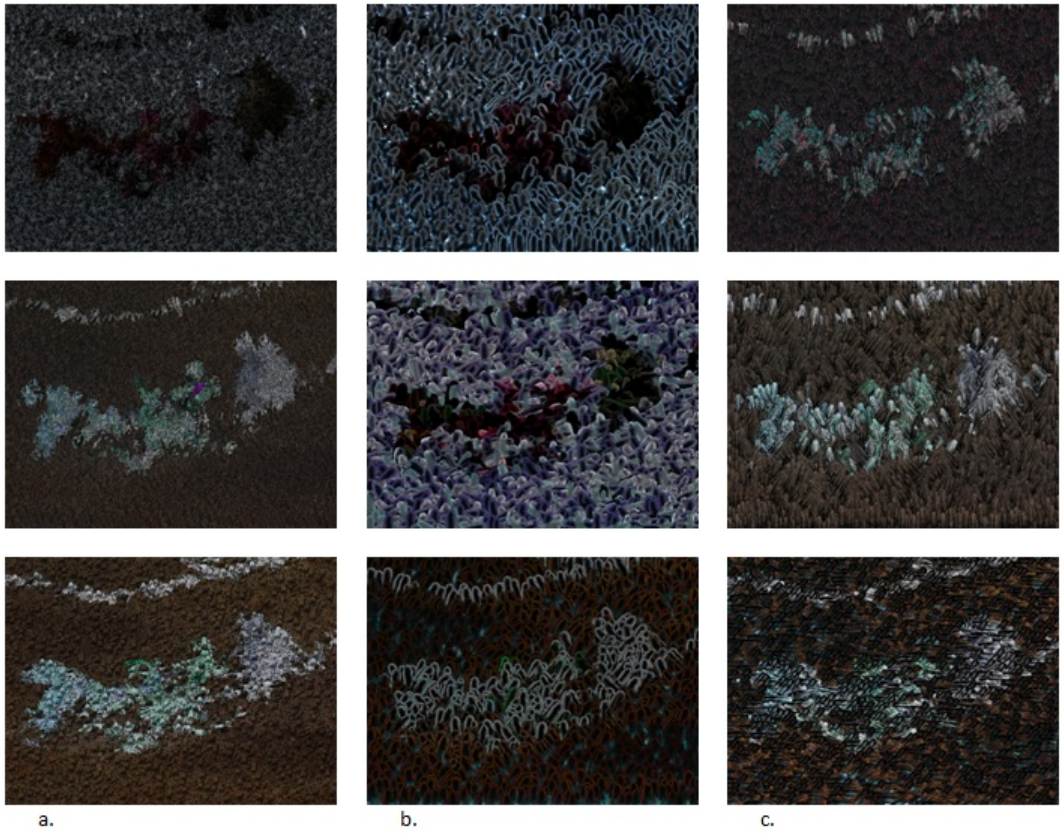


Figure A.18: First sample: The best of the last generation of three runs using different brush bitmap styles.

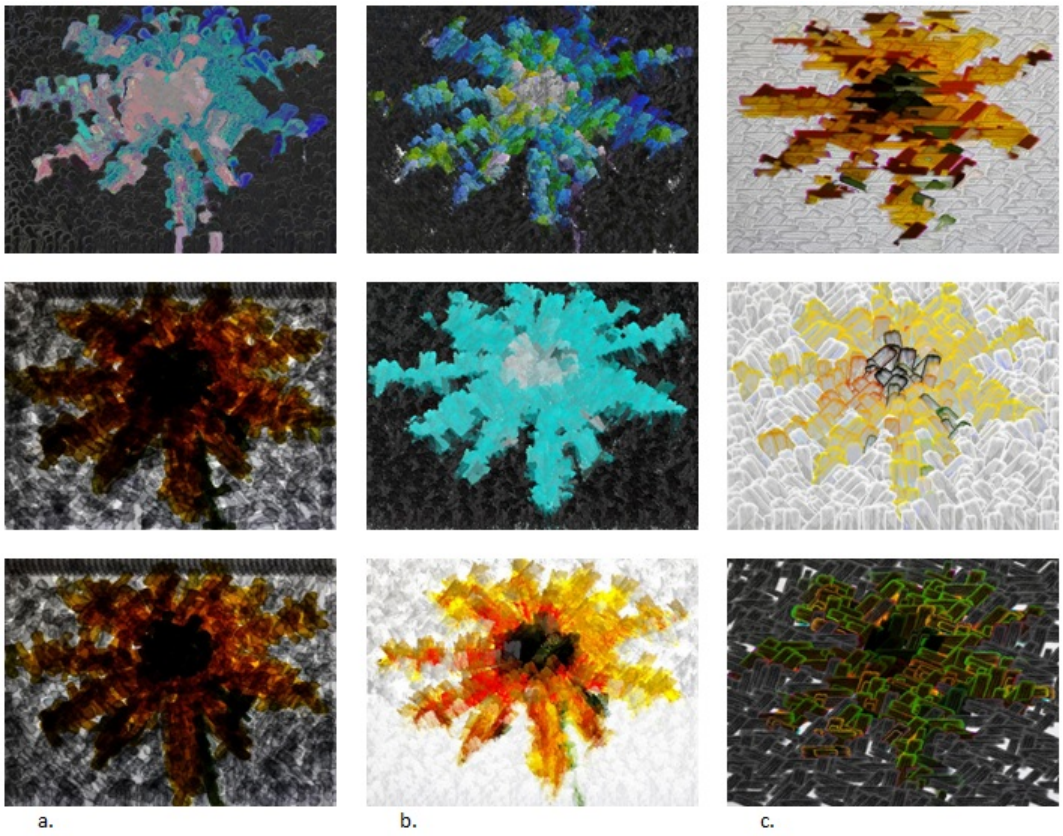


Figure A.19: First sample: The best of the last generation of three runs using different brush bitmap styles.

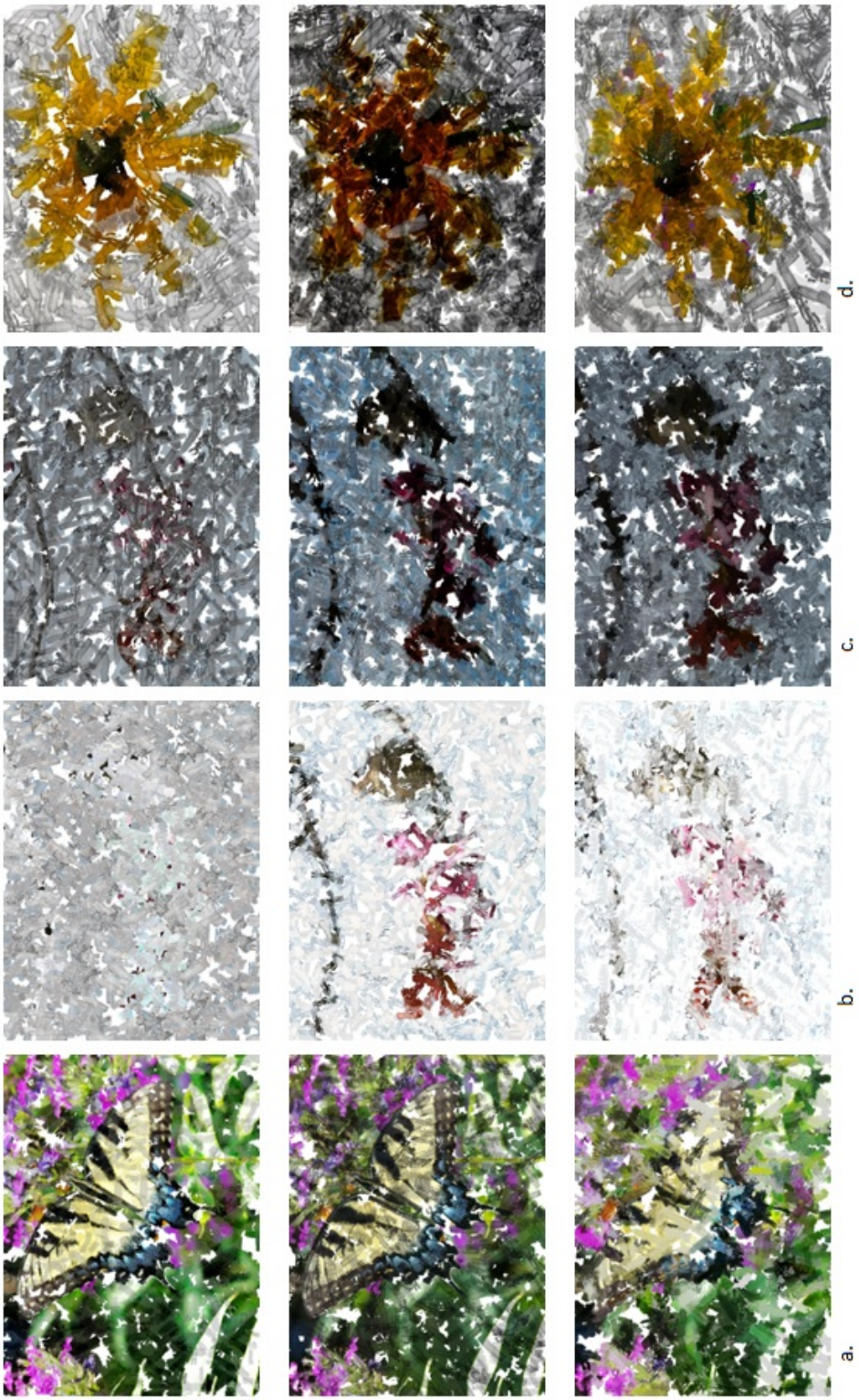


Figure A.20: The best of the last generation of three runs using random pixel selection method.

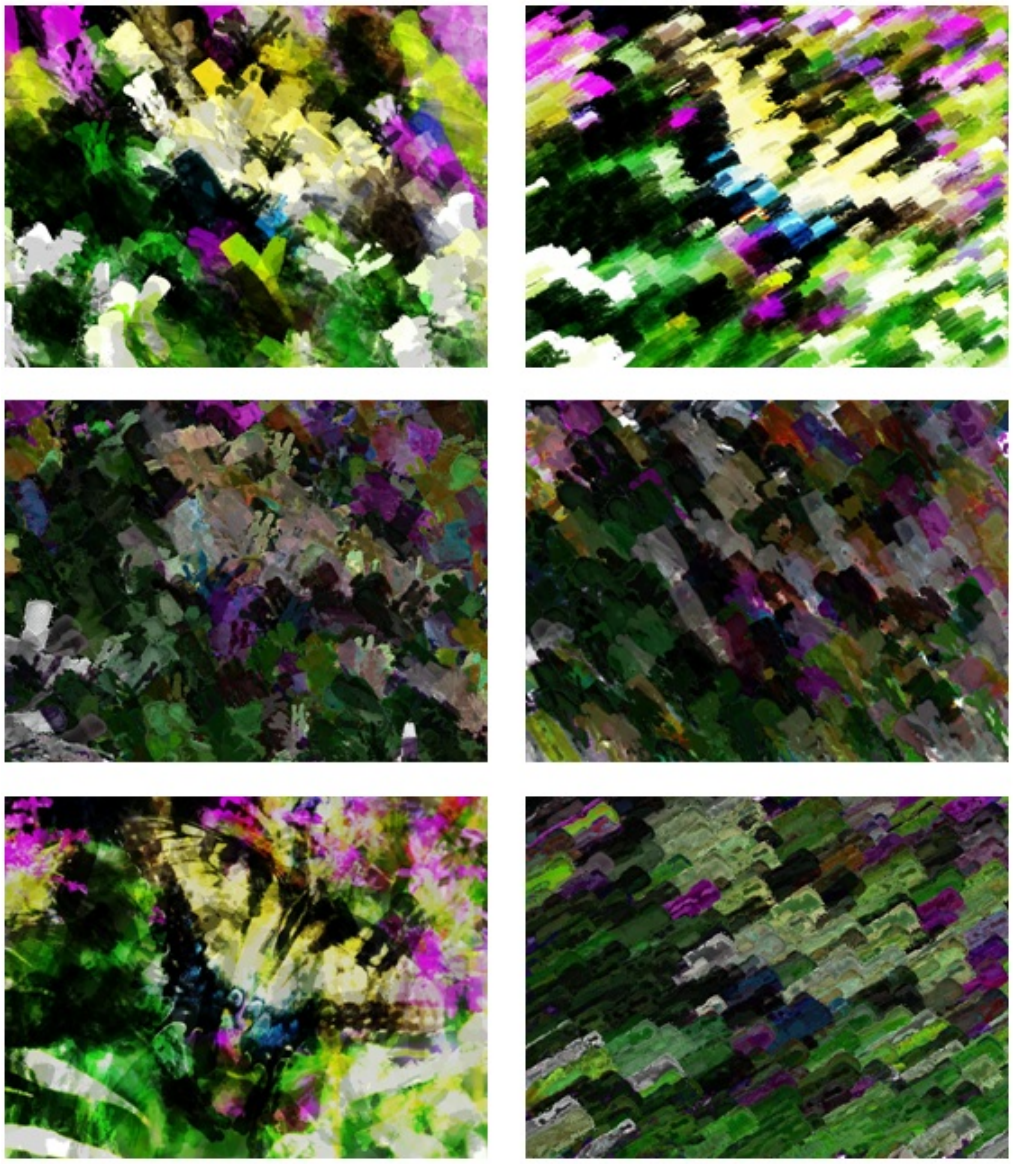


Figure A.21: First sample: The best of the last generation of three runs using white canvas and source image.

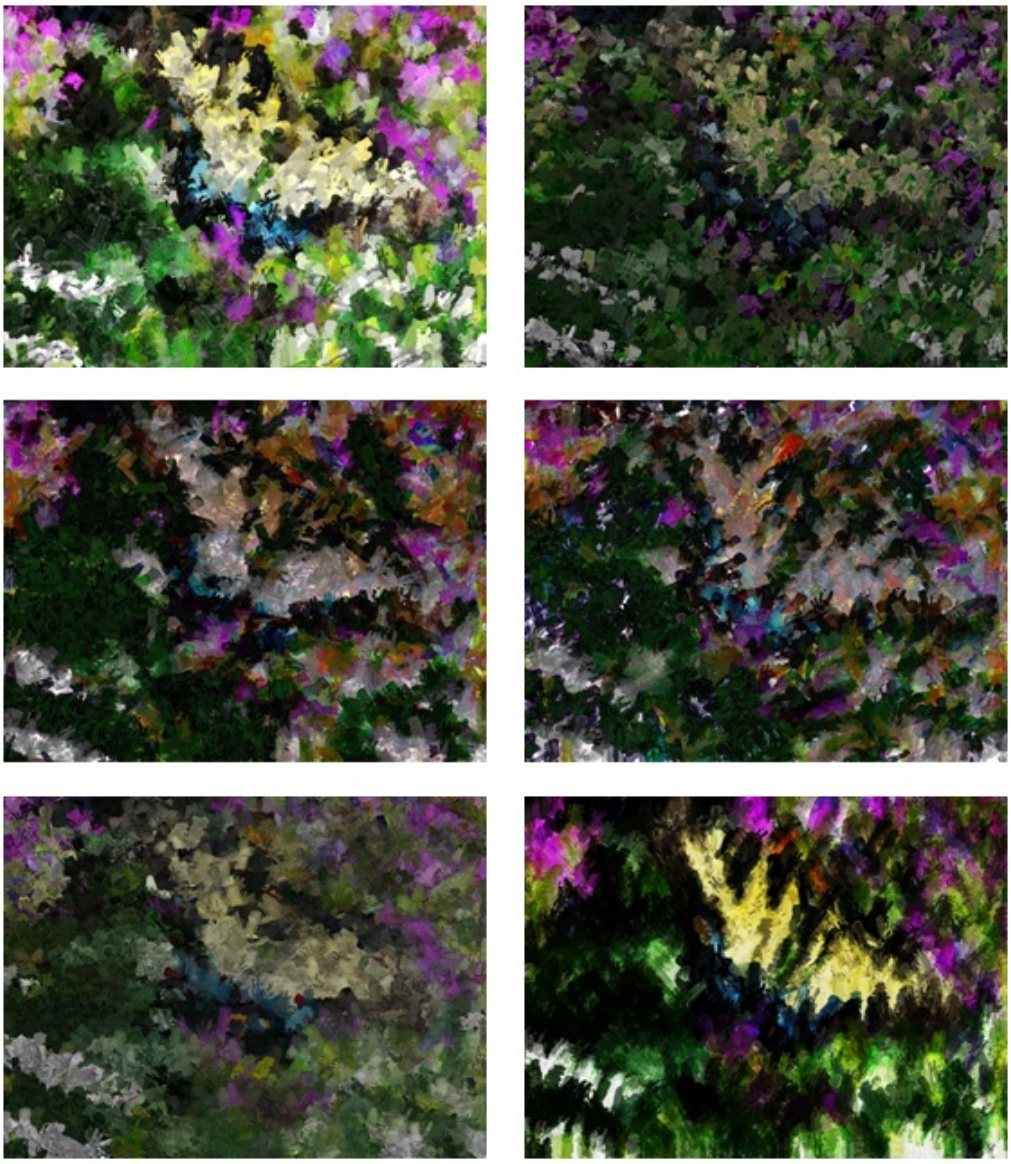


Figure A.22: Second sample: The best of the last generation of three runs using white canvas and source image.

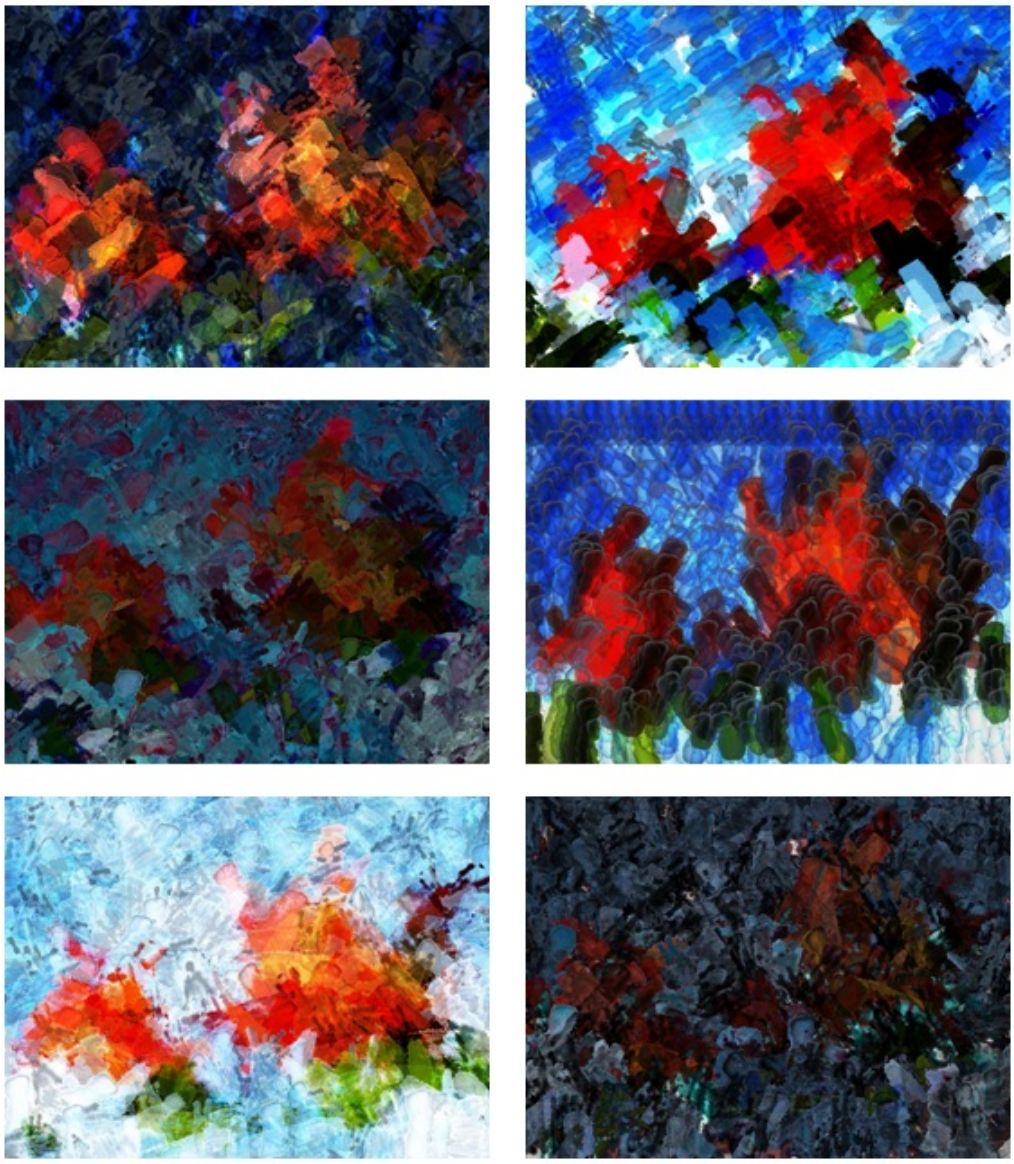


Figure A.23: Third sample: The best of the last generation of three runs using white canvas and source image.

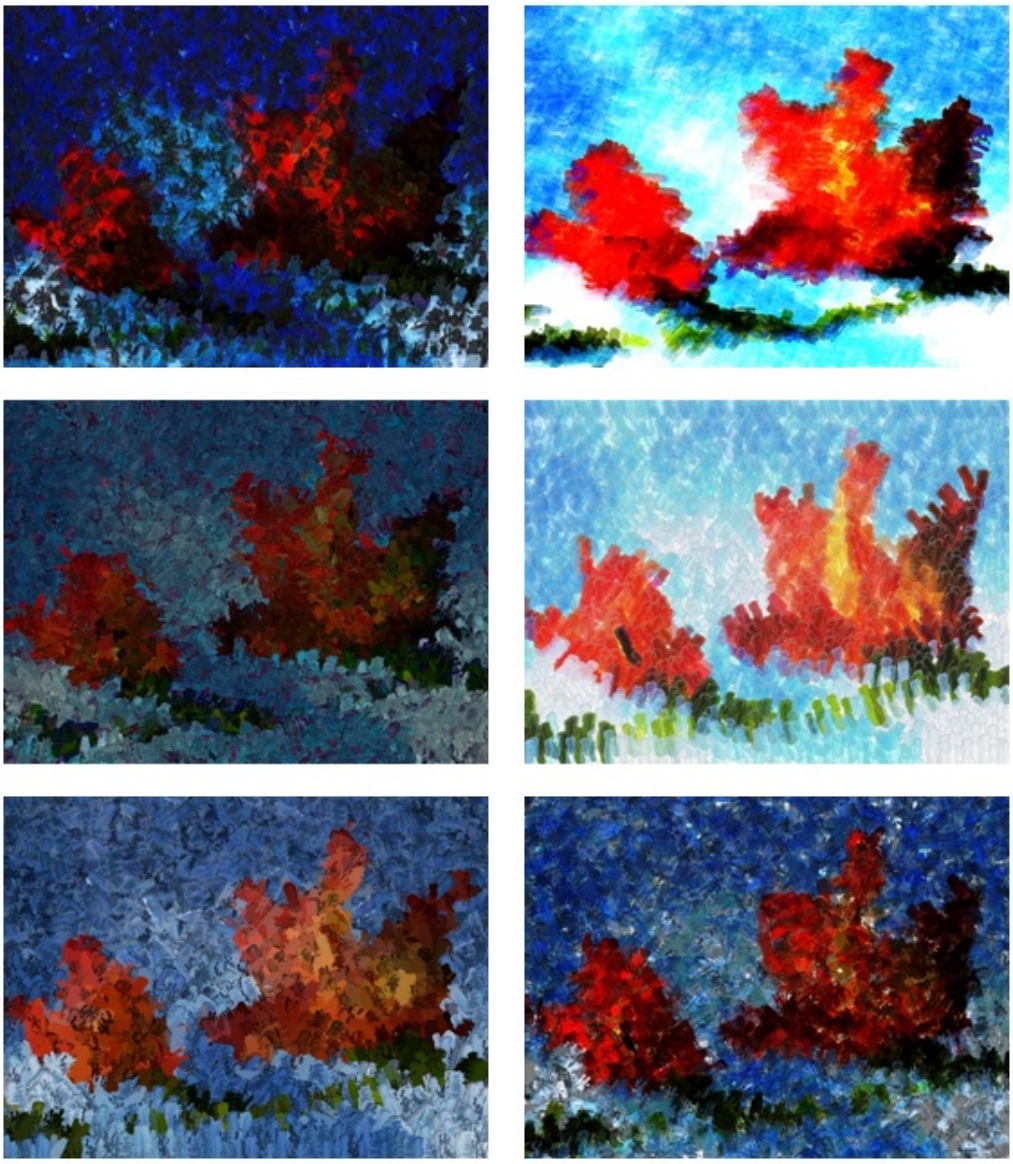


Figure A.24: Fourth sample: The best of the last generation of three runs using white canvas and source image.

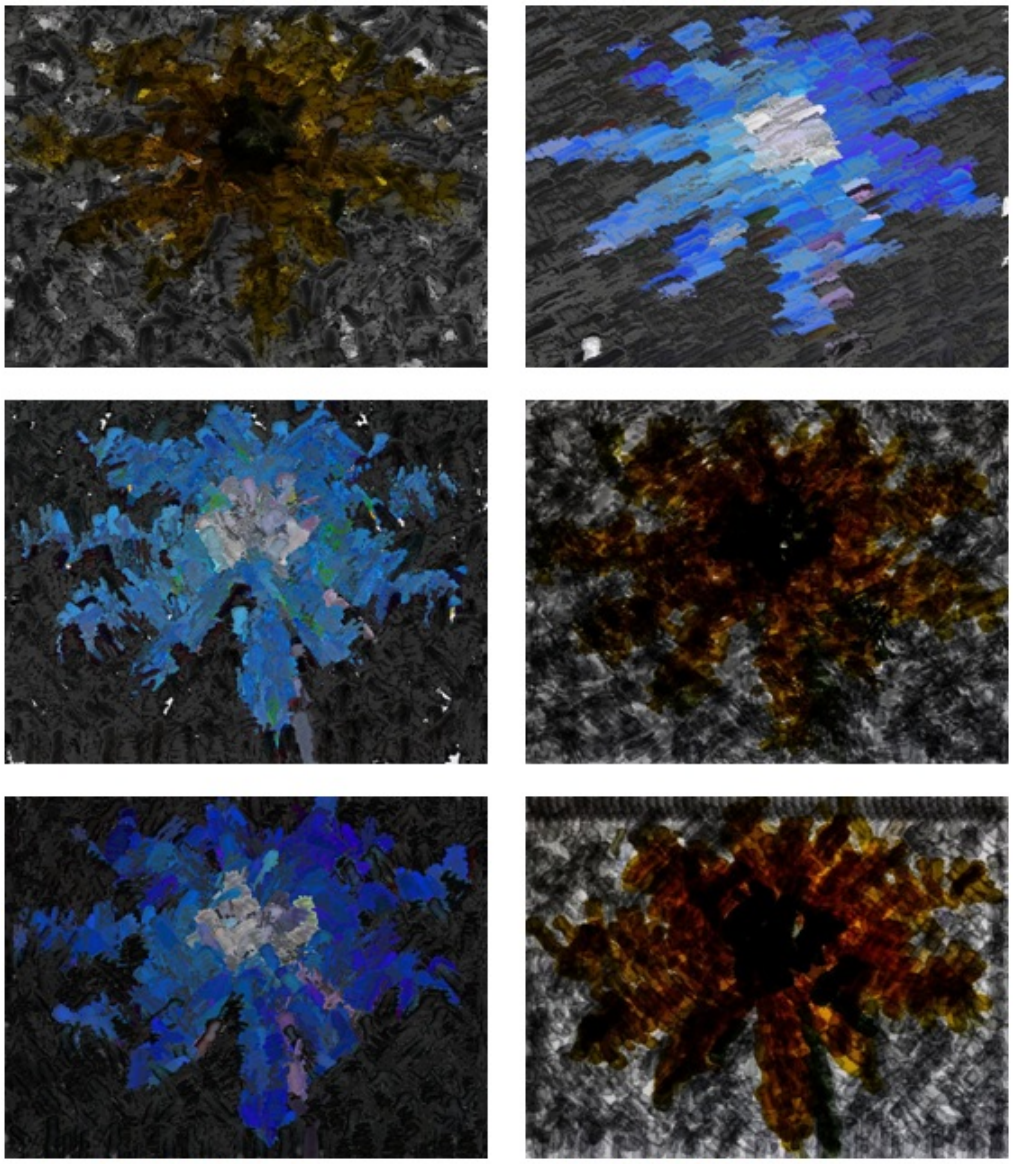


Figure A.25: Fifth sample: The best of the last generation of three runs using white canvas and source image.

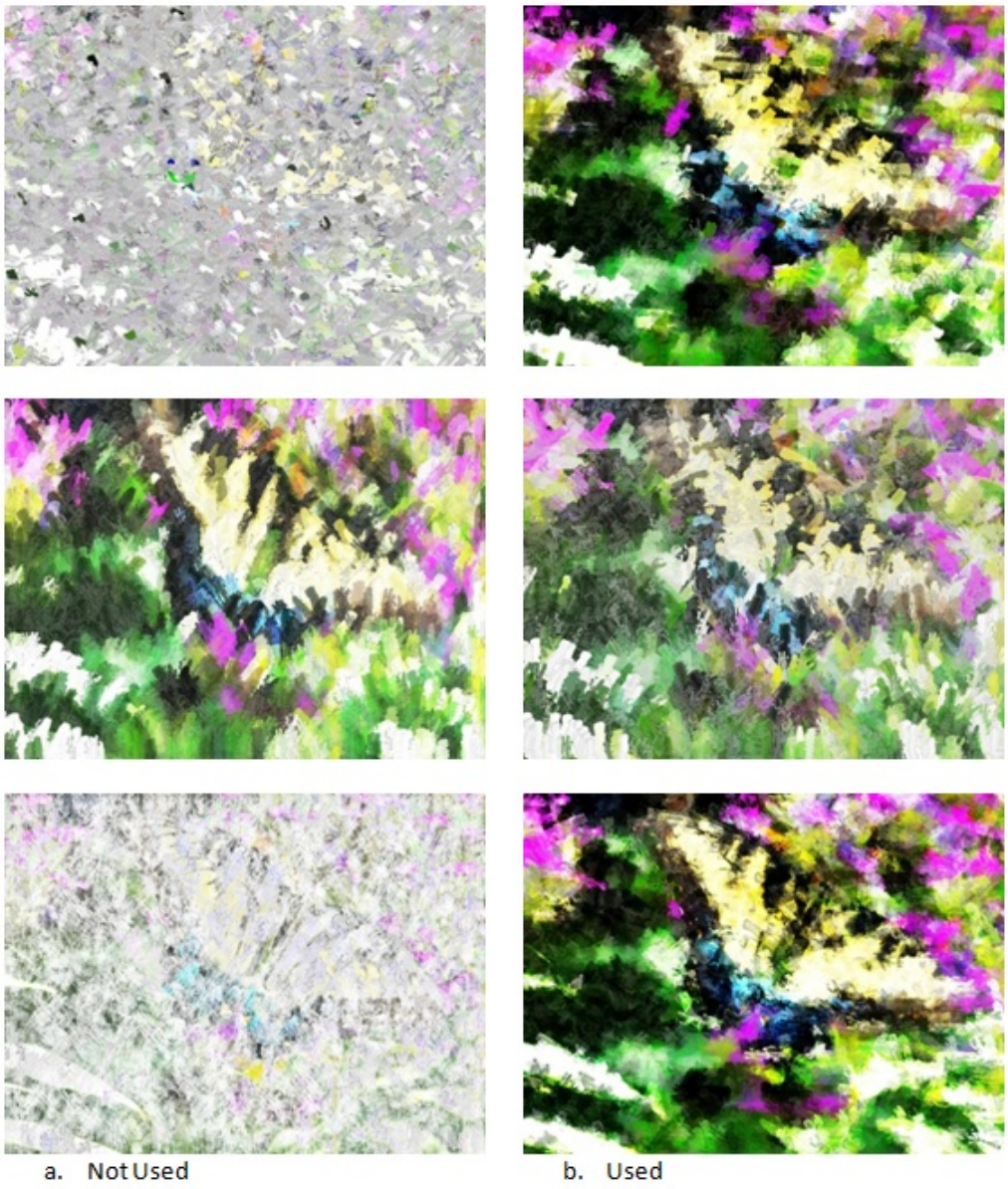


Figure A.26: First sample: The best of the last generation of three runs using luminance direct match.

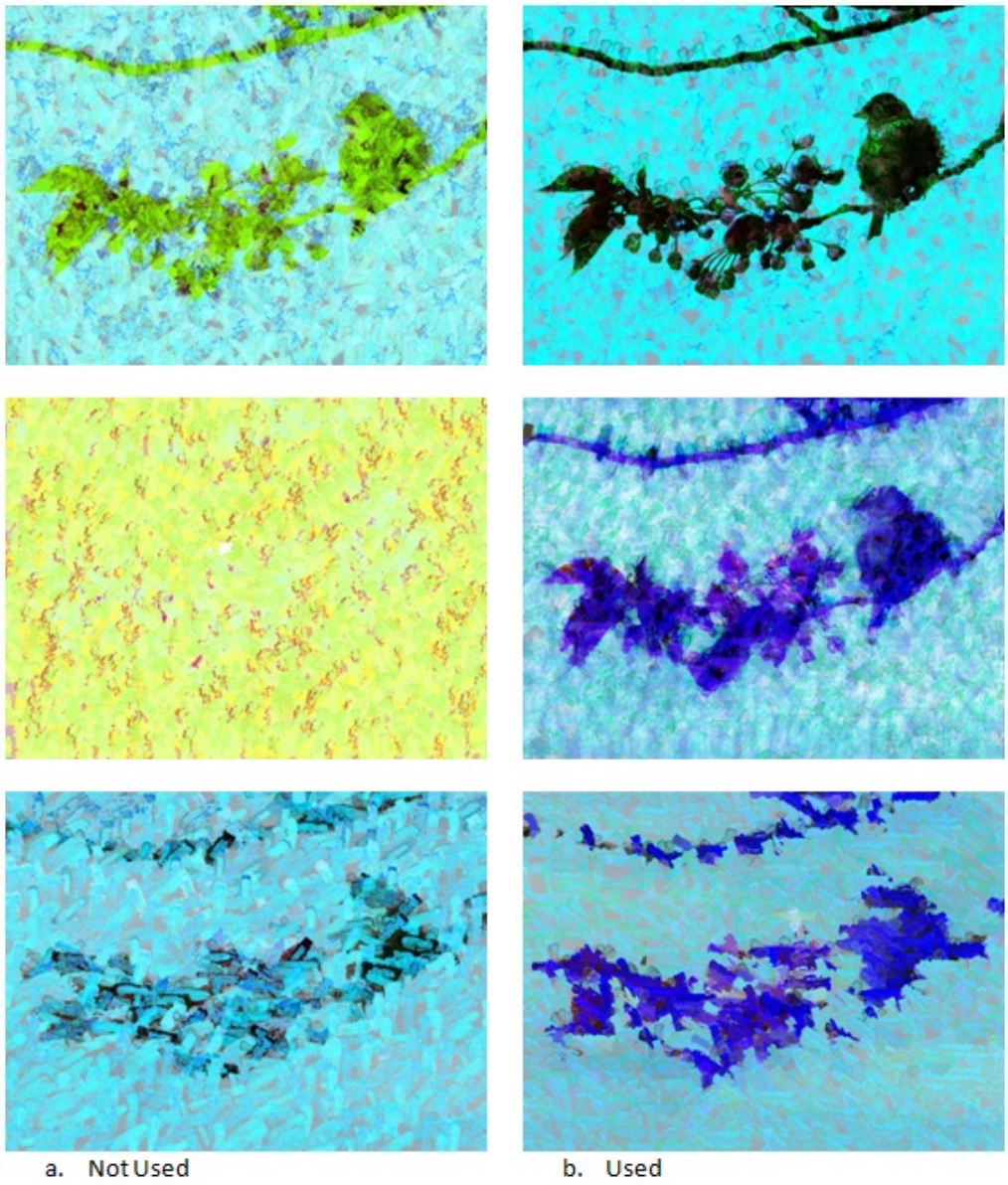


Figure A.27: Second sample: The best of the last generation of three runs using luminance direct match.

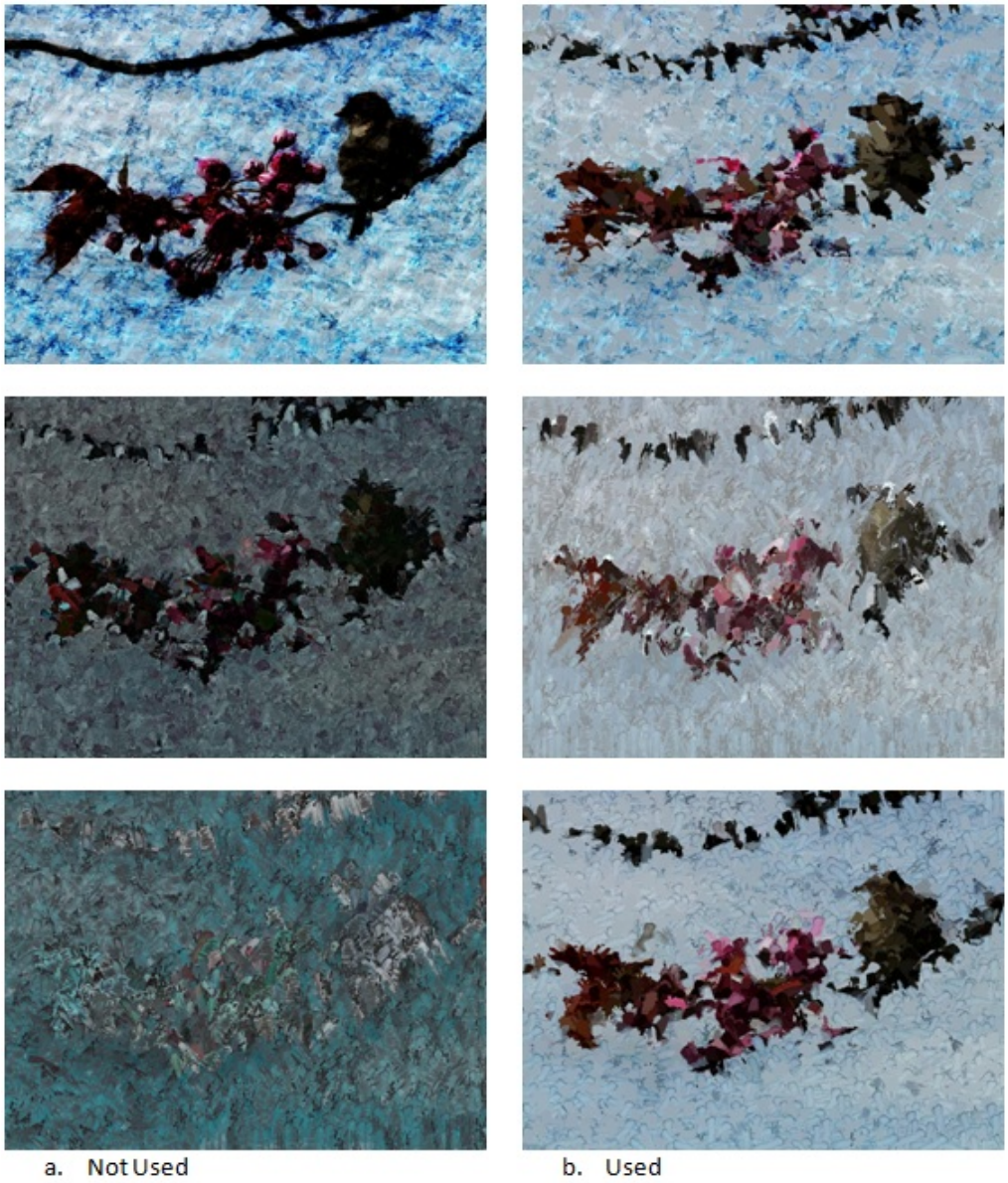


Figure A.28: Third sample: The best of the last generation of three runs using luminance direct match.

Figure A.29: Fourth sample: The best of the last generation of three runs using luminance direct match.

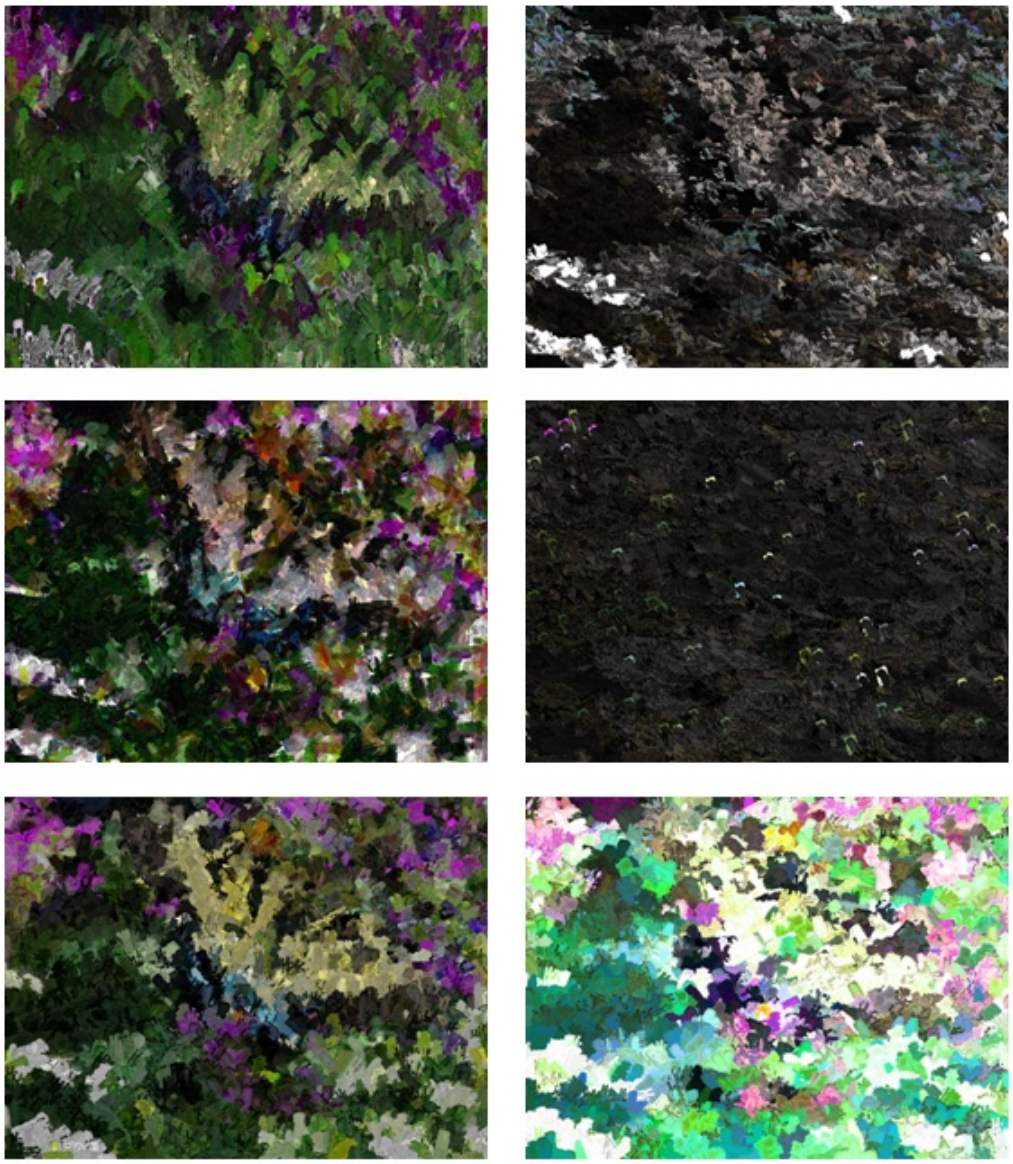


Figure A.30: First sample: The best of the last generation of three runs using RGB and HSV modes.

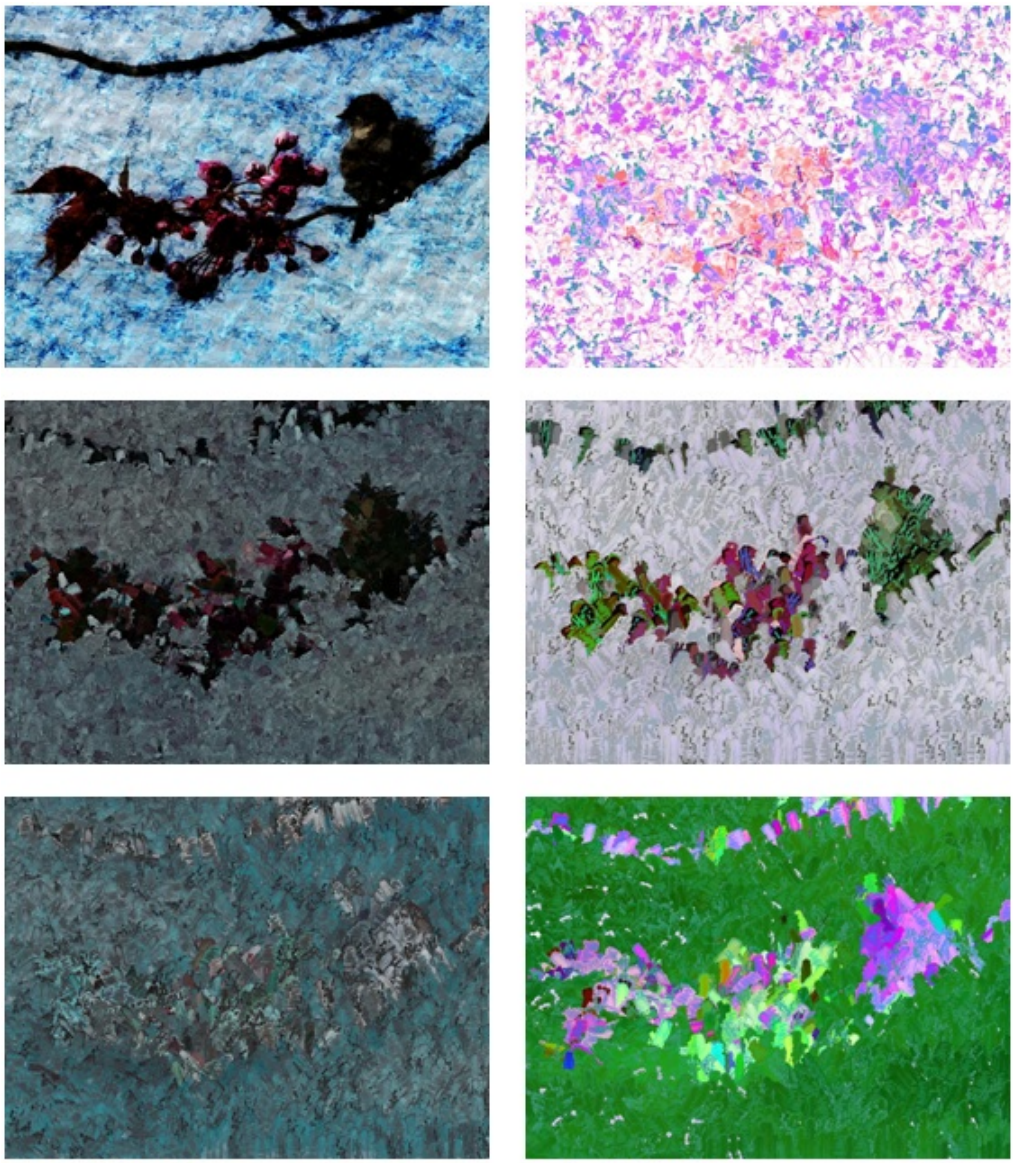


Figure A.31: Second sample: The best of the last generation of three runs using RGB and HSV modes.

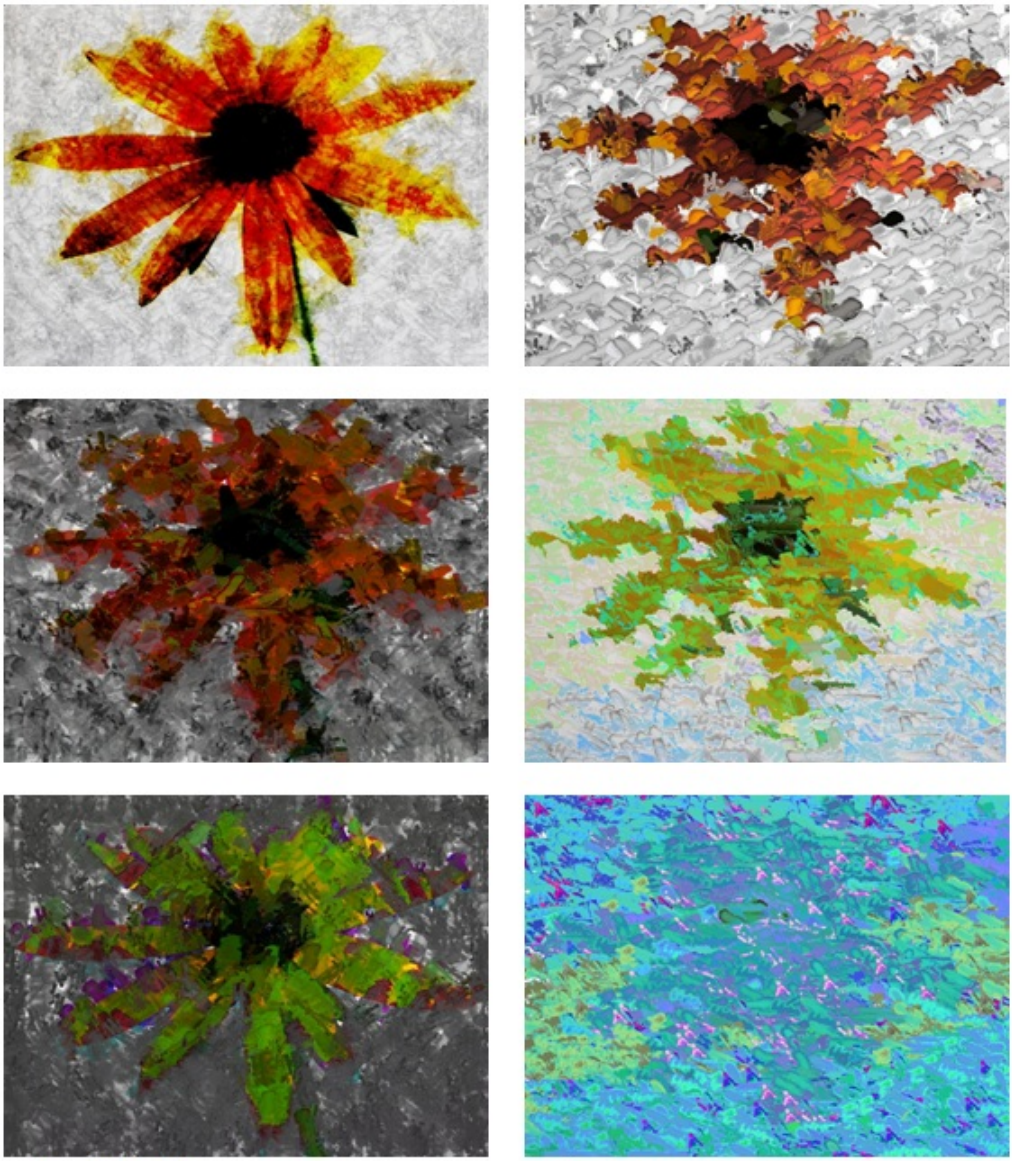


Figure A.32: Third sample: The best of the last generation of three runs using RGB and HSV modes.

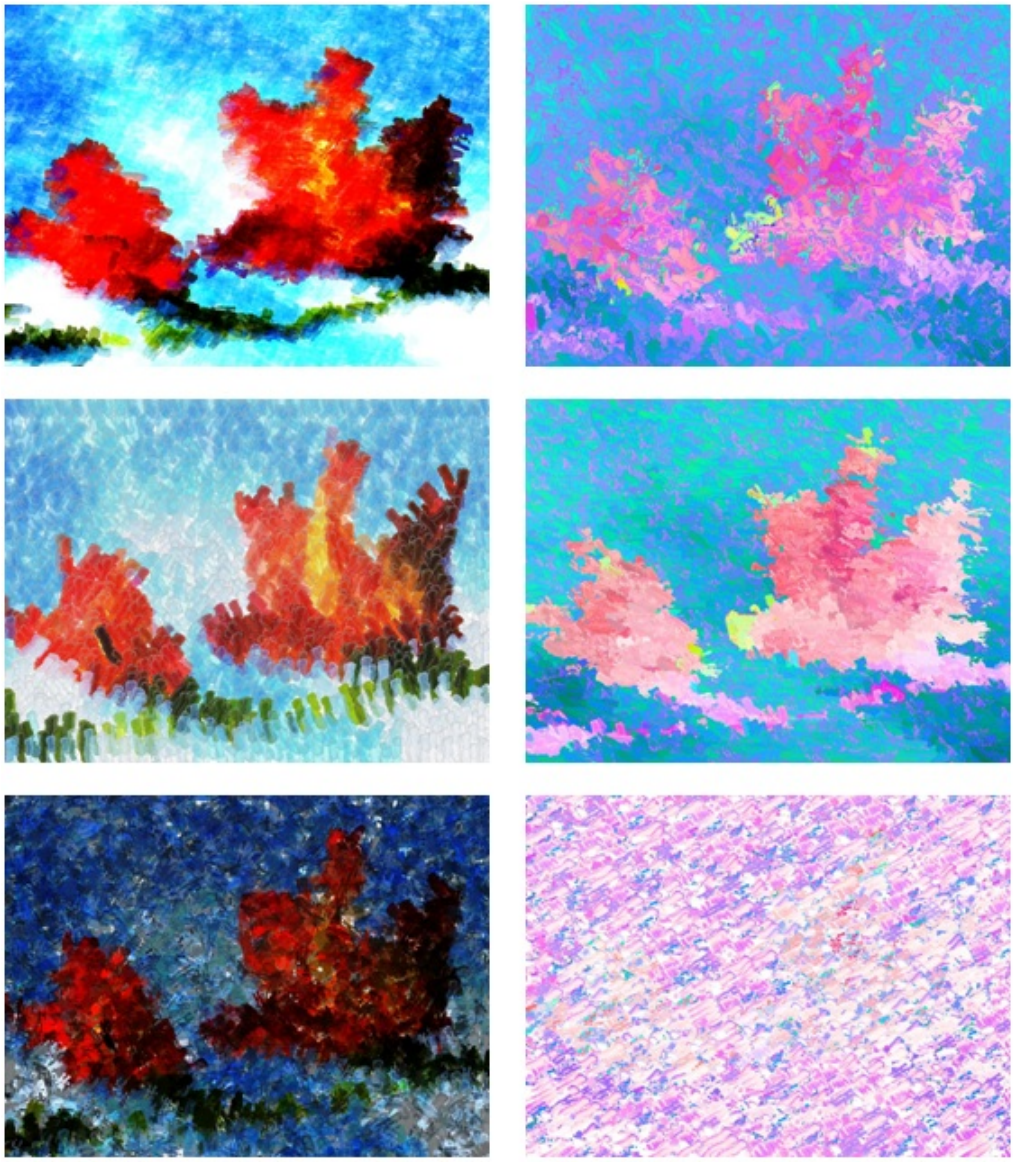


Figure A.33: Fourth sample: The best of the last generation of three runs using RGB and HSV modes.

```
(P4 (VtoF_R CANVAS_C) (Cos (- (VtoF_B SOURCE_C)
     (Sin (- L Min5)))) (- (Sin (- (Sin (- (Cos
     (- Y X)) Min5)) (Sin L))) (- (Sin (VtoF_G
     (P4 Max5 THETA Mean5 SOURCE_C))) Std5)) (MixOBld
     (MixSin (MixAvg CANVAS_C CANVAS_C Bm)) (MixCos
     (MixDBld SOURCE_C CANVAS_C Bm))))
```

Figure A.34: The tree expression with the resulting image

```
(P_RGB Median5 Mean9 (% Median9 Std5) (MixAvg
     CANVAS_C (MixAvg SOURCE_C (Mix+ (Mix- (MixAvg
     SOURCE_C CANVAS_C Bm) (MixAvg CANVAS_C (MixAvg
     CANVAS_C (Mix/ (Mix+ CANVAS_C CANVAS_C (Mixabs
     Bm)) (Mix+ (Mix* CANVAS_C SOURCE_C Bm) (Mix/
     CANVAS_C CANVAS_C Bm) (Mixabs Bm)) Bm) (Mixabs
     Bm)) (Mixabs Bm)) (Mixabs (Mixabs Bm))) CANVAS_C
     (Mixabs Bm)) Bm) Bm) SOURCE_C CANVAS_C)
```

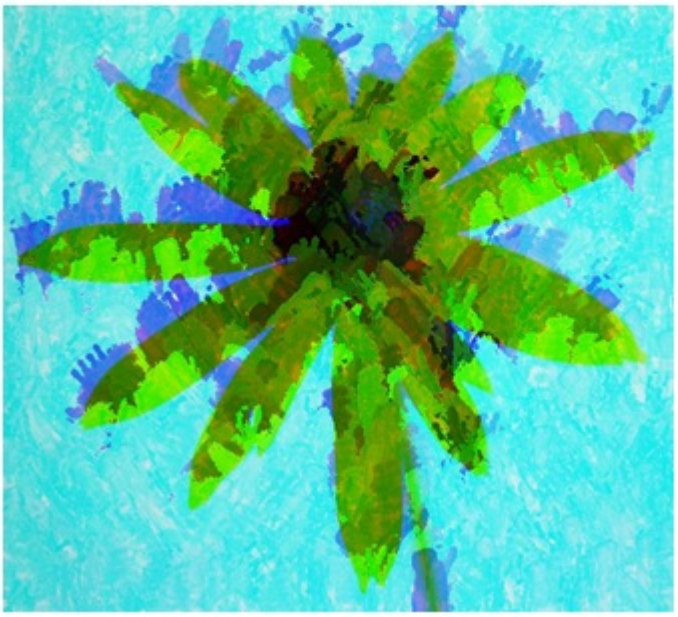


Figure A.35: The tree expression with the resulting image