Contents

Chapter 1

Introduction

This report describes the development of an application that blends images in the Fourier spectrum by using various image processing techniques. Evolutionary computing is used to find good paramaters for filters that produce the best output images. The project falls under two main categories in Computer Science namely Image processing and Computational Creativity. This introductory chapter describes some of the factors that inspired the project as well as stating the main objectives and report layout.

1.1 Motivation

Computational creativity is a subfield of Artificial Intelligence (AI) and is a relatively new and exciting research area with many different branches. Current research spans from the creation of art and music to analogical reasoning, linguistics and even joke generation. Evolutionary computation (EC) is a method that can be used to evolve aesthetically pleasing images, this approach is referred to as evolutionary art. The evolved images can then be assessed by a human observer or by the system itself. This type of algorithm can often explore areas of the solution space that would not necessarily be searched by other means. Computer programs can search this space much faster than humans and by doing so they have the ability to find unique and compelling results.

Digital image processing is a very useful topic of study in it's own right with practical applications in many areas of research and industry. There are many professional image editing packages available, such as Adobe Photoshop. These programs enable users to blend images in a variety of ways using different blending modes and superimposition. However, creating new images by combining select portions of existing images has been done, either manually or with the help of Photoshop-like tools. To our knowledge, nobody has explored the use of Fourier representations as a basis for creating new "blended" images. Blending images using their Fourier representation differs significantly from the "cut and paste" types of image mixing, as Fourier representations do not represent pixels uniquely, so combining different regions of Fourier representations should (in theory) generate images that are significantly different from their "cut and paste" counter-parts.

1.2 Project Objectives

The general objective of the project was to create an evolutionary algorithm that tests the hypothesis of whether a computer program can be used to create new and interesting images. This general objective can be broken down into several more specific goals as follows:

- To explore several methods for blending images using different processing techniques and transforms.
- Become more familiar with some of these techniques through experimentation in order to make confident decisions in the design stage of the system.
- Design and implement an algorithm that evolves digitally blended images through the use of filters in the Fourier domain with the ambition to create new and interesting art.
- Devise and apply several fitness functions that have been designed to judge the aesthetic quality of the resulting images.

- Test the system by comparing the ratings given by humans of the images produced to that of the Systems evaluation.
- Analyse and reflect upon the results of the experiment to aid future research and future ideas for the generation of digital art.

1.3 Method

Due to the experimental nature of the project the application needed to be built using a modular structure in order to incorporate predicted future adjustments. The addition and modification of new and existing filters and fitness functions would need to be incorporated as seamlessly as possible.

The algorithm is implemented using Octave an open source high-level interpreted language which is quite similar to Matlab. Octave provides a powerful environment for image processing due in part to the matrix being the fundamental data type.

1.4 Report Layout

The report has the following main chapters:[THIS NEEDS TO BE FIXED]
Image Blending Concepts - Covers preliminary concepts regarding image processing and evolutionary algorithms.
The Image Blender Application - Details the design and implementation of the system from initial planning to the final program.
Results - Outputs are discussed in detail, the effect of different inputs are examined
Conclusions - Reflection and future work is discussed.

Chapter 2

Image Blending Concepts

In this chapter some fundamental concepts involved with image processing, Fourier Transforms and Evolutionary algorithms are explained. Additionally a review of the literature pertinent to the area of Evolutionary Art and Computational Creativity is performed, introducing some relevant categories in these areas.

2.1 Image Processing

Image processing is a form of signal processing where the input and output signals are images. The same standard one-dimensional signal processing techniques can be applied to images by simply treating the image as a two-dimensional signal [1] which can be represented by a two dimensional array or matrix.

2.2 Fourier Transform

The Fourier Transform (FT), named after the mathematician Joseph Fourier, is a process that transforms a signal in the time or spatial domain to a signal in the frequency domain. The frequency domain describes the signal in terms of sinusoidal components. All Fourier

analysis techiques try to identify the sinusoids that make up the given signal. The FT, like many other types of transforms, is used to rearrange a signal to make it easier to process [2]. The FT for a continuous signal is expressed with Equation 2.1 where $F(\alpha)$ denotes the resulting FT of the signal, $\mathcal{F}$ represents the application of the FT and $f(x)$ is the time domain signal.

$$F(\alpha) = \mathcal{F}\{f(x)\} = \int_{-\infty}^{\infty} f(x)e^{-i2\pi\alpha x}\,dx \tag{2.1}$$

The inverse FT transforms a function of frequency into a function of time and differs only in the sign of the exponent of the above equation. Applying the forward FT twice results in the original time domain signal reflected across the origin, it is therefore referred to as its own inverse [3].

To represent a continous signal digitally, the signal needs to be sampled at discrete intervals. Each pixel in a digital image represents a sample of the real world image or scene. The Discrete Fourier Transform (DFT) is the discrete version of the FT and is used for discrete signals. The DFT gives an approximation of the continuous FT and is shown in Equation 2.2.

$$F[n] = \sum_{k=0}^{N-1} f[k]e^{-i2\pi nk/N} \tag{2.2}$$

Calculating the DFT in this way can be slow to compute for large signals, such as images and several people derived faster algorithms, Gauss being the first. The most common fast Fourier Transform (FFT) algorithm used today was described by John Tukey and John Cooley [4].

Filters can be used to attenuate or amplify specific frequencies in the Fourier domain. For example low pass, high pass, bandpass, and linear filters are example filters that can be used to target specific frequencies in a signal. With regards to images, the higher frequencies represent the edge information in the image where the low spatial frequencies represent the non-changing or blurry parts of the image.

2.3 Evolutionary Computation

Evolutionary computation is used to solve problems by mimicking biological evolution. Evolutionary Algorithms (EA) are used to find good solutions to optimisation problems and are particularly useful when the solution space is vast and complex. An EA will usually consist of the following steps:

1. Generate initial population:
   This is the first step and involves creating a random population of potential solutions to the problem.

2. Evaluate Fitness:
   Each individual is assessed by one or more fitness measures. Fitness functions are used to score an individual which can later be used to compete for reproduction.

3. Selection:
   Certain individuals are chosen to parent a new population. There are many different selection algorithms that can be implemented depending on the nature of the problem being solved. Example selection algorithms include rank, tournament and roulette wheel.

4. Reproduction:
   The selected parents are then used to create the next generation of individuals. Like selection there are a number of algorithms which use crossover to obtain a new population.

5. Mutation:
   Mutation usually involves swapping a gene in an individual with a random or ancestor gene. Mutation is applied using a certain rate of probability. The rate is usually very small but provides vital diversity to the algorithm and helps to stop convergence to local extrema [5].

2.4 Literature Review

Early pioneers in evolutionary art included Karl Sims and William Latham. Their work, using concepts derived by Richard Dawkins, created artistic images using evolutionary computation [6, 7]. Sims evolved symbolic expressions to create images that can be used in computer graphics and animation, an example of his work can be seen in Figure 2.1. Latham, who worked with Stephen Todd, created programs such as ESME and "Form Grow" to create artistic virtual sculptures, an example of which is shown in Figure 2.2.

Figure 2.1: Image by Karl Sims

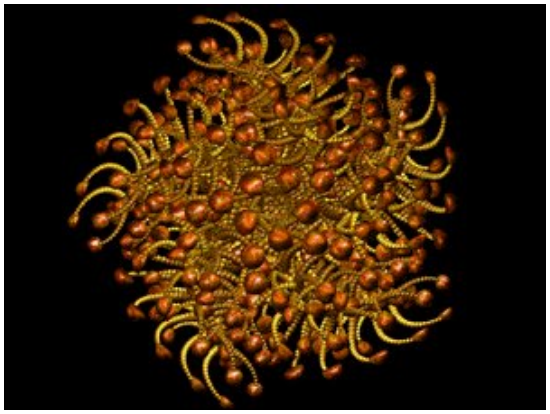

Figure 2.2: Virtual Sculpture - Latham

Like Latham and Sims, many evolutionary art systems rely on human interaction to assess fitness. However there is ongoing research to design automated systems to assess aesthetic value. Forsythe and collaborators discuss how visual complexity contributes to perceived beauty in art in [8]. Their study showed a correlation between Gif compression and human judgments of complexity using a wide range of artistic imagery. Their research also examined the use of the fractal dimension as a method for measuring beauty. Taylor [9] claimed that the famous painter Jackson Pollock's paintings had specific fractal qualities, arguing Pollock was able to adjust the fractal dimension value of his paintings. Simon Colton is an active researcher in the field of computational creativity and is the creator of "The Painting Fool". Figure 2.3 shows a portrait created by this artificial artist. Colton's goal with the project [10] is to see whether computer software can be accepted as being creative. Colton's fully autonomous program can be self critical, obtain

its own images [11] and can even read news articles to obtain a "mood". DARCI [12] is another digital artist program which has been invented by a team led by Dan Ventura. A website allows humans to label images from DARCI's database with adjectives. These adjectives are then used to form a description to render provided images with a variety of image processing filters. Figure 2.4 shows a peaceful render of an image given to DARCI as input which is described in [13].

Figure 2.3: Portrait by The Painting Fool

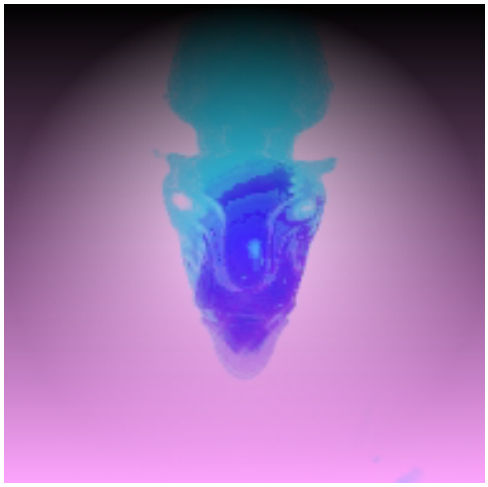

Figure 2.4: Image by DARCI

It is not just software that is being developed by researchers in this ar, over the years machines have also been developed to physically paint pictures that could arguably match human artistic ability. A robot called e-David [14] has been created in Germany to mimic how a human paints. A painting by e-David can be seen in Figure 2.5. In the early 1970's, Harold Cohen invented the robotic artist AARON, who's paintings captivated visitors to the Computer Museum in Boston [15].

Figure 2.5: Painting by e-David

A review of the literature would not be complete without discussing the work of Margaret Boden who addresses the conceptual problems with the definition of creativity. In [16], Boden states there are three forms of creativity: exploratory, combinational and transformational. Combinational creativity produces new ideas from existing ones, our analogical reasoning ability would be an example of this. Conceptual spaces, first described by Peter Gärdenfors [17], are structured styles of thought, ideas and memories. New ideas from within this thinking style are creative in the exploratory sense. Transformational creativity gives rise to new, rare ideas that do not conform to this established way of thinking and can produce far more radical and suprising results. AI approaches can be used to help us learn more about human thought processes and creativity.

Chapter 3

The Image Blender Application

3.1 Planning

[HOUSEKEEPING STUFF GOES HERE, PROJECT GANTT ETC. SOME POINTS TO COVER ARE LISTED BELOW]

1. Gather Images, prepare images (crop and scale to 256 and 1024 batch image processer used - Phatch)
2. Study Signal processing concepts and Fourier Transforms
3. Experiment using different filters and transforms
4. Meetings with different academics Tom Naughton, Charles Markham,

3.2 Preparation

[THIS SECTION COVERS THE DIFFERENT EXPERIMENTS DONE E.G. FIRST BLEND, MAG/PHASE BLEND, DIFFERENT FILTERS, ROTATIONS, BLENDS USING BLENDED IMAGES ETC.] Before approaching the design phase for the EA, experiments needed to be carried out to decide which image processing techniques would be used to blend the images. Most of the experiments were done using the Fourier transform,

however others were also considered, including the Hough, Z and Radon transforms. The steps carried out for the first attempt to blend two images were as follows:

1. FA = Fourier Transform of Image A
2. FB = Fourier Transform of Image B
3. FC = Left FA + Right FB
4. Blended Image = Inverse Fourier Transform of FC

3.1 shows these steps

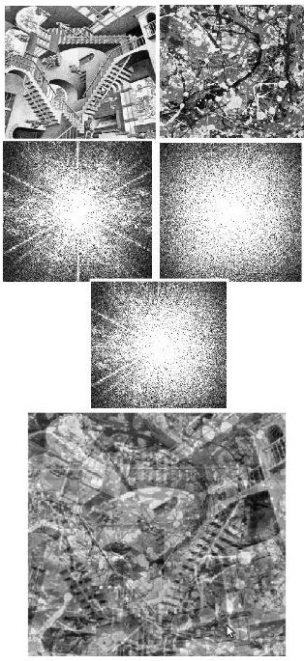

Figure 3.1: The First Blend

3.3 Design

Before implementing an EA it is necessary to consider how the population of candidate solutions will be represented and also how each individual will be assessed. This can often lead to the most difficult design decisions when it comes to EA's and was no exception

in this case. In particular it was decided to automate the fitness functions, meaning the program would need to assess image aesthetic quality which is a known difficulty for evolutionary art projects [18].

3.3.1 Representation

Choosing the correct representation for the genome is an integral part in the design of an EA. This was not a straightforward process due to the various filters, which characterise a gene, having a different number of parameters. The most common representation for a genome or chromosome in an EA is a series of bit values. Each bit or series of bits represents a property of the proposed solution. Figure 3.2 shows a sample chromosome using bit representation.

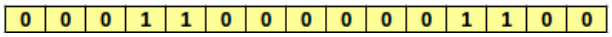

Figure 3.2: Common Representation

However this type of representation, for this project, would have fuelled unnecessary constraint violations during the crossover process; therefore a different design was implemented. For each filter the filter type and the parameters are needed. Each filter type is denoted by a letter and depending on the letter is followed by 1 or more numerical parameters. Each filter is stored in a cell and is therefore intact during crossover and mutation. For example Figure 3.3 depicts a simple chromosome where the first gene is a bandpass filter that takes two parameters (2 radii).

Figure 3.3: Example Representation

3.3.2 The Fitness Functions

From an early stage in the project it was decided to automate the process of determining the fitness of an image. Other evolutionary art projects [19, 20] use target images to evaluate fitness. This approach relies on knowing the desired result, which is not always possible. Dawkins describes real evolution as not having the ability to foresee the future [21] thus a different evaluation method was investigated. However, target metrics were used for some of the fitness evaluation. The objective of the EA is then to minimise these fitness functions.

There were two fitness functions introduced in the early stages of development which measured the jpeg and gif compressed size of the newly created image and assessed fitness by comparing the resulting file size to a predefined metric. The metric was obtained by calculating the mean file size of a variety of well known paintings after the images had been processed and compressed in the same manner as the evolved images. As the project advanced, three more fitness functions were introduced which measured entropy, edge difference from input image 1 and also input image 2.

Jpeg is a lossy form of compression and uses the discrete cosine transform(DCT) [22] which has similarities to the DFT but will not be discussed in detail in this report. This method was chosen due to it being based on what the human visual system can perceive, the higher frequencies (edge information) in the image are lost during compression, in a photograph or complex image this usually goes unnoticed by the observer. The Jpeg fitness function uses the following equation to score the individual where x is the jpeg file size of the image being assessed.

$$y = (abs(target - x)/1000)^2 \tag{3.1}$$

In contrast Gif is a lossless form of compression. How well the method compresses an image is based on the number of repetitions that are present in the image. Each pixel is comprised of 8 bits which limits the number of colours available to a maximum of 256. However, Gif's retain edge sharpness which makes this form of compression a suitable choice for images containing text, simple shapes with sharp edges and big blocks of solid

colour [23]. Due to to these characteristics a decent judgement can be made from the file size after compression as to the type of image in question. A similar function to the Jpeg method is used to obtain a score for the Gif fitness function but allows slightly more variation.

$$y = (abs(target - x)/2000)^2 \tag{3.2}$$

3.3.3 Pareto Optimality

The evolutionary algorithm implemented for this project has multiple objectives meaning the fitness of an image is based on more than one attribute. For multi-objective optimisation (also known as pareto optimisation) it is good practice to keep a set of fittest individuals rather than simply a single best individual. The pareto front comprises of individuals who cannot be dominated by any other single individual across each fitness attribute. An algorithm for keeping track of the pareto frontier can be complicated and would usually involve multiple nested iterations. Each individual in the current population needs to be compared with each member of the Pareto frontier for each of the attribute scores which are determined by the fitness functions. To improve on efficiency and to make use of Octave's speed on vector operations, an algorithm was developed that can be described briefly as follows (the code for this function can be found in Appendix A):

1. Loop through current population scores.
2. Check current individual scores against every individual in the pareto front table.
3. Case 1 is trivial: When current individual does not have at least one better score than each entry in the pareto table, the individual is not added and the Pareto tables remains unchanged.
4. Case 2: If the current individual has at least one better score than each of the pareto entries then the individual will be added to the table. If the current individual is better on each of the fitness scores than any entry in the pareto table the individual replaces the entry.

The pseudocode presented in Figure 3.3.3 depicts the algorithm. Note the outer loop is necessary and cannot be vectorised since the pareto table needs to be updated before checking the next individual in the current population[WILL FIX].

```
Input: CurrentPareto, CurrentPopScores
Output: UpdatedPareto
for i=1:popCount do
    Compare CurrentPopScores(i) to CurrentPareto;
    if CurrentPopScores(i) contains dominant score then
        Insert and check for inferior entries;
    end
    UpdatedPareto = update(CurrentPareto);
end
```

Figure 3.4: Function: UpdatePareto()

Figure 3.5 shows a plot of every individual's jpg score (x-axis) versus gif score (y-axis) in a 30 generation run of the program. The Pareto frontier is shown in red. Only 2 fitness functions were used for graphing purposes.

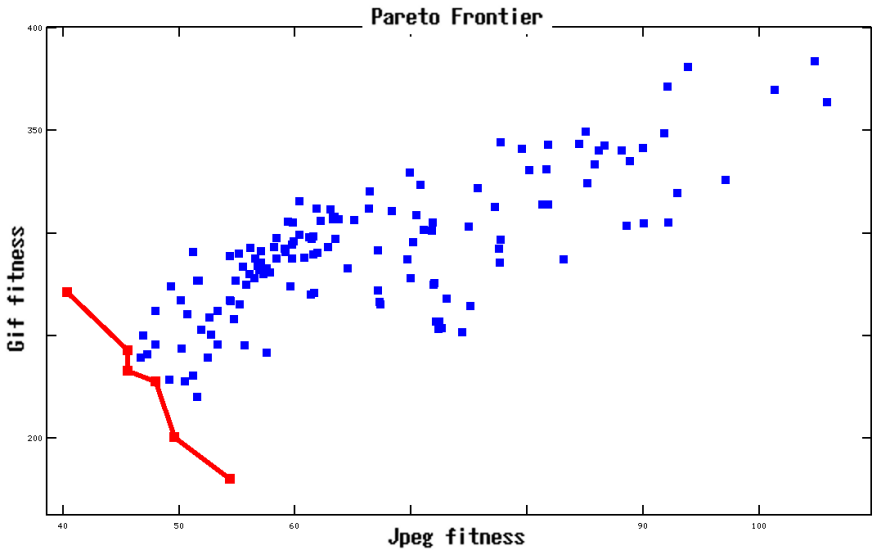

Figure 3.5: Pareto Frontier

3.4 The Algorithm

To start the program parameters are set in the main function that runs the application. These include the images to be blended, the population size and the desired Chromosome length. The application begins by creating a random population of image filters. A matrix representation of the filters are then created by parsing the individual chromosome, each individual is associated with one final binary filter or mask, an example of which is shown in Figure 3.6.

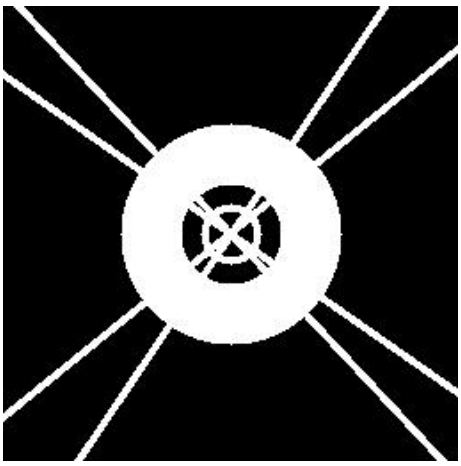

Figure 3.6: Example Binary filter

This mask is then mulitplied by the FT of the first image. The inverse of the mask is applied to the second image. The two filtered FT's are then added together to produce the frequency domain representation for the new image. The inverse FT can then be applied to obtain the image in the spatial domain. Figure 3.7 shows an example of a blended image, with the two original images on either side.

Figure 3.7: Example Blend

The images are then assessed by the fitness functions to obtain a list of scores for each individual. The individuals then compete for entry to the pareto table. [DESCRIBE SELECTION, CROSSOVER, MUTATION.....]

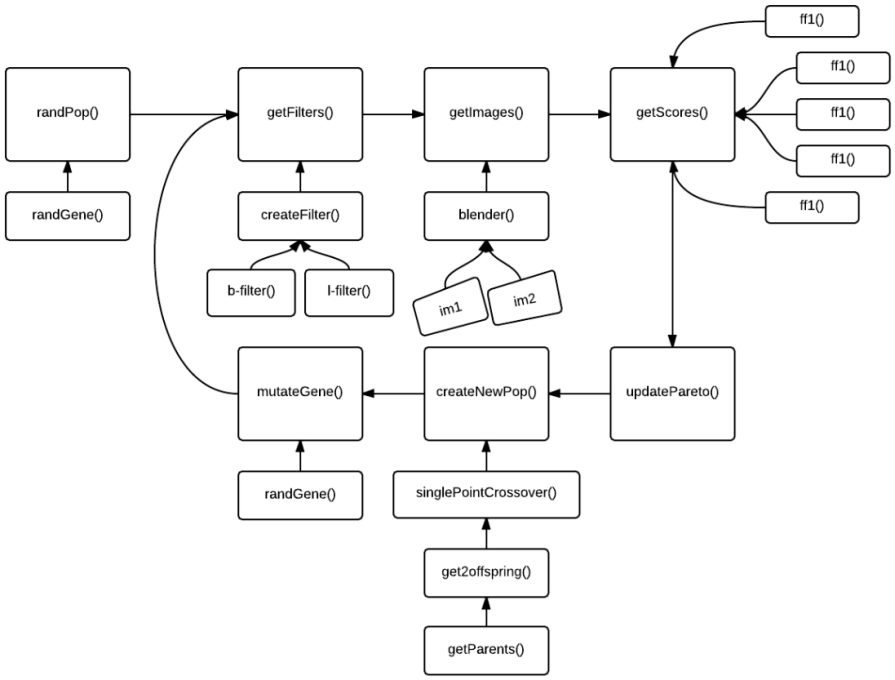

Figure 3.8: Algorithm Flow Chart

Chapter 4

Results

LIBRARY SURVEY RESULTS

BASIC SHAPES, PHOTOGRAPHS AS INPUT, ARTIST PAINTINGS,

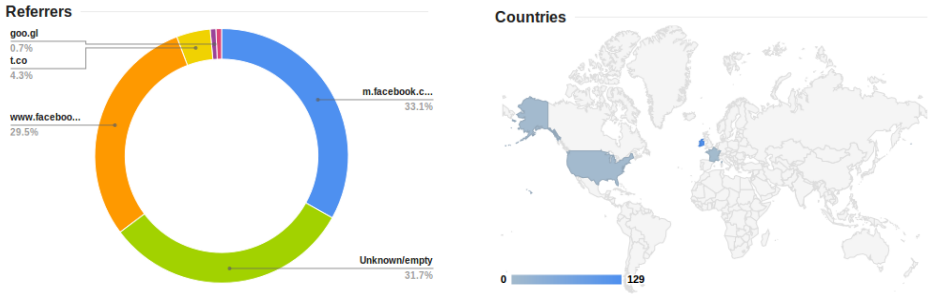

Figure 4.1: Respondent Demographics

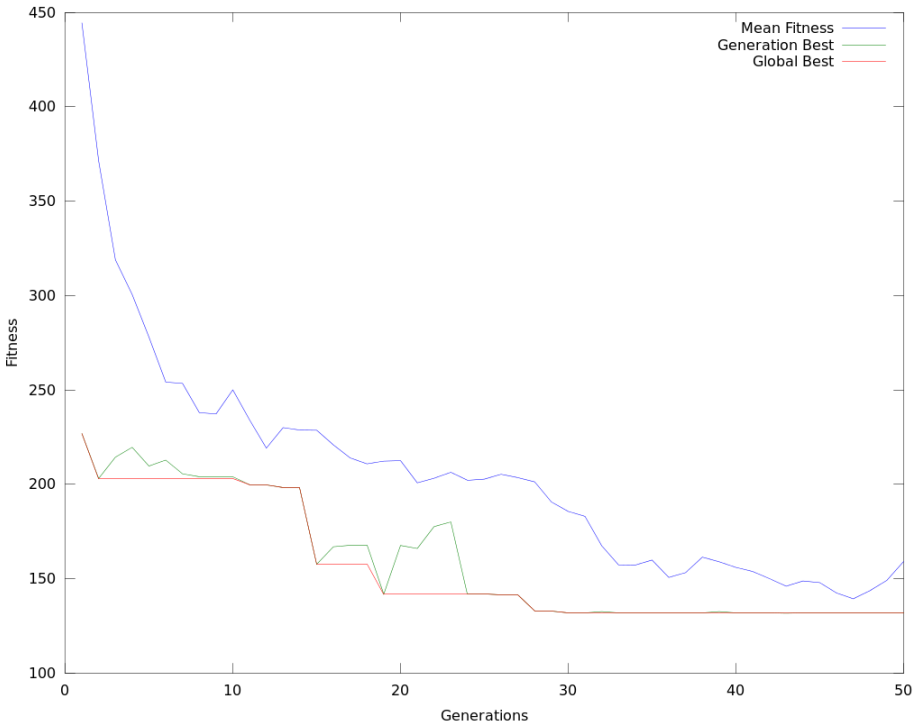

Figure 4.2: Fitness Euclidean-distance

Chapter 5

Conclusions and Future Work

Encouraging response from library exhibition

- Fractal Compression
- Symmetry and golden ratio
- Phase and Magnitude splitting
- More filters, different transforms
- Using fractal dimension to predict beauty

Appendix A

Code Samples

```
function newp = updateParetoTable(currscores,currpareto,chromLen)
epsilon = 1e-8;
fitno = columns(currscores) - chromLen;
newp = currpareto;
for i=1:rows(currscores)

    scores=cell2mat(currscores(i,chromLen+1:end));
    scores = repmat(scores,rows(newp),1);
    pscores=cell2mat(newp(:,chromLen+1:end));
    %vector to store when current pop scores beat current pareto
    currwins = sum(scores < pscores,2);
    checkadd = sum(currwins > 0);
    if checkadd == rows(newp)
        %vector to sum when current scores equal pareto scores
        eqscores = sum((scores .- pscores)<epsilon,2);
        %vector to score when current pareto beats current
        pwins = sum(pscores < scores,2);
        %vectors to see which pareto scores should stay
        goodlist1= (currwins != fitno);
        goodlist2=(pwins != 0);
```

```
        goodlist3=(eqscores != fitno);
        goodlist = goodlist1&goodlist2&goodlist3;
        keeplist = find(goodlist == 1);
        newp = newp(keeplist,:);
        newp = cat(1,newp,currscores(i,:));
    endif


endfor
endfunction
```

Appendix B

Gallery of Blended Images

Figure B.1: Sample Blends using Gaussian filter gene

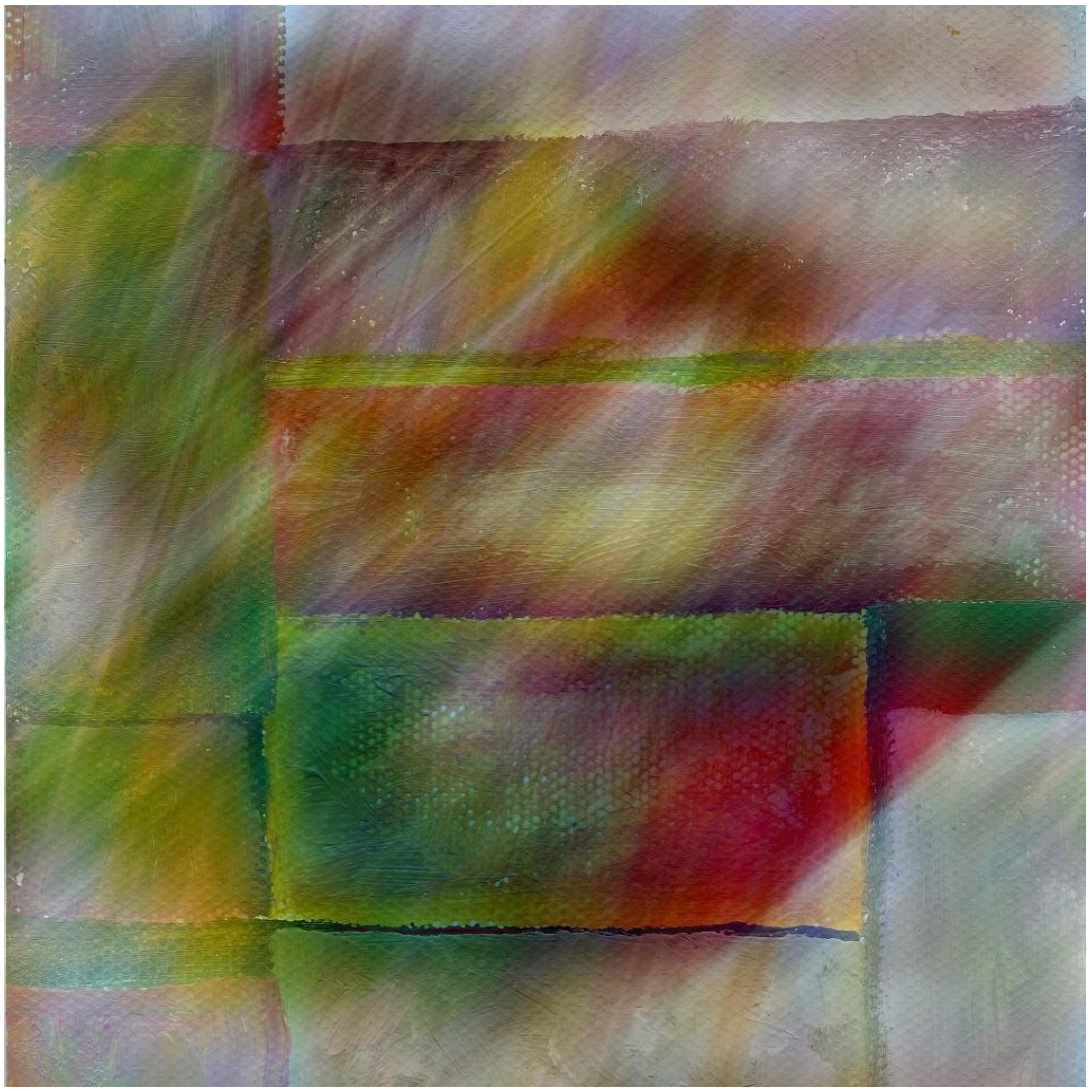

Figure B.2: Image Blender: author's own paintings

Figure B.3: Image Blender: author's own paintings

Figure B.4: Image Blender: Degas and Klee

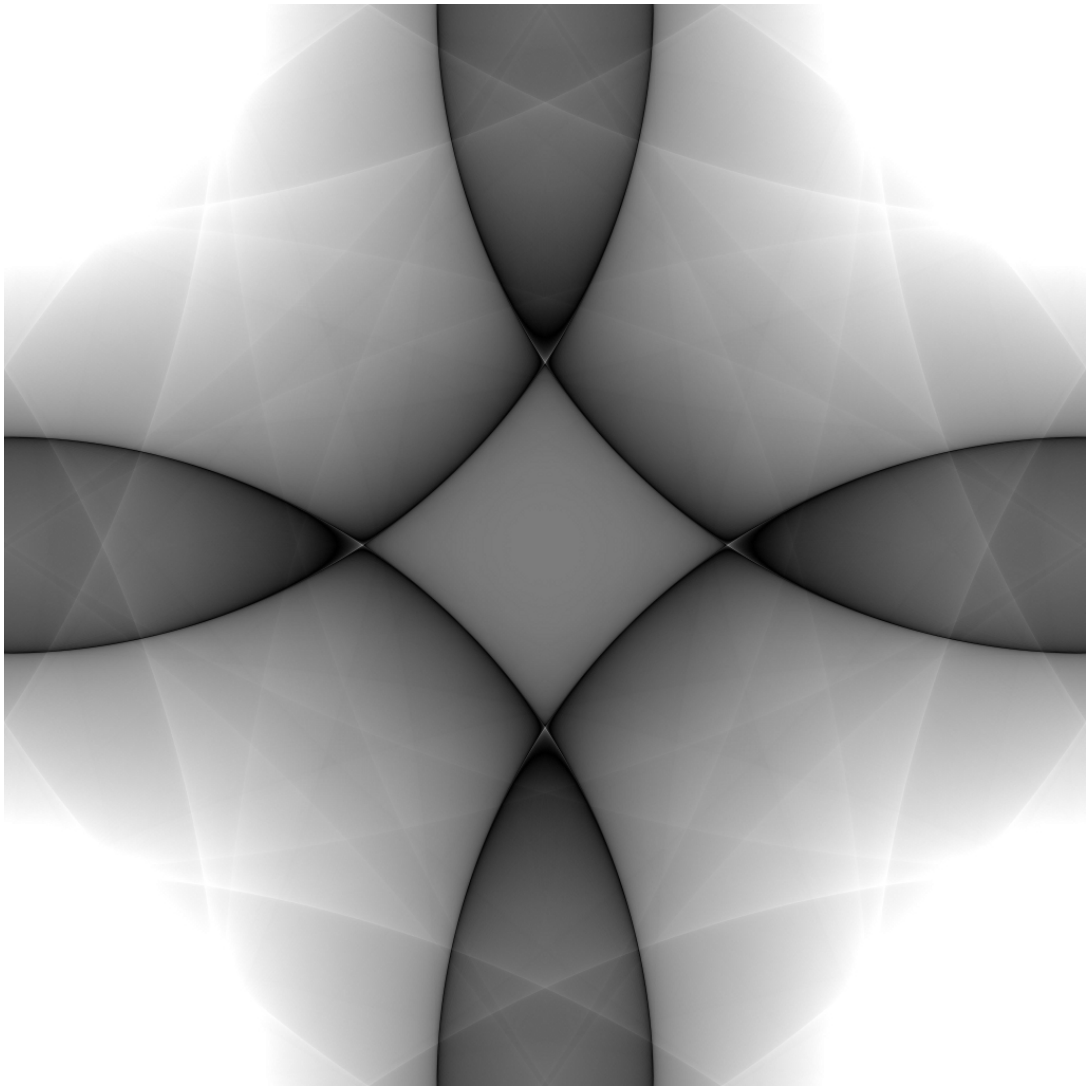

Figure B.5: Magnitude and Phase Blend

Figure B.6: Image Blender: Pollock and Film still from "The Black Stallion"

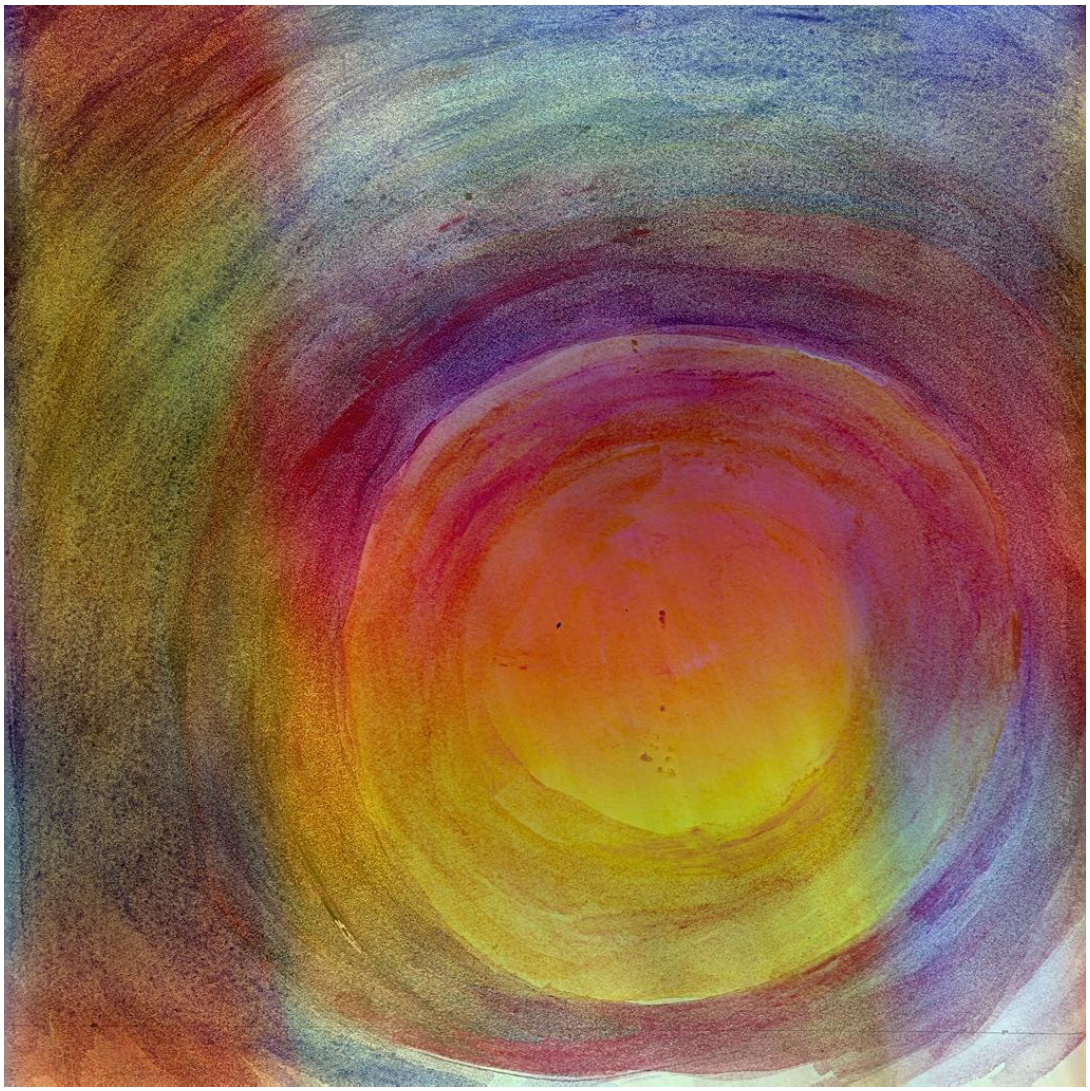

Figure B.7: Image Blender: author's own paintings

Figure B.8: Image Blender: family paintings

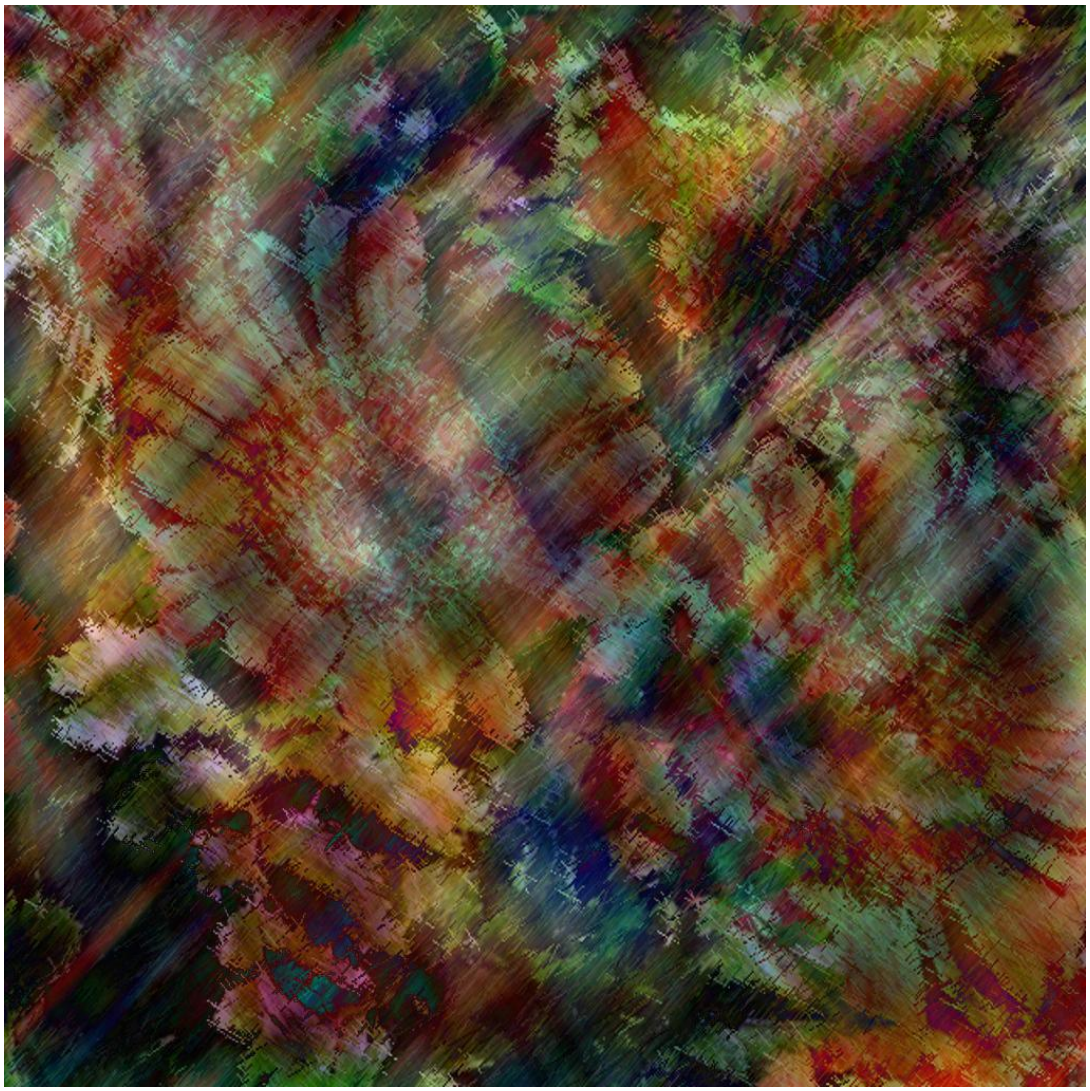

Figure B.9: Image Blender: Kandinsky and flower photograph

Figure B.10: Image Blender: Hopper and Rothko

Figure B.11: Image Blender: family paintings

Figure B.12: Image Blender: author's own and photograph

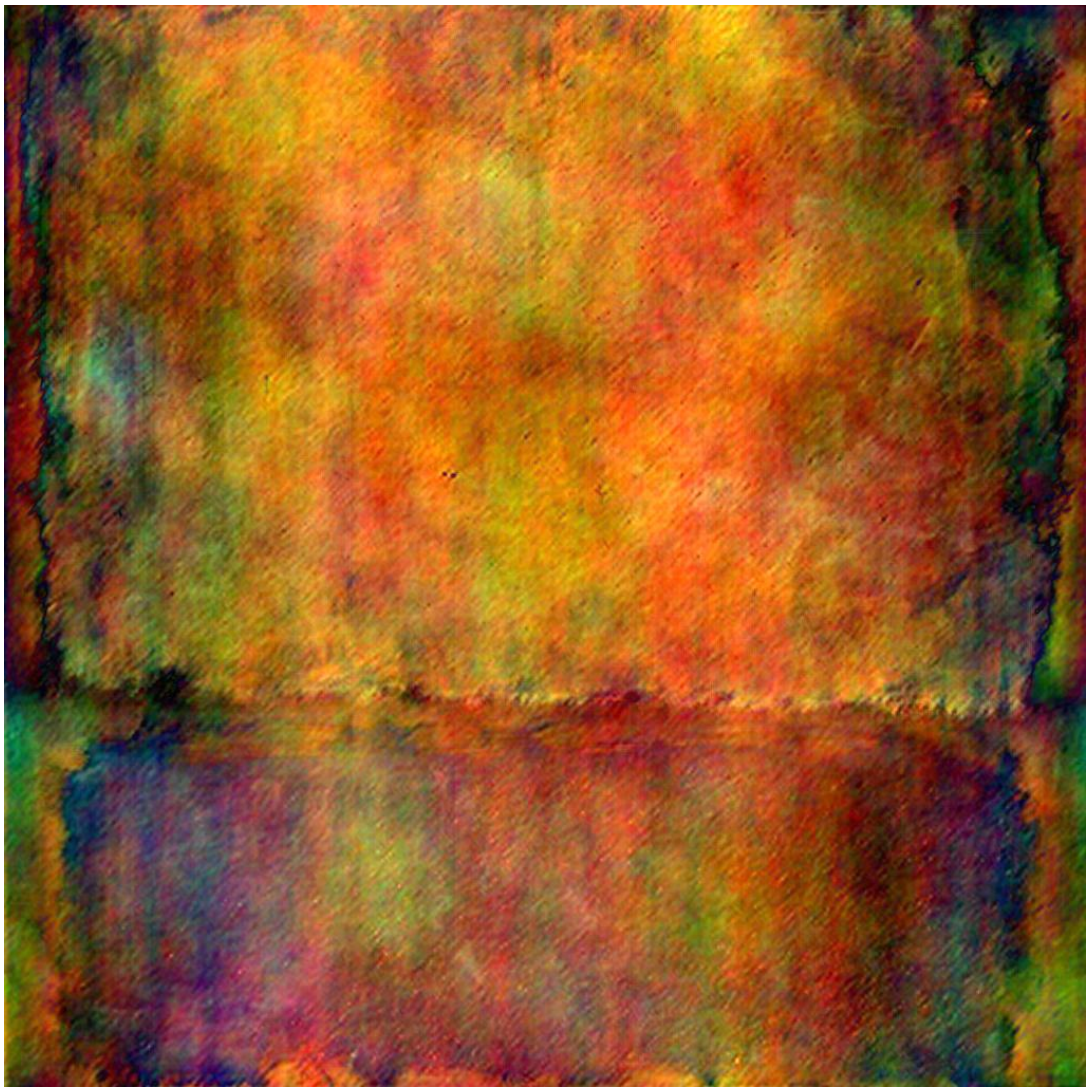

Figure B.13: Magnitude and phase blend using Modigliani and Rothko paintings

Figure B.14: Image Blender: Klee and family member's painting

Figure B.15: Image Blender: Pollock and film still from "The Horse Whisperer"

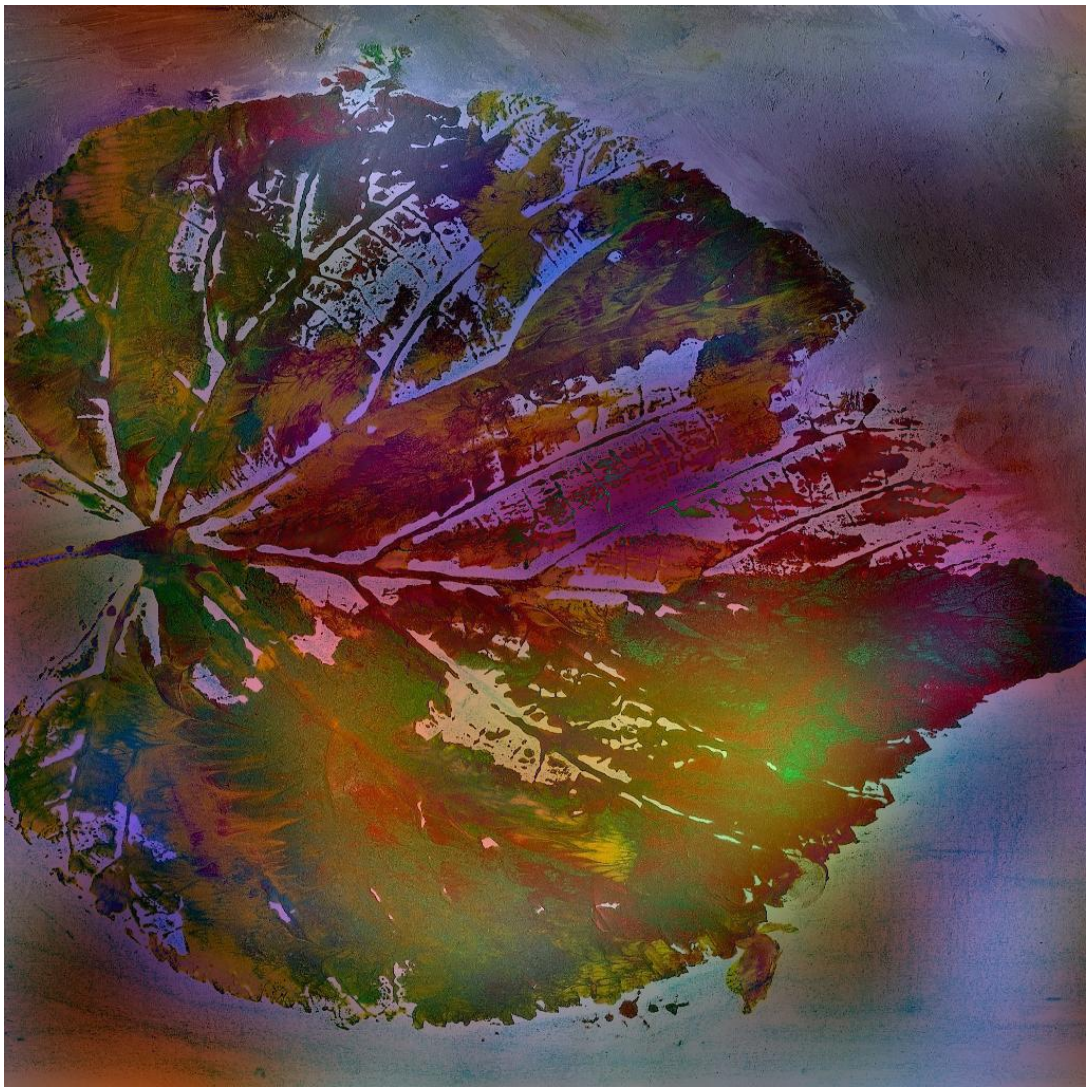

Figure B.16: Image Blender: author's own paintings

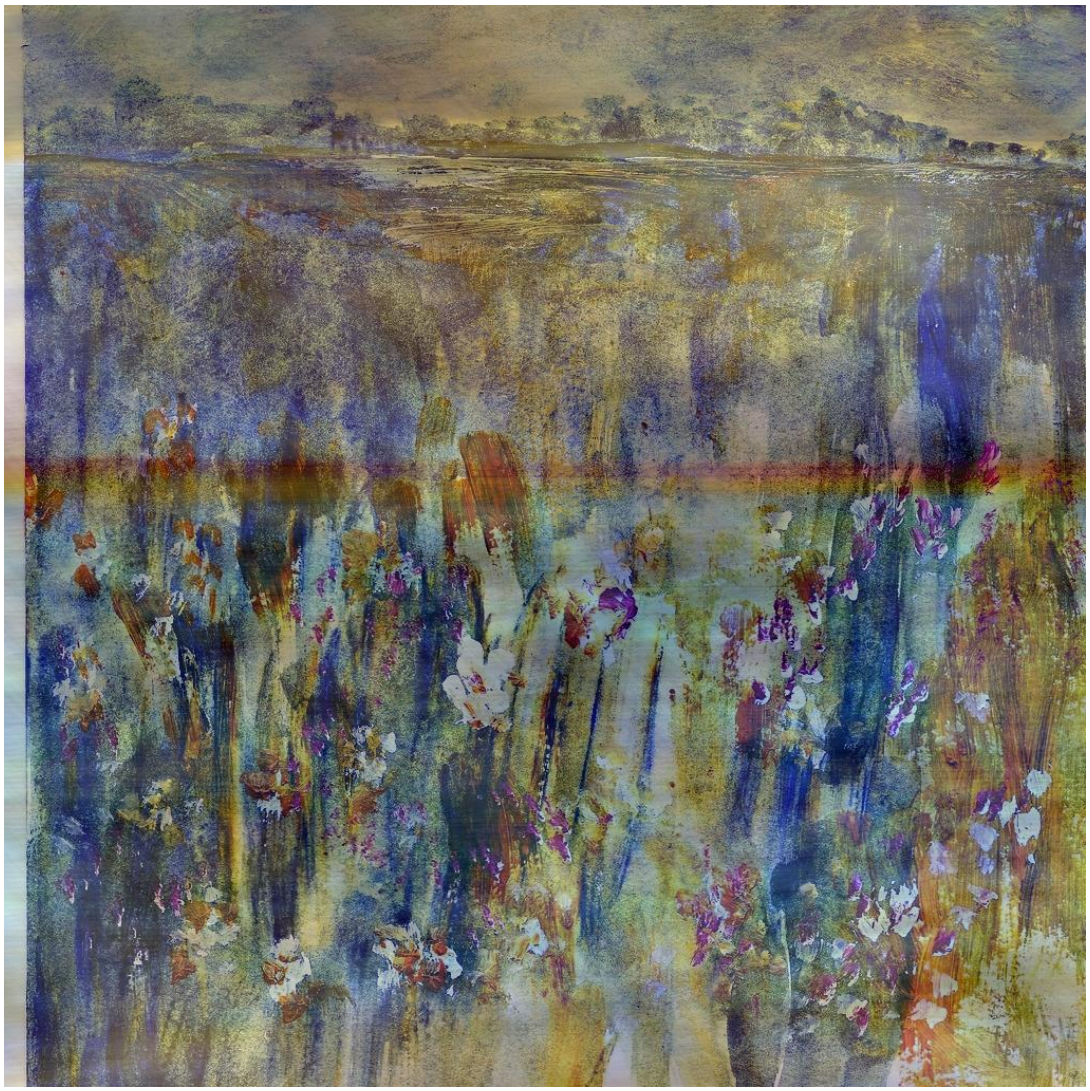

Figure B.17: Image Blender: family paintings

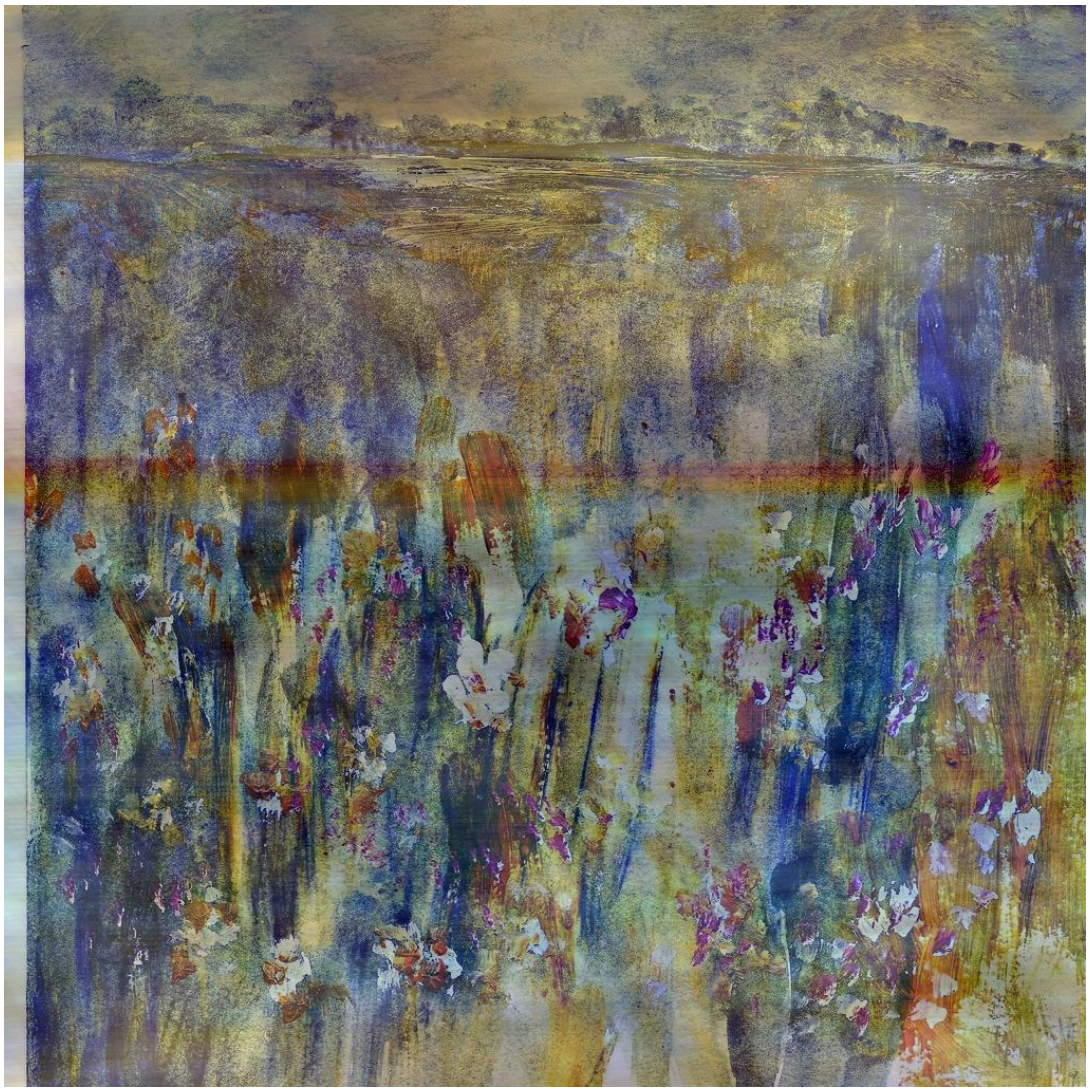

Figure B.18: Image Blender: family paintings

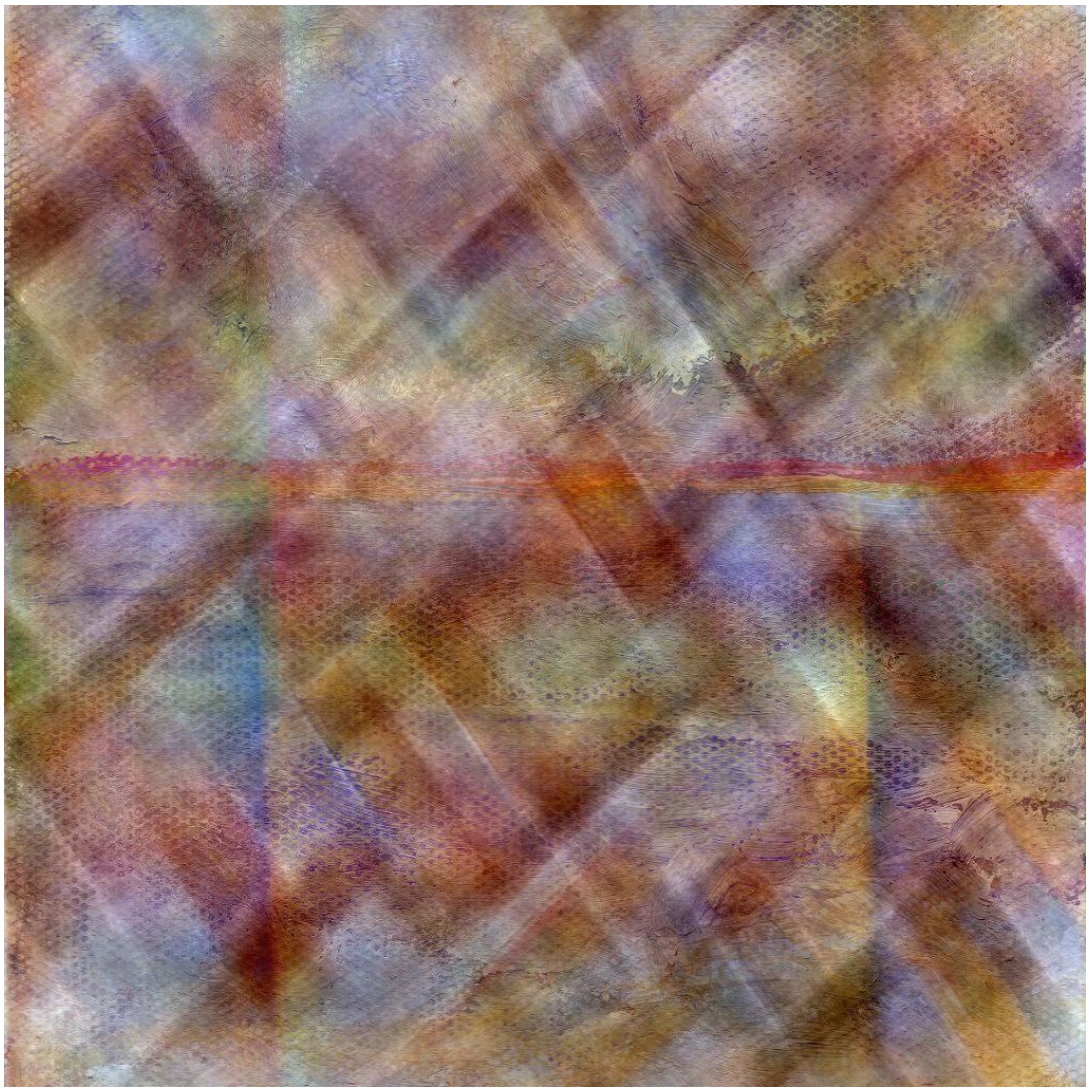

Figure B.19: Image Blender: family paintings

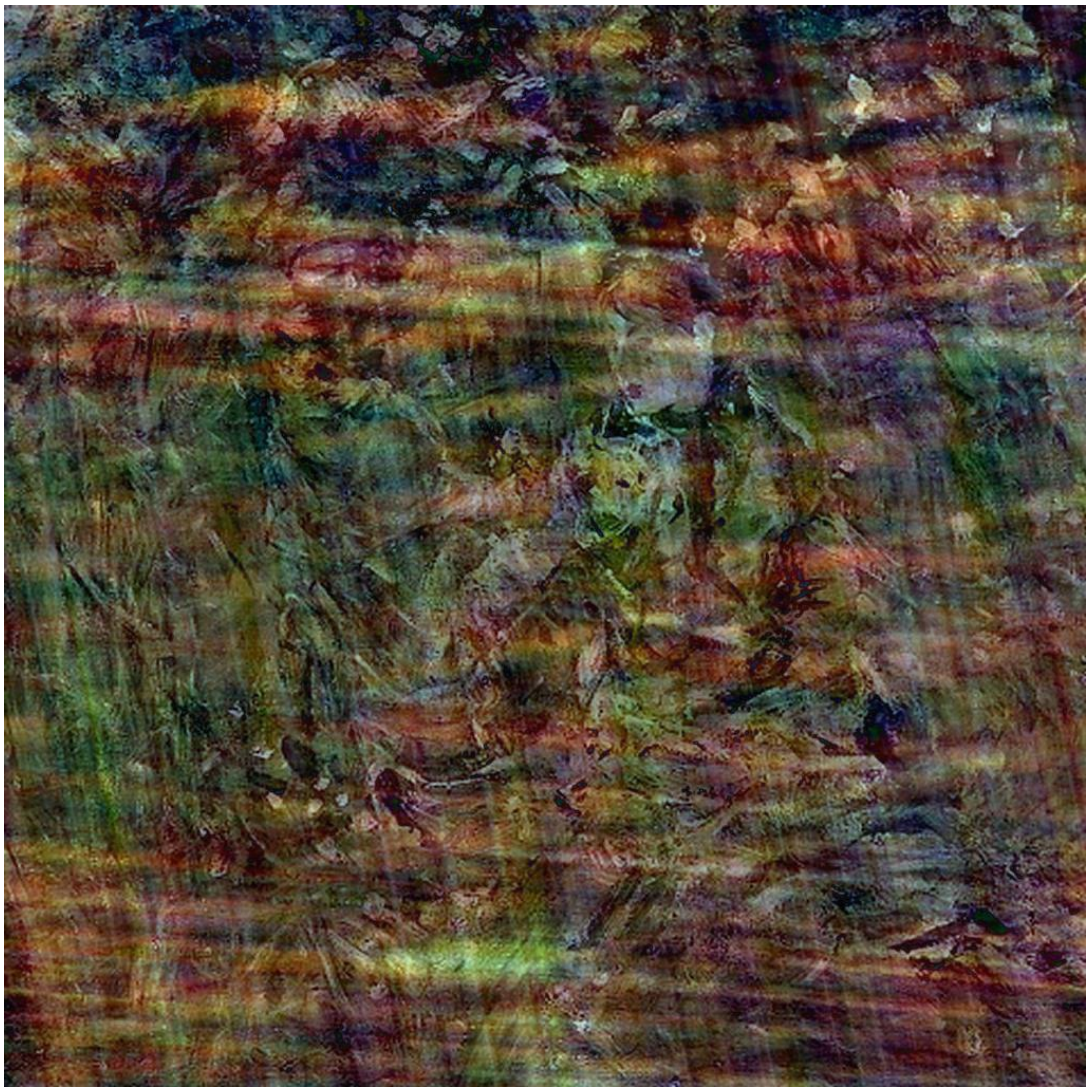

Figure B.20: Image Blender: Morisot and Pollock

Figure B.21: High Pass Filter with red layer

List of Figures

Bibliography

[1] Maria Petrou and Costas Petrou, *Image Processing: The Fundamentals*, pp. 1–46, John Wiley & Sons, Ltd, 2010.

[2] T.J. Naughton, "Cs356 signal, image and optical processing lecture notes," 2013, Department of Computer Science, National University of Ireland Maynooth.

[3] "Roots of polynomials of degrees 3 and 4," *ArXiv e-prints*, Sept. 2010.

[4] James Cooley and John Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of Computation*, vol. 19, no. 90, pp. 297–301, 1965.

[5] M. Mitchell, *An Introduction to Genetic Algorithms*, A Bradford book. MIT Press, 1998.

[6] Karl Sims, "Artificial evolution for computer graphics," *SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 319–328, July 1991.

[7] S.J.P. Todd and W. Latham, *Evolutionary art and computers*, Academic Press, 1992.

[8] A. Forsythe, M. Nadal, N. Sheehy, C. J. Cela-Conde, and M. Sawey, "Predicting beauty: Fractal dimension and visual complexity in art," *British Journal of Psychology*, vol. 102, no. 1, pp. 49–70, 2011.

[9] Richard P Taylor, "Order in pollock's chaos.," *Sci Am*, vol. 287, no. 6, pp. 116–21, 2002.

[10] Simon Colton, "The painting fool: Stories from building an automated painter," in *Computers and Creativity*, Jon McCormack and Mark dInverno, Eds., pp. 3–38. Springer Berlin Heidelberg, 2012.

[11] Michael Cook and Simon Colton, "Automated collage generation - with more intent," in *Proceedings of the Second International Conference on Computational Creativity*, Dan Ventura, Pablo Gervás, D. Fox Harrell, Mary L. Maher, Alison Pease, and Geraint Wiggins, Eds., México City, México, apr 2011, p. 1–3.

[12] David Norton, Derrall Heath, and Dan Ventura, "An artistic dialogue with the artificial," in *Proceedings of the 8th ACM Conference on Creativity and Cognition*, New York, NY, USA, 2011, C&C '11, pp. 31–40, ACM.

[13] David Norton, Darrell Heath, and Dan Ventura, "A collaboration with darci," in *ACM C&C Workshop on Semi-Automated Creativity*, 2011.

[14] Oliver Deussen, Thomas Lindemeier, Sören Pirk, and Mark Tautzenberger, "Feedback-guided stroke placement for a painting machine," in *Proceedings of the Eighth Annual Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging*, Aire-la-Ville, Switzerland, Switzerland, 2012, CAe '12, pp. 25–33, Eurographics Association.

[15] Mass.) Computer Museum (Boston, *The Robotic Artist: Aaron in Living Color : Harold Cohen at The Computer Museum*, Computer Museum, 1995.

[16] Margaret A. Boden, "Computer models of creativity," *The AI Magazine*, vol. 30, no. 3, pp. 23–34, 2009.

[17] P. Gärdenfors, *Conceptual Spaces: The Geometry of Thought*, A Bradford book. Bradford Bks, 2004.

[18] Juan Romero, Penousal Machado, Adrin Carballal, and Olga Osorio, *Aesthetic Classification and Sorting Based on Compression.*, vol. 6625 of *Lecture Notes in Computer Science*, Springer, 2011.

[19] Steven Bergen and BrianJ. Ross, "Automatic and interactive evolution of vector graphics images with genetic algorithms," *The Visual Computer*, vol. 28, no. 1, pp. 35–45, 2012.

[20] Jacob Hoffman, "Evoart: Creating art using genetic algorithms," `http://jdevh.com/2012/05/19/evoart-creating-art-using-genetic-algorithms`, Accessed: 2014-03-04.

[21] R. Dawkins, *The Blind Watchmaker*, Penguin Books Limited, 2013.

[22] W.B. Pennebaker and J.L. Mitchell, *JPEG: Still Image Data Compression Standard*, Chapman & Hall digital multimedia standards series. Springer, 1993.

[23] J. Miano, *Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP*, ACM Press Series. Addison Wesley, 1999.