THE UNIVERSITY OF ADELAIDE

DOCTORAL THESIS

Algorithms and Machine Learning for Evolving Images

Author:
Aneta Neumann

Principle Supervisor:
Dr. Bradley Alexander

Co-Supervisor:
Prof. Dr. Zbigniew Michalewicz

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

School of Computer Science

August 8, 2019

Declaration of Authorship

I, **Aneta Neumann**, certify that this work titled, "Algorithms and Machine Learning for Evolving Images " contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint-award of this degree.

I give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

I acknowledge the support I have received for my research through the provision of an Australian Government Research Training Program Scholarship.

Signed:

Date: August 8, 2019

THE UNIVERSITY OF ADELAIDE

Abstract

Faculty of Engineering, Computer and Mathematical Sciences
School of Computer Science

Doctor of Philosophy

Algorithms and Machine Learning for Evolving Images

by **Aneta Neumann**

Evolutionary algorithms have been used in a wide range of areas to discover novel solutions. The primary aim in the research area of evolutionary algorithms and art is to evolve artistic and creative outputs through an evolutionary process. In this thesis, we investigate how algorithms and machine learning methods can assist in evolving images. We begin by designing different mutation operators based on uniform, biased random and quasi-random walks, and subsequently study how their combination with a baseline mutation operator can lead to interesting image transition processes in terms of visual effects and artistic features. Motivated by our investigations of evolutionary image transition, we present an approach for the composition of new images from existing ones, that retain some salient features of the original images. We use evolutionary algorithms to create new images based on a fitness function that incorporates feature covariance matrices associated with different parts of the images. Furthermore, evolutionary algorithms can be employed to obtain a diverse set of solutions and this field has gained increasing attention over recent years. Diversity optimization in terms of features on the underlying problem allows one to obtain a better understanding of possible solutions to the problem at hand and can be utilised for algorithm selection when concerned with combinatorial optimization problems. We introduce discrepancy-based diversity optimization approaches for evolving diverse sets of images. Furthermore, we propose a new approach for evolutionary diversity optimization based on well-established multi-objective performance indicators. It bridges the areas of evolutionary diversity optimization and evolutionary multi-objective optimization and shows how techniques developed in evolutionary multi-objective optimization can be used to produce diverse sets of solutions of high quality for a given single-objective problem.

Contents

x

List of Figures

List of Tables

Underlying Publications

The content of this thesis is based on the following publications:

1. A. Neumann, B. Alexander, F. Neumann (2016): The Evolutionary Process of Image Transition in Conjunction with Box and Strip Mutation. In: Neural Information Processing, ICONIP (3) 2016, Lecture Notes in Computer Science 9949: 261-268.

2. A. Neumann, B. Alexander, F. Neumann (2017): Evolutionary Image Transition Using Random Walks. In: Computational Intelligence in Music, Sound, Art and Design, EvoMUSART 2017, Lecture Notes in Computer Science 10198: 230-245. An extended version was submitted for a journal publication.

3. B. Alexander, J. Kortman, A. Neumann (2017): Evolution of Artistic Image Variants Through Feature Based Diversity Optimisation. In: Genetic and Evolutionary Computation Conference, GECCO 2017, ACM Press: 171-178.

4. A. Neumann, Z. L. Szpak, W. Chojnacki, F. Neumann (2017): Evolutionary Image Composition Using Feature Covariance Matrices. In: Genetic and Evolutionary Computation Conference, GECCO 2017, ACM Press, 817-824.

5. A. Neumann, F. Neumann (2018): On the Use of Colour-based Segmentation in Evolutionary Image Composition, In: IEEE Congress on Evolutionary Computation, CEC 2018, IEEE Press: 1-8.

6. A. Neumann, W. Gao, C. Doerr, F. Neumann, M. Wagner (2018): Discrepancy-based Evolutionary Diversity Optimization. In: Genetic and Evolutionary Computation Conference, GECCO 2018, ACM Press: 991-998.

7. A. Neumann, C. Pyromallis, B. Alexander (2018): Evolution of Images with Diversity and Constraints Using a Generator Network. In: Neural Information Processing, ICONIP (6) 2018, Lecture Notes in Computer Science 11306: 452-465.

8. A. Neumann, F. Neumann, T. Friedrich (2018): Quasi-random Agents for Image Transition and Animation. In: submitted for publication. Available as CoRR abs/1710.07421.

9. A. Neumann, W. Gao, M. Wagner, F. Neumann (2018): Evolutionary Diversity Optimization Using Popular Multi-Objective Indicators. In: Genetic and Evolutionary Computation Conference, GECCO 2019, ACM Press: 837-845. Paper has been nominated for Best Paper Award at GECCO 2019 (CORE A).

To Frank and Michelle

Chapter 1

Introduction

Evolutionary algorithms (EAs) have been widely and successfully applied in the areas of music and art [5, 147, 221]. The advantage of evolutionary computation methods derives from the well-known fact that they do not require exact knowledge about the optimization problem. For instance, this encompasses the classical task of computing high performing solutions for combinatorial optimisation problems, new designs in the area of engineering, as well as creative solutions in the areas of music and art.

The use of evolutionary algorithms for the generation of images has attracted research interest since the 1980s [175, 267]. In the seminal work the *Blind Watchmaker* Dawkins [30] investigated the process of evolution. He evolved figures called biomorphs in order to explore and to visualize the power of evolution. The biomorphs' appearance diverges broadly from the original parent over time demonstrating a simple model of bio-inspired evolution. Inspired by Dawkins' work Sims [232] created complex and aesthetically pleasing images. He used an expression-based approach to evolve images using a mathematical expression as the genotype and applied crossover and mutation operators. Todd and Latham [247] introduced the framework called *Mutator* to generate art and evolve new 3D biomorphic forms. The *Mutator* creates a series of complex branching organic forms and surreal virtual sculptures, and animated videos through the process of "surreal" evolution. The potential of Charles Darwin theory of the mechanism of evolution have been employed by evolutionary algorithms and used for the generation of digital art [30, 232, 234, 247].

Over time, the growing interest led to the expansion of the area of evolutionary art, providing coherent and solid basis for further research [2, 24, 37, 224, 234]. By following these steps, computer graphics and evolutionary methods have been successful combined to create artworks [87, 96, 147, 221]. Different representations have been used to create works of greater complexity in 2D and 3D [89, 165, 232, 247], and in image animation [96, 232, 248]. Furthermore, several frameworks have been introduced to create artworks based on machine learning [118, 136, 183] as artificial intelligence

has become an essential part of the technology industry [24]. Moreover, in order to judge the quality of such artistic images produced by evolutionary computation methods, aesthetic features have been introduced to measure their properties in an objective way [97, 119, 172]. Image features play also a crucial role in computer vision to classify images according to their properties [7, 265, 275].

Diversity plays a crucial role in evolutionary computation. Traditionally, diversity is used to avoid premature convergence and it is generally assumed that crossover-based evolutionary algorithms need a diverse population in order to produce good results. During the last 10 years, using evolutionary algorithms to produce a diverse set of solutions has gained increasing attention, particularly discovering novel designs for various engineering problems. Ulrich and Thiele [253] introduced evolutionary computation approaches that are able to produce diverse sets of solutions by evolving a population according to quality criteria and diversity measures. Diversity has been used to design evolutionary algorithms as it often prevents the algorithms from premature convergence. In recent years, evolutionary diversity optimization has gained increasing attention [3, 74, 194, 252]. Evolutionary diversity optimization uses an evolutionary algorithm in order to compute a diverse set of solutions that all fulfil given quality criteria. Presenting decision makers with such alternative designs that are all of good quality gives them a variety of design choices and helps to better understand the space of good solutions for the problem at hand. Related to evolutionary diversity optimization is the concept of novelty search [220, 238]. Here, evolutionary algorithms are used to discover new designs without focusing on an objective. The goal of novelty search is to explore designs that are different to the ones previously obtained. In this work, firstly, we consider diversity optimization approaches that builds on a simple diversity measure that measures diversity according to a given feature. We extend this approach to more than one feature and employ a diversity measure weightening for the different features. Furthermore, we use the population of an evolutionary algorithm to evolve images that are close to a given one in terms of an error measure, and that are diverse in respect to different features.

This thesis considers evolutionary algorithms and machine learning methods for evolving images. One of the main goal in the research area of evolutionary algorithms and digital art is to evolve artistic and creative outputs through an evolutionary process.

We now give a brief overview of our parts and chapters that are covered in this work. We begin this work with parts containing descriptions of basic of evolutionary algorithms, approaches in diversity optimization, and introduction to the field and

image features. In the second part we describe evolutionary image transition and animation that use random walks and quasi-random agents. The third part contains the description of evolutionary image composition using feature covariance matrices and colour-based segmentation. The last part contains the description of evolutionary diversity optimisation approaches for images based on features, discrepancy, multi-objective optimization indicators and Generative Adversarial Networks.

In Chapter 2, we introduce preliminaries for considering evolutionary algorithms and diversity optimisation problems. Here, we describe evolutionary algorithms, and subsequently discuss the basic modules of evolutionary algorithms and their basic operators. We give a summary of random walk and quasi random walk concepts in mathematics. We provide an introduction of the basic approaches in diversity, discrepancy theory in respects to the mathematics advantages and describe indicators used in multi-objective optimization. Finally, we give a brief introduction of the machine learning method called Generative Adversarial Networks [84].

In Chapter 3, we give a brief introduction to the field of evolutionary art. For 40 years many new approaches have been proposed for evolving images. We highlight the works of a few specific researchers, in order to examine the various approaches and their selected representants. Furthermore, we present different approaches to evolutionary image transition. We discuss random walk, biased random walk and quasi-random walks for image transition and animation. We present general and aesthetic features used in image analysis. In this chapter, we also briefly explain the concept of the region covariance descriptor and their several features. We discuss the advantages of using the covariance matrix and describe the different distance measurements.

In Chapter 4, we present a study demonstrating how various mutation operators and random walk algorithms can be used for evolutionary image transition. We introduce box and stripes mutation operators which are specifically designed for evolutionary image transition. We design different mutation operators based on uniform and biased random walks, and study how their combination with a baseline mutation operator can lead to novel image transition processes in terms of visual effects and artistic features. Using feature-based analysis, we study the evolutionary image transition behaviour in respect to different features and evaluate the images constructed during the image transition process. Afterwards, we investigate how modifications of our biased random walk approaches can be used for evolutionary image painting. We introduce an evolutionary image painting approach whose underlying biased random walk can be controlled by a parameter influencing the bias of the random walk and thereby creating different artistic painting effects. Our experimental results show that

the process of an evolutionary algorithm in combination with these mutation operators can be used as a valuable way to produce unique generative art.

In Chapter 5, we explore the use of quasi-random walks for evolutionary image transition and animation. In the previous chapter we demonstrated how evolutionary processes have been employed to create artistic image transition processes using random walks. Quasi-random walks have similar features to standard random walks, but with much less randomness. We utilize this established model derived from discrete mathematics and show how agents carrying out quasi-random walks can be used for evolutionary image transition and animation. The key idea is to generalize the notion of quasi-random walks and let a set of autonomous agents perform quasi-random walks that paint an image. Each agent has one particular target image that they paint when following a sequence of directions for their quasi-random walk. The sequence can easily be chosen by an artist and allows them to produce a wide range of different transition patterns and animations.

In Chapter 6, we present a novel approach for the evolutionary image composition (EIC) of new images from existing ones, that retain some salient features of the original images. We introduce evolutionary algorithms that create new images based on a fitness function that incorporates feature covariance matrices associated with different parts of the images. This approach is very flexible in that it can work with a wide range of features and enables specific regions in the images to be targeted. For the creation of the new images, we propose a population-based evolutionary algorithm with mutation and crossover operators based on random walks. Our experimental results reveal a spectrum of aesthetically pleasing images that can be obtained with the aid of our evolutionary process.

In Chapter 7, we introduce the use of image segmentation methods in evolutionary image composition. Here, a composition of two images is created by evolving an image that is similar to the given two images S and T and an error function that takes into account the similarity of the resulting images to S and T in terms of a covariance feature-based distance measure. An important aspect of creating visually pleasing compositions of images is a good method that characterizes important regions of the given images as that the composition takes into account these aspects. We show that the use of different weightings in the fitness function plays a crucial role in obtaining image compositions.

In Chapter 8, we focus on the concept of diversity optimization in the field of evolutionary computation. While diversity has been mainly used to prevent the population of an evolutionary algorithm from premature convergence, the use of evolutionary

algorithms to obtain a diverse set of solutions has gained increasing attention in recent years. Diversity optimization in terms of features on the underlying problem allows one to obtain a better understanding of possible solutions to the problem at hand and can be used for algorithm selection when dealing with combinatorial optimization problems. Moreover, in Chapter 8, we present a search framework to evolve diverse variants of a source image in one and two feature dimensions. Those measures aim to maximise the diversity of images. We provide the resulting images that show the spectrum of effects from transforming images to score across the range of each feature. We demonstrate the correlations between different feature metrics obtained from populations for two feature dimensions.

After having presented the use of diversity optimization in the previous chapter, we now consider discrepancy-based diversity optimization approaches for evolving diverse sets of images. In Chapter 9, we present a star discrepancy measure that measures the regularity in respect to all axis-parallel boxes, that are anchored in the origin. This measure provides insights into how evenly the points are distributed. We investigate the applicability of the star discrepancy measure in evolutionary diversity optimization for the diversity optimization of images. Our experimental investigations compare several diversity optimization approaches and show that a discrepancy-based diversity optimization approach in conjunction with the tie-breaking rule provides the best results in terms of the star discrepancy measure. The tie-breaking rule is based on weighted differences between surrounding feature points in respect to two and three features.

In Chapter 10, we bridge the areas of evolutionary diversity optimization and evolutionary multi-objective optimization. We utilise evolutionary diversity optimization which aims to compute a diverse set of solutions where all solutions meet a given quality criterion and combine with different indicators from evolutionary multi-objective optimization. We show how popular indicators frequently employed in the area of multi-objective optimization can be utilised for evolutionary diversity optimization. Our experimental investigations for evolving diverse sets of images according to various features show that two of the most prominent multi-objective indicators, namely the hypervolume indicator and the inverted generational distance, provide excellent results in terms of visualization and various diversity indicators.

Generative Adversarial Networks (GANs) have become well known for the synthesis of realistic images. In Chapter 11, we consider the training of Generative Adversarial Networks with the aim of generating novel images that are created based on optimisation of feature scores in one or two dimensions. Recent developments in deep

learning have enabled a generation of compelling images using Generative Adversarial Networks that encode images with lower-dimensional latent space. We use search in the latent space of the GAN to generate images scoring high or low on feature value. Our approach successfully generates image variations with two datasets, namely faces and butterflies. The work gives insights into how feature measures promote diversity of images and how different measures interact in the content of Generative Adversarial Networks.

Many people have supported my work during the last few years. Firstly, I would like to thank my supervisor Bradley Alexander and my co-supervisor Zbigniew Michalewicz for supporting my research on evolutionary algorithms and machine learning for evolving images. I would also like to thank my collaborators for the discussions we have had over the last few years: Wojciech Chojnacki, Carola Doerr, Tobias Friedrich, Wanru Gao, James Kortman, Frank Neumann, Christoph Pyromallis, Zygmunt Szpak, Markus Wagner. I am very grateful to my husband Frank and our daughter Michelle for their support and love.

Part I

Basics

Chapter 2

Evolutionary Algorithms and Machine Learning

In this section, we present a brief introduction of the field of evolutionary computation and pay attention to key diversity approaches. Evolutionary algorithms are inspired by evolutionary processes in nature and follow the principles of the Charles Darwin theory of evolution through natural selection [29]. We describe the basic modules and the different approaches developed in the field of evolutionary optimization in Sections 2.2 and 2.3. Throughout this work we mainly consider simple evolutionary algorithms, namely the (1+1) EA and the $(\mu + \lambda)$ EA. In Section 2.4, we study random walks and quasi-random walks approaches. Diversity is used to avoid premature convergence and plays an important role in the exploration of the diverse population in the solution space. We introduce the basic approaches in diversity and their measure in Section 2.5. We present multi-objective optimization indicators for diversity optimisation. In this chapter, we also explain the concept of Generative Adversarial Networks that have become a popular technique for unsupervised deep learning in recent years.

2.1 Evolutionary Algorithms

Evolutionary algorithms (EAs) have become very popular over the last 50 years. They owe this success in large part to their effective way of solving a wide range of real-world optimization problems. Evolutionary algorithms are inspired by the evolutionary process in nature. The aim of evolutionary algorithms is to solve problems by evolving sets of search points and to obtain satisfying results. Especially in real-world applications, the function that we optimize is often unknown and the function values are obtained through experiments. On the other hand, experiments are costly and time consuming. The underlying idea of evolutionary algorithms is to produce the best individuals according to survival processes as mirrored by evolution in nature.

Evolutionary algorithms work with a set of solutions at each time step. We call this the population of an evolutionary algorithm. In the next step, this population produces a set of solutions that we call the offspring population. This process is known as the reproduction. After the new population is created, we begin with the selection process. New parent populations are formed from the previous parent and offspring population. During the selection process, a candidate solution will be accepted. A comprehensive description of different types of evolutionary algorithms is given in [60, 178]. In this work we mainly consider simple evolutionary algorithms.

2.2 Basic Modules of Evolutionary Algorithms

In this section, we give a brief overview of the different modules of evolutionary algorithms. In order to define an evolutionary algorithm, we describe its basic components. Specifically, we summarize the following components: representation, fitness function, variation operators, and selection mechanisms.

2.2.1 Representation

The representation of a given problem is the stepping stone toward the definition of individuals used for solutions [60, 225]. This process can be compared to the mapping from the problem in the real word to the problem in solving space. The goal is to define the context of the problem within the possible solution space, taking into account the abstraction of real word structures. This prompts the question of how feasible solutions can be specified on the computer. In the first step, we map a set of genotypes from the phenotypes. Within the evolutionary algorithms, the individuals that specify possible solutions within context problems are called phenotypes. On the other hand, the individuals that are encoded from phenotypes are called genotypes.

A common way of representing solutions is bitstring of length n, $x \in \{0,1\}^n$. For example, for graph problems solutions can also be represented in different ways. The representation of a given undirected connected graph with n vertices and m edges is a set of n - 1 edges and is known as spanning trees by edge sets [214]. We can represent a spanning tree as bitstrings of length m. Each bit corresponds to an edge that is included in the solution and set to 1 and to 0 alternatively.

Another example is an optimisation problem that needs to be solved where the possible feasible solutions are represented by the RGB color model. The palette RGB refers to a system for three levels that correspond to the range of decimal numbers from 0 to 255. It is possible to represent the decimal numbers by their binary digits. In the case

of the color crimson, the RGB values are $(220, 20, 60)$ defined as a phenotype, and the contra-part in binary code is (11011100, 10100, 111100) as a genotype. Furthermore, the search space is defined as the genotype space, although the phenotype space can differ from the genotype space. We refer the reader to [60] for a more detailed description.

2.2.2 Fitness function

The fitness function evaluates the quality of a solution. The fitness function is used to guide the population towards a good or optimal solution. Initially, after a new population of solutions is generated, each of the solutions is evaluated by the fitness function. In this way, we determine how well is this solution. In other words, we estimate the success of each population of candidate solutions in terms of their adaptation towards the best possible solution for the particular problem. This mirrors the natural selection process based on the equivalent observations from the real world phenomena.

2.2.3 Variation Operators

In evolutionary computation, variation operators are important components of evolutionary algorithms and play a key role as they producing new solutions. Variation operators are divided into *mutation* and *recombination operators* and crucially promote the diversity within the population by generating new individuals. We illustrate important variation operators for binary strings $x \in \{0,1\}^n$.

Mutation Mutation operators depend on the structure of the individuals and introduce changes at random positions in the search space. The most common behaviour of all mutation operators induces some small, unbiased change to an individual's genotype. We now describe two common mutation operators. In the case of bit-strings of length n, mutation is performed by flipping each bit of an individual independently with the probability $p_m = 1/n$. Thus, the most common mutation probability is called *standard bit mutation*, by flipping only one bit on average. So, the expected number of positions occurs by the differences between individuals, the parent and the offspring, equals n. Similarly, in the case of bit-strings of length n, *b-bit mutation* is created by flipping exactly b bits. The positions of b bits are chosen uniformly at random.

Recombination As mentioned above, we classify variation operators based on the number of arguments they modify. Variation operators involving two individuals are called *crossover operators* or *recombination operators*. The crossover operator modifies one of the parents, or both parents according to the current status of the other/both parents.

By modifying one or two offspring with different features, a crossover operator may be able to produce offspring that combine these features. This implies that if both parents share a common value at some bit, the offspring will receive the same value at this position. If both parents have different bit values then the bit is randomly assigned in the offspring. Here, we describe three crossover operators that construct one or two offspring. Let $x, y \in \{0,1\}^n$ be the two parents of length n, with $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$. In the case of *1-point crossover* by crossing two parents, a value i, $i \in 1, \ldots, n$ is chosen uniformly at random. The *1-point crossover* operator creates an offspring by choosing the parent at position i and taking other parts from a different parent. The resulting offspring from the crossover is: $z = (x_1, \ldots, x_i, y_{i+1}, \ldots, y_n)$. The case can be generalized by using *k-point crossover* where the points are chosen at random. Let the points be $p_1 < p_2 < \ldots < p_k$. Next, the k-point crossover alternates k-times between the two individuals. As such, the recombination operator depends on the ordering of bits. This is in contrast to *uniform crossover* where each bit value is chosen from a parent uniformly at random.

2.2.4 Selection Mechanisms

In evolutionary computation, we distinguish between two selection mechanisms, namely *parent selection* and *survivor selection*. Parent selection is used to select a set of parents from the current population to generate offspring. The survivor selection is used to decide on the individuals from the combined parent and offspring population based on their quality, which can then build the next generation. The survivor selection takes place after the creation of the offspring. We will present four selection methods in the following section. A simple kind of selection is a *uniform selection* where the individuals are selected uniformly at random. Another selection method is *fitness-proportional selection*. Let us assume that the fitness function f has to be maximized and that all function values are positive. When the population contains μ individuals $X_1, \ldots, X_\mu$ the individual X_i the probability of the individual being chosen in each step is $f(X_i)/\sum_{j=1}^{\mu} f(X_i)$. We are able to choose a duplicate in each selection step. Another strategy is a *tournament selection* that simulates a tournament among randomly selected individuals. More precisely, tournaments of size $q \in \{1, \ldots, \mu\}$ are chosen. Here, in each tournament q individuals compete against each other. Subsequently, the winner of the tournament with the highest fitness value is selected. Finally, this individual is chosen for the next generation. *Cut selection* is a strategy that deterministically selects the individuals with the best fitness. Note that survivor selection drives

the search process, which implies that it often has a strong focus on selecting a high quality solution from the parent and offspring.

2.3 Main Types of Evolutionary Algorithms

In this section, we present a brief overview of the main types of evolutionary computation techniques. A comprehensive description of the different types of evolutionary algorithms can be found in [60, 178].

Genetic algorithms (GAs) have been introduced by Holland [105]. The use of binary strings of length n to represent possible solutions. In the next step, mutation, recombination, and selection can be applied to generate possible solutions. They use crossover as the main variation operator. The simple GA (SGA) works with a population of size μ and in each iteration λ children are produced. SGA uses a 1-point crossover and in the mutation step each bit is flipped with a fixed probability p where $p = 1/n$ is often chosen. The major survivor selection method for GAs is fitness proportional selection (see Section 2.2.4).

Evolution strategies (ES) were proposed and developed by Rechenberg [216, 217] and Schwefel [230, 231]. These are used to solve continuous optimization problems. In ES, the individuals are represented by real-valued vectors. In contrast to genetic algorithms, that rely on recombination, evolution strategies use the primary mutation as an operator for creating offspring. The survivor selection in ES is based on fitness rankings. Two common evolutionary strategies are: $(\mu + \lambda)$ ES and (μ, λ) ES. Here, μ denotes the parent population and λ denotes the size of the offspring population. In $(\mu + \lambda)$ ES, in each iteration λ offspring are produced, and the fitness is calculated. In survivor selection the bestμ individuals from the current parent population and offspring are selected. In the case of (μ, λ) ES, λ offspring are generated from the parent population. The new parent population is created by selecting μ best individuals from the offspring population whilst ignoring the old parent population.

Evolutionary programming (EP) was first discussed in Fogel's work [65] during the early 60s and was subsequently introduced by Fogel, Ownes, and Walsh [66]. EP does not require a specific form of representation. Instead, it allows the user to select the representation depending on the particular optimization problem. The main variation operator is mutation. In a standard approach, a parent population of the size μ produces μ children by mutation. In contrast to GA, where the crossover is used exchange information between possible solution, there is no focus on crossover. EP makes use

of fitness proportional selection. Evolutionary programming insures that the best solution is included in the new parent population as part of the optimization process [63, 64].

Genetic programming (GP), invented by Koza [139–141, 143, 144], is an evolutionary computation approach that originally takes the form of Lisp S-expressions [174, 239]. Over the past decades, GP has become popular as it successfully produces human-competitive performance in the area of real-world application [142, 206]. GP aims to construct good computer programs by following the evolutionary approach. In this case, individuals are computer programs and are usually represented as trees mirroring represented expressions. These trees are evolved during the evolutionary process. Traditionally, GP evolves structures of a variable-length as the size of a possible good solution may not be known. Here, a parent population creates an offspring population using mutation and crossover. The fitness is calculated by its performance in respect to the evaluation of some test cases. Fitness-proportional selection is often chosen in order to select individuals from the parents and offspring for the new parent population.

Ant Colony Optimization (ACO) [15, 46–48, 50, 51] is a randomized search heuristic for combinatorial optimization problems. Dorigo, Maniezzo, and Colorni [49] introduced the Ant System (AS), namely the first ACO algorithm, and shown how this approach can be successful applied in solving combinatorial optimization problems. As such, ACO algorithms are inspired by the biological example of the behaviour of ants who use pheromones created from the glands of their body as a communication medium whilst searching for a source of food. By analogy, the artificial ants communicate with another agents of the colony with help of artificial pheromone trails. In an ACO algorithm each artificial ant of the colony probabilistically construct graph in order to search for an optimal solution and adapt during the search obtained by random walks of ants with a help of memory that are store individual. In addition, ACO algorithms allows to use of metaheuristic to guide the random walks and to solve combinatorial optimization problems.

Particle swarm optimization (PSO) is a metaheuristic inspired by the social behaviours observed in animals, namely fish schools, bird flocks, insects and humans for solving combinatorial optimization problems. Originally developed by Kennedy and Eberhart [127] PSO is a popular and efficient bio-inspired optimization technique that can be applied for solving various optimization problems [16, 17, 125, 126]. PSO uses social behaviours and cognition information in order to maintain a population of candidate solution. These swarms of particles are able to iteratively be improved

Algorithm 1 (1+1) EA for maximizing $f : \{0,1\}^n \to \mathbb{R}$.

1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. Produce x' by flipping each bit of x independently of the other bits with probability $1/n$.
3. Replace x by x' if $f(\,x'\,) \geq f(\,x\,)$.
4. Repeat Steps 2 and 3 forever.

Algorithm 2 $(\mu + \lambda)\ EA$ for maximizing.

1. Initialize the population P with μ search points.
2. Let $C \subseteq P$ where $|C| = \lambda$.
3. For each $x \in C$, produce an offspring x' of x by mutation and add x' to P.
4. While $|P| > \mu$, remove x from P a lowest fitness values.
5. Repeat step 2 to 4 until termination criterion is reached.

through interactions based on their own and their neighbour's position.

2.3.1 Simple Evolutionary Algorithms

In this section, we describe simple evolutionary algorithms namely called (1+1) EA and $(\mu + \lambda)$ EA. We consider a simple evolutionary algorithm called (1+1) EA that has been extensively studied in the area of runtime analysis. The approach for maximizing a fitness function f is given in Algorithm 1. Firstly, we begin with a randomly chosen bitstring x of length n for the search space $\{0,1\}^n$. More precisely, the algorithm creates a child/search point by independently flipping each bit of x with a fixed mutation probability $1/n$. The interesting property of (1+1) EA is that the probability of moving from x to y in one iteration is strictly positive for any $x, y \in \{0,1\}^n$. The (1+1) EA uses a population size of exactly one and then produces exactly one single child/search point at each iteration step.

We now describe a population-based algorithm as $(\mu + \lambda)$ EA. Here, we state μ as notation for the size of the parent population. We denoted the number of offspring created in one generation by λ. In this case, we do not only keep a single best individual, but store up the μ best individuals. The approach for maximizing a fitness function f

is given in Algorithm 2. We start with initialization of the population by choosing a multiset of μ individuals uniformly at random from $\{0,1\}^n$. In the parent selection, we select set C from P uniformly at random. Each selected parent is mutated by flipping each bit of x with probability $1/n$. The new offspring is added to the population P. In the selection, individuals with the lowest fitness are iteratively removed until the population once more contains μ individuals.

2.4 Random Walks

Random walks are one of the most studied topics in probability theory [208]. The intuition of the simplest random walk process is as follows: the walker moves along a street, and at each intersection he or she flips a fair coin to decide the direction for the next step. However, in our work we consider random walks on the two dimensional integer lattice. Adequately, the walker in two dimensions moves on a grid of streets, and at each intersection he or she uniformly and randomly decides on one of the four possible directions. This implies that the walker can move up, down, left, or right with equal probability. Figure 2.1 illustrates the path of one random walk with s steps, s_1 = (100, 1000, 1000), and s_2 = (200, 2000, 2000). For example, the cover time of the uniform random walk on a $n \times n$ torus is upper bounded by $4n^2(\log n)^2/\pi$ [33] which implies that the expected number of steps of the uniform random walk until the torus is obtained is upper bounded by $4n^2(\log n)^2/\pi)$. In our work, we also consider a *biased random walk* where the probability of choosing a direction is dependent on the weights. Weighted random walks have been used in a similar way in the context of image segmentation [85]. We will utilize the biased random walk for evolutionary image transition in Chapter 4.

2.4.1 Quasi-random Walks

We investigate a *quasi-random walk* that is a deterministic analogue of random walks, studied in [59, 210, 262]. The quasi-random walks were also described as a Propp machine by Cooper and Spencer [26] and as "deterministic random walk" later on by the same authors [25, 41]. The quasi-random walk is a model of a rotor-router that aggregates particle in $\mathbb{Z}^2$. The rotor can point right, down, left, and up. At the beginning, a particle starts from a chosen location, then follows the direction of the router to its current position p. Subsequently, the particle is rotated clockwise by 90 degrees. Analyses of the behaviour in comparison to classical random walks are provided in [70,

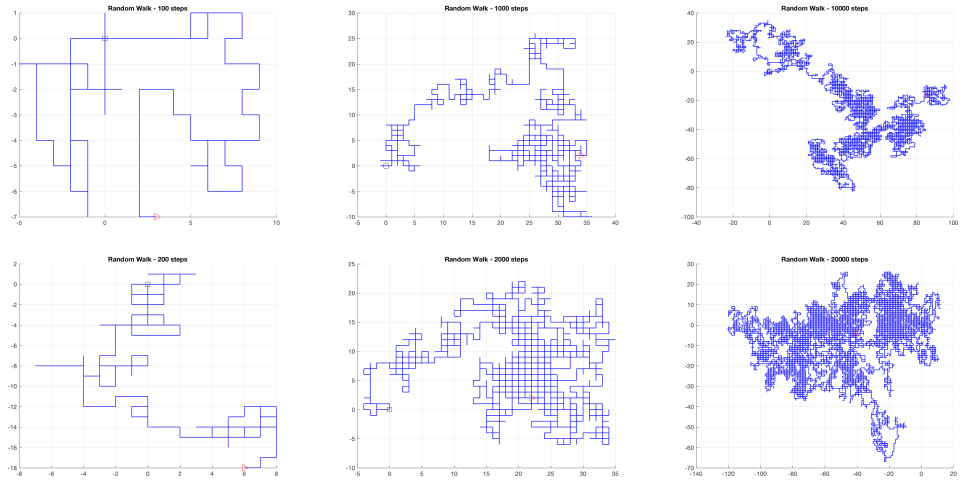


FIGURE 2.1: The trajectories of one random walk with steps s_1 = (100, 1000, 10000) (top) and s_2 = (200, 2000, 20000) (bottom).

133, 153, 211]. These investigations operate under the assumption that the router sequence on each position follows one specific pattern alternating every time between the directions right, down, left and up. The quasi-random model has recently gained significant attention in the context of rumor spreading [40, 42, 43].

The original Propp machine distributes the particle on an infinitive two-dimensional grid. Furthermore, the particle of the Propp machine is always moving in the direction that the current particle is pointing to. Through this we achieve an interesting behaviour from the particles as the rotor rotates according to the fixed permutation of the four directions. This process ensures that the particles are distributed evenly among the neighbours. In recent years, the Propp machine has attracted attention from mathematical and computer science researchers due to its similarity to random walks.

A quasi-random walk is best described on the two-dimensional infinite grid $\mathbb{Z}^2$. Imagine a single walker on some arbitrary vertex $(x, y) \in \mathbb{Z}^2$. The walker is allowed to move to one of the four neighbours by moving right, down, left, or up, that is, to $(x + 1, y)$, $(x, y - 1)$, $(x - 1, y)$, or $(x, y + 1)$. A random walk would independently and uniformly choose one of these four neighbours at random and move there. In order to use less randomness, the quasi-random walk assigns a permutation of the four directions right, down, left, and up to each vertex of the grid. These permutations are fixed initially and deterministically govern the behaviour the quasi-random walkers. To store the current status of the permutation, each vertex has a rotor to store which neighbour it should visit next. Each time a walker leaves a vertex, it moves in the direction of the rotor and updates the rotor according to the fixed permutation of the

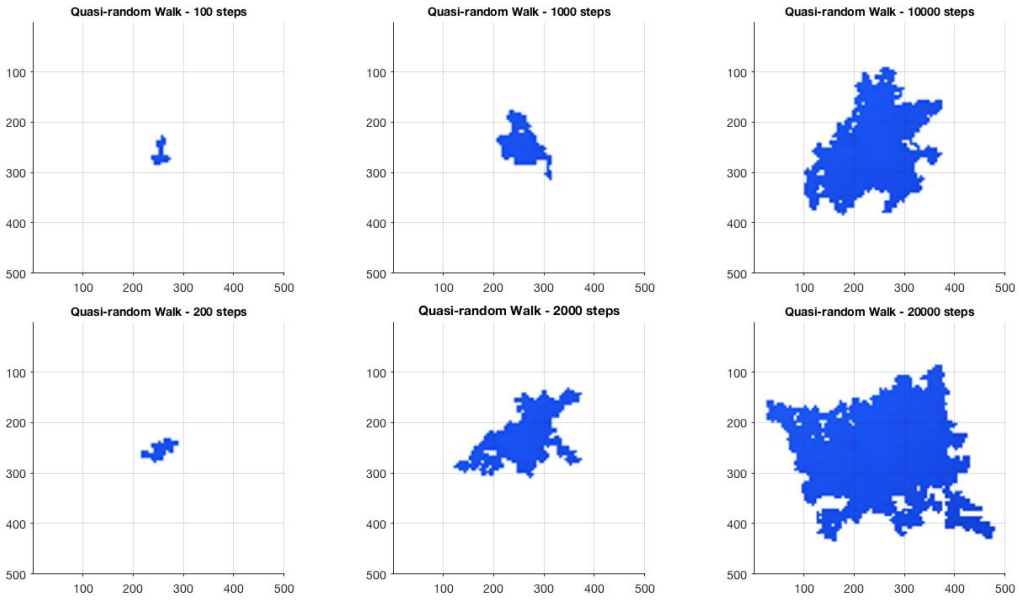


FIGURE 2.2: The trajectories of one quasi-random walk on the two-dimensional infinite grid $\mathbb{Z}^2$ for steps s_{q1} = (100, 1000, 10000) (top) and s_{q2} = (200, 2000, 20000) (bottom).

vertex. When the rotor reaches the last position of permutation, it wraps around and starts again at the beginning of the permutation. The advantage of the rotor-router model is that it does not require randomness and the process is completely determined by the sequence used and the starting position of the rotor.

For example, the simplest rotor sequences are $s_{q1} = (\text{right}, \text{down}, \text{left}, \text{up})$ or $s_{q2} = (\text{right}, \text{left}, \text{down}, \text{up})$. In both cases the quasi-random walker visits the four neighbours as uniformly as the standard random walk in expectation. Note that the first clockwise rotor sequence is circular whilst the second rotor sequence is non-circular. Figure 2.2 illustrates the trajectories of one quasi-random walk starting at the position [0, 250] x [0, 250] on the two-dimensional grid. We present the paths of quasi-random walks for s_{q1} = (100, 1000, 10000) in the top left-hand corner, and s_{q2} = (200, 2000, 20000) in the bottom left-hand corner. It has been proven that such quasi-random walkers behave very similarly to the expected behaviour of a standard random walk [41]. In fact, there is even a slight difference between both aforementioned rotor sequences: The second non-circular rotor sequence is proven to behave closer to the expected random walk than the first circular rotor sequence.

2.5 Basic Approaches in Diversity Optimization

Optimisation problems are ubiquitous in real-word applications. For many years researchers have been interested in developing robust algorithms in the area of combinatorial optimization. Traditionally, algorithms are developed based on the improvements of a single solution in the whole search space. However, in the last thirty years, population-based optimisation algorithms have steadily grown in popularity. This interest is due to the awareness of certain advantages of the population-based algorithms. The focus of population-based algorithms lies in the diversity of the solutions of the population. Most evolutionary algorithms work towards integrating the diversity mechanisms by ensuring that there is a diverse set of individuals among the population [22, 259]. The existence of the diverse population plays an important factor in adequately exploring the solution space. The diversity of a population-based algorithm allows us to analyze different factors of the population. Specifically, the diversity in evolutionary algorithms measures the variety of individuals in respect to the values of the evaluation functions.

2.5.1 Diversity in Solution Space

In general, evolutionary algorithms explore the information of the previous generation in order to extend the search into the most promising region in regards to the best performance [82]. It is crucial to include a diverse set of candidate solutions in the early stage of the search so that the algorithms are able to explore the whole search space. A good diversity mechanism prevents an evolutionary algorithm from encountering premature convergence and the potential for it to be trapped in local optimal solutions [152]. Many different mechanisms are used to preserve objective space diversity and to balance exploration in the search space, such as crowding [170], sharing [179], niching [171, 228], have been proposed.

2.5.2 Diversity in Decision Space

Ulrich and Thiele [253] have introduced the framework for evolutionary diversity optimization. They studied how to evolve diverse sets of instances for single-objective problems in the underlying search space. Furthermore, this diversity optimization approach has been introduced into multi-objective search [252]. In [74], an evolutionary diversity optimization process was introduced to evolve TSP instances based on given problem features. This approach evolves TSP instances that are either difficult or easy

to solve for a given algorithm. Here, diversity is measured according to a weighted distribution in terms of the differences in feature values.

2.5.3 Diversity Measurements

In order to prevent premature convergence, it is beneficial to use decision space diversity [251, 252]. In the following, we show the most well-know classes of diversity measures, specifically used in the field of biodiversity. We focus on the possibilities for the application in EAs. We consider the following properties of a diversity measure first introduced in [236]:

1. Monotonicity in Varieties. The diversity of a set of solutions S increases by adding an individual b which is not in S i.e., $D(S \cup b) > D(S)$ if $min_{a \in S}$ $d(a, b) > 0$. With respect to biodiversity this requirement assures that the process of increasing species richness also reflects itself through the diversity measure [114].

2. Twinning. Diversity stays constant in case that an individual c that is already in S is added, i.e., $D(S \cup c) = D(S)$. Ulrich at al. [252] has defined diversity as the coverage of a space by a set of solutions. According to this definition, the coverage will not increase when duplicates are added.

3. Monotonicity in Distance. The diversity of a set of solutions S does not decrease if all pairs of solutions are at least as dissimilar as previously discussed i.e., $D(S') \geq D(S)$, if $d(a'_i, a'_j) \geq d(a_i, a_j), \forall a_i, a_j \in S, a'_i, a'_j \in S'$. Note that dissimilar solutions are better.

2.5.4 Discrepancy Theory in Mathematics

Traditionally, diversity has been used to avoid premature convergence as discussed in Section 2.4 and it is generally assumed that crossover-based evolutionary algorithms require a diverse population in order to produce good results.

Discrepancy theory studies the *irregularity of distributions* in the following sense. Given a metric space S and some n points $s_1, \ldots, s_n \in S$, the discrepancy of the set $X := \{s_1, \ldots, s_n\}$ is measured as the largest deviation from a perfectly evenly distributed point set. When, as in our case, $S = [0, 1]^d$ is the d-dimensional unit cube, we can measure the discrepancy with respect to all axis-parallel boxes $[a, b] := [a_1, b_1] \times \ldots \times [a_d, b_d]$. In an ideal situation, we would like the number of points of X that are inside such a box $[a, b]$ to be proportional to its volume. In other words, we

would like the difference $\mathrm{Vol}([a,b]) - |X \cap [a,b]|/n$ to be as small as possible for all possible boxes $[a,b]$. The discrepancy is set to be the largest deviation; i.e.,

$$D(X,\mathcal{B}) := \sup\{\mathrm{Vol}([a,b]) - |X \cap [a,b]|/n \mid a \leq b \in [0,1]^d\},$$

where we abbreviate $a \leq b$ if and only if for every component $i \in d$ the inequality $a_i \leq b_i$ holds. The smaller the discrepancy of a point set, the more regular its distribution with respect to all axis-parallel boxes will be.

2.5.5 Advantages of Using Discrepancy

Discrepancy theory plays an important role in numerical integration, where (under certain circumstances), low discrepancy point sets are known to provide very good estimates for the integral of an unknown or difficult-to-analyze function. Classical Monte Carlo integration is therefore often replaced by a so-called Quasi-Monte Carlo integration, which uses low discrepancy point sets instead of purely random ones, cf. [173] for an illustrated introduction to discrepancy theory.

In the context of evolutionary computation, low discrepancy points sets such as Sobol and Halton sequences [93, 235] have been used in the sampling routines of evolution strategies [9, 229], CMA-ES variants [243–245], and other genetic algorithms [131, 132]. Low discrepancy point sets are able to increase the efficiency of the sampling, which consequently becomes more effective than pure random sampling. On the other hand, evolutionary algorithms have been used to compute point sets of low discrepancy values [45, 215], an optimization problem not admissible by traditional analytical approaches. Finally, randomized search heuristics also play a crucial role in the computation of discrepancy values of point sets in high dimensions [81].

2.5.6 Star Discrepancy

The arguably, the most intensively studied discrepancy notion is the so-called *star discrepancy*, which measures the regularity with respect to all axis-parallel boxes $[0,b]$, $b \in [0,1]^d$ that are anchored in the origin (see Figure 2.3). This is also the measure for which Sobol and Halton sequences have been designed for. We use this star discrepancy measure to evaluate how evenly the points are distributed.

At first sight, one might conjecture that a regular $\sqrt{n} \times \sqrt{n}$ grid has a good and regular distribution. Its star discrepancy, however, is rather large: we easily convince ourselves there are boxes of volume $1/\sqrt{n}$ which do not contain any point, so that the

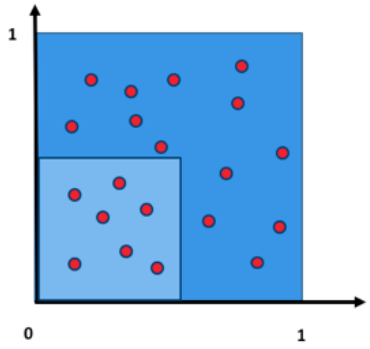


FIGURE 2.3: Illustration of discrepancy in $S = [0, 1]^2$.

star discrepancy is of at least this order. Random point sets also achieve a discrepancy value of order $1/\sqrt{n}$ only. In contrast, the low-discrepancy sequences mentioned above achieve a discrepancy value of order $\log^{d-1}/n$, and are thus much more evenly distributed with in terms of discrepancy [36, 55, 173, 202].

Apart from numerical integration and the mentioned applications in evolutionary computation, low discrepancy sequences also play an important role in statistics, computer graphics, and stochastic programming.

2.5.7 Multi-objective Optimization Indicators for Diversity Optimization

One of the most prominent areas of evolutionary computation where a diverse set of solutions is sought is evolutionary multi-objective optimization [31]. Evolutionary algorithms have been widely applied to multi-objective optimization problems and it is one of the key success areas for applying evolutionary algorithms. Over the years, many evolutionary multi-objective algorithms have been developed and making them applicable to the area of evolutionary diversity optimization provides huge potential for high performing evolutionary diversity optimization approaches.

In this section, we present three quality indicators evaluating the quality of a given set of objective vectors S. For a given set of search points P (called the population) and a function $g\colon X \to \mathbb{R}^d$, we define $S = \{g(x) \mid x \in P\}$ as the set of objective vectors of P.

2.5.7.1 Hypervolume

The hypervolume (HYP) is the volume covered by the set of objective vectors S with respect to a given reference point r. The hypervolume indicator measures the volume

of the dominated space of all solutions contained in a set $S \subseteq \mathbb{R}^d$. This space is measured with respect to a given reference point $r = (r_1, r_2, \ldots, r_d)$. The hypervolume $HYP(S, r)$ of a given set of objective vectors S with respect to r is then defined as

$$HYP(S, r) = VOL\left(\cup_{(s_1,\ldots,s_d)\in S} [r_1, s_1] \times \cdots [r_d, s_d]\right),$$

with $VOL(\cdot)$ being the Lebesgue measure [68]. Moreover, under the assumption of the exponential time hypothesis, which state that there is no algorithm with runtime $n^{o(d)}$ HYP can be solved on average in time $O(d^{d^{2/2}}n+dn^2)$. HYP performs better on average than the worst-case scenario $n^{\Omega(d)}$ [19]. Note the points are distributed at random on a d-dimensional simplex.

2.5.7.2 Inverted Generational Distance

The inverted generational distance (IGD) measures S with respect to a given reference set R. It calculates the average distance of objective vectors in R to their closest points in S. In general, IGD assesses the quality of approximations to the Pareto front accomplish/perform by multi-objective optimization algorithms [14, 260, 278]. We have

$$IGD(R, S) = \frac{1}{|R|} \sum_{r\in R} \min_{s\in S} d(r, s),$$

where $d(r, s)$ is the Euclidean distance between r and s in the objective space.

2.5.7.3 Additive Epsilon Approximation

The additive epsilon approximation (EPS) measures the approximation quality of the worst approximated point in R that S achieves. For finite sets $S, R \subset \mathbb{R}^d$, the additive approximation of S with respect to R (assuming all objectives are to be minimized) is defined as

$$\alpha(R, S) := \max_{r\in R} \min_{s\in S} \max_{1\leq i\leq d} (s_i - r_i).$$

To get a sensitive indicator that can be used to guide the search, we consider instead the set $\{\alpha(\{r\}, S) \mid r \in R\}$ of all approximations of the points in R. We sort this set decreasingly and call the resulting sequence $S_\alpha(R, S) := (\alpha_1, \ldots, \alpha_{|R|})$ (see [263]).

While other indicators could also be used for driving diversity optimization, we do not intend to highlight differences of the indicators (which has been subject to many papers). In Chapter 10 we will focus on demonstrating that they can in-fact be used as a tool out-of-the-box to explore the space of combinations of instance features.

2.6 Machine Learning

During the last few years machine learning played an important role in the field of image analysis and computer vision. There are many applications on the rise that require machine learning methods such as autonomous driving, virtual/augmented reality systems, navigation, image classification and generation, and video recommendation [35, 112, 123, 151, 204, 212, 271, 272, 274, 276].

Deep learning uses multiple neuronal network layers in order to represent the abstractions of data and to build computational models. Deep learning algorithms like Generative Adversarial Networks (GANs), Convolutional Neural Networks (CNNs), and model transfers have changed our perception of image processing. In recent years, Generative Adversarial Networks have become popular technique for unsupervised deep learning [130, 157, 162, 181, 227, 274].

2.6.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) were first introduced by Ian Goodfellow et al. in the seminal work [84] in 2014. GANs are a new framework for estimating generative models via an adversarial process. The main idea behind a GAN is to have two competing neural network models that are simultaneously trained. A generative model G captures the data distribution and generates samples whilst the discriminative model D estimates the probability that occurs when a sample is delivered from the training data. The discriminator D receives samples from both the generator and the training data and has to subsequently be able to distinguish between the two sources. This means that we train the discriminator D to maximize the probability of assigning the correct label.

This process refers to the the training examples in addition to the training samples from G [83, 205]. In a nutshell, D and G play the two-player minimax game with the value function $V(G, D)$:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))], \tag{2.1}$$

where, $D(\boldsymbol{x})$ represents the probability that $\boldsymbol{x}$ came from the data rather than the generator's distribution p_g. We train D to maximize the probability of assigning the correct label to both: real images from training examples and fake samples samples from G. Furthermore, we train G to minimize $\log(1 - D(G(\boldsymbol{z})))$ [57, 84, 102, 205].

Chapter 3

Evolutionary Art and Image Features

In this chapter, we provide a brief introduction of existing evolutionary approaches for the generation of images. Evolutionary computing methods have been widely applied during the last 40 years in the area of digital art. We outline various approaches and frameworks investigating the evolutionary process for creating artworks in Section 3.1. Afterwards in Section 3.2, we describe an evolutionary image transition process where an arbitrary image can be transformed into to a given target image. The goal of this transition is to produce new images and video sequences and to study how different operators can influence the appearance of images. In Section 3.3, we summarize several general and aesthetic features used in our experiments that described in the rest of this thesis. In Section 3.4, we finish with a description of a region covariance descriptor that is able to capture essential characteristic of classes of images.

3.1 Introduction

Evolutionary algorithms have been used in a wide range of application domains to provide novel solutions to various problems. This includes the classical task of computing high performing solutions for combinatorial optimisation problems, new designs in the area of engineering, as well as creative solutions in the areas of music and art [5, 147, 175, 221]. In the research area of evolutionary algorithms and art, the primary aim is to evolve artistic and creative outputs through an evolutionary process [2, 87, 164, 165, 169, 224, 232, 248]. The use of evolution for the composition of images has gained some attention in literature and mainly focuses on using textures. In these works, genetic programming is used to synthesise procedural texture formulas [224]. Furthermore, genetic programming has been used to evolve functions constructing images based on aesthetic features [98, 99].

In the seminal work the *Blind Watchmaker*, Dawkins [30] investigated the process of evolution. He evolved figures called biomorphs in order to explore and to visualize the power of evolution. The biomorphs' appearance changes/diverges broadly from the original parent over time demonstrating a simple model of bio-inspired evolution. Inspired by Dawkins' work, Sims [232] created complex and aesthetically pleasing images. He used an expression-based approach to evolve images using a mathematical expression as the genotype and applied crossover and mutation. In the interactive media installation *Galapagos*, visitors were able to evolve virtual creatures based on Darwinian evolution principles whilst taking into account their own preferences [233]. The abstract organisms were displayed on computer screens and users interactively chose virtual organisms for simulated growth by stepping onto selected pads. This exhibition was a visualization example of a collaboration process between the visitors and the machine.

Latham [149] created a black and white lithographic artwork called *Black Form Synth* consisting of hand-drawn "evolutionary trees of complex forms" in the form of a flow diagram arranged with a set of transformation rules. Following the expression-based approach, Latham [150] extended the rule-based evolutionary approach in order to invent new complex 3D forms from geometric primitives. Todd and Latham [247] introduced the framework called *Mutator* to generate art and evolve new 3D biomorphic forms. The *Mutator* creates a series of complex branching organic forms and surreal virtual sculptures, and animated videos through the process of "surreal" evolution. At each iteration the artist selects phenotypes based on the idea of evolved organisms that are able to "breed and grow".

Over time, the growing interest in an expression-based approach led to expansion of the area of evolutionary art, providing coherent and solid basis for further research. Hart [96] used an expression-based approach with the focus of evolving a set of images with a different appearance to the previously works. This system's interface allowed a greater range of control over the colors and forms of the evolved images. Complex and detailed images were created in [223] using an expression-based approach guided by aesthetic selection, in particular towards the evolution of color space.

Unemi [254, 256] continued to explore the expression-based approach to evolutionary art by evolving images and animations as the progression of color volumes and novel geometric forms with varied variables. The images are evolved based on aesthetic measures and have been displayed on the internet for decades using different versions of the SBART framework [255]. Machado and Cardoso [164] introduced

computer-aided software, called *NEvAr*. This is an evolutionary art tool which uses: genetic programming, user-guided evolution, and automatic fitness assignment. The model uses a fitness function that permits aesthetically pleasing and visually complex images. Draves [54] introduced *Electric Sheep*, a large and ongoing evolutionary art project using collective human evaluation. It was implemented as a distributed screen-saver allowing users to approve or disapprove phenotypes in order to evolve artificial life and is focused on the continuing behavior of the distributed system. Greenfield [88] describes simulated robots embedded into an evolutionary framework for the purpose of painting a new artwork. The system implements optimization towards identifying robot paintings with higher aesthetic properties and takes into account the behavior of the simulated robots.

Aesthetic feature measures have been widely applied to generate new artistic images [154]. The evolutionary art system *NEvAr*, Machado and Cardoso [164] used aesthetic measures and evolutionary computation methods, and thus developed automatic seeding procedures to generate images. Greenfield [91] evolved images using computational aesthetic functions that are based on a color segmentation algorithm. Moreover, Heijer and Eiben [99] studied aesthetic measures in unsupervised evolutionary art using aesthetic measures as fitness functions.

In *Interactive Evolutionary Computation* (IEC), the traditional objective-function is replaced by a human subject who guides the selection [242]. In IEC, each solution is evaluated by a human judgment. Those subjective choice provides the basis for the selection of solutions in the evolutionary process. IEC founds its biggest application in the creative domains in which including humans 'in the loop' can lead to novel solutions [62, 104, 222, 242]. In this case, the fitness function is based on the individual user's experiences and preferences towards interesting or aesthetic results. This means that the fitness of the evolutionary algorithm (EA) is strongly subjective.

In contrast to this, a different approaches evaluate artistic images by automate fitness evaluation in order to generate novel images. Baluja, Pomerleau, and Jochem [10] attempted to automate the process of image evolution by using neural networks in order to produce images. The primary idea was to learn the user's preferences, and to apply this insight to generate aesthetically pleasing images.

Moreover, Heijer and Eiben [99] presented an approach for evolving images without human interaction. Instead of human interaction, they used one or more aesthetic measures as fitness functions to guide the search. They investigated the correlation between aesthetic scores in order to calculate which aesthetic measure generate images that are evaluated positively by the other aesthetic measures.

More recently, deep neural networks have been used to create artistic images through the transfer of artistic style from one image to another, facilitating novel forms of image manipulation [75]. A new neural style transfer approach preserves the colors of the original image using simple linear methods for the transferring style [79]. Gatys et al. [77] extended the existing method by introducing control over colour information, spatial scale, and spatial location.

Johnson et al. [118] combined the benefits of both approaches, trained feed-forward convolutional neural networks and perceptual loss functions based on high-level features extracted from pretrained networks and proposed the use of perceptual loss functions for training feed-forward networks for image transformation. The transfer of style from one image to stable video sequences was proposed by Ruder et al. [226] by the application of new initialization and loss functions applicable to videos.

Non-photorealistic rendering (NPR) is an area of computer graphics that focuses on enabling a wide variety of expressive styles for digital art [240]. In contrast to traditional computer graphics, which has focused on photorealism, NPR is inspired by artistic styles such as painting, drawing, technical illustration, and animated cartoons.

Litwinowicz [156] described transformations from ordinary video segments into animations that appear similar to hand-painted techniques. In particular, they used modified off-the-shelf image processing and rendering techniques in order to process images and videos for an impressionist effect. Kubelka [145] introduced the optical properties model in order to simulate the optical effect of different layers of artwork. Hertzmann [101] generated hand-painted images by using series of different brush stroke sizes and orientations to generate images. In recent years, Barile, Ciesielski, and Trist [11], Izadi, Ciesielski, and Berry [113], Kang et al. [121], and Machado and Pereira [166] introduced evolutionary approaches for the production of non-photorealistic renderings of images.

Furthermore, extensive research has been carried out on swarm painting. Urbano [257] investigated choices inside a group of agents in order to design swarm art with interesting random patterns. Monmarché, Slimane, and Venturini [182] used the stochastic and exploratory principles of an ant colony to automatically discover clusters in data without prior knowledge of the structure of the data.

A swarm-based system was used as a method to create visualizations of data, and in order to combine information aesthetics with data visualization [163]. Boyd, Hushlak, and Jacob [18] created a collaborative project called *Swarm Art* that incorporated the swarm-based simulation and projected the artwork onto a large screen. Greenfield [90] used a colony optimization model to evolve ant paintings and investigated

the effects of different fitness measures on the generation of different artistic styles. Inspired by natural phenomena, namely the use of pheromone substances for mass recruitment in ants' pattern behaviour, Urbano [258] generated collective artistic work by embedding a pheromone medium pattern into the painters' behaviour.

3.2 Evolutionary Image Transition

Evolutionary algorithms have been used for image transition in various ways. Earlier work by Sims [232] described methods for cross-dissolving of images by changes in an expression genotype. This approach employs evolutionary techniques in order to create complex organic structures, textures, and motions that can be applied in computer graphics and animation. Moreover, 3D plant structures can be evolved using fixed sets of genetic parameters. Sims has argued that this process of artificial evolution could be utilised for supporting user through creative explorations in larger search space, and help us gain knowledge about human aesthetics.

Graf and Banzhaf [86] used interactive evolution to help determine parameters for image morphing. They introduced systems of selection and variation based on recombination and/or mutation operators. In their work, the interactive evolution can be used to control the development of the most preferred designs in various application areas such as product visualisation of several vehicles types, engineering components and/or construction projects, interactive plotting computer aided design, drafting and manufacturing, simulation and animation for scientific visualisation, architecture, etc. By combining interactive evolutionary computation with the concept of warping and morphing from computer graphics they were able to evolve images for computer graphics and animation. Specifically, they used the recombination of two $2D$ and $3D$ bitmap images through image interpolation. They showed that artificial evolution has the system's ability to achieve flexibility and complexity in image design. Moreover, the system requires only a moderate amount of input from the user.

Furthermore, Karungaru et al. [124] used an evolutionary algorithm to automatically identify features for morphing face images. The method enables one to morph two face images by automatically detecting five control points which are extracted from the source and target image based on facial features. The evolutionary algorithm along with the face detection neural network extracted the facial features that are associated with size and orientation changes.

The evolutionary algorithms and the morphing methods are applied by Wong et al. [269] to evolve attractive face images. Human subjects scored each faces image

from randomly selected images on an attractiveness scale. The aim of this approach is to create the most attractive synthetic faces and to explore the correlation between attractiveness scores and morphometric measurements.

3.3 Features

In this section, we describe general and aesthetic features used in our experiments [237, 270]. We employ single features in several experiments, and subsequently examine the combinations of two and three features (see Chapters 8 to 11). The features are briefly defined as follows. Note that a more detailed descriptions can be found in the original papers.

3.3.1 General Features

The general features we use in our experiments are *Mean-hue*, *Standard-deviation-hue*, *Mean-saturation*, and *Smoothness*. We describe each in turn. *Mean-hue*, (Hue), is the mean value of the hue of all pixels of image X. The range of *Hue* is $[0, 1]$ with both 0 and 1 corresponding to the colour red. *Standard-deviation-hue*, $(SDHue)$, is the mean standard-deviation of the hue of all pixels of image X and is a measure of the degree of diversity in the image's hue [122]. *Mean-saturation*, *(Saturation)*, is the mean value of saturation for all pixels of image X and is the degree of dominance of hue mixed in the images color [122]. The range is $[0, 1]$ with 0 representing low saturation and 1 representing high saturation. Finally, *Smoothness* measures image smoothness. $Smoothness(X)$ is calculated for an image X with N pixels as:

$$1 - \sum_{i=1}^{N} \sum_{c=1}^{3} gradient(X_{ic})/3N,$$

where *gradient* is the gradient magnitude image produced by MATLAB's *intermediate* image gradient method which calculates gradients between adjoining pixel values on each colour channel. Source images which have many sharp edges will have brighter gradient images than those with smooth graduations of colour. Thus, $Smoothness(X)$, is simply the mean intensity of the gradient image of X subtracted from 1.0.

3.3.2 Aesthetic Features

The set of aesthetic features we use are, in order of appearance, *Benford's Law* [119], *Global Contrast Factor* [172], *Information Theory* [219], *Ross Bell-Curve* [224], *Colorfulness* [97], and *Reflectional Symmetry* [99]. We describe each of the following aesthetic

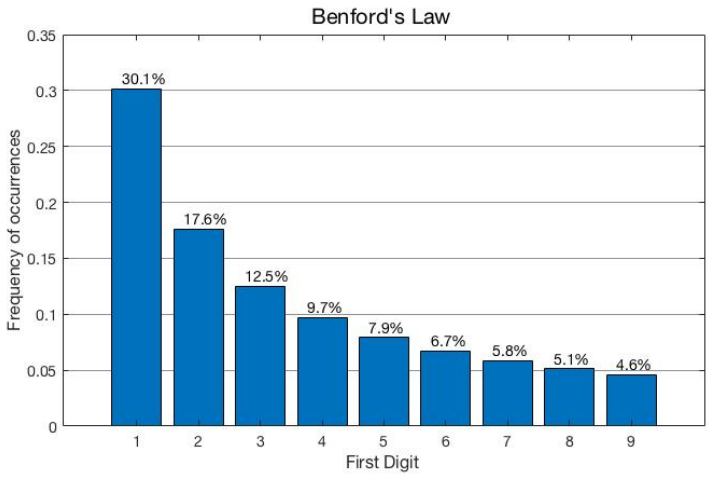


FIGURE 3.1: Histogram of first digits vs. the Benford's distribution.

features.

The *Benford's Law*, (*Ben*) feature is a measure of naturalness in an image X. Jolion et al. [119] observed that the sorted histogram of luminosities in natural images followed the shape of Benford's Law distribution of first digits (see Figure 3.1). Here we use the encoding of the Benford's Law feature based on the one used by den Heijer and Eiben [99]. To calculate $Ben(X)$ we first calculate a nine-bin histogram H_X of the luminosities of X. The bins of H_X are then sorted by frequency and scaled to sum to 1.0. We define

$$Ben(X) = 1 - d_{total}/d_{max},$$

where

$$d_{total} = \sum_{i=1}^{9} H_X(i) - H_{benford}(i),$$

and $H_{benford}$ is a 9-bin histogram, encoding Benford's Law distribution, with the bin frequencies $0.301, 0.176, 0.125, 0.097, 0.079, 0.067, 0.058, 0.051, 0.046$. The value

$$d_{max} = (1 - H_{benford}(1)) + \sum_{i=2}^{9} H_{benford}(i)$$

is the maximum possible value for d_{total} which is the largest possible deviation of H_X from $H_{benford}$.

Global Contrast Factor, (*GCF*) is a measure of mean contrast between neighbouring

pixels at different image resolutions. To calculate *GCF*(X) we calculate the local contrast at each pixel at a given resolution

$$r : lc_r(X_{ij}) = \sum_{X_{kl} \in N(X_{ij})} |lum(X_{kl}) - lum(X_{ij})|,$$

where $lum(P)$ is the perceptual luminosity of pixel P and $N(X_{ij})$ are the four neighbouring pixels of X_{ij} at resolution r. The mean local contrast at the current resolution is defined

$$C_r = (\sum_{i=1}^{m} \sum_{j=1}^{n} lc_r(X_{ij}))/(mn).$$

From these local contrasts, *GCF* is calculated as

$$GCF = \sum_{r=1}^{9} w_r \cdot C_r.$$

The pixel resolutions correspond to different *superpixel* sizes of 1, 2, 4, 8, 16, 25, 50, 100 and 200. Each superpixel is set to the average luminosity of the pixel's it contains. The w_r are empirically derived weights of resolutions introduced by Matkovic et al. [172] giving highest weight to moderate resolutions. In order to illustrate the differences obtained by *GCF* measures, we present in Figure 3.2 exemplary two images with corresponding values 0.0133 and 0.0231 (from left).

Colorfulness, (*Color*) is a measure of the perceived variety of color in of an image X. We use simplified metric introduced by Hasler and Suesstrunk [97] for calculating colorfulness. This measure quantifies spreads and intensities of opponent colors by calculating for the RGB values in each pixel X_{ij} the red-green difference: $rg_{ij} = |R_{ij} - G_{ij}|$, and the yellow-blue difference: $yb_{ij} = |(R_{ij} + G_{ij})/2 - B_{ij}|$. The means: μ_{rg}, μ_{yb} and standard-deviations: σ_{rg}, σ_{yb} for these differences are then combined to form a weighted magnitude estimate for colorfulness for the whole image:

$$Color(X) = \sqrt{\sigma_{rg}^2 + \sigma_{yb}^2} + 0.3\sqrt{\mu_{rg}^2 + \mu_{yb}^2}.$$

Information Theory, (*Info*) is the Information Theory metric from Rigau et al. [219] as described in [99]. *Info*(X) calculates the Shannon entropy of the intensity the 256-bin intensity histogram for the image X. It is defined:

$$Info(X) = -\sum_{i=0}^{N} p(x_i) \times \log(p(x_i)),$$

FIGURE 3.2: Images with different GCF values, 0.0133 and 0.0231 (from left).

where $p(x_i)$ is the probability of pixel intensity $x_i \in \{0, \ldots, 255\}$ in image X. This is calculated by dividing the count of pixels in bin x_i by N – the total number of pixels in image X.

Ross-Ralph-Zong Bell-Curve, (*Bell*) measures the conformance of the distribution of colour gradients of an image X to normal distribution. This feature stems from the observation by Ross et al. [224] that colour gradients in fine-art images tend to conform to normal distributions. To calculate *Bell* we first calculate the colour gradient $S_{i,j}$ for each colour channel at pixel $X_{i,j}$:

$$R_{i,j} = \sqrt{|\nabla r_{i,j}|^2 + |\nabla g_{i,j}|^2 + |\nabla b_{i,j}|^2}/2,$$

where the gradient $|\nabla c_{i,j}|^2$ at each colour channel c is:

$$|\nabla c_{i,j}|^2 = (c_{i,j} - c_{i+1,j+1})^2 + (c_{i+1,j} - c_{j,j+1})^2)/sf^2,$$

where sf^2 is a scaling factor used to allow fair comparisons of images of different size. Once the R_{ij} are calculated they are formed into histogram and *Bell* is calculated as the deviation from normality of this histogram. The details of this calculation are given in [224]. Note, that because the result returned is a deviation from normality, *low* values of *Bell*(X) should indicate an aesthetically pleasing picture.

Reflectional Symmetry, (*Symmetry*) is a measure based on that defined by den Heijer et al. [99] to measure the extent to which the image reflects itself, horizontally, vertically and diagonally. *Symmetry* divides an image X into four quadrants as shown in Figure 3.3. *Symmetry* is defined for image I as:

$$Symmetry(X) = S_h(X) + S_v(X) + S_d(X)/3,$$

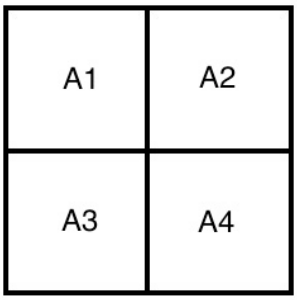


FIGURE 3.3: The four quadrants used by the Reflectional Symmetry aesthetic measure.

where $S_h(X)$, $S_v(X)$, and $S_d(X)$ are, respectively, measures of horizontal, vertical, and diagonal symmetry. Horizontal symmetry contrasts the left and right halves of the image: $S_h(X) = s(A1 \cup A3, A2 \cup A4)$. Vertical symmetry contrasts the top and bottom halves of the image: $S_v(X) = s(A1 \cup A2, A3 \cup A4)$. Diagonal symmetry compares diagonally opposite quadrants: $S_d(X) = (s(A1, A4) + s(A2, A3))/2$. The s function is a similarity operator between two quadrants defined as:

$$s(A_i, A_j) = \sum_{x=0}^{w} \sum_{y=0}^{h} |A_i(x, y) - \overline{A}_j(x, y)|)/w \times h \times 255,$$

where w is the width of the image, h is the height of the image, and $\overline{A}_j$ is the mirror image of A_j. Note that our definition of s is different from the measure in [99] in that it does not threshold corresponding pixel differences to 1 or 0 value. This more continuous response permits us to more easily evolve images in both directions for this feature.

3.4 Region Covariance Descriptor

In computer vision various image descriptors have been proposed and used to capture essential characteristics of particular classes of images. Amongst these the *region covariance descriptor*, introduced by Tuzel et al. [250], has proved particularly useful for various computer vision tasks including object tracking [209], pedestrian detection [249], action recognition [92], and medical imagining [129].

3.4.1 The Region Covariance Matrix

The region covariance descriptor is constructed based on a feature mapping that associates with each pixel in the image a finite-dimensional vector of numerical features

such as intensity, colour, gradients, or filter responses. Once a feature mapping is specified, any region in the image gives rise to the covariance matrix of the feature mapping restricted to this region. More formally, let $X = (X_{ij})$ be a colour image with a corresponding carrier $\Omega = \{(i,j) \mid 1 \leq i \leq m \text{ and } 1 \leq j \leq n\}$ of pixel locations. Each matrix entry X_{ij} is a triplet of integers representing colour information as coordinates in some colour space. Given a region of interest $\mathcal{R} \subset \Omega$ and a feature mapping $\phi\colon \Omega \to \mathbb{R}^p$, the corresponding *region covariance* matrix is given by

$$\Lambda_{\mathcal{R}} = \frac{1}{|R|-1} \sum_{(i,j)\in\mathcal{R}} (\phi(i,j) - \mu_{\mathcal{R}})(\phi(i,j) - \mu_{\mathcal{R}})^{\mathsf{T}},$$

where $\mu_{\mathcal{R}} = |\mathcal{R}|^{-1} \sum_{(i,j)\in R} \phi(i,j)$ and $|\mathcal{R}|$ denotes the number of pixels in the region of interest. The matrix $\Lambda_{\mathcal{R}}$ describes the variations of the length-n feature vectors $\phi(i,j)$ as (i,j) varies over the region $\mathcal{R}$. An example feature mapping, used for human detection, is

$$\phi(i,j) = \Big[i, j, I_{ij}, \left(\tfrac{\partial I}{\partial i}\right)_{ij}, \left(\tfrac{\partial I}{\partial j}\right)_{ij}, \left(\tfrac{\partial^2 I}{\partial i^2}\right)_{ij}, \left(\tfrac{\partial^2 I}{\partial j^2}\right)_{ij}, \sqrt{\left(\tfrac{\partial I}{\partial i}\right)^2_{ij} + \left(\tfrac{\partial I}{\partial j}\right)^2_{ij}}, \tan^{-1}\left(\left|\left(\tfrac{\partial I}{\partial i}\right)_{ij}\right| / \left|\left(\tfrac{\partial I}{\partial j}\right)_{ij}\right| \right) \Big]^{\mathsf{T}},$$

where I_{ij} is the image intensity at (i,j), $\left(\frac{\partial I}{\partial i}\right)_{ij}$, $\left(\frac{\partial^2 I}{\partial i^2}\right)_{ij}$, ... are intensity derivatives at (i,j), and the last two terms are the magnitude of edge response and the edge orientation at (i,j). In line with the convention adopted by MATLAB's `rgb2gray` function, the image intensity can be suitably defined by

$$I_{ij} = 0.2989 X^R_{ij} + 0.5870 X^G_{ij} + 0.1140 X^B_{ij},$$

where $X_{ij} = [X^R_{ij}, X^G_{ij}, X^B_{ij}]^{\mathsf{T}}$ is the decomposition of the colour vector X_{ij} into RGB components.

3.4.2 The Set of Features

We present a set of features and describe the corresponding description in Table 3.1. In Chapter 6, we will explore several candidate feature mappings obtained by selecting component elements from different features.

TABLE 3.1: Description of Potential Features for ϕ

	Notation	Description
ij	i	vertical spatial coordinate
	j	horizontal spatial coordinate
rgb	r	red channel
	g	green channel
	b	blue channel
∂	$\left\|\frac{\partial I}{\partial i}\right\|$	magnitude of first-order partial derivative in horizontal direction
	$\left\|\frac{\partial I}{\partial j}\right\|$	magnitude of first-order partial derivative in vertical direction
∂^2	$\left\|\frac{\partial^2 I}{\partial i^2}\right\|$	magnitude of second-order partial derivative in horizontal direction
	$\left\|\frac{\partial^2 I}{\partial j^2}\right\|$	magnitude of second-order partial derivative in vertical direction
	$\left\|\frac{\partial^2 I}{\partial i \partial j}\right\|$	magnitude of second-order mixed partial derivative
edge	$\sqrt{\left(\frac{\partial I}{\partial i}\right)^2 + \left(\frac{\partial I}{\partial j}\right)^2}$	magnitude of edge response
	$\tan^{-1}\left(\left\|\frac{\partial I}{\partial i}\right\| / \left\|\frac{\partial I}{\partial j}\right\|\right)$	edge orientation
hsv	h	hue (HSV colour space)
	s	saturation (HSV colour space)
	v	value (HSV colour space)

3.4.3 Influences of Using Covariance Matrices

There are several advantages of using covariance matrices as region descriptors. The feature mapping proposes a natural way of fusing multiple features which might be correlated. A single covariance matrix extracted from a region is usually enough to match the region in different views and poses. The noise that may corrupt individual samples is largely filtered out through averaging which is intrinsic to the process of covariance computation. The covariance descriptors are low dimensional—each matrix $\Lambda_{\mathcal{R}}$ has only $p(p+1)/2$ different entries (p is often less than 10), which is a number significantly smaller than the number of histogram bins (going into hundreds) or of raw pixels (going into thousands) used by other descriptors. Moreover, each $\Lambda_{\mathcal{R}}$ does not preserve information regarding the ordering and the number of underlying grid points. This implies a certain degree of scale and rotation invariance over the regions in different images. However, it should be noted that this near invariance is largely reduced if the feature mapping contains explicit information regarding the orientation of points, such as the gradient of image intensity. The same is true for illumination.

3.4.4 Distance Measures

Covariance matrices are positive-definite and one can speak about a distance between a pair of covariance matrices once a distance measure is defined between members of the set of all real positive-definite matrices. Let $Sym(p)$ denote the set of all $p \times p$ symmetric real matrices, and let $Sym_+(p)$ denote the subset of $Sym(p)$ comprised of all $p \times p$ positive-definite matrices in $Sym(p)$. $Sym_+(p)$ can be endowed with a variety of distance measures [116, 117]. In what follows we shall consider three specific distances. One is the *Euclidean* metric given by

$$\mathrm{dist}_{\mathrm{E}}(P, Q) = \|P - Q\|_{\mathrm{F}} \,,$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm, and P and Q are members of $Sym_+(p)$. Another is the *Log-Euclidean* metric given by

$$\mathrm{dist}_{\mathrm{L}}(P, Q) = \|\log P - \log Q\|_{\mathrm{F}} \,,$$

where $\log$ denotes the principal matrix logarithm [6]. (For an invertible real matrix without eigenvalues on the negative real axis, there always exists a unique real matrix logarithm, called the principal logarithm, whose eigenvalues lie in the strip $\{z \in \mathbb{C} \mid -\pi < \mathrm{Im} z < \pi\}$; cf. [103, Theorem 1.31].) Yet another distance measure is the *affine-invariant* metric given by

$$\mathrm{dist}_{\mathrm{A}}(P, Q) = \left\|\log(P^{-1}Q)\right\|_{\mathrm{F}} = \left\|\log\left(P^{-1/2}QP^{-1/2}\right)\right\|_{\mathrm{F}}$$

(cf. [148, Chap. XII]). The label "affine-invariant" reflects the fact that $\mathrm{dist}_{\mathrm{A}}$ is invariant under each mapping of the form $P \mapsto APA^{\mathsf{T}}$, where A is a real invertible matrix A; that is,

$$\mathrm{dist}_{\mathrm{A}}(P, Q) = \mathrm{dist}_{\mathrm{A}}(APA^{\mathsf{T}}, AQA^{\mathsf{T}})$$

for all $P, Q \in Sym_+(p)$ and all invertible $p \times p$ matrices A. The affine-invariant metric can alternatively be written as

$$\mathrm{dist}_{\mathrm{A}}(P, Q) = \left(\sum_{i=1}^{p} \log^2 \lambda_i(P^{-1}Q)\right)^{\frac{1}{2}}, \tag{3.1}$$

where $\lambda_i(P^{-1}Q)$, $1 \le i \le p$, are the eigenvalues of $P^{-1}Q$. As the matrix $P^{-1}Q$ is similar to the symmetric matrix $P^{-1/2}QP^{-1/2}$, the eigenvalues $\lambda_i(P^{-1}Q)$ are all positive and hence the right-hand side of (11.1) is well defined for all P and Q in $Sym_+(p)$.

Part II

Evolutionary Image Transition

Chapter 4

Evolutionary Image Transition and Painting Using Random Walks

4.1 Introduction

In this chapter, we focus on study how random walk algorithms can be used in the evolutionary image transition process. The key idea in this chapter is to use the evolutionary process *itself* in an artistic way. We consider the well-studied (1+1) EA, popular random walk algorithms and provide a new approach to evolutionary art by using theoretical approaches for evolutionary image transition.

The transition process consists of evolving a given starting image S into a given target image T through random decisions. Considering an error function which assigns to a given current image X a number of pixels, where these agrees with T and maximizes this function boils down to the classical ONEMAX problem for which numerous theoretical results on the runtime behaviour of evolutionary algorithms are available [115, 241, 268]. An important topic related to the theory of evolutionary algorithms are random walks [33, 160]. We consider random walks on images where each time the walk visits a pixel its value is set to the value of the given target image. By biasing the random walk towards pixels that are similar to the current pixel we can study the effect of such biases, which might be more interesting from an artistic perspective. After observing these two basic random processes for image transition, we study how they can be combined to give the evolutionary process interesting new properties. We study the effect of running random walks for short periods of time as part of a mutation operator in a (1+1) EA. Furthermore, we consider the effect of combining them with the asymmetric mutation operator for evolutionary image transition introduced in [189]. Our results show that the area of evolutionary image transition based on random walks provides a rich source of artistic possibilities for creating video art. All our approaches are pixel-based and the created videos are based on the evolutionary processes that

FIGURE 4.1: Starting image S (Yellow-Red-Blue, 1925 by Wassily Kandinsky) and target image T (Soft Hard, 1927 by Wassily Kandinsky).

show frames corresponding to images and were created every few hundred generations. After introducing these different approaches to evolutionary image transition based on random walks, we study their behaviour with respect to different aesthetic features. Feature-based analysis of heuristic search methods has gained increasing interest in recent years [176, 177, 187]. In other application areas feature-based analysis is an important method to increase the theoretical understanding algorithm performance and is particularly useful for algorithm selection and configuration [186, 197]. For evolutionary image transition, we study how artistic features behave during the transition process. This allows the measurement of the evolutionary image transition process in a quantitative way and provides a basis to compare our different approaches in respect to artistic measures.

In this chapter, we firstly investigate the impact of random walk lengths for image transition and the impact of the bias of the random walks controlled by the parameter α. Large values of α increase the probability of moving to similar pixel and different values of α lead to different image transition processes. Secondly, we investigate how random walk algorithms can be used to carry out the painting of images. The key idea is to use a biased random walk starting at a given pixel and then color all pixels visited by the walk with the color of the starting pixel. We use the biased random walk mutation approach introduced for evolutionary image transition and combine it with the asymmetric mutation operator. Our approach starts a biased random walk for each pixel that has to be changed by asymmetric mutation. To achieve different effects of evolutionary image painting, we consider the parameter α which allow us to control the bias of the random walk towards similar pixels. Our experimental results show the effect of the setting of α for images of various types.

The outline of this chapter is as follows. In Section 4.2, we introduce the evolutionary transition process. Section 4.3 gives results of how creative mutation operators, namely *stripes mutation, combined stripes mutation* and *block mutation* can be used in

Algorithm 3 Asymmetric mutation

- Obtain Y from X by flipping each pixel X_{ij} of X independently of the others with probability $c_s/(2|X|_S)$ if $X_{ij} = S_{ij}$, and flip X_{ij} with probability $c_t/(2|X|_T)$ if $X_{ij} = T_{ij}$, where $c_s \geq 1$ and $c_t \geq 1$ are constants, we consider $m = n$.

FIGURE 4.2: Image Transition using asymmetric mutation with $c_s = 100$ and $c_t = 50$ at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

the evolutionary image transition process. In Section 4.3.1 we investigate the use of *asymmetric mutation* as part of mutation operators and study their combinations with pixel-based mutations during the evolutionary process. Moreover, in Section 4.4 we study how variants of random walks can be used for the image transition process. We examine the use of random walks as part of mutation operators and study their combinations with asymmetric mutation during the evolutionary process in Section 4.5. Furthermore, we extend our investigation of the impact of different random walk lengths in biased random walk mutation. We examine the impact of the chosen α in biased random walk mutation. In Section 4.6, we analyse the different approaches for evolutionary image transition in respect to aesthetic features. Our evolutionary painting algorithm using biased random walks is introduced and evaluated in Section 4.7. Finally, we finish with some concluding remarks.

4.2 Evolutionary Image Transition

We now describe the evolutionary image transition process. It transforms a given image $S = (S_{ij})$ of size $m \times n$ into a given target image $T = (T_{ij})$ of size $m \times n$. This is done by producing images X for which $X_{ij} \in \{S_{ij}, T_{ij}\}$ holds. Given a starting image $S = (S_{ij})$, a target image $T = (T_{ij})$, and a current image $X = (X_{ij})$, we say that pixel X_{ij} is in state s if $X_{ij} = S_{ij}$, and X_{ij} is in state t if $X_{ij} = T_{ij}$. The goal is to study different ways of using random walk algorithms for evolutionary image transition.

Algorithm 4 (1+1) EA for evolutionary image transition

- Let S be the starting image and T be the target image.
- Set X:=S.
- Evaluate $f(X,T)$.
- while (not termination condition)
  - Obtain image Y from X by mutation.
  - Evaluate $f(Y,T)$
  - If $f(Y,T) \geq f(X,T)$, set $X := Y$.

Throughout this work, we assume that $S_{ij} \neq T_{ij}$ as pixels with $S_{ij} = T_{ij}$ can not change values and therefore do not have to be considered in the evolutionary process. To illustrate the effect of the different methods presented in this work, we consider the work Yellow-Red-Blue, 1925 by Wassily Kandinsky as the starting image and the work T Soft Hard, 1927 by Wassily Kandinsky as the target image (see Figure 4.1). In principle, this process can be carried out with any starting and target image. Using artistic images in this setting has the advantage that artistic properties of images are transformed during the evolutionary image transition process. We will later on in this work study how the different operators used in the algorithms influence artistic appearance in terms of different artistic features. At the beginning of the 20th century the well-known artist Wassily Kandinsky was part of the famous Bauhaus movement [56]. Kandinsky was a unique university teacher in Weimar and Dessau and an iconic promoter of a theory of geometric figures and their relationships. In the work "Point and Line to Plane", [120] show an innovative approach to artistic expression and to the creation of abstract paintings.

We use the fitness function for evolutionary image transition used in [189] and measure the fitness of an image X as the number of pixels where X and T agree. This fitness function is equivalent to the ONEMAX problem when interpreting the pixels of S as 0's and the pixels of T as 1's. Hence, the fitness of an image X with respect to the target image T is given by

$$f(X,T) = |\{X_{ij} \in X \mid X_{ij} = T_{ij}\}|.$$

We consider simple variants of the classical (1+1) EA in the context of image transition. The algorithm is using mutation only and accepts an offspring if it is at least

as good as its parent according to the fitness function. The approach is given in Algorithm 4. Using this algorithm has the advantage that parents and offspring do not differ too much from the number of pixels. This ensures a smooth process for transitioning the starting image into the target. Furthermore, we can interpret each step of the random walks flipping a visited pixel to the target as a mutation step which, according to the fitness function, is always accepted.

As the baseline mutation operator, we consider the asymmetric mutation operator which has been studied in the area of runtime analysis for special linear functions [115] as well as the minimum spanning tree problems [198]. Using this mutation operator instead of standard bit mutations for ONEMAX problems avoids the coupon collector's effect [135, 180]. In the transition process, the couponcollector's effect implies that the last (even small) fraction of the pixels that need to be flipped to the target need more time to be flipped than all the pixels previously set to the target. More precisely, flipping each bit with probability $1/n$ as done in standard bit mutations, implies that the waiting time to flip the last pixel to the target is $\Theta(n)$ whereas increasing the number of target pixels is $\Theta(1)$ if the number of target pixels is still a constant fraction of all the pixels. Such a slow-down at the end of the transition process where there is no progress for many iterations is not desirable.

We use the generalization of this asymmetric mutation operator already proposed in [189] and shown in Algorithm 3. Let $|X|_T$ be the number of pixels where X and T agree. Similarly, let $|X|_S$ be the number of pixels where X and S agree. Each pixel in starting state s is flipped with probability $c_s/(2|X|_S)$ and each pixel in target state t is flipped with probability $c_t/(2|X|_T)$. The special case of $c_s = c_t = 1$ has been mathematically analyzed with respect to the runtime behaviour on other pseudo-Boolean functions.

We set the parameters as follows: $c_s = 100$ and $c_t = 50$. This allows both a decent and sufficient speed for the image transition process and enough exchanges of pixels for an interesting evolutionary process. We should mention that obtaining the last pixels of the target image may take a long time compared to the other progress steps when using large values of c_t. However, for image transition, this only effects steps when there are at most $c_t/2$ source pixels remaining in the image. From a practical perspective, this means that the evolutionary process has almost converged towards the target image and setting the remaining missing target pixels to their target values provides an easy solution.

All experimental results for evolutionary image transition in this paper are shown for the process of moving from the starting image to the target image given in Figure 4.1

FIGURE 4.3: Starting image S (Kinkaku-Ji Temple Kyoto) and target image T (Daigo-ji Temple Kyoto).

where the images are of size 200×200 pixels. The algorithms have been implemented in MATLAB. In order to visualize the process, we show the images obtained when the evolutionary process reaches 12.5%, 37.5%, 62.5% and 87.5% pixels of target image for the first time. We should mention that all processes except the use of the biased random walks are independent of the starting and target image which implies that the use of other starting and target images would show the same effects in terms of the way that target pixels are displayed during the transition process.

In Figure 4.2 we show the experimental results of the asymmetric mutation approach as the baseline. On the first image from left we can see the starting image S with lightly stippling dots in randomly chosen areas of the target image T. Consequently, the area of the yellow dimensional abstract face disappears, and black abstract figure appears. Meanwhile the background has adopted a dot pattern, where a nuance of dark and light develops steadily. In the last image, we barely see the starting image S and the target image T appearing permanently with the background becoming a darker blue tone, whereby the stippling effect shown in the middle two frames decreases. Interesting images in respect to aesthetic and evolutionary creativity emerge for the pictures at 37.5% and 62.5% of the evolutionary processes. In the third picture, we can observe elements of both images compounded with a very special effect as a result of the image transition process.

4.3 Block and Stripes Mutation

In this section, we introduce specific mutation operators for image transition. The design of our three creative mutation operators, namely *stripes mutation*, *combined stripes mutation* and *block mutation*, is strongly oriented towards production of interesting artistic images. To illustrate the effect of the different methods presented in this section, we consider two images of famous Kyoto Temples (see Figure 4.3).

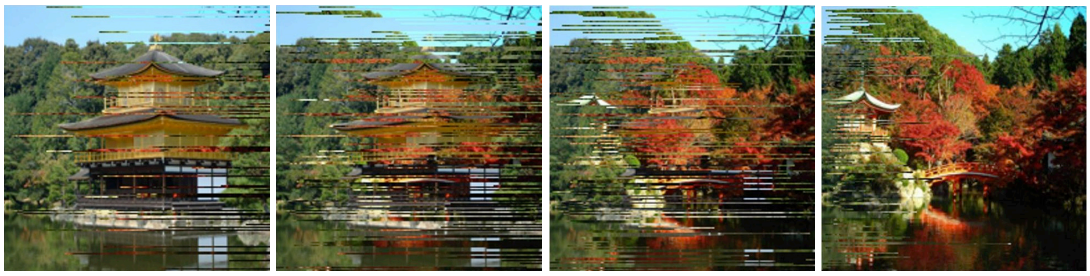


FIGURE 4.4: Image transition for stripes mutation at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

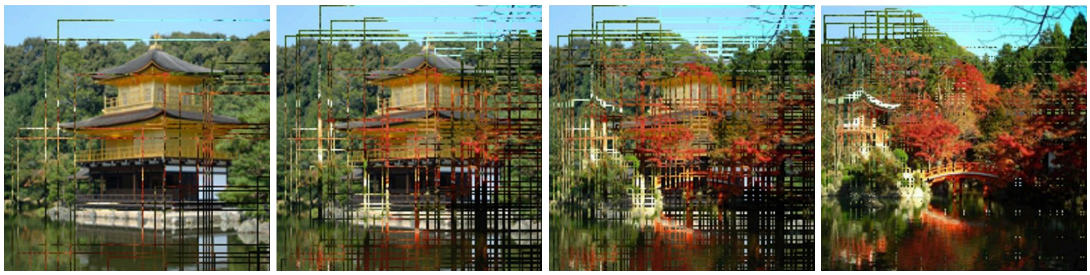


FIGURE 4.5: Image transition for combined stripes mutation at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

All of these mutation operators transition a region of the current image X to an area of the target image T, starting at randomly chosen pixel position X_{ij}. For our process we assume that $S_{ij} \neq T_{ij}$ as pixels with $S_{ij} = T_{ij}$ can not change values and therefore do not have to be considered in the evolutionary process. *Stripes mutation* mutates vertical stripes of pixels from X_{ij} for a length of 180 pixels or to the boundary of the image whichever comes first. *The stripes mutation* imitates well-known technique in generative art called a glitch. This effect intentionally corrupts data of an image by encoding the JPEG process. All experiments show that in each new generation the initial image and the target image overlay through a series of randomly placed stripes. In Figure 4.4 we show the experimental results of *stripes mutation*. The effect of *stripes mutation* creates generative art with highly interesting components. Furthermore, we have conceptualised a *combined stripes mutation* operator where both horizontal and vertical stripes gradually overlay the original image in random locations. The parameter settings for *the combined stripes mutation* are 200 pixels in width and 40 pixels in height for the horizontal stripes and one pixel in width and 200 pixels in height for the vertical orientation.

Our third mutation operator is *block mutation*. This operator randomly selects a position in the space of the matrix $S(i, j)$ and flips a block of 3×3 pixels. Because the position chosen for mutation can be any point X_{ij}, we can observe that mutated blocks can overlap. Figure 4.6 shows the experimental results of *block mutation*. We

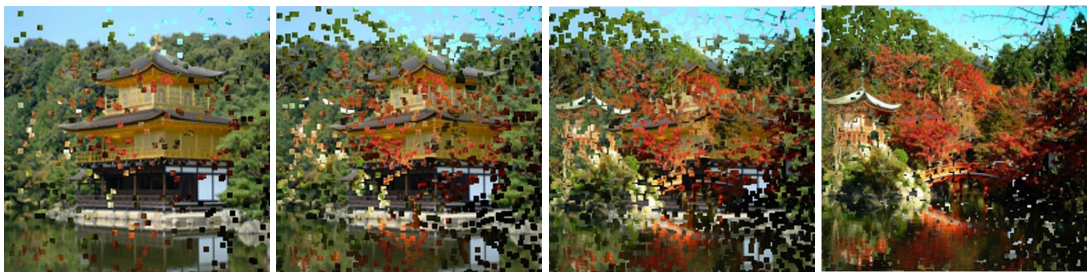


FIGURE 4.6: Image transition for block mutation at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

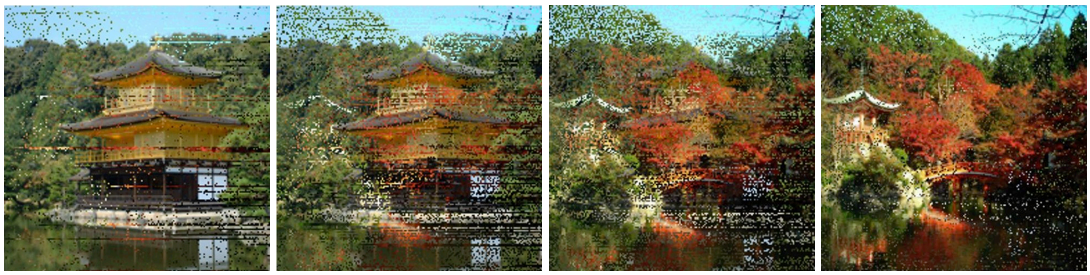


FIGURE 4.7: Image transition for combined asymmetric and stripes mutation at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

have executed a smooth transition so as to interest the viewer. Additionally, we have made the changes in the image clear visible.

4.3.1 Combined Approaches with Stripes and Block Mutation

The asymmetric, stripes, combined stripes and *block mutation* operators have quite distinct behaviours when applied to image transition. We now study the effect of combining the approaches for evolutionary image transition in order to obtain a more artistic evolutionary process. We explore the combination of *the asymmetric mutation* operator and *stripes, combined stripes* and *block mutation*. Here we run *the asymmetric mutation* operator as described in Algorithm 3. The process alternates between *asymmetric mutation* and one of either *stripes*, *combined stripes* or *block mutation*.

Figure 4.7 shows the results of the experiments for *combined asymmetric* and *stripes mutation*. The transitional images are significantly different to the images produced in *stripes mutation* or *block mutation*.

Interesting blurred elements randomly emerge over the image during the transition stage. Similar elements are shown as a characteristically steadily changing, whereas the elements of the target image continuously appear. On the last image we can almost completely see the details of the target image. The blurred elements seem to be created through the bright autumn colour in the target image in this stage of the sequence.

FIGURE 4.8: Image transition for combined asymmetric and combined stripes mutation at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

FIGURE 4.9: Image transition for combined asymmetric and block mutation at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

Figure 4.8 shows the results of the experiments for *combined asymmetric* and *combined stripes mutation*. The lower sequence of the picture is similar to the top in terms of the behaviour of the patches. More differences can be seen as the transition progresses, but the picture continues to be partially unrecognisable during the transition. New elements can be discovered on the affected image which slowly appear and create an element of surprise.

We see less of the dotted image pattern, but now observe combined blurred elements randomly occurring in the upper sequence of the images. The image provides an interesting comparison to the old linen pictures of Poussin and Rembrandt [100, 266]. In the advanced stages, the patches are well integrated into the images using *asymmetric mutation*. Finally, the last image we see is well integrated and shows more detail in all areas of the image.

Figure 4.9 shows the results of the experiments for *combined asymmetric* and *block mutation*. In the advanced stages, the patches are well integrated into the images using *asymmetric mutation* to produce a smooth transition process.

Algorithm 5 Uniform Random Walk

- Choose the starting pixel $X_{ij} \in X$ uniformly at random.
- Set $X_{ij} := T_{ij}$.
- while (not termination condition)
  - Choose $X_{kl} \in N(X_{ij})$ uniformly at random.
  - Set $i := k$, $j := l$ and $X_{ij} := T_{ij}$.
- Return X.

Algorithm 6 Biased Random Walk

- Choose the starting pixel $X_{ij} \in X$ uniformly at random.
- Set $X_{ij} := T_{ij}$.
- while (not termination condition)
  - Choose $X_{kl} \in N(X_{ij})$ according to probabilities $p(X_{kl})$.
  - Set $i := k$, $j := l$ and $X_{ij} := T_{ij}$.
- Return X.

4.4 Random Walks for Image Transition

Our evolutionary algorithms for image transition build on random walk algorithms and use them later on as part of a mutation step. Specifically, we investigate the use of random walk algorithms for image transition which move, at each step, from a current pixel X_{ij} to one pixel in its neighbourhood.

We utilize the neighbourhood $N(X_{ij})$ of X_{ij} following von Neumann's definition of neighbourhood [199] as

$$N(X_{ij}) = \{X_{(i-1)j}, X_{(i+1)j}, X_{i(j-1)}X_{i(j+1)}\},$$

where we work modulo the dimensions of the image in the case that the values leave the pixel ranges, $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$. This implies that from a current pixel, we can move up, down, left, or right. Furthermore, we wrap around when exceeding the boundary of the image. We do this in order to have a more interesting process. Moreover, it supports the effectiveness of our biased random walks in the case that

FIGURE 4.10: Image Transition for Uniform Random Walk (top) and Biased Random Walk (bottom) at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

they would be biased towards the image boundary. Not wrapping around in this case would imply that pixels opposite of the focused boundary can hardly be reached.

The classical random walk chooses an element $X_{kl} \in N(X_{ij})$ uniformly at random [208]. We call this the *uniform random walk* in the following. The cover time of the uniform random walk on a $n \times n$ torus is upper bounded by $4n^2(\log n)^2/\pi$ [33] which implies that the expected number of steps of the uniform random walk until the target image is obtained (assuming $m = n$) is upper bounded by $4n^2(\log n)^2/\pi)$.

4.4.1 Biased Random Walk

We also consider a *biased random walk* where the probability of choosing the element X_{kl} is dependent on the difference in RGB-values for T_{ij} and T_{kl}. We favour a neighbor $X_{kl} \in N(X_{ij})$ if T_{kl} is close to T_{ij} in respect to RGB-values. Weighted random walks have been used in a similar way in the context of image segmentation [85].

We denote by T_{ij}^r, $1 \leq r \leq 3$, the rth RGB value of T_{ij} and define

$$\gamma(X_{kl}) = \left(\max \left\{ \sum_{r=1}^{3} |T_{kl}^r - T_{ij}^r|, 1 \right\} \right)^{-\alpha}.$$

In our random walk, we want to prefer X_{kl} if $\gamma(X_{kl})$ is small compared to the other elements in $N(X_{ij})$. In order to compute the probability of moving to a new neighbour we consider $(1/\gamma(X_{kl})) \in [0, 1]$ and prefer elements in $N(X_{ij})$ where this value is large.

In the biased random walk, the probability of moving from X_{ij} to an element $X_{kl} \in N(X_{ij})$ is given by

$$p(X_{kl}) = \frac{(1/\gamma(X_{kl}))}{\sum_{X_{st} \in N(X_{ij})}(1/\gamma(X_{st}))}.$$

Our standard biased random walk works with $\alpha = 1$ and we extend our investigations to other choices of α as part of our random walk mutation operator in Section 4.3. Furthermore, we use different choices of α for image painting in Section 4.7.

The biased random walk is dependent on the target image when carrying out mutation or random walk steps and the importance of moving to a pixel with similar color. By introducing the bias in terms of pixels that are similar, the bias can cause the evolutionary image transition process to take exponentially long as the walk might encounter effects similar to the gambler's ruin process [180]. For our combined approaches described in the next section, we use the random walks as mutation components which ensures that the evolutionary image transition is carried out efficiently.

In Figure 4.10 we show the experimental results of the uniform random walk and biased random walk. At the beginning, we can observe the image with the characteristic random walk pathway appearing as a patch in the starting image S. Through the transition process, the clear recognisable patches on the target image T emerge. In the advanced stages, the darker patches from the background of the target image dominate. The effect in animation is that the source image is scratched away in a random fashion to reveal an underlying target image.

The four images of the biased random walk are clear different to the images of the uniform random walk. During the course of the transition, the difference between these processes becomes more prominent, especially in the background where at 87.5% pixels of the target image there is nearly an absolute transition to the target image T. In strong contrast, the darker abstract figure of the images stays nearly untouched, so that we see a layer of the yellow face in starting image S in the center of the abstract black figure in target image T. In this image, the figure itself is also very incomplete with much of the source picture showing through. These effects arise from biased probabilities in the random walk which makes it difficult for the walk to penetrate areas of high contrast to the current pixel location.

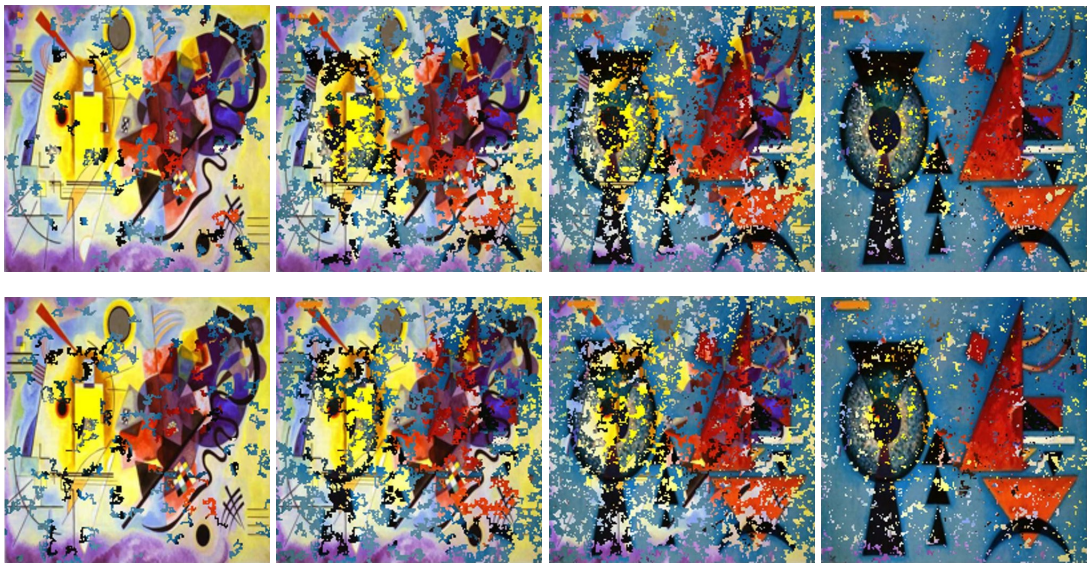


FIGURE 4.11: Image Transition for EA-UniformWalk (top) and EA-BiasedWalk (bottom) at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

4.5 Random Walk Mutation and Combined Approaches

The asymmetric mutation operator and the random walk algorithms display quite different behaviour when applied to image transition. We now study additional ways of carrying out mutations as part of the image transition process. Our goal is to obtain a new evolutionary process using mutations based on random walks and biased random walks. Furthermore, we investigate the effect of combining them with the asymmetric mutation operator.

4.5.1 Random Walk Mutation

Firstly, we explore the use of random walks as mutation operators and call this a *random walk mutation*.

The *uniform random walk mutation* selects the position of a pixel X_{ij} uniformly at random and runs the uniform random walk for $t_{\max}$ steps (iterations of the while-loop). We call the resulting algorithm EA-UniformWalk. Similarly, the *biased random walk mutation* selects the position of a pixel X_{ij} uniformly at random and runs the biased random walk for $t_{\max}$ steps. This algorithm is called EA-BiasedWalk. For our experiments, we set $t_{\max} = 100$.

Figure 4.11 shows the results of the experiments for EA-UniformWalk and EA-BiasedWalk. The transitions produce significantly different images compared to the previous ones. In both experiments we can see the target image emerging through a

FIGURE 4.12: Image Transition for EA-AsymUniformWalk (top) and EA-AsymBiasedWalk (bottom) at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

series of small patches at first, then steadily changing through a more chaotic phase where elements of the source and target image appear with roughly equal frequency. The last image of each experiment is most similar to the target image.

The images from EA-BiasedWalk appear visually similar to those of the EA-UniformWalk in the beginning but differences emerge during the final stages of the transition. In EA-BiasedWalk, elements of the source image still show through in areas of high contrast in the target image, which the biased random walk finds difficult to traverse. At a more local scale, this mirrors the effects of bias in the earlier random walk experiments. At a global scale, it can be seen that the blue background, which is a low contrast area, is slightly more complete in the final frame of EA-BiasedWalk than in the same frame in EA-UniformWalk.

4.5.2 Impact of Random Walk Length

Furthermore, we investigate the impact of the random walk length in biased random walk mutation. We want to explore how different lengths of biased random walk influence the creative process of evolving images. We also aim to understand the influence that the length of the biased random walk has on the results.

In general, we want to investigate the importance of different random walk lengths in the evolutionary process. Also want to understand how this parameter shapes the final result and how we are able to control those effects in a systematically manner. The

artistic techniques that appear from evolutionary image transition using different random walks can be compared to a French style of abstract painting, namely Tachisme (synonymous with art informel). Tachisme features the intuitive and spontaneous gestures of the artist's brushstrokes that involve the use of dabs, drips, or splotches of colour.

In our previous experiments described in Section 4.2, we used the random walk for evolutionary image transition and set the length of the random walk and the biased random walk to 100. Now, we investigate different choices of $t_{\max}$: small ($t_{\max} = 10, 50$), middle ($t_{\max} = 200, 400$) and large ($t_{\max} = 2000, 10000, 20000, 50000$). In Figure 4.13 we present the experimental results. In case of using different length of random walk, the special characteristic of biased random walk play most important role as the biased random walk has tendencies moving to the similar colors.

At the beginning, we observe that the use of parameter $t_{\max} = 10$ and $t_{\max} = 50$ results in images with smaller patches. In the later stages, the darker patches from background dominated the image. Apparently differences between image transition with small and medium $t_{\max}$ occurs in two stages. It can be observed that the small and medium random walk lengths produce various distinctive images.

When the process of the evolutionary image transition reaches 12.5% and 37.5% pixels of target image, we can observe more chaotic random walk patches that are present over image with tendency to form isolated structures. At the same stage of the evolutionary image processes, the images produced with medium $t_{\max}$ are less patchy and have a less irregular appearance. The reason for this is that the biased random walk build longer pads during the evolutionary transition.

Furthermore, we notice differences in image during the evolutionary processes when 62.5% and 87.5% pixels of target image occur. This time the image transition for biased random walk with medium $t_{\max}$ seems to have more patches incorporated into target image T. Thus, the image occurs more discompose. In the examples described above, we can assume that the different length of the biased random walk has a clear impact on the final image.

In Figure 4.14 we can recognize the experimental results of the biased random walk with length of 2000, 10000, 20000 and 50000, respectively. We can observe that the resulting images obtained with larger $t_{\max}$ are visual different from the images obtained with a smaller length of random walk.

At the beginning, when the process of the evolutionary image transition reaches 12.5% and 37.5% pixels of target images, we can observe only few longer random

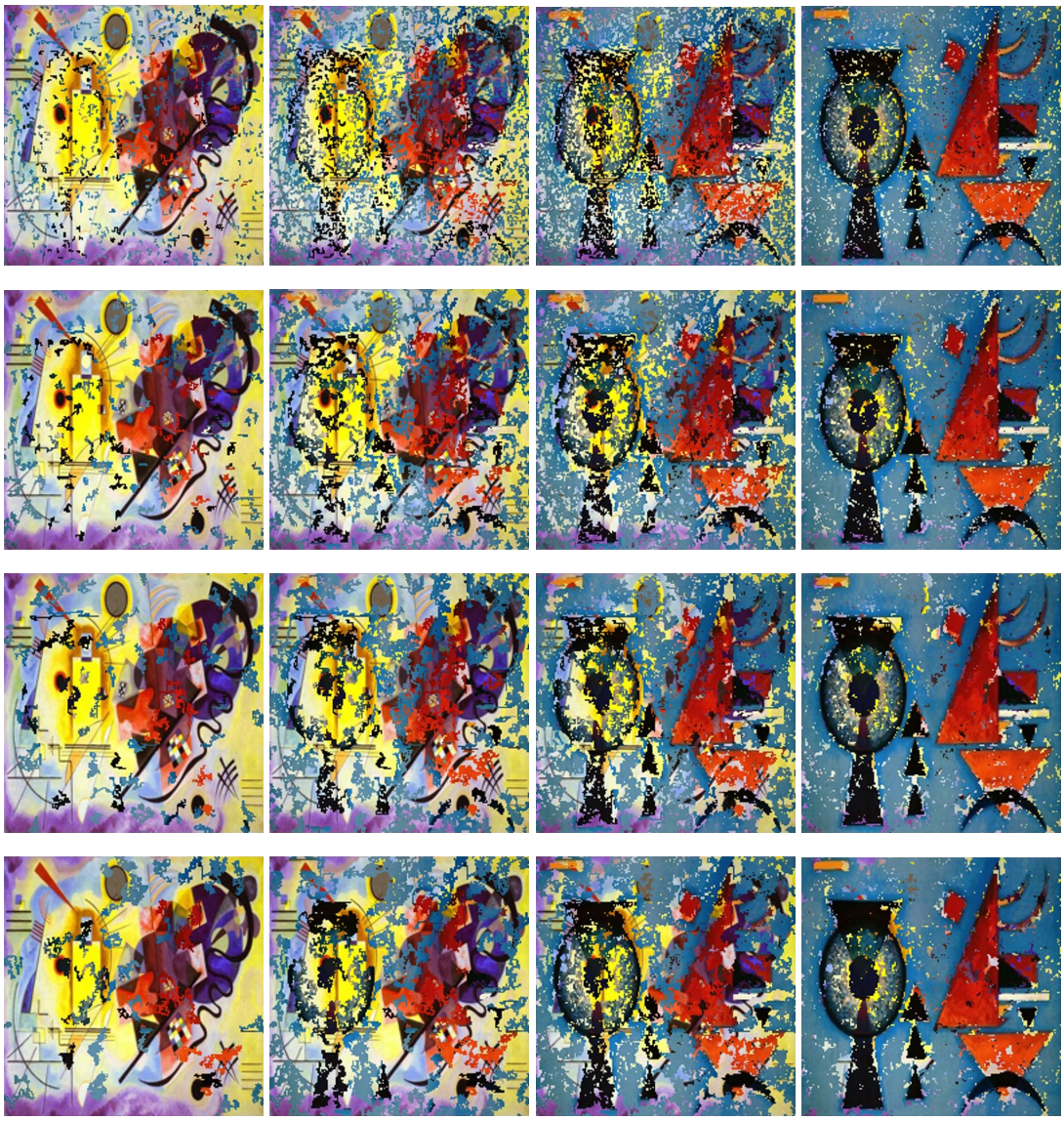


FIGURE 4.13: Image Transition for biased random walk with $t_{\max} = 10, 50, 200, 400$ (from top to bottom) and at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

walks. As consequence of this, we see start image S dominates during the evolutionary image transition. In the later stages, when 62.5% and 87.5% pixels of target image occur, we see the darker abstract figure is overall untouched. This is the result of the bias occurring during the transition with biased random walk mutation. In contrast to the experiments conducted with small and medium $t_{\max}$, the images, at this stage of evolution, are mostly completed with more recognizable elements from starting S and target T image. Employing biased random walk with a longer $t_{\max}$ into evolutionary image transition can create images that maintained content of both images without losing the style of the paintings.

FIGURE 4.14: Image Transition for EA-BiasedWalk with $t_{\max} = 2000, 10000, 20000, 50000$ (from top to bottom) and at 12.5%, 37.5%, 62.5% and 87.5% of the target image (from left to right).

4.5.3 Impact of the Choice of α in the Biased Random Walk

Additionally, we investigate the impact of the choice of the α in biased random walk mutation. We want to examine how the different α of biased random walk influence the generation of artistic images. We aim to understand the influence that the choice of different α has on the results.

In our previous experiments described in Section 4.2, we used the biased random walk for evolutionary image transition and set the α to 0.

From now on, we investigate different choices of α: small values ($\alpha = 0.25, 0.50$),

middle values ($\alpha = 0.75, 1.0, 1.25$) and large values ($\alpha = 1.5, 2.0$). In Figure 4.21, we observe the experimental results of our investigation.

The most differences we can noticed in images during the evolutionary processes when 62.5% pixels of target image occur. The evolutionary image transition for our algorithm with smaller value of α imply to have more patches incorporated into target image T. Thus, the image occurs more finished. In the examples described above, we can assume that the different values of α has predominantly influence in rather smaller degree with the chosen parameter $t_{\max}$ = 2000, on the final stage of the evolutionary image transition algorithm.

4.5.4 Combination of Asymmetric and Random Walk Mutation

Furthermore, we explore the combination of the asymmetric mutation operator and random walk mutation. Here, we run the asymmetric mutation operator as described in Algorithm 3 and a random walk mutation every τ generations. We explore two combinations, namely the combination of the asymmetric mutation operator with the uniform random walk mutation (leading to the algorithm EA-AsymUniformWalk) as well as the combination of the asymmetric mutation operator with the biased random walk mutation (leading to Algorithm EA-AsymBiasedWalk). We set $\tau = 1$ and $t_{\max} = 2000$ which means that the process is alternating between asymmetric mutation and random walk mutation where each random walk mutation carries out 2000 steps.

In Figure 4.12, we show the results of EA-AsymUniformWalk and EA-AsymBiasedWalk. From a visual perspective both experiments combine the stippled effect of the asymmetric mutation with the patches of the random walk. In EA-AsymBiasedWalk there is a lower tendency for patches generated by random walks to deviate into areas of high contrast. As the experiment progresses, the pixel transitions caused by the asymmetric mutation have a tendency to degrade contrast barriers.

However, even in the final frames there is clear more background from the target image in EA-AsymBiasedWalk than in EA-AsymUniformWalk. Moreover, there are more remaining patches of the source image near the edges of the base of the abstract figure, creating interesting effects.

As it can be observed through the experimental results, the user obtains a large variety of effects for image transition through the choice of the different parameters. Large random walk lengths in the mutation operator lead to large patches in the transition process whereas small values of $t_{\max}$ imply that the are many smaller patches widely distributed across the image. This implies that the image transitions appears more

evenly across the whole image. The combination with asymmetric mutation which flips single pixels allows for a further smoothening of the image transition process.

4.6 Feature-based Analysis

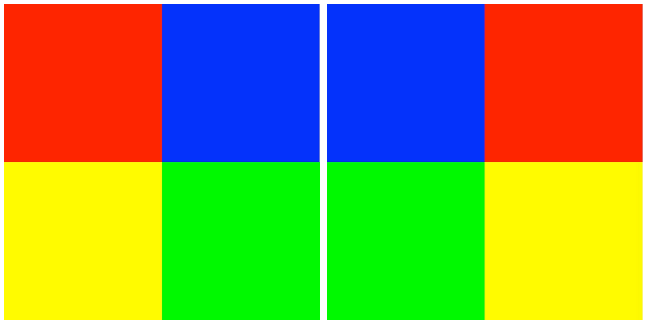
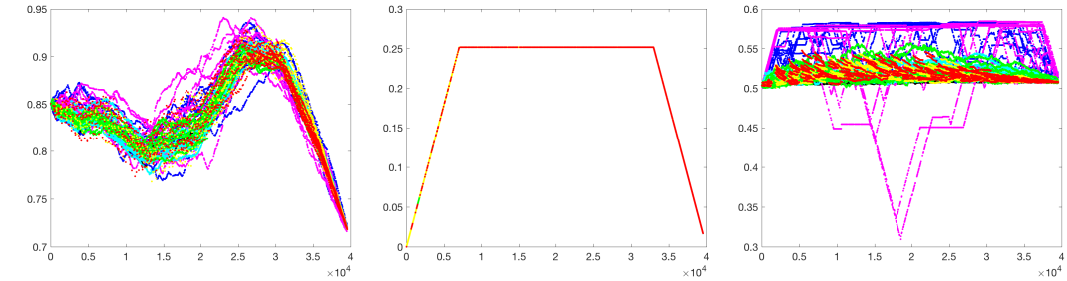


FIGURE 4.15: Starting image S (Color1) and target image T (Color2).

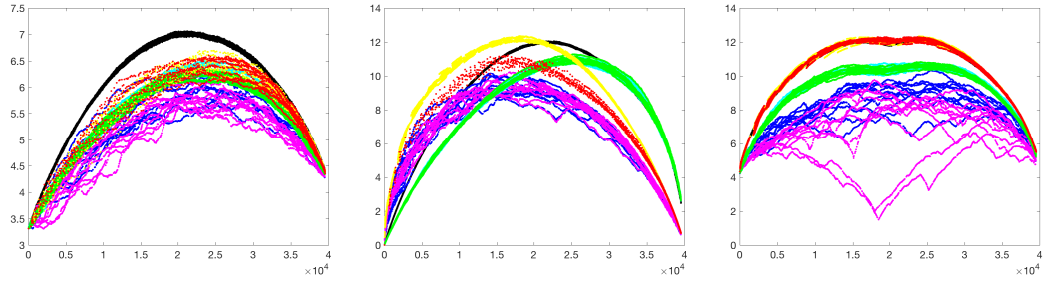
We now analyze the different introduced approaches for evolutionary image transition with respect to some features that measure aesthetic behaviour. Our goal is twofold. First, we analyze how the aesthetic feature values change during the process of transition. Furthermore, we compare the different approaches against each other and show where they differ with respect to the examined features when used for evolutionary image transition. For our investigations, we examine the starting and target image of Figure 4.1, the transition of a black starting image into a white target image, and the transition of the starting image Color1 into the target image Color2 as shown in Figure 4.15. Taking the last two pairs of images allows us to get additional systematic insights into the process of evolutionary image transition. Note that the images of Figure 4.15 are only swapping the colored squares.

The set of features we use are, in order of appearance, *Benford's Law* [119], *Global Contrast Factor* [172], *Hue*, and *Colorfulness* [97]. We describe each of them in the following.

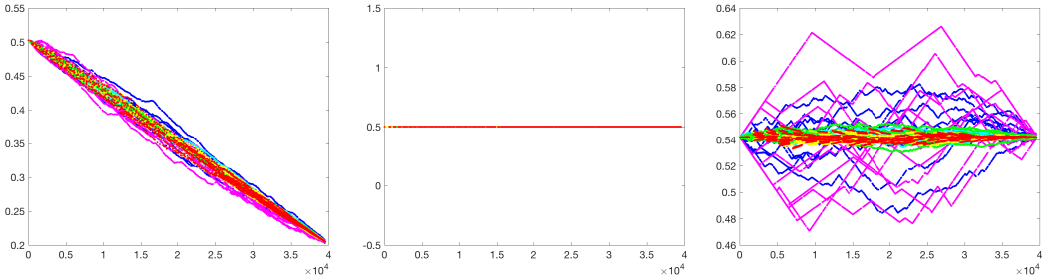
Figure 4.16 shows how the features evolve over time during the image transition process. The first column refers to the transition process of the starting and target image given in Figure 4.1. The second column shows the transition of a complete black image starting image to a complete white target image, and the third column shows the transition of the color starting image to the color target image of Figure 4.15. Each figure shows the results of 10 runs for each algorithm that we have considered for evolutionary image transition.



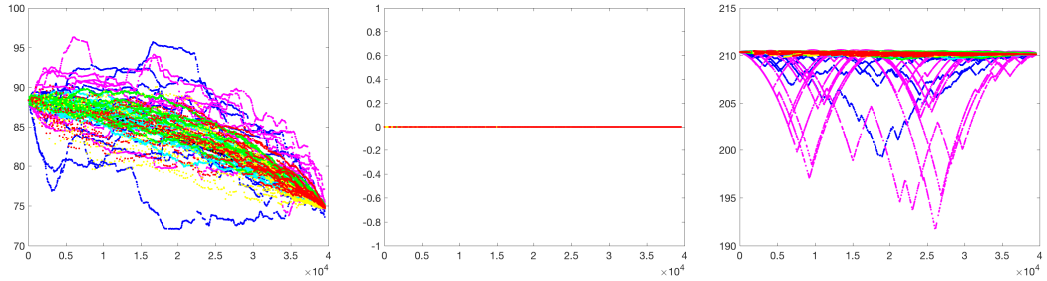
(A) $Benford'sLaw$



(B) $GlobalContrastFactor$



(C) Hue



(D) $Colorfulnessr$

FIGURE 4.16: Features during transition for images for Asymmmetric Mutation (●), Uniform Random Walk (●), Biased Random Walk (●), EA-UniformWalk (●), EA-BiasedWalk (●), EA-AsymUniformWalk (●) and EA-AsymBiasedWalk (●) for images from Figure 1 (left), Black-White (middle), Figure 4.15 (right). Generation number is shown on the x-axis and features values on the y-axis.

Considering the results for the images of Figure 4.1 (left column), it can be observed that the feature values for Benford's Law reduce at the first half of the transition process and increase afterwards. Furthermore, the value for the target image is quite low, but the evolutionary image transition process produces images where the value for Benford's Law is significantly higher than the one for the starting and the target image in the last third of the image transition process. In terms of global contrast, it can also be observed that the transition process creates images of higher feature value than the ones of the starting and target image. All considered algorithms follow the same pattern for these two features, but it can be observed that the pure random walk algorithms overall achieve higher values for Benford's Law and the combined approaches are able to obtain a trajectory of higher values for Global Contrast Factor.

Considering the features Mean Hue and Colorfulness, the features values are following a more direct trajectory from the value of the starting image to the one of the target image. For Hue, this trajectory is also very concentrated around the linear function connecting these two values whereas for Colorfulness a strong deviation, especially for the pure random walk algorithms, can be observed.

The transition process for the images of Figure 4.15 carries out a process where the features values of the starting and target image are of the same value. Again, it can be observed that the algorithms obtain higher values for Benford's Law and Global Contrast Factor during the transition for most of the runs. An exception is the biased random walk algorithm that sometimes produces lower values for these two figures during the transition. Mean Hue and Colorfulness again exhibit a more direct trajectory between the starting and target feature value with the random walk algorithms showing a stronger fluctuation and in particular lower values with respect to Colorfulness.

Considering the transition for Black to White images, it can be observed that Benford's Law and Global Contrast Factor increase during the transition process. The concentrated behaviour for Benford's Law is due to the calculation of this feature as the feature value is fully determined by the number of black and white pixels. Furthermore, there are no changes during the transition process for Mean Hue and Colorfulness.

4.7 Evolutionary Image Painting

We now consider how to use variations of the evolutionary image transition process for evolutionary image painting. The key idea is to make use of the biased random

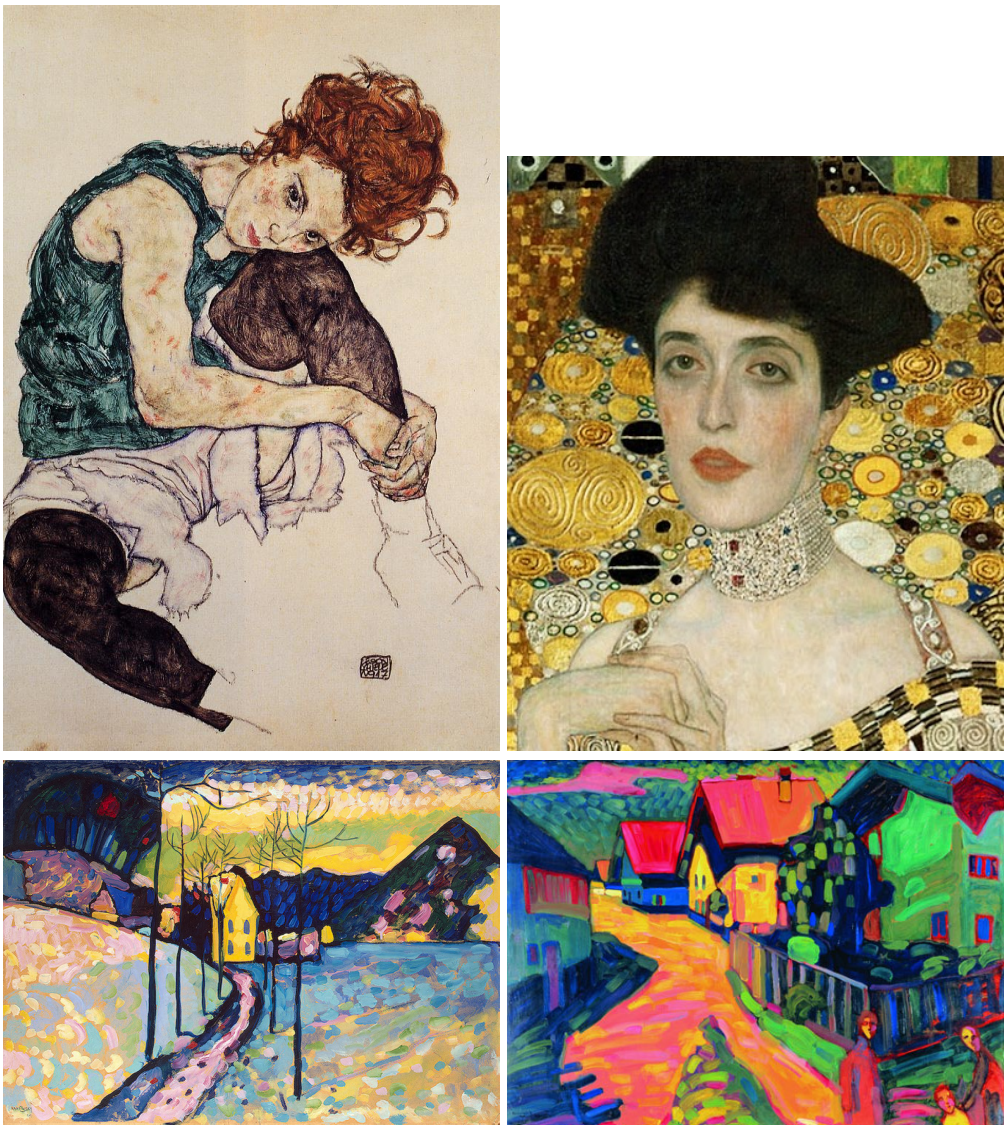


FIGURE 4.17: Target images for Evolutionary Image Painting.

walk and use its behaviour of favouring similar colors. We use this property to change pixel values of a "blank" image such that it becomes similar to a given target image. A biased random walk mutation in the painting process uses for each step the same colour which is determined by the pixel of the target image where it started. Due to this, we call this process "painting" of the target image.

The evolutionary image painting algorithm is given in Algorithm 7. It is similar to the evolutionary transition algorithm and uses a starting image S and the target image T to be painted. Again, we assume $S_{ij} \neq T_{ij}$ for all pixels as pixels with $S_{ij} = T_{ij}$ can be viewed as already painted. The algorithm minimizes the number of pixels that the current image X agrees with S and the pixels of X with $X_{ij} = S_{ij}$ can be viewed as

Algorithm 7 Evolutionary image painting

- Let S be the starting image and T be the target image.
- Set $X := S$.
- while (not termination condition)
  - $Y := X$.
  - For each $Y_{ij} \in Y$ with $(Y_{ij} == S_{ij})$.
    - Do Y:=PaintMutation$(Y_{ij}, Y, S, T, \alpha, t_{\max})$ with probability $\min\{c_s/(2|X|_S), 1\}$.
  - Set $X := Y$.

pixels that have not yet been visited. For our investigations, we use a white starting image S as we are mainly interested in the final painted image.

The mutation operator uses the biased random walk for a given starting pixel X_{ij} (see Algorithm 8). As we are considering painting of an image, the starting pixel of the biased random walk determines the color in which the random walk paints the part of the image that it is visiting. The mutation operator in the evolutionary painting algorithm uses this biased painting random walk for each pixel that is still in the starting state S_{ij} with probability $\min\{c_s/(2|X|_S), 1\}$ and therefore adapting the asymmetric mutation operator. It ensures that only biased random walks are started at pixels that have not changed their state yet. The idea is that pixels that have changed their state are considered as being painted and should therefore not change their color again.

If a biased random walk is started at a pixel X_{ij}, the color $C := T_{ij}$ to be used during this random walk is given by the value of the pixel T_{ij} in the target image T. During the biased random walk only pixels that have not changed their value yet are painted with the color C which is again motivated by not painting pixels of the image that have already been painted. Each iteration of Algorithm 7 does not increase the number of pixels where X and S agree. If at least one biased random walk happens, an offspring Y with $f(Y, S) < f(X, S)$ (see definition of f in Section 4.2) is obtained. The implies that the algorithm minimizes the fitness function $f(X, S)$ and an image X^* with $f(X^*, S) = 0$ is considered to be fully painted.

We now introduce a novel mutation operator called painting mutation operator for image transition developed to one specific purpose, in particular to produce interesting artistic images imitating modernist movement in the art called expressionism. The position of the first pixel X_{ij} is chosen exactly the same how in our previous mutation

Algorithm 8 PaintMutation(X_{ij}, X, S, T, α, $t_{\max}$)

- Set $C := T_{ij}$.
- Set $X_{ij} := C$.
- $c := 0$.
- while ($c \leq t_{\max}$)
  - $c := c + 1$
  - Choose $X_{kl} \in N(X_{ij})$ according to probabilities $p(X_{kl}, \alpha)$.
  - Set $i := k$, $j := l$.
  - If ($X_{ij} == S_{ij}$) then $X_{ij} := C$.
- Return X.

operators, randomly. The painting mutation operator make transition of the current starting image X to an area of the target image T. Note that the paint mutation operator given in Algorithm 8 only paints a pixel X_{ij} with the chosen colour C if the considered pixel is in the state of the starting pixel. A different option would be to always paint with the colour C irrespective of whether the pixel X_{ij} is in the starting or target state. This would allow that pixels already painted with a colour are able to change their colour during the process.

In our approach, we choose randomly one pixel from starting image C and using the biased random walk described in detail in Section 3.2. The transition process occurs from starting image C, in our case we decided the image is white, to the target image, our artistic image. Through the transition from white image to another artistic image we can clear represent processes the occur during biased random walk mutation. It is also possible to choose two artistic images when the colors are opposite of the color theory and achieve interesting effects in terms of stronger contrast. Our investigation has additionally the goal to give adequate insight in to the evolutionary processes during the transition processes.

4.7.1 Impact of Choice of the α in Evolutionary Image Painting

We have developed the special case of biased random walk mutation algorithm with constant α. The classical random walk is running with $\alpha = 0$ without deliberately currying about the α. On the assumption biased random walk with $\alpha > 0$ the algorithm

will with high probability perform along the edges of the similar colors as opposed to going over the edges to the colors with greater RGB values differences.

For our experimental investigations, we set $c_s = 200$ which implies that several random walks painting different parts of the image are started in each generation. The goal of our experimental investigations is to study the effect of α in $p(X_{kl}, \alpha)$ (introduced in Section 5.3) for evolutionary image painting. This parameter allows to focus on similar pixels during the biased random walk and one would expect a painted image close to the given target if α is large enough.

We study our evolutionary painting approaches on two landscape pictures and two portraits. The four target images used for evolutionary painting (see Figure 4.17) are as follows: Seated woman with bent knee, 1917 by Egon Schiele; Adele Bloch-Bauer, 1907 by Gustav Klimt; Winter Landscape, 1909 by Wassily Kandinsky; Murnau street with women, 1908 by Wassily Kandinsky. These images have been selected for our studies due to the wide recognition of these images as examples of fine art in Western culture [111]. For our experimental investigations, we consider evolutionary painting with $\alpha = 0.25, 0.5, 0.75, 1.0$. The results for evolutionary image painting with these different parameters of α are shown in Figure 4.18 and Figure 4.19.

Our experiments show that the elements emerge over the image during the transition stage resulting in differently occurring images at the end of the generation processes. Figure 4.18 and Figure 4.19 show four various images with unique artistic value. Firstly, we see the target image T, following the four adjustment for $\alpha = 0.25, 0.5, 0.75, 1.0$. We have executed mostly varied appearance of the target image T. Each of these pictures represent different stages of the painting. Additionally, for comparison we have chosen 4 different pictures as portray, landscape, abstract art and nature, in the stage of transition process when the painting process is completed. We can observe the impact on the evolutionary process cause through the different settings.

As α controls the bias of the underlying random walks, it can be observed that small values of α obtain paintings that are much less precise than the target image. Thus, it's creating coarse-grained painting effects. This effect decreases with increasing α and the painted image becomes much closer to the given target image. For all considered images, $\alpha = 1$ already gives a painting being close to the target which is the reason why larger values of α are not investigated. Comparing the different images, it can be observed that $\alpha = 0.5, 0.75$ creates aesthetically pleasing paintings for the considered landscape pictures whereas a value of $\alpha = 0.75$ produces novel results when applied to the considered portraits.

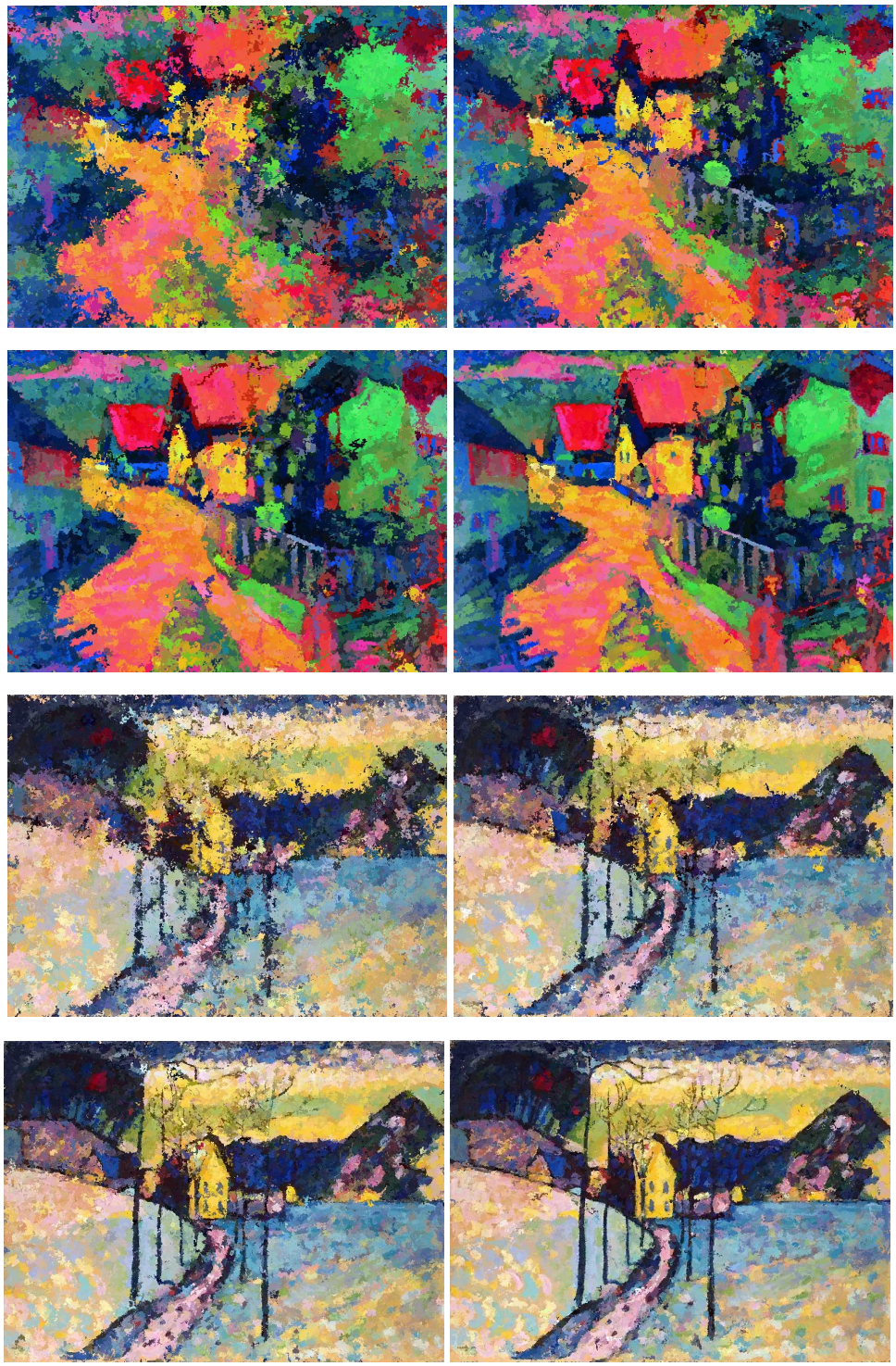


FIGURE 4.18: Evolutionary Image Painting with $t_{\max} = 500$ and $\alpha = 0.25, 0.5, 0.75, 1.0$.

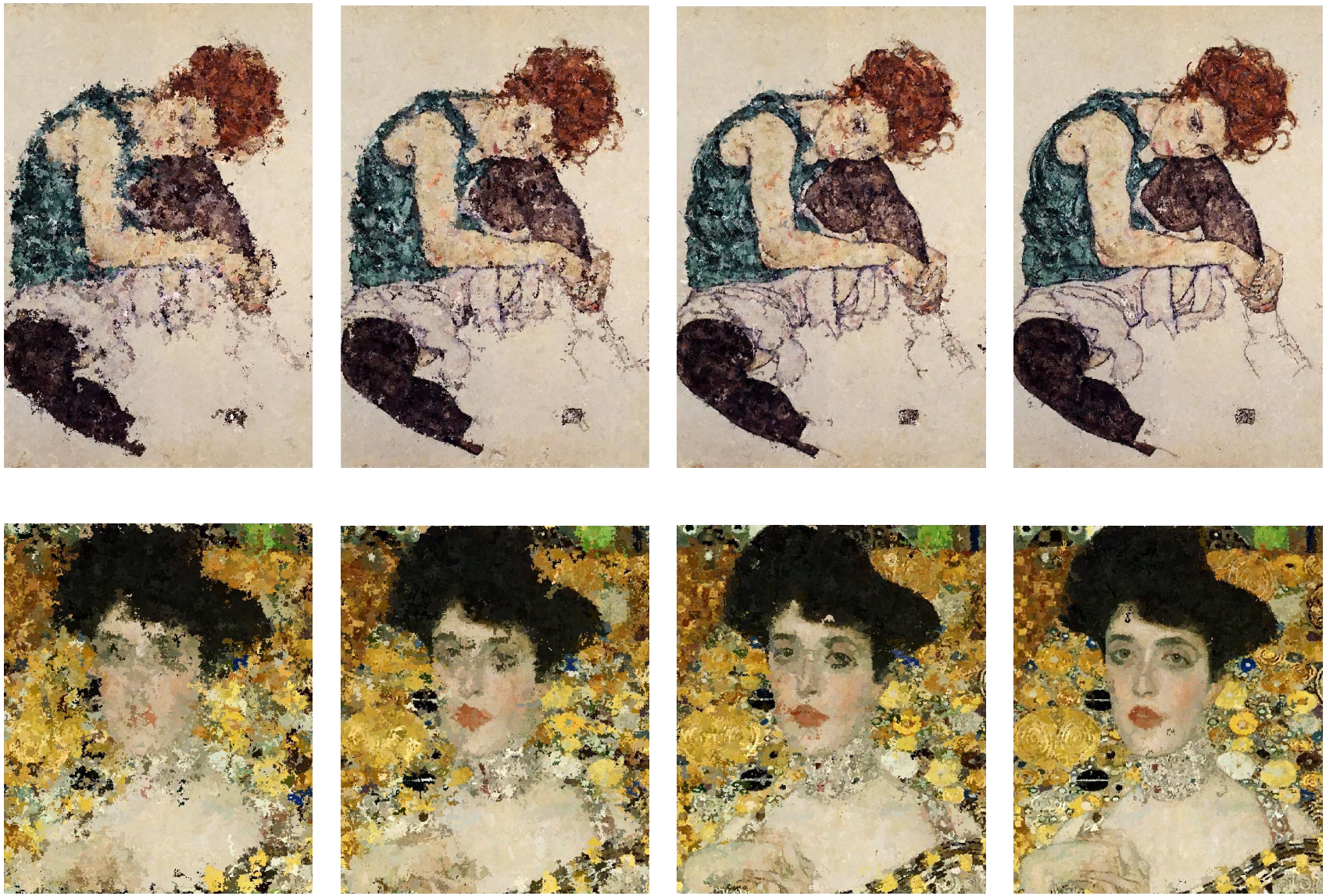


FIGURE 4.19: Evolutionary Image Painting with $t_{\max} = 500$ and $\alpha = 0.25, 0.5, 0.75, 1.0$ (from left to right).

4.7.2 Impact of Random Walk Length in Evolutionary Image Painting

Now we investigate the impact of random walk length on our evolutionary painting approaches. For our experimental investigations, we consider evolutionary painting algorithm with random walk length with $t_{\max} = 10, 100, 200, 1000, 4000, 10000$, and $\alpha = 0.25, 0.5, 0.75, 1.0$. We choose one of the landscape pictures presented in the Figure 4.17 and the abstract pictures presented in the Figure 4.1, what give us a clear comparison to the previous experiments described in Section 4.7.

Also, we set $c_s = 200$ what again implies that several random walks painting different parts of the image are started in each generation. The goal of our experimental investigations is to study the effect of random walk length in comment $p(X_{kl}, \alpha)$ introduced in Section 5.3 for evolutionary image painting. This parameter allows to focus on similar pixels during the biased random walk and one would expect a painted image close to the given target if α is large enough.

The results for evolutionary image painting with four different values of α and several lengths of random walk, which imply values of the range between large $t_{\max} = 10000$ and small $t_{\max} = 10$, are shown in Figure 4.20 and Figure 4.21.

In the first row, we observe the image with $t_{\max} = 10$ and $\alpha = 0.25$, following the three adjustments for $\alpha = 0.5, 0.75, 1.0$. Each of these pictures represent different stages of the painting. Comparing the different images, it can be observed that $\alpha = 0.25, 0.5$ and $t_{\max} = 2000, 4000, 10000$ creates more patch-based appearance of the image. We can observe the impact on the process cause through the different settings of random walk length in evolutionary image paining. In summary, the random walk length and the choice of α influences how "precisely" the target image is painted. The different parameter choices exhibit a wide range of possibilities for painting images with different effects.

4.8 Conclusions

Evolutionary image transition uses the run of an evolutionary algorithm to transfer a starting image into a target image. In this work, we have investigated how random walk algorithms can be used in the evolutionary image transition process. We have shown that mutation operators that utilize different ways of incorporating uniform and biased random walks lead to different effects during the transition process. Furthermore, we have studied the impact of the different approaches with respect to different artistic features and observed that the process creates images which significantly differ from the starting and target image in respect to these features. Specially,

we have shown how *the asymmetric mutation* operator, introduced originally for the optimization of OneMax, can be applied to our problem. We examined how the varying mutation settings influences our results through *block mutation, stripes mutation, combined stripes mutation* and combined approaches with *asymmetric mutation*. By investigating combinations of the different approaches, we have shown a variety of interesting evolutionary image transition processes.

Using biased random walks for image painting, we have shown that this approach creates artistic paintings which can be varied by the parameter α controlling the bias of the underlying random walk.

All our investigations are based on a fitness function that is equivalent to the well-known ONEMAX problem. For future research, it would be interesting to study more complex fitness functions and their impact on the artistic behaviour of evolutionary image transition.

A conference version has been published in the Proceedings of the Computational Intelligence in Music, Sound, Art and Design - 6th International Conference (EvoMUSART) 2017 (see A. Neumann, B. Alexander and F. Neumann (2017) [188]). A conference version containing the results of Section 4.3 has been published in the Proceedings of Neural Information Processing - 23rd International Conference (ICONIP) 2016 (see A. Neumann, B. Alexander and F. Neumann (2016) [189]). A journal version of this chapter has been submitted for publication.

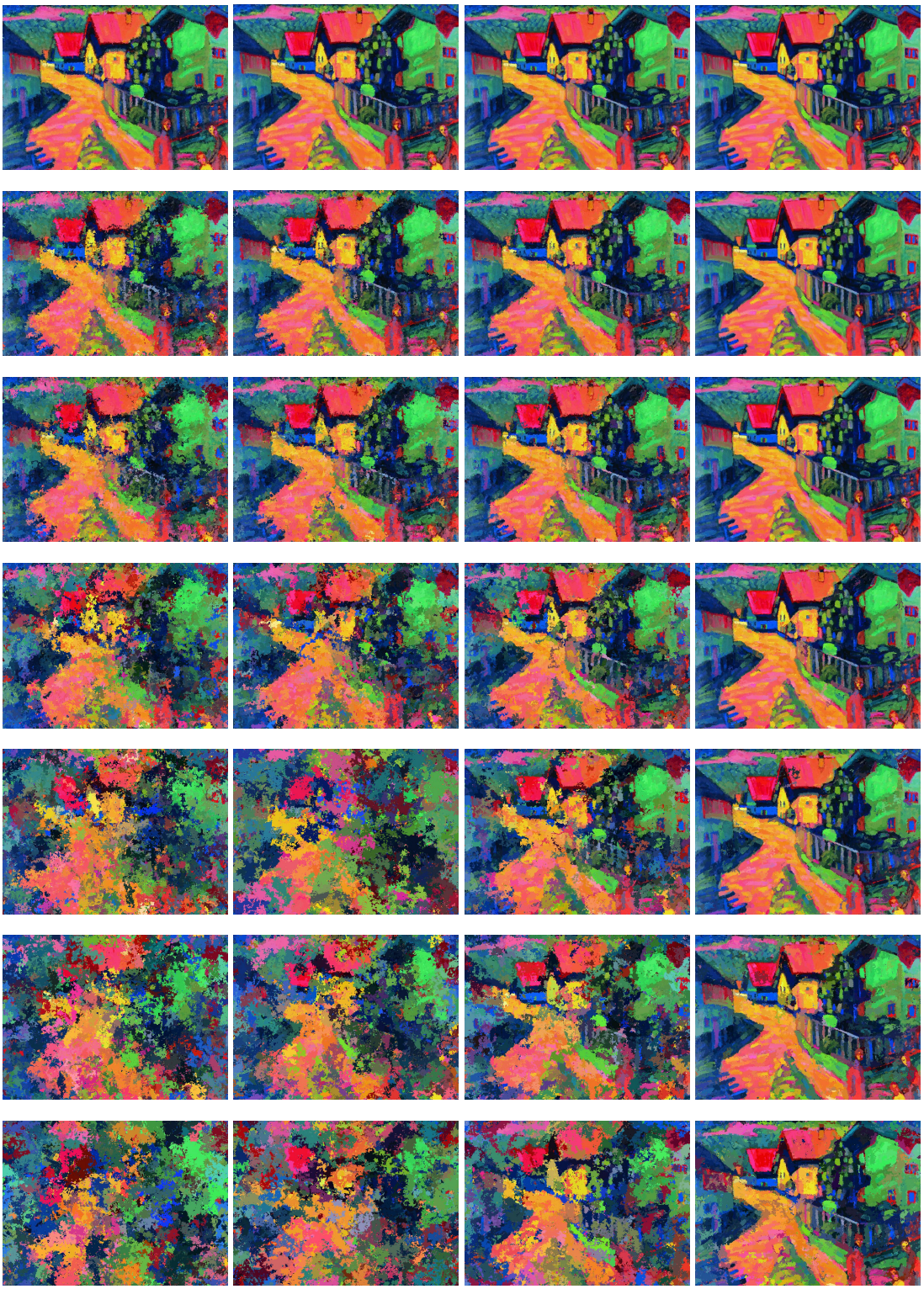


FIGURE 4.20: Evolutionary Image Painting with $t_{\max} = 10, 100, 200, 1000, 2000, 4000, 10000$ (from top to bottom) and $\alpha = 0.25, 0.5, 0.75, 1.5$ (from left to right).

FIGURE 4.21: Evolutionary Image Transition with $\alpha = 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75$ at $12.5\%, 37.5\%, 62.5\%$ and 87.5% of the target image (from top to bottom) and $t_{\max} = 2000$.

Chapter 5

Quasi-random Agents for Image Transition and Animation

5.1 Introduction

In this chapter, we study the processes of creating artistic art work in the context of evolutionary image transition and animation. The key idea in this area is to use *randomness* to create interesting visual effects. Evolutionary dynamics and random walks can be closely linked and random walks are often used to explain evolutionary behavior [4]. Random walks have been previously used in the evolutionary image transition context [188]. Specially random walks biased on the given images can create very interesting effects in respect to evolutionary image transition. However, the drawback of these is that the behavior is difficult to control for the user/designer of the application.

We present a new approach for evolutionary image transition using quasi-randomness and extends the framework to evolutionary animations. Quasi-randomness has already been used in evolution computation and it has been shown that using this may lead to better performing and better understandable evolutionary algorithms [243–245]. We replace the classical random walk with a so-called *quasi-random walk*. In this model, the decisions of the walker are not chosen uniformly at random, but deterministically. Instead of moving to a random neighbor, the walker visits all neighbors in a fixed order. This model is well-studied in discrete mathematics for investigating how much randomness is required to achieve similar properties of regular random walks, but with less randomness. The quasi-random walk was rediscovered independently several times in the literature. In order of appearance, this concept has been called "Eulerian walker" [210], "edge ant walk" [262], "whirling tour" [59], "Propp machine" [26, 133], "deterministic random walk" [25, 41], and the "rotor-router model" [69, 71]. To show the relationship to standard random walks, we

mostly use the term "quasi-random walk" in the rest of the work. We will synonymously use "rotor-router model" to sometimes emphasize the inner workings.

Evolutionary algorithms have been used in various ways to create art and music. In the context of art, various evolutionary computing methods have been used to create artistic images [88, 169, 232, 247, 256]. Recently, the notion of evolutionary image transition [188] has been introduced and the goal of this line of research is to use the evolutionary process itself to create evolutionary processes that are interesting from an artistic perspective and potentially inspiring to artists using evolutionary computation. Furthermore, the underlying process of evolutionary image transition has been used for evolutionary image composition taking into account features of the given images [196].

We use quasi-random walks to design algorithms for image transition and animation [1]. In our algorithms, a set of agents perform quasi-random walks where the characteristic of the random walk can easily be determined by a user. The process starts with a given starting image (which may be white or blank). Each agent has an image that it paints. It does so by carrying out its quasi-random walk as determined by the router sequence chosen by the user. The r agents perform their quasi-random walks in parallel painting different images (in the case of animation) or the same target image in the case of transition.

The user can easily determine the behavior of each agent by setting the sequence of directions that is followed at every pixel. An example sequence could be (left, down, right, up, left). Each pixel has its current active direction in the sequence. If the agent visits a pixel p, it paints it with the corresponding pixel of its target image. Afterwards, the agent moves to the next pixel as directed by the current active direction at p in its sequence and updates the active direction at p to the next one in the sequence.

The use of different sequences for the agents allows to create various types of interesting transition and animation processes. Analyzing the images created during the animation process with respect to different artistic features, we show that quasi-random animation using different rotor sequences allows to create animation processes with a wide range of different artistic behaviors.

The remainder of this chapter is organized as follows. In the next section, we give a brief introduction into quasi-random walks. We present our approach for quasi-random image transition and animation in Section 5.2. Afterwards in Section 5.3, we examine the wide range of image transitions and animations that can be created using various types of agents. We carry out a feature-based analysis of our animations using

[1] Videos are available at `https://vimeo.com/anetaneumann`

artistic features in Section 5.5 and finish with some concluding remarks.

5.2 Quasi-random Transition and Animation

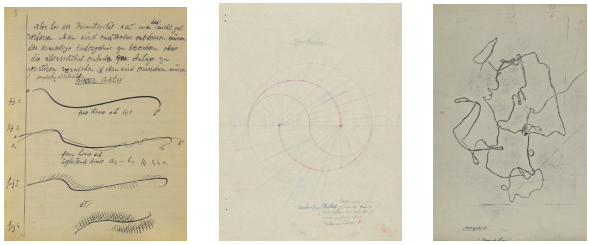
We now present our method for carrying out quasi-random image transition and animation. The concept of quasi-random walks we described in greater detail in Chapter 2.

5.2.1 Quasi-random Walks

The quasi-random walk is defined on the two-dimensional infinite grid $\mathbb{Z}^2$. To visualise this process, a single walker on an arbitrary vertex $(x, y) \in \mathbb{Z}^2$ can move to one of the four neighbors right, down, left, or up, that is, to $(x + 1, y)$, $(x, y + 1)$, $(x - 1, y)$, or $(x, y - 1)$. A random walk would choose independently and uniformly at random one of these four neighbors and move there. In order to use less randomness, the quasi-random walk assigns a permutation of the four directions right, down, left, up to each vertex of the grid. More importantly, these permutations are fixed initially and deterministically determine the behavior the quasi-random walkers. The advantage of the rotor-router model is that it does not require randomness. The process is completely determined by the sequence used and the starting position of the rotor.

The simplest rotor sequences are $(\text{right}, \text{down}, \text{left}, \text{up})$ or $(\text{right}, \text{left}, \text{down}, \text{up})$. The quasi-random walks move regarding to the information that are store as the current status of the permutation in each vertex. In this way the vertex has a rotor information to store which neighbor can be visited next. Each time a walker is leaving a vertex, it moves in the direction of the rotor and updates the rotor according to the fixed permutation of the vertex. When the rotor reaches the last position of permutation, it wraps around and starts again at the beginning of the permutation.

Following the idea of a "stack walk" [106], we generalize the concept of the quasi-random walk and allow not only permutations and rotor sequences of length four, but arbitrary sequences of the four cardinal directions. If some directions occur more frequently than others, this results in a bias of the walker in a particular direction. For example, the rotor sequence (right, left, up, down, right, left) will much more explore horizontally than vertically. This bias implies a deviation from the expected behavior of the standard random walk but allows for more interesting artistic effects.



(A) (B) (C)

FIGURE 5.1: Works of Paul Klee.

5.2.2 Related Artistic Work

At the beginning of the 20th century the well-known artist Paul Klee was part of the famous Bauhaus movement. Klee was a unique university teacher in Weimar and Dessau and an iconic promoter of a theoretical approach to making art. In the Pedagogical Sketchbook [134] the artist shows his innovative approach to artistic expression. One of his most important messages about art and design is "an active line on the walk, moving freely, without goal." He describes the line as the most human mark and characterizes the various types of a line. Figure 5.1 (a) shows "the linear line they are subdivided in some modus in contrast to the primitiveness as a free line, a free line with accompanies of lines"'. In (b), we see the drawing of "construction with middle curve from two centers radiated and in associated region shaft-wide halved". Finally, in (c), we see the drawing with the title Honigschrift as "a flowing line". Paul Klee's notes include 3900 pages and demonstrate in multiple ways how a point can become a line and the line the plane can result in "free irregularity". Quasi-random walks essentially create lines in the space where they perform the walk and we take this process as an inspiration for designing our transition and animation methods.

5.2.3 Our Approach

We take the technical behavior of quasi-random walks and combine it with Klee's idea to create artistic work. *Quasi-random animation* follows the idea of the rotor-router model in order to produce artistic ideas by agents carrying out quasi-random walks. The pseudo-code for quasi-random animation is given in Algorithm 9.

We consider a set of r agents where each of them has the goal to paint one particular image I^k. Each agent k, $1 \leq k \leq r$, works with a sequence S^k which consists of entries from $\{\text{right}, \text{down}, \text{left}, \text{up}\}$. For example, we could have $S^k = (\text{right, left, down, up})$ as in the standard rotor-router model, but also sequences such as $S^k = (\text{right, left, up, down, right, left})$ which are not symmetric with respect to the

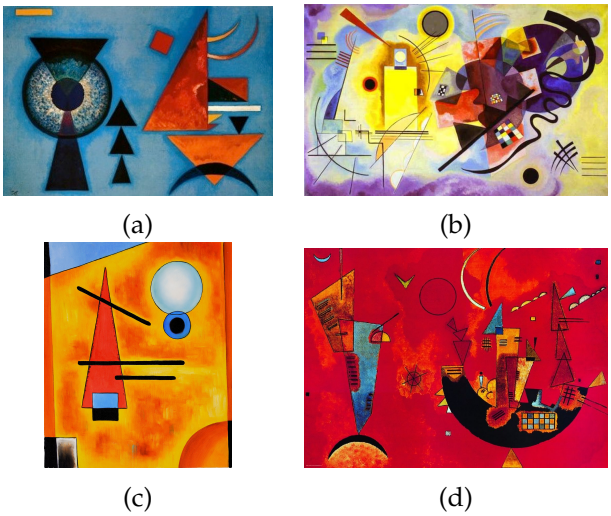
Algorithm 9 QUASI-RANDOM ANIMATION

Require: Start image Y of size $m \times n$. For each agent k, $1 \leq k \leq r$, an image I^k of size $m \times n$, sequence S^k and position counters $c^k(i,j) \in \{0, \ldots, |S^k|\}$, $1 \leq i \leq m$, $1 \leq j \leq n$.

$X \leftarrow Y$
for each agent k, $1 \leq k \leq r$ **do**
    choose $P^k \in m \times n$ and set $X(P^k) := I^k(P^k)$.
$t \leftarrow 1$
while $(t \leq t_{\max})$ **do**
    for each agent k, $1 \leq k \leq r$ **do**
        Choose $\hat{P}^k \in N(P^k)$ according to $S_k(c(P^k))$.
        $X(\hat{P}^k) \leftarrow I^k(\hat{P}^k)$
        $c^k(P^k) \leftarrow (c^k(P^k) + 1) \mod |S^k|$.
        $P^k \leftarrow \hat{P}^k$.
    $t \leftarrow t + 1$

different directions. Using such asymmetric sequences creates a bias in the walk performed by the agent and leads to various interesting effects dependent on the choice of S^k. At each time step each agent k moves from its current position P^k to one of its pixel neighbors $\hat{P}^k \in N(P^k)$ where the direction $S^k(c(P^k))$ is determined by the current active pointer at position P^k (wrapping around at the border of the image). Afterwards the current image X gets updated by setting the pixel value at position $\hat{P}^k$ to the value of image I^k at position $\hat{P}^k$. The move of agent k is completed by increasing the sequence counter $c^k(P^k)$ in a modulo fashion and updating the current position P^k to $\hat{P}^k$.

Quasi-random image transition is a special case of quasi-random animation where all I^k are set to one particular target image I. The key aspects when using this algorithm to create interesting transitions and animations is the number of agents, the images that they are using for painting and the router sequence for each agent. The choice of the router sequence S^k determines the random walk behavior of the agent k. A sequence can be an arbitrary sequence of directions. We mainly study sequences that are rather short in this work. The reason is that this already gives a large variety of different effects. Furthermore, short sequences are user-friendly as they can be easily adapted and understood by the user. The potential of longer sequences (partially explored in our feature-based analysis) allows to design more fine-grained animations. In addition, the approach may be extended to have for each pixel its own sequence of



(a) (b)

(c) (d)

FIGURE 5.2: Starting and target images.

directions which would allow to resemble the behavior of biased random walks used for example in image segmentation [85].

A sequence of length four containing each direction exactly once such as (right, down, left, up) corresponds to the classical roter-router model leading to a cover time that is close to the one of a classical random walks which moves at each to a neighboring pixel selected uniformly at random. However, the setting leads to a much smoother transition and animation being generated as it avoids the unbalance in terms of the different directions that occurs in the classical random walk. An unbalanced sequence such as (right, down, left, up, right) where each direction does not appear the same number of times, introduces a bias into the direction that appears more often. Let r be the total sequence length and r_i the number of times direction i appears in this sequence. Assume that the agent has done M steps, where M is large. Then the number of steps it has taken into direction i is roughly $M \cdot \frac{r_i}{r}$.

This means that the sequence length r and the number of appearances r_i of direction i can be used to create arbitrary fractions of moves of four cardinal directions. Given that each agent is working with its own sequence and therefore its own fractions of the different directions, complex behavior can be created by using agents with different types of sequences.

5.3 Experimental Investigations for Quasi-random Image Transition

We now show our results for quasi-random image transition where all agents paint the same image I. We use the pairs of images shown in Figure 5.2 for our experiments.

To illustrate the effect of the diverse methods presented in this paper, we consider the Soft Hard, 1927 (a) and the Cool, 1927 by Wassily Kandinsky (c) as a starting image. We consider the Yellow-Red-Blue, 1925 (b) and the With and Against, 1929 by Wassily Kandinsky (d) as a target image for image transition. In the experiments, each agent performs 5 000 000 steps and the images are of size 400 x 400 pixels. This setting holds for all experiments reported on in this work. We should mention that the use of other starting and target images would show the same algorithmic behavior as the performance of the agents in Algorithm 9 is independent of the choice of the images. However, using different images allows to create different artistic effects and we will use different sets of images to illustrate a wide range of effects we observed.

5.3.1 One Agent

In our first set of experiments, we consider one agent carrying out the image transition. For the first experiment, we consider the sequence S^1 = (right, down, left, up) which resembles a standard roter-router model. The results in Figure 5.3 (top row) shows the quasi-random image transition process in a circle-patch style transforming image (a) into image (b). As we set the starting point of router at the middle of a given image (a), we observe the movement of router circulated in right direction over the image. The quasi-random image transition produces visually pleasing results with bridging parts of both images in a new abstract composition. Compared to image transition using classical random walks [188], this produces much smoother images.

In the second experiment, we expand our approach to non-standard roter-router settings and use an asymmetric sequence for the transition. We add only one additional position to the sequence which introduces a bias into the added direction. For the experiment, we consider the sequence S^1 = (right, down, left, up, right). Figure 5.3 (bottom row) shows the results. In contrast to the previous setting, the behavior of the quasi-random image transition leads to horizontal stripes over the whole image. These stripes gather over time leading to interesting cumulative effects with the target image finally appearing. It also results in a much faster transition process. This is due to the additional direction "right" in the chosen sequence.

FIGURE 5.3: Image transition with one Agent. Top row shows symmetric sequences (images after 100 000, 500 000, 2 000 000 and 3 500 000 steps). Bottom row shows asymmetric sequences (images after 100 000, 200 000, 300 000 and 500 000 steps).

5.3.2 Two Agents

We now consider the combination of two agents performing the image transition. The starting positions of the two agents are chosen as $[m/4, n/4]$ and $[3m/4, 3n/4]$. We consider the sequences S^1 = (right, down, left, up) and S^2 = (up, left, down, right) for the agents. The results in Figure 5.4 (top row) show two parts of the target image around the agents starting positions emerging. The process of quasi-random image transition with an arrangement of two agents appears more dynamic than the one with one agent using a symmetric sequence.

Finally, we study the behavior of two agents with asymmetric sequences for image transition. We consider the sequences S^1 = (right, down, left, up, right) and S^2 = (up, left, down, right, up). The results in Figure 5.4 (bottom row) show that the quasi-random image transition progresses similar to the one agent approach with an asymmetric sequence. In addition, there are vertical stripes "walking/running" from left to right due to the second agent. The whole transition occurs by horizontal and vertical quasi-random walks on the image.

FIGURE 5.4: Image transition with two Agents. Top row shows symmetric sequences (images after 200 000, 600 000, 1 000 000 and 1 900 000 steps). Bottom row shows asymmetric sequences (images after 30 000, 80 000, 140 000 and 210 000 steps).

5.4 Experimental Investigations for Quasi-random Animation

We now present results obtained for quasi-random animation. We showcase the results using interesting and characteristic images of the considered animation process. Furthermore, the feature-based analysis in the next section provides an analysis over the whole animation process with respect to artistic features.

5.4.1 Two Agents

We use two agents starting at positions $[m/4, n/4]$ and $[3m/4, 3n/4]$ and images from Figure 6.2. Image (c) is used as the starting image and as image I^1 for the first agent, and image (d) is used image as I^2 for the second agent. Firstly, we consider the behavior when using the symmetric sequences S^1 = (right, down, left, up) and S^2 = (up, left, down, right). Figure 5.5 (top row) shows the results. The second agent starts to paint the image in circle appearances according to the symmetric agent movement. The animation obtains a more dynamic character when the regions painted by the agents meet and the agent operate at a "larger radius". This leads to the effect that the agents operate from all direction on the image. It looks as if the agents naturally pervade the image and effortlessly collaborate together.

We now consider two agents using the asymmetric sequences S^1 = (right, down, left, up, right) and S^2 = (up, left, down, right, up). Figure 5.5 (bottom row) shows the

FIGURE 5.5: Quasi-random animation with two Agents. Top row shows symmetric sequences (images after 3 600 000, 3 900 000, 4 000 000 and 4 900 000 steps). Bottom row shows asymmetric sequences (images after 300 000, 500 000, 1 600 000 and 3 700 000 steps).

results. In contrast to the previous animation, the images obtained are a mixture of the two images of the agents in all parts of the image. This is due to the first agent painting its image in form of horizontal stripes and the second agent painting its image using vertical stripes. As the agents move over the image, their current horizontal (for the first agent) and vertical (for the second agent) position determines which image of the agents shines through in which part at any given point in time of the animation.

5.4.2 Four Agents

We now consider results that are achieved when using four agents. We obtain tree additional images by changing the color spectrum of the image shown in Figure 5.2 (c) to blue, green, yellow. We use (c) and these tree additional images as the images I^k painted by the agents. This means that we can identify the image of an agent by its color. The setting allows us to design animations where the underlying images have the same structure and only differ in terms of their color spectrum. The image in Figure 5.2 (d) is chosen as the starting image. The four agents start at positions $[m/4, m/4]$, $[m/4, 3n/4]$, $[3m/4, n/4]$ and $[3m/4, 3n/4]$, respectively.

For the symmetric experiment, we use the sequences $S^1 = S^3$ = (right, down, left, up), and $S^2 = S^4$ = (up, left, down, right). The results in Figure 5.6 (top row) show that the images of the agents appear in circle-patch styles. It can be observed that parts

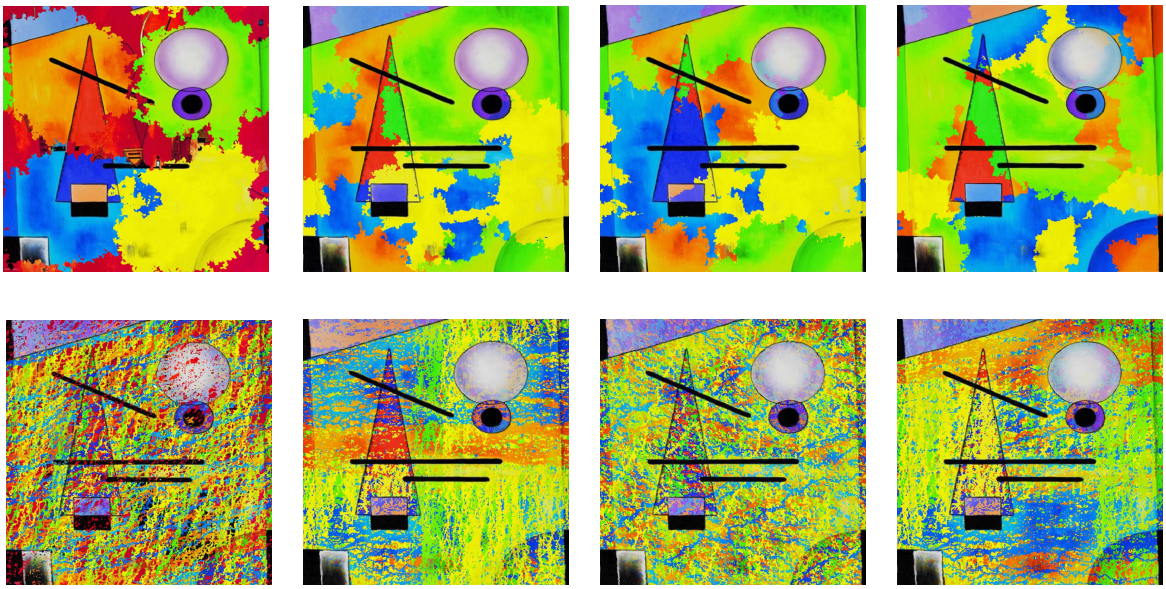


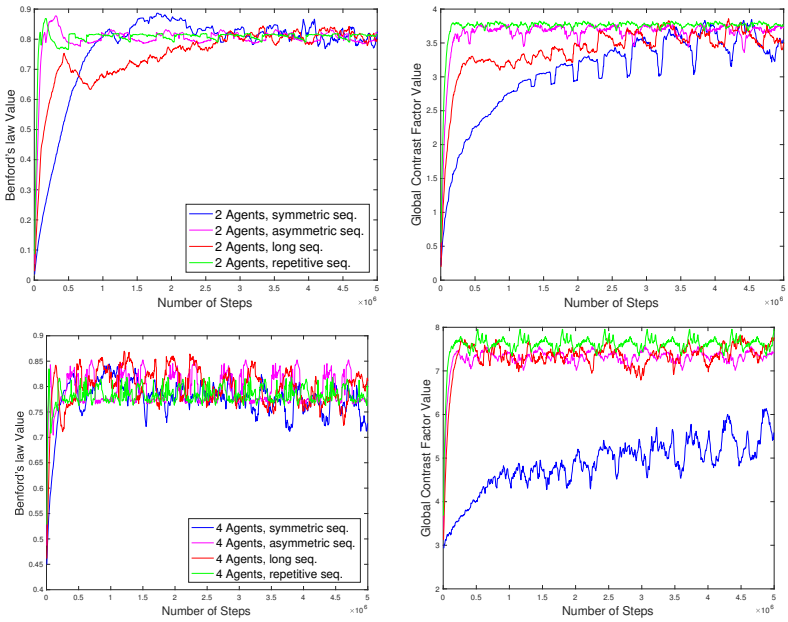
FIGURE 5.6: Quasi-random animation with four Agents. Top row shows symmetric sequences (images after 500 000, 2 100 000, 2 400 000 and 4 200 000 steps). Bottom row shows asymmetric sequences (images after 100 000, 1 200 000, 4 000 000 and 4 300 000 steps).

of the different images painted by the agents appear at different time steps at different parts of the image. Thus, the algorithm allows the agents to paint part of their own image. The quasi-random image animation produces a visually pleasing result and combines the different parts of all images in a new abstract composition.

We now consider four agents using the asymmetric sequences $S^1 = S^3$ = (right, down, left, up, right), and $S^2 = S^4$ = (up, left, down, right, up). Figure 5.6 (bottom row) shows the results. In contrast to the symmetric sequences, the different images obtained provide an interesting mixture of the images of the different agents which is due to two agents moving horizontally and two agents moving vertically. There are no parts that can be clear attributed to one of the agent's image. Instead of this, the animation produces interesting overlapping effects where the dominance of the colors (corresponding the agent's images) in the different parts of the image change over time.

5.5 Feature-based Analysis

We now analyze the different animation processes obtained by our quasi-random animation algorithm. As the process creates different images over time, we measure the images created with respect to different artistic features. We consider features from the



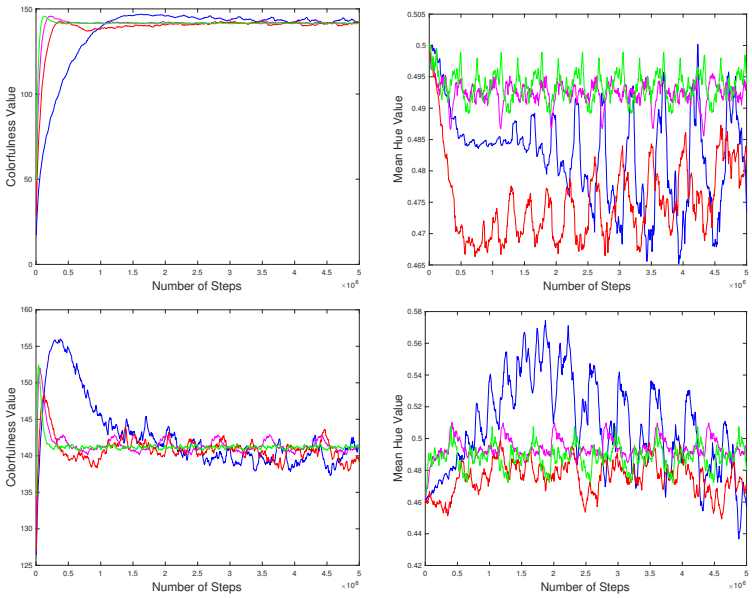
(a) Benford's law (b) Global Contrast Factor

FIGURE 5.7: Features during quasi-random animation: (a), (b).

literature that have been used to measure artistic images [1, 97, 99, 172, 188, 218] in order to enable an objective evaluation.

For our analysis, we examine all animation settings with two and four agents studied in the previous section. Furthermore we investigate the following additional settings for two agents: In the *long sequences* experiment, we consider S^1 = (right, down, left, up), S^2 = (right, down, left, up, right, down, left, up, right, down, left, up, right, down, left, up, right, up) and in the *repetitive sequences* experiment we use S^1 = (up, right, right, right), S^2 = (down, right, right, right). For four agents we consider the following two additional settings: In the *long sequences* experiment, we use $S^1 = S^3$ = (right, down, left, up), $S^2 = S^4$ = (right, down, left, up, right, down, left, up, right, down, left, up, right, down, left, up, right, up). For the *repetitive sequences* experiment, we use $S^1 = S^3$ = (up, right, right, right), and $S^2 = S^4$ = (down, right, right, right). The results for all animations according to the features *Benford's law*, *Global Contrast Factor*, *Colorfulness*, *Mean Hue* are shown in Figure 5.7. Here the top row shows the results for animations with two agents whereas the bottom row shows the results for four agents.

Subfigure (a) shows the results of the different animations for the feature *Benford's law* [12, 97]. It can be observed that the curve for two agents with symmetric sequences between 1 500 000 and 2 000 000 steps achieve the highest value. High values mean that results are less natural. Furthermore, the two agents with long sequences show



(c) Colorfulness (d) Mean Hue

FIGURE 5.8: Features during quasi-random animation: (c), (d).

the most natural results, as the values in comparison to the other agents are the smallest. During the animation, the values increase slowly. The values for two agents with repetitive and asymmetric sequences show a more stable behavior during the animation. The feature values for the animations that involves the use of four agents with long and asymmetric sequences show regularly appearing differences in values during the animation. They also have the overall the highest feature values among all animations. This means that those animations are less natural and explains the chaotic appearance of the animation. Finally, it can be observed that the curve for four agents with asymmetric sequences is clearly lower than those obtained for four agents with symmetric sequences.

In Subfigure (b), the results for the feature *Global Contrast Factor* show a great difference between symmetric sequences and all other choices. *Global Contrast Factor* measures the richness of contrast and corresponds to the human perception of contrast in terms of visual attention. [172] show that humans prefer higher scores for *Global Contrast Factor* if they are looking at images. High values during the animation can be observed for the animations with two and four agents using repetitive sequences. For the animation with two and four agents with symmetric sequences, a characteristic zig-zag curve is obtained where the value slowly increases over time. In contrast to this, the animations with two and four agents using asymmetric, long and repetitive sequences have less variation in term of the feature values.

Subfigure (c) shows the results of the different animations for the feature *Colorfulness*. [97] computed perceived colorfulness to evaluate the effect that processing of the natural images has on their colour. [218] used measurements of perceived colorfulness of website screenshots to develop computational models. It can be observed that all curves of the *Colorfulness* feature obtain values of around 140-145 after the animation has run for 1 000 000 steps. Additionally, all curves have a very steady behavior except for the four agents with symmetric sequences where we can observe regular changes over time. This changes make the process more interesting as it shows a wider variety of animations.

In Subfigure (d), the results for the feature *Mean Hue* are shown. The animation with four agents using symmetric sequences reveal the largest variety in terms of this feature value. It can be observed that the curve is alternating between high and low values after approx. 2 500 000 steps. This expresses the chaotic character of those agent's movement during the animation. In contrast to this, the curve obtained during the animation for two agents with long sequences has a zig-zag pattern with tendency to increasing those values during the animation. On the other hand, all agents with repetitive and asymmetric sequences tend to have a continuous and regular behavior during the quasi-random animation.

5.6 Conclusions

Quasi-random methods allow to create interesting random behaviors and can easily be determine by a user who only has to set a few parameters within a quasi-random algorithm. We have presented a new approach to carry out image transition and animations. Our approach builds on theoretical insights of quasi-random walks and generalizes this concepts to multiple agents performing quasi-random walks for image transition and animation. The approach allows the creation of different forms of transition and animation by determining a sequence for each agent to be used in the quasi-random walk. Choosing these sequences is very easy and allows an artist to experiment with different complex behaviors resulting in interesting ways of carrying out transition and animation. Our feature-based analysis shows that the resulting animations exhibit quite different behavior in terms of important artistic features.

A conference version that contains the results of this chapter has been submitted for publication (see A. Neumann, F. Neumann and T. Friedrich [192]).

Part III

Evolutionary Image Composition

Chapter 6

Evolutionary Image Composition Using Feature Covariance Matrices

6.1 Introduction

In this chapter, we present a new approach for the composition of new images from existing ones that retain some salient features of the original images. We introduce evolutionary algorithms that create new images based on a fitness function that incorporates feature covariance matrices associated with different parts of the images. This approach is very flexible in that it can work with a wide range of features and enables the targeting of specific regions in the images. For the creation of the new images, we propose a population-based evolutionary algorithm with mutation and crossover operators based on random walks.

Recently, evolutionary processes have been used to create artistic videos through evolutionary image transition. In evolutionary image transition, a source image is transferred into a target image by an evolutionary algorithm, with the images constructed during this process forming frames of a video sequence [188, 189]. In Chapter 3 and 4, we discussed evolutionary image transition in more detail.

Now we describe a variant of evolutionary image transition. We use the evolutionary process to create a composition of two images S and T. The key element to achieve this goal is to use the concept of a so-called region covariance descriptor, which is well known and well-studied in the field of computer vision. Chapter 3 details some prerequisites concerning a covariance descriptor. Region covariance descriptors can be used to describe an image based on various sets of features related to different regions of the image. Given two images S and T, we aim to evolve an image X which consists of a mixture of pixels from S and T and is "close" to both S and T in terms of covariance descriptors for both images. One of the main contributions of this paper is in finding

a way to incorporate region covariance descriptors into an appropriate fitness function. We investigate the impact of different choices of image features on the aesthetic properties of images created via evolutionary image composition. We also explore the aesthetic impact of weights with which various region covariance descriptors can be combined together to form a fitness function.

The fitness function based on region covariance descriptors is minimised using an evolutionary algorithm which constructs sets of composed images. A key element in our evolutionary algorithm are random walk algorithms used previously in the context of evolutionary image transition. A random walk can be run as part of a mutation phase to obtain transitional images between S and T. In the case of crossover, a random walk can be used to generate a crossover mask which is then used to produce an offspring from two parent individuals. The random walk operator introduced in [188] depends on a parameter $t_{\max}$ which determines how many steps a mutation operator performs. Here, with a view to developing an efficient process, we allow for self-adaptation of $t_{\max}$ during the evolutionary computation. This permit increase of $t_{\max}$ when the algorithm is making sufficient progress, and decrease of $t_{\max}$ when the progress is low. As a result, our evolutionary algorithm adapts to the different stages of the optimisation process. Self-adaption is a key element of evolutionary algorithms for continuous optimisation problems. Recently, Doerr and Doerr [39] have modified this notion for the discrete case and shown that a cogent use of the modification significantly speeds up the optimisation for simple benchmark functions. We use the self-adaptation mechanism in such a fashion that $t_{\max}$ decreases by a factor when an offspring is rejected and increases by a factor when an offspring is accepted.

In our experimental investigations we explore how different parameters influence the artistic merit of the resulting images. We explore the role of image features, distances between covariance matrices and region weighting schemes. One of our key innovations is to utilise a measure of visual attention to promote the incorporation of salient regions from both images into the new synthesised images.

In this chapter, we present in Section 6.2 a fitness function that we propose for image composition. In Section 6.3, we introduce an evolutionary algorithm including the mutation and crossover operators based on random walks. We present in Section 6.4, the results of our experimental investigations, and finally finish with a short discussion and some conclusions.

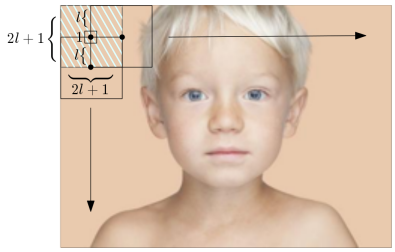


FIGURE 6.1: Illustration of overlapping square regions of interest.

6.2 Covariance-based Fitness Function

Given two images S and T both of size $m \times n$, we define the fitness of an image X of size $m \times n$ with respect to S and T. The fitness of the mixed image X is evaluated using region covariance descriptors that characterise how similar parts of the generated image are to both input images. We briefly recap the main aspect of the covariance descriptors from Chapter 2. The location and shape of the regions associated with the covariance descriptors can be chosen arbitrarily and are based on the personal preference of the artist. In this work we limit ourselves to square regions of interest arranged in a grid-like manner. In particular, we consider regions $\mathcal{R}_{(c,d)} = \{(i,j) \mid |i-c| \leq l, |j-d| \leq l\}$, which represent squares of size $(2l+1) \times (2l+1)$ centred at (c,d), where (c,d) runs over a grid of pixels $\mathcal{G}$. The grid is defined as

$$\mathcal{G} = \left\{ (c,d) \;\middle|\; \begin{array}{l} c = (l+1) + pl,\ p = 0, 1, \ldots, \left\lfloor \dfrac{m-l}{l} \right\rfloor - 1 \\ d = (l+1) + ql,\ q = 0, 1, \ldots, \left\lfloor \dfrac{n-l}{l} \right\rfloor - 1 \end{array} \right\},$$

which results in half-overlapping square regions of interest (see Figure 6.1).

To measure the similarity between the generated image and the input images we propose the fitness function

$$f(X,S,T) = \sum_{(c,d)\in\mathcal{G}} \Big(w^S_{(c,d)} \mathrm{dist}\left(\Lambda^X_{\mathcal{R}_{(c,d)}}, \Lambda^S_{\mathcal{R}_{(c,d)}}\right) + w^T_{(c,d)} \mathrm{dist}\left(\Lambda^X_{\mathcal{R}_{(c,d)}}, \Lambda^T_{\mathcal{R}_{(c,d)}}\right)\Big),$$

where dist is one of the distance functions described in Section 3, and $w^S_{(c,d)} \in [0,1]$ and $w^T_{(c,d)} \in [0,1]$ are weights associated with the region $\mathcal{R}_{(c,d)}$ that can be used to emphasise the local contribution of the images S and T in the mixing process.

In addition to the fitness function, we put forward a constraint. Let $c_S(X) = |\{X_{ij} \mid X_{ij} = S_{ij}\}|$ be the number of pixels in X that are set to S and $c_T(X) = |\{X_{ij} \mid X_{ij} =$

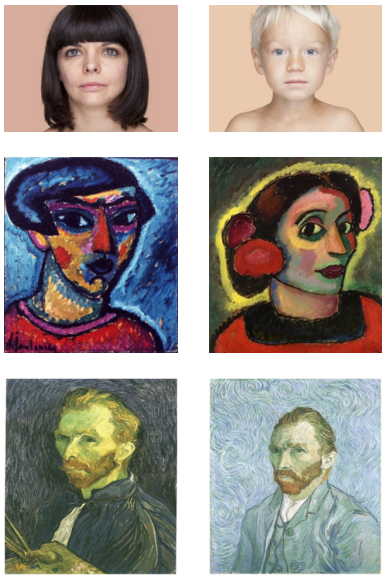


FIGURE 6.2: Pairs of images S (left column) and T (right column). Image credit (top row): ©Angélica Dass.

$T_{ij}\}|$ be the number of pixels where X and T agree. We minimise the fitness function f subject to the constraint

$$c(X) = |c_S(X) - c_T(X)| \leq B,$$

where B is an upper bound on how much the number of pixels in X from the two images S and T are allowed to differ.

6.3 Evolutionary Algorithm for Image Composition

In this section, we introduce our algorithm for creating artistic images using region covariance descriptors and an evolutionary image transition process. Our method takes as input two images, $S = (S_{ij})$ and $T = (T_{ij})$ of size $m \times n$, and produces a new images $X = (X_{ij})$ of the same size, that is a mixture of the input images, i.e. $X_{ij} \in \{S_{ij}, T_{ij}\}$ for each $1 \leq i \leq m$ and each $1 \leq j \leq n$. The mixed images are generated using the genetic algorithm given in Algorithm 10. The algorithm uses a parent population of μ images and produces in each iteration one image Y by crossover with probability p_c or by mutation with probability $1 - p_c$. The initial population is a multi-set of images of the given images S and T. More precisely, each initial individual P_i in the initial population is chosen uniformly at random from $\{S, T\}$. In order to produce a diverse set of images, an offspring Y is only competing against the first parent P_i for survival.

Algorithm 10 $(\mu + 1)$ GA for evolutionary image composition

Require: S and T are images

Initialise population $\mathcal{P} = \{P_1, \ldots, P_\mu\}$
while not termination condition **do**
  Select an individual $P_i \in \mathcal{P}$ uniformly at random
  if $rand() < p_c$ **then** ▷ Crossover
    Select $P_j \in \mathcal{P} \setminus P_i$ uniformly at random
    if $rand() < 0.5$ **then** ▷ See Section 6.3.2 for t_{cr}
      $Y \leftarrow$ RANDOMWALKMUTATION(X,Z,t_{cr})
    else
      $Y \leftarrow$ RECTANGULARCROSSOVER(P_i,P_j)
    $P_i \leftarrow$ SELECTION(P_i,Y)
  else ▷ Mutation
    if $rand() < 0.5$ **then**
      $Y \leftarrow$ RANDOMWALKMUTATION(P_i,S,$t_{\max}$)
    else
      $Y \leftarrow$ RANDOMWALKMUTATION(P_i,T,$t_{\max}$)
    $P_i \leftarrow$ SELECTION(P_i,Y)
    Adapt $t_{\max}$ ▷ See Section 6.3.1.
return $\mathcal{P}$ ▷ Result is a population of evolved images.

This leads to a low selection process and produces a diverse set of images in the final population. We describe the different components of the genetic algorithm in the following section.

6.3.1 Self Adaptive Random Walk Mutation

Our genetic algorithm uses a variant of the random walk mutation introduced in [188]. The mutation operator is shown in Algorithm 11. Given an image Z the random walk starts at a random pixel of the current image X and produces an offspring Y from X by setting each visited pixel Y_{ij} to the value of Z_{ij}.

The random walk moves from a current pixel X_{ij} either left, right, up, or down to visit the next pixel and does this for $t_{\max}$ steps. To formalise this, we define the neighbourhood $N(X_{ij})$ of X_{ij} as

$$N(X_{ij}) = \{X_{(i-1)j}, X_{(i+1)j}, X_{i(j-1)}X_{i(j+1)}\}$$

Algorithm 11 RANDOMWALKMUTATION($X, Z, t_{\max}$)

Require: X and Z are images

1: $Y \leftarrow X$
2: $Y_{ij} \in Y$ ▷ Choose starting pixel uniformly at random
3: $Y_{ij} \leftarrow Z_{ij}$
4: $t \leftarrow 1$
5: **while** $t \leq t_{\max}$ **do**
6: Choose $Y_{kl} \in N(Y_{ij})$ uniformly at random
7: $i \leftarrow k$, $j \leftarrow l$ and $Y_{ij} \leftarrow Z_{ij}$
8: $t \leftarrow t + 1$
9: **return** Y ▷ Result is a mutated image.

and choose in each step an element of $N(X_{ij})$ uniformly at random. We use modulo the dimensions of the image which implies that the walk may wrap around the boundaries of the image.

Given a current image X, our genetic algorithm uses the random walk mutation to either paint all the visited pixels with the same values as in S or T. Whether to choose S or T is decided uniformly at random for each mutation operation. In this way, the algorithm is able to produce images that are mixtures of S and T and minimise the given fitness function. Each random walk mutation is run for $t_{\max}$ steps.

Self Adaptation

When the algorithm successfully minimises the fitness function, we choose to increase the length of random walks by increasing $t_{\max}$. On the other hand, if progress can only be achieved by small random walk mutations, we decrease $t_{\max}$. In particular, we employ the approach for adjusting discrete parameters recently introduced by Doerr and Doerr [39]. Their approach adapts the classical $1/5$-rule adaptation for evolution strategies in [8] to the discrete setting. Our approach increases $t_{\max}$ in the case of a success and decreases $t_{\max}$ if the new offspring is not accepted. With the benefit of a self-adjustable mechanism, $t_{\max}$ can take on values in $t_{\mathrm{LB}} \leq t_{\max} \leq t_{\mathrm{UB}}$, where t_{LB} is a lower bound on $t_{\max}$ and t_{UB} is an upper bound on $t_{\max}$. This differs from the approach in [39], where the offspring population size is reduced in the case of a success and increased in the case of a failure.

For a successful mutation, we set

$$t_{\max} \coloneqq \min\{F \cdot t_{\max},\, t_{\mathrm{UB}}\}$$

and for an unsuccessful mutation, we set

$$t_{\max} \coloneqq \max\left\{F^{-1/k} \cdot t_{\max},\, t_{\mathrm{LB}}\right\},$$

where $F > 1$ is a real value and $k \geq 1$ an integer which determines the adaptation scheme.

The scheme implements a $1/(k+1)$-rule where F determines the size of the step changes dependent on the current value of $t_{\max}$. For our experimental investigations, we set $t_{\mathrm{LB}} = 50$, $t_{\mathrm{UB}} = 5000$, $F = 2$, $k = 8$ based on preliminary experimental investigations. Note that the use of $t_{\max}$ in Algorithm 11 does not require it to be an integer. For a positive real value $t_{\max}$, the mutation operator will carry out $\lfloor t_{\max} \rfloor$ steps of the random walk.

6.3.2 Crossover

We consider two crossover operators which take two parents and produce one child.

The first crossover operator produces an offspring Y by taking the image of the first parent P_i and performing a random walk for t_{cr} steps on the image of the second parent P_j. This means that the offspring Y can be produced by calling RANDOMWALKMUTATION(P_i, P_j, t_{cr}). For our experimental investigations in this paper, we set $t_{\mathrm{cr}} = 10000$, i.e. each random walk crossover mask is generated by a random walk consisting of 10000 steps. Note, that this value is independent of $t_{\max}$ which is adapted during the mutation step. The reason is that we always want to make sure that crossover reduces an offspring which has sufficiently large parts from both parents.

The second crossover operator that we dub Rectangular Crossover produces an offspring where all entries are copied from parent P_i except from a randomly specified rectangle R. The upper left point of the rectangle is chosen uniformly at random from the image. The width and the height of the image is chosen as a random integer in $\{1, \ldots, m/10\}$ and $\{1, \ldots, n/10\}$, respectively. Values where the width and/or height would exceed the image boundaries are ignored.

Both crossover operators are focused on a local part of the target image. This facilitates the creation of offspring that have an identified part of the image of the second parent substituted into the image of the first parent. The hope is that the optimisation process benefits from this structured way of creating offspring.

6.4 Experimental Investigations

FIGURE 6.3: Image composition with different features. Rows 1, 2 and 3 correspond to Feature Sets 1, 2 and 3, respectively. Note the structure that emerges with the first feature set.

Our experiments were designed to meet a threefold objective: (1) we wished to investigate the impact of different region covariance features on the resulting images; (2) we wanted to discover how different weighting schemes for covariance matrices influence the results; and (3) we wished to understand the influence that the distance measures have on the final results.

The pairs of images S and T for all of our experiments are depicted in Figure 6.2. For each setting investigated in this section, we ran our genetic algorithm for 2000 generations with a population size of $\mu = 4$ and and crossover probability $p_c = 0.2$.

6.4.1 Impact of Features

Our first set of experiments were focused on characterising how the choice of features may influence the resulting image. With reference to Table 3.1 we chose 3 sets of features as follows:

Set 1: $\left[i, j, r, g, b, \sqrt{\left(\frac{\partial I}{\partial i}\right)^2 + \left(\frac{\partial I}{\partial j}\right)^2}, \tan^{-1}\left(\left|\frac{\partial I}{\partial i}\right| / \left|\frac{\partial I}{\partial j}\right|\right)\right]^{\mathsf{T}}$;

Set 2: $[i, j, h, s, v]^{\mathsf{T}}$;

FIGURE 6.4: Image composition with different covariance weighting schemes. Rows 1, 2 and 3 correspond to $w^S_{(c,d)}$ set to 0.25, 0.5 and 0.75 and $w^T_{(c,d)}$ set to 0.75, 0.5 and 0.25, respectively. In the last row the weights were set using an image saliency algorithm. The saliency algorithm strikes a consistent balance between notable regions in both images.

Set 3: $\left[h, s, v, \sqrt{\left(\frac{\partial I}{\partial i}\right)^2 + \left(\frac{\partial I}{\partial j}\right)^2}, \tan^{-1}\left(\left|\frac{\partial I}{\partial i}\right| / \left|\frac{\partial I}{\partial j}\right|\right)\right]^{\mathsf{T}}$.

For all experiments in this section we considered a grid of equispaced square region covariance descriptors ($l = 25$ pixels). We set both sets of weights $w^S_{(c,d)}$ and $w^T_{(c,d)}$ to 0.5, and utilised the Euclidean distance as our measure of similarity between covariance matrices.

The results in Figure 6.3 demonstrate that the three populations evolve differently depending on the feature set. Feature Set 1 produced the most visually pleasing results, with the composite image incorporating facial features from both subjects. We attribute this to the correlations that are captured between pixel locations and edge features (magnitude and orientation) in the first feature set, and not in the other two feature sets.

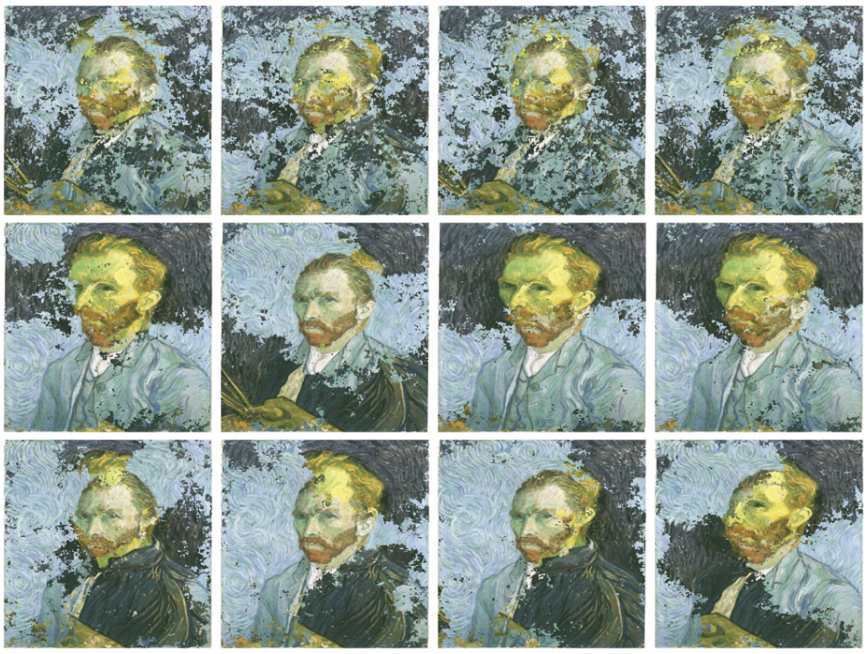


FIGURE 6.5: Image composition with different covariance distances. Rows 1, 2 and 3 correspond to distance metrics $\mathrm{dist}_{\mathrm{E}}$, $\mathrm{dist}_{\mathrm{A}}$ and $\mathrm{dist}_{\mathrm{L}}$, respectively. Note how the Euclidean distance yields patchy results, whereas the other distance measures result in more structured images.

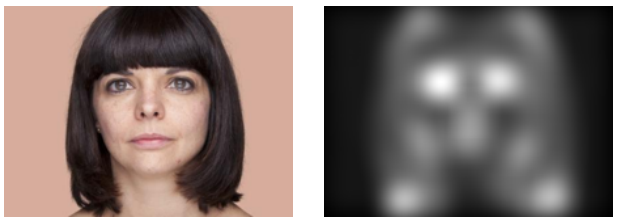


FIGURE 6.6: An image and its corresponding saliency mask.

6.4.2 Impact of Different Weightings

In the second set of experiments we explored the consequence of using different weights for the region covariance matrices. We limited ourselves to two weighting schemes: uniform and saliency-based. In the uniform regime all of the weights $w^S_{(c,d)}$ associate with S were set to the same value, and, similarly, all the weights $w^T_{(c,d)}$ associated with T were also fixed. In particular, for simulations 1, 2 and 3, we set $w^S_{(c,d)}$ to 0.2, 0.5 and 0.75 and $w^T_{(c,d)}$ to 0.75, 0.5 and 0.25, respectively.

In contrast, for a simulation with saliency-based weighting, we utilised the image saliency algorithm of [107] to assign weights for each region covariance descriptor. The purpose of the saliency algorithm is to predict human fixation points, which in turn are used as a measure of visual attention [213]. The saliency algorithm takes as

FIGURE 6.7: Image composition using Feature Set 1 with saliency-based weighting and the Log-Euclidean distance measure.

input an image, and produces a new image of the same size which can be interpreted as a probability map designating which regions of an image a human would pay most of their attention to (see Figure 6.6). We ran the saliency algorithm on both S and T. The weights for $w^S_{(c,d)}$ and $w^T_{(c,d)}$ were taken from the saliency maps associated with S and T, respectively.

For all experiments in this section we used Feature Set 1 and employed a grid of equispaced square region covariance descriptors ($l = 20$ pixels). We measured the similarity between covariance matrices using the Log-Euclidean distance.

The results of our experiment are depicted in Figure 6.4. As expected, when uniform weights are used the global relative contribution of the image S can be traded-off against the global relative contribution of the image T. The benefit of the saliency-based weighting scheme is that it can automatically produce visually pleasing results that strike a balance between notable regions in both images.

6.4.3 Impact of Distance Measures

In the third and final set of experiments we investigated the utility of different metrics for covariance matrices. Once again, a notable structure emerges. The Euclidean distance measure produced inferior results, with the images taking on a very patchy and

noisy appearance. In contrast, the images generated using the Log-Euclidean or affine-invariant distance were much more cohesive and interesting. The results presented in Figure 6.5 are based on grid of equispaced square region covariance descriptors ($l = 20$ pixels) using saliency-based weights and Feature Set 1.

6.5 Conclusions

Our experimental examinations suggest that the best results are obtained when Feature Set 1 is used in conjunction with saliency-based weighting and a Log-Euclidean distance measure. To verify this hypothesis, we deployed our genetic algorithm one more time with these parameters on all three image pairs. The results presented in Figure 6.7 confirm our hypothesis. In all cases, our algorithm produced a population of new aesthetic images that incorporate pertinent regions from both input images. These images mimic the distinctive styles of the self-portraits, combining the explosive colours of the fauvism movement and the multiple emotions displayed in the cubism abstract art style.

We have introduced a new approach of image composition based on feature covariance matrices. This approach facilitates the composition of novel artistic images. We have introduced a genetic algorithm evolving a set of images by crossover and mutation operators based on random walk. For our experimental investigations, we have considered different pairs of images and compared the final populations showing the resulting images with respect to different parameters.

A conference version containing the results of this chapter has been published in the Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2017 (see A. Neumann, Z. Szpak, W. Chojnacki and F. Neumann (2017) [196]).

Chapter 7

Evolutionary Image Composition Using Colour-based Segmentation

7.1 Introduction

In the previous chapter, we introduced an evolutionary image composition approach based on feature covariance matrices [196] which allows for two images into a new one based on their feature characteristics to be composed. When using evolutionary image composition, it is important to obtain a good weighting of interesting regions of the two images. In this chapter, we consider colour-based segmentation based on K-Means clustering to come up with such a weighting of the images. Our results show that this preserves the chosen colour regions of the images and leads to composed images that preserve colours better than the previous approach based on saliency masks [196]. Furthermore, we evaluate our composed images in terms of aesthetic features and show that our approach based on colour-based segmentation leads to higher feature values for most of the investigated features.

In this chapter, we investigate the use of image segmentation methods in evolutionary image composition. We introduce segmentation-based evolutionary image composition (SEIC) which makes use of image segmentation methods to discover with suitable weightings of different regions of the given images in order to produce a good composition of the images. Our segmentation method uses the importance of colours in order to come up with good weightings. The perception of colour is important for the perception of images [146] and the use of colours can be seen as a product of our language [161] and culture [72].

We use image segmentation based on K-Means which converts a given colour image as shown in Figure 7.1 into a *Lab* colour space image and separates the positions of a set of colours. The segmentation is then used in the image composition process in

FIGURE 7.1: Image S (Spanish Woman, 1911 by Alexej von Jawlesnky) and image T (Head in blue, 1912 by Alexej von Jawlesnky).

FIGURE 7.2: Images with corresponding segmentations regions with blue-coloured schema in image T and yellow-coloured schema in image S (from top left, respectively).

order to come up with images that minimize the error in terms of the weighted covariance featured-based error function where weights are set according to the colour-based segmentation.

We evaluate our approach by feature-based analyses which reveal that aesthetic feature values for almost all the images are higher using our segmentation approach compared to the evolutionary image composition approaches investigated in [196].

This chapter is structured as follows. In Section 7.2, we introduce the evolutionary image composition approach that we investigated. In Section 7.3, we introduce the different image segmentation methods that we used to obtain the weighting for evolutionary image composition. In Section 7.4, we report our experimental results and compare the different weightings based on the investigated image segmentation methods. Finally, we finish with some conclusions.

7.2 Evolutionary Image Composition

Evolutionary image composition creates a composition of two given images taking into account important properties of these images. Given two images, $S = (S_{ij})$ and $T = (T_{ij})$ of size $m \times n$, it produces a new image $X = (X_{ij})$ of the same size, that is a mixture of the input images, i.e. $X_{ij} \in \{S_{ij}, T_{ij}\}$ for each $1 \leq i \leq m$ and each $1 \leq j \leq n$.

We investigate the evolutionary image composition approach based on a using feature covariance matrices given in [196]. The approach uses a $(\mu + 1)$-EA which aims to produce a set of μ images by mutation and crossover based on random walks. A detailed description of the algorithm we presented in Chapter 4.

The key part of the image composition approach in [196] is the used fitness function, which is very flexible and can take various features of the images into account. Given two images S and T both of size $m \times n$, the goal is to minimize the fitness of a composed image X of size $m \times n$ given by

$$f(X,S,T) = \sum_{(c,d)\in\mathcal{G}} \sum_{(i,j)\in\mathcal{R}_{(c,d)}} \Big(w^S_{(i,j)} \mathrm{dist}\left(\Lambda^X_{(i,j)}, \Lambda^S_{(i,j)}\right) + w^T_{(i,j)} \mathrm{dist}\left(\Lambda^X_{(i,j)}, \Lambda^T_{(i,j)}\right)\Big).$$

The fitness function f measures the error of X with respect to S and T. Here dist is a distance function, and $w^S_{(i,j)}$ and $w^T_{(i,j)}$ are weights for S and T associated with pixel (i,j), $1 \le i \le m$, $1 \le j \le n$. $\mathcal{R}_{(c,d)}$ denotes a region of the image and $\Lambda^I_{(i,j)}$ the local covariance matrix of image I centered at pixel (i,j).

We now briefly recap the main aspect of a region covariance matrix from Chapter 6. We consider square regions in a grid-like manner determined by parameter l called the half-window size. The regions $\mathcal{R}_{(c,d)} = \{(i,j) \mid |i-c| \le l, |j-d| \le l\}$ are squares of size $(2l+1)\times(2l+1)$ centered at points (c,d) given by a grid of pixels $\mathcal{G}$. Formally, the grid is defined as

$$\mathcal{G} = \left\{ (c,d) \;\middle|\; \begin{array}{l} c = (l+1) + pl,\ p = 0,1,\ldots,\left\lfloor \dfrac{m-l}{l} \right\rfloor - 1 \\ d = (l+1) + ql,\ q = 0,1,\ldots,\left\lfloor \dfrac{n-l}{l} \right\rfloor - 1 \end{array} \right\},$$

which results in half-overlapping square regions.

In order to make sure that each composed image X has a certain number of pixels from S and T the following constraint is used. Let $c_S(X) = |\{X_{ij} \mid X_{ij} = S_{ij}\}|$ be the number of pixels in X that are set to S and $c_T(X) = |\{X_{ij} \mid X_{ij} = T_{ij}\}|$ be the number of pixels where X and T agree. The fitness function f is minimized subject to the constraint

$$c(X) = |c_S(X) - c_T(X)| \le B,$$

which says that the number of pixels in X belonging to S and T can differ by at most B.

In previous chapter, we have shown that the chosen distance function, the chosen features, and the weighting of the different pixels of the image play a crucial role for the outcome of the considered evolutionary image composition approach. In particular, a good weighting of regions of interest in the two given images S and T is crucial and the best results have been obtained in [196] by using a saliency mask to obtain the weights $w^S_{(i,j)} \in \mathbb{R}^{m \times n}$ and $w^T_{(i,j)} \in \mathbb{R}^{m \times n}$ for the given images S and T. We will explore the use of image segmentation methods to come up with suitable weightings of the different regions.

7.3 Colour-based Image Segmentation

Clustering is considered an important unsupervised machine learning problem. Data elements are partitioned into clusters that represent approximate collections of data elements based on a distance function. Clustering analysis has found many applications in areas such as data mining [58, 61] and data compression [207].

It has been shown that different weighting schemes of the image have an important role in determining the weighting of the final outcome of the evolutionary image composition. In this chapter, we investigate different weightings of regions using the segmentation method. We denote two given images S and T of the size $m \times n$ that we can randomly select from a given image database. We utilize different segmentation masks w^S and w^T that are calculated for both image S and T in order to distinguish different images using evolutionary image composition. We use the weighting matrices $w^S \in \mathbb{R}^{m \times n}$ and $w^T \in \mathbb{R}^{m \times n}$ that are associated with images S and T. The weights are obtained by colour-based K-Means segmentation methods which we describe in the following.

7.3.1 K-Means Clustering for Colour-based Segmentation

K-Means is one of the popular algorithms for clustering unsupervised data [7]. We use Lloyd's algorithm, which is best known as the K-Means algorithm, to solve the K-Means clustering problem [159]. The algorithm is based on the observation that the optimal distribution of a center occurs at the centroid of the associated cluster [94]. An important component of a clustering problem is the distance between data points. In order to group similar points together, we can use the *Squared Euclidean Distance* metric (or any other suitable distance metric) to calculate the distance between points.

Given n data points, the data is partitioned in k clusters where k is a parameter chosen by the user. Each cluster has a cluster center which is given by the mean of

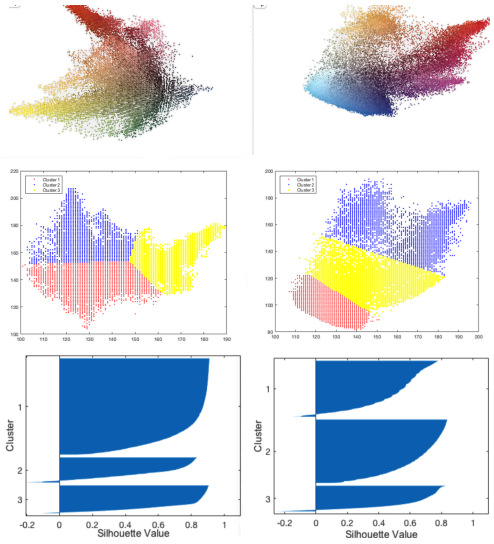


FIGURE 7.3: *Lab* colour classes of image S and image T, cluster assignments, and silhouette values from clustered data (from top left, respectively).

the points in the cluster. The algorithm assigns each of the n data points to the closest cluster center. Centers are updated by taking the mean of each cluster and the process is iterated. The K-Means algorithm aims to find the positions i, $i = 1, \ldots, k$ on the cluster which minimize the distance from those points to the cluster.

In Figure 7.3, we visualize the *Lab* colour space for images S and T. This enables us to quantify the visual differences. We show the adequate cluster assignments and plot the silhouette values from the clustered data for each image. Silhouette values from both images give us information on how similar these points are compared to other points in its own and the other two clusters.

A key concept for our approach is the use of the K-Means clustering algorithm for colour-based segmentation in order to separate various colours in an automated procedure. This happens by using the *Lab* colour space in the K-Means clustering algorithm. The *Lab* colour space is expressed by a luminosity 'L', also called the brightness layer, the chromaticity layer 'a' that specifies where colours take place along the red-green axis, and the chromaticity layer 'b' which determines where colours take place along the blue-yellow axis. The steps of the processes are as follows.

Firstly, we convert the image from *RGB* colour space to *Lab* colour space. Then, we classify the colours in the $a \times b$ space using K-Means clustering. The color information remains in the $a \times b$ space. We use K-Means to cluster the objects into three clusters using the Euclidean distance metric. As input into the K-Means algorithm we consider our objects that are the pixels of the given image with a and b values. We have to

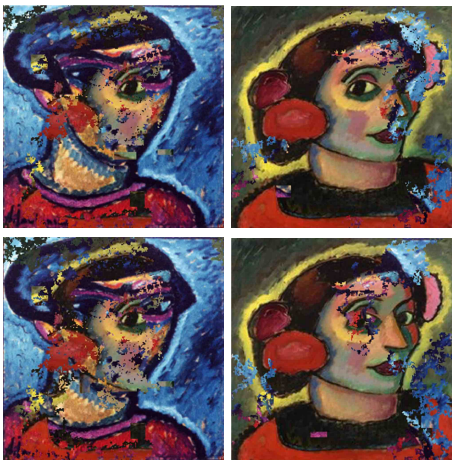


FIGURE 7.4: Two Pairs of evolved images after 2000 generations conducting K-Means segmentation correspond to blue-coloured clustering schema.

specify the number of clusters that we want to use in K-Means. For our experiments, we use $K = 3$ which results in 3 clusters.

Afterwards, we label every pixel in the image with using the results from K-Means with its own index corresponding to a cluster. We separate objects in the image by color. Note that for our setting the results consist of three images as we set $K = 3$ in the K-Means algorithm.

For our experiments, we use the K-means segmentation method implemented in Matlab using the Statistics and Machine Learning Toolbox [110]. K-Means assigns index numbers 1, 2, and 3 to pixels. The pixels in a particular given image are segmented into three colour spectra: yellow, blue, and red. We do this for the images S and T as shown in Figure 7.2. In the K-Means algorithm, the procedure uses the squared Euclidean distance metric. The weights for $w^S_{(i,j)}$ and $w^T_{(i,j)}$ are given by the segmented images of a chosen color associated with S and T, respectively.

7.4 Experimental Investigations

In this section, we carry out our experimental investigations. We compare the use of colour-based segmentation in evolutionary image composition to the one using saliency masks, which is the best performing method given in [196]. Furthermore, we investigate different parameter settings such as half-window size and different distance measure for evolutionary image composition in conjunction with colour-based segmentation.

FIGURE 7.5: Two pairs of evolved images S and T after 2000 generations using K-Means segmentation. Row 1 corresponds to yellow-coloured clustering schema, and row 2 correspond to red-coloured clustering schema.

We use the pairs of images S and T of the size 240 x 240 pixels for all experiments illustrated in Figure 7.1. For each set of the experiments, we use the same setting as in [196] and run our algorithm for 2000 generations, t_{max} = 1024, population size of $\mu = 4$, and crossover probability $p_c = 0.2$. Also, the upper bound B for the constraint $c(X)$ was assumed as 5000 pixels. At last, we inherited a grid of squared region covariance descriptors ($l = 20$ pixels). We compare the results using colour-based segmentation to the best performing approaches in [196] which uses a saliency mask to obtain the weighting of the images. The experiments have been carried out on the supercomputer Phoenix, on a single node of a Lenovo NeXtScale M5 Cluster including two Intel® Xeon® E5-2600 v4 series 16 core processors with 64GB of RAM. We run experiments with varying preferences for 30 times to see the effect of different settings on artistic outcomes as described below.

7.4.1 Impact of Colour-based Segmentation

We carry out three experiments where our main objective is to identify how the choice of different clustering sets with respect to the *Lab* values affect the final outcome. The *Lab* colour space per definition outlines all colours in the three dimensions [109]. For the experiments, we choose a grid of square region covariance descriptors and set the half-window size $l = 20$ (see Section 7.2). As we can see our population took the evolutionary process differently based on segmentation outcome for the S and T images

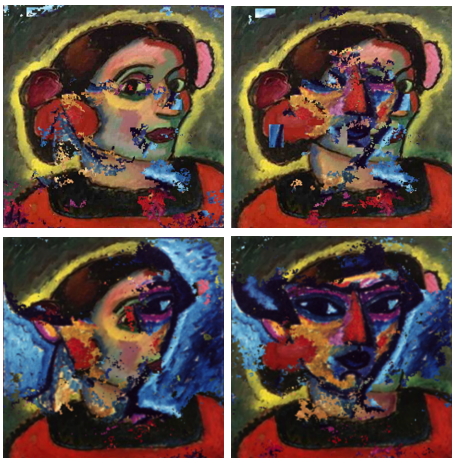


FIGURE 7.6: Population of evolved images S and T after 2000 generations using K-Means segmentation and yellow-coloured clustering schema (top), and one pair of evolved images S and T after 2000 generations using the saliency weighting schema from [196] (bottom).

in comparison to the result presented in the research paper [196]. Our result preserve parts of both images in a special way as the faces are mostly visible and in the background we can only see the second given image. Inspired by the visually different outcomes of our approaches, we investigate two new variants of our algorithm. We employ the K-Means segmentation method and cluster our given images with respect to the red and yellow pixel values in *Lab* colour space. In Figure 7.5, we see two pairs of evolved images S and T with respect to the yellow- and red-coloured clustering schema. The created images are different to each other and to the one obtained in the first experiment. Interesting properties are occurring on the red-coloured clustering methods, as new forms reveal a surprising composition of the image. In the first row of Figure 7.6, we see two images created with our approach using the yellow-coloured schema. In the second row, we see two images created with the evolutionary image composition approach using the saliency weighting schema [196]. The images in the first row contain visible yellow parts from the given image. In contrast, images in the second row appear more distracted. We can not recognise the dominant yellow lines. Our algorithm preserves the yellow colour from a given image during the evolutionary processes. This makes it possible to evolve images towards the colour that we wish to be most present in newly created images.

7.4.2 Impact of Distance Measures

We now describe experiments carried out when using different distance measure. We use three distance measures for minimization of K-Means segmentation as follows:

FIGURE 7.7: Evolved images with K-Means segmentation with different distance measures. Row 1 corresponds to distance measures cityblock and row 2 to cosine, respectively. Note the structure that emerges with the first window set.

square Euclidean, *cityblock* and *cosine* [67]. The K-Means segmentation algorithm computes centroid clusters uniquely for each of the type of distance measurements. At this point, we shortly outline the main properties of the distance measurements. Note x is an observation specified as a row of particular X whilst c is a centroid specified as a row vector. *Squared Euclidean Distance Measures* are standardized measures where each centroid c is the mean of the points in an explicitly cluster. *Cityblock Distance Measures* is a sum of absolute differences. This means that each centroid c is the component-wise median of the points in the cluster.

Cosine Function Distance Measures is one minus the cosine of the all enclosed angle between points. This means that the particular centroid c is the mean of the points in the cluster [21]. In Figure 7.7, we see a different series of images where the structure of both images is changed. The distance measures have the primary role in the process of creating the most diverse results.

7.4.3 Impact of Different Sizes of the Regions

We now explore how the use of different half-window sizes l influence the generation of images. Figure 7.8 shows that the window sizes have a clear influence on the final images. We can distinguish between two categories where smaller half-window sizes $l = 5, 20$ produce clearer images. The larger half-window sizes $l = 40, 120$ produce more chaotic patches over different parts of the images. Choosing an appropriate value

FIGURE 7.8: Evolved images with K-Means segmentation with different windows sizes. Rows 1, 2, 3 and 4 correspond to windows size 5, 20, 40 and 120, respectively.

for the half-window size l can therefore be used to influence the final outcome with the most desired characteristic.

7.4.4 Feature-based Analysis

We now analyze the different colour-based segmentation methods for evolutionary image composition with respect to features that measure aesthetic values. Firstly, we analyze the final aesthetic feature values for different colour-based segmentation settings. Furthermore, we compare the different set of parameters against each other.

The set of features we use are, in order of appearance, *Global Contrast Factor* [172], *Ross-Bell-Curve* [224], *Benford's Law* [119], *Saturation*, *reflectionary Symmetry* [99], *Mean Hue*, *Symmetry*, *Standart Deviation Hue* and *Smoothness* [203] (see Chapter 3).

We carry out several aesthetic measurements on the final images that were evolve

with previous EIC and our colour-based SEIC algorithm. Table 7.1 shows the comparison of the results of the evolved images obtained with segmentation in respect to blue, yellow and red colour space, using cityblock and cosine distance measures. Table 7.2 shows different window sizes of the covariance regions and the image S and T. The columns in the tables present the mean and standard deviation of the different settings for each feature obtained from the first image from the final population in each run. It should be noted that the algorithms do not have any bias regarding particular images in the population.

The first two columns in Table 7.1 give the mean and standard deviation for the approach using the saliency mask [196]. The next three columns give the feature values obtained by using the blue, yellow and the red-based segmentation schema (ECLU.1, ECLU.2, ECLU.3). The last columns in Table 7.1 list the features values for cityblock and cosine distance measures. Images evolved with cosine measurement distance and blue colour-based schema achieve by most of the aesthetic features the highest feature values. However, different aesthetic features give us valuable insights into how the user can create aesthetic images with more control over the evolutionary process.

Table 7.2 list the features values obtained by using different window sizes of the covariance regions, the image S and T. In the Table 7.1 and 7.2 we see the correlation between different sizes of regions, different distance measures and different colour-based schema segmentation. We can observe that images evolve with the windows size 20, 40, 120, the cosine distance measure and the blue, yellow and red colour-based schema have the highest value for *reflectional Symmetry*. This mean that regarding this feature the images have also the highest aesthetic value. Similar, the strongest performance for *GCF* achieve images evolved with the windows size 120, the cosine measurements distance and the blue colore-base schema. In contrast, the lowest score for *Benfords'law* has only the yellow colour-based schema. Overall, the images evolve with SIEC have higher *Saturation* values excepts window size 120 and the yellow colour-based schema.

Furthermore, we conduct human subject experiments in order to evaluate the image results presented in our work and in EIC [196]. The goal of the experiment is to gain knowledge about user preferences for images. We carry out a human-based experiment by using Amazon Mechanical Turk [20] with 20 participating users. We present each human two images at a time that correspond to EIC with the saliency weighting schema and the yellow-coloured schema SEIC shown in Figure 7.6. The results reveal that the human subjects preferred 95% of the time images created with our SEIC methods. It would be interesting to examine in further study the preferences of the human subject by adding all evolved images with different parameters. It would

TABLE 7.1: Results of aesthetic measurements using previously EIC approach with saliency wighting and colour-based SEIC using ECLUE.1 blue-coloured, ECLU.2 yellow-coloured, ECLUE.3 red-coloured spaces, cityblock and cosine distance measures respectively. Note bold text values have the highest score comparing the two approaches.

Features	**Saliency [196] mean**	**Saliency std**	ECLU.1 mean	ECLU.1 std	ECLU.2 mean	ECLU.2 std	ECLU.3 mean	ECLU.3 std	city. mean	city. std	cosine mean	cosine std
Glob. Contrast Factor	0.0395	0.0004	**0.0396**	0.0004	0.0391	0.0015	0.0393	0.0006	0.0395	0.0002	**0.0396**	0.0003
Ross Bell Curve	0.0129	0.0004	**0.0164**	0.0030	**0.0174**	0.0049	0.0168	0.0026	**0.0149**	0.0012	**0.0147**	0.0001
Benford's Law	0.8471	0.0170	**0.8660**	0.0207	0.8464	0.0594	**0.8674**	0.0141	**0.8513**	0.0116	**0.8652**	0.0197
Saturation	0.3375	0.0067	**0.3433**	0.0284	0.3311	0.0441	**0.3403**	0.0286	**0.3441**	0.0193	**0.3572**	0.0339
Ref.Symmetry	0.4069	0.0040	**0.4168**	0.0178	**0.4324**	0.0511	**0.4132**	0.0188	**0.4075**	0.0086	0.4058	0.0078
Hue	0.3295	0.0064	0.2855	0.1143	0.2869	0.0792	0.2790	0.1011	0.3129	0.0534	**0.3469**	0.0834
Symmetry	0.8244	0.0025	0.8231	0.0084	0.8220	0.0110	0.8195	0.0074	0.8199	0.0034	0.8193	0.0060
SDHue	0.7010	0.0097	0.6206	0.1031	0.6699	0.0401	0.6146	0.0931	0.6607	0.0356	0.6800	0.0333
Smoothness	0.9451	0.0010	**0.9504**	0.0047	0.8716	0.1273	**0.9505**	0.0044	**0.9477**	0.0025	**0.9486**	0.0014

TABLE 7.2: Results of aesthetic measurements using colour-based K-Means SEIC approach for window size 5, 20, 40, 120, the image S and T, respectively. Note bold text values have the highest score comparing the two approaches.

Features	W-5 mean	W-5 std	W-20 mean	W-20 std	W-40 mean	W-40 std	W-120 mean	$W-120$ std	Image S	Image T
Glob. Contrast Factor	0.0393	0.0005	0.0392	0.0002	0.0393	0.0004	**0.0400**	0.0002	0.0198	0.0212
Ross Bell Curve	**0.0230**	0.0103	**0.0172**	0.0029	**0.0171**	0.0035	**0.0136**	0.0003	0.9897	0.5715
Benford's Law	**0.8555**	0.0129	**0.8504**	0.0097	**0.8584**	0.0115	**0.8534**	0.0111	0.7940	0.7752
Saturation	**0.3614**	0.0259	**0.3430**	0.0213	**0.3457**	0.0292	0.2207	0.1184	0.4564	0.5844
Ref.Symmetry	0.3897	0.0412	**0.4085**	0.0161	**0.4135**	0.0167	**0.4071**	0.0004	0.2295	0.1684
Hue	0.2877	0.1160	0.2709	0.1006	0.2843	0.1236	0.3012	0.0017	0.1875	0.6329
Symmetry	0.8200	0.0075	0.8203	0.0066	0.8225	0.0063	0.8224	0.0010	0.8006	0.7495
SDHue	0.6231	0.0837	0.6322	0.0830	0.6070	0.1079	0.6814	0.0126	0.4623	0.4697
Smoothness	**0.9518**	0.0042	**0.9504**	0.0044	**0.9518**	0.0053	0.9426	0.0010	0.9624	0.9427

also be valuable to have more insights into the human subjects in terms of correlation between their preferences, age, education, sex etc.

7.5 Conclusions

Colours play an important role when creating appealing images. We have shown how to use colour-based image segmentation within the evolutionary image composition approach given in [196] and showed that the use of colour-based segmentation allows one to create composed images based on their colour characteristics. The evaluation based on the feature-based analysis shows that aesthetic feature values for the images created with our colour-based segmentation approach have a higher value in various aesthetic features compared to the previous evolutionary image composition based on saliency masks. Additionally, we studied the effects of different parameters in our algorithm. For future research, it would be interesting to explore more complex fitness functions and their special effect on the evolutionary processes that can produce new abstract images.

A conference version has been published in the Proceedings of Congress on Evolutionary Computation (CEC) 2018 (see A. Neumann and F. Neumann (2018) [191]).

Part IV

Diversity Optimization

Chapter 8

Evolutionary Diversity Optimization for Images

8.1 Introduction

In this chapter, we present the search framework to evolve diverse variants of a source image in one and two feature dimensions. This approach aims to directly evolve populations for diversity in one or more image features. We use the $(\mu + \lambda) - EA_D$ algorithm defined in [74]. This algorithm was originally used to evolve hard and easy TSP instances. Here, we use $(\mu + \lambda) - EA_D$ to evolve diverse image instances. By directly targeting feature diversity we gain a broader picture of what each image feature "sees" by being able to view images that score across the range of that feature. We can also see how different image features combine to produce interesting image effects. Finally, we can also study how different combinations of image features correlate when images are evolved to maximise diversity. Such correlations can point to the combinations of features which permit the evolution of diverse images across each feature dimension.

A primary aim of evolutionary art is to generate aesthetically pleasing artistic images. One approach to achieving this end is to evaluate the fitness of candidate images automatically during evolution according to one or more aesthetic feature metrics [99, 138, 224]. An aesthetic feature metric is a mapping of an image to a scalar value that serves as a proxy for aesthetic property of that image. In evolutionary art, features that are used to evaluate images include those that measure colour and intensity distributions [119, 219, 224] and those that measure the structural properties of an image [99, 172].

Different combinations of aesthetic feature metrics and evolutionary frameworks give rise to different artistic images. Some work in this field has investigated the relationship between feature metrics and images produced [28, 99]. More recent work has

explored how feature metrics correlate within the images produced by the evolutionary process [99].

By learning more about how these feature metrics are related, practitioners can make informed decisions about which combinations of features are likely to produce the most diverse images. However, work done to date on correlating features has either concentrated on sampling feature values in images evolved to *maximise* one or more of these feature values. There has been no work done on how features relate across their spectrum including for medium and low values in the range of each feature. Moreover, there has also been no work on assessing the effect of selecting a population of images primarily on their contribution to diversity.

The primary contributions of this work are: applying a methodology for evolving image variants to maximise diversity in image feature metrics; showing how image variants change across the spectrum of feature values; demonstrate how correlations between feature metrics can be obtained mapping populations evolved for multiple features; and presenting preliminary finding of what combination of feature metrics produce the most interesting artistic images. The results presented here form a proof-of-concept that lay the groundwork for future statistical studies.

In evolutionary art, aesthetic and general feature metrics have been applied to the production of new images using several evolutionary frameworks [28, 99, 138, 167]. More recent work has correlated features in the artworks produced by evolutionary search [99] to determine how much aesthetic feature metrics agree with each other (and themselves) when applied to evolved images. In a more general feature setting Machado [28] used features embedded in cascading classifiers to create images from learned categories. Other recent work has focused on tracking feature values during an evolutionary image transition process [188].

There is also much work in the domain of using feature search to produce image variants. Recent examples of such work include the generation of art from image transitions [188, 189]; Gaty's work using deep learning to transfer artistic style to existing images [75]; and the use of priors from a Deep Generative Network to generate image variants within a defined category [201].

Finally, there is related work that aims to improve the diversity of populations in evolutionary art. Such work includes the use of island models to improve exploration [99]; measures that favor image novelty [138, 261]; and work that favours individuals that spawn novel offspring [137]; and work using coevolutionary artist/critic models to improve novelty [168].

Algorithm 12 The $(\mu + \lambda) - EA_D$

input: an image S.
output: a population $P = \{I_1, \ldots, I_\mu\}$ of image variants. ▷ Initialise with μ mutated copies of source image
$P = \{mutate(S), \ldots, mutate(S)\}$
repeat
    randomly select $C \subseteq P$ where $|C| = \lambda$
    for $I \in C$ **do**
        produce $I' = mutate(I)$
        if $valid(I')$ **then**
            add I' to P
    while $|P| > \mu$ **do**
        remove TEXT an individual I = **arg min**$_{J \in P} d(J, P)$
until termination condition reached.

Our approach differs from this previous work because we aim to maximise a population diversity measure directly in the feature space rather than indirectly through searching for areas of novelty in the feature or image space. In contrast, our approach targeting diversity directly in order to maximise the coverage of the feature space by a population of individual images.

The remainder of this chapter is structured as follows. After giving a motivation for our approach, we introduce in Section 8.2 feature-based evolutionary diversity optimization for images. Section 8.3 presents our experimental investigations in single and two-dimensional feature space. Finally, Section 8.4 describes our conclusions and discusses future work.

8.2 Feature-based Diversity Optimization

The evolutionary algorithm we use here is the $(\mu+\lambda)-EA_D$ algorithm defined in [74]. A version of $(\mu + \lambda) - EA_D$, adapted for the production of images, is shown in Algorithm 12.

The algorithm is structured as a $(\mu + \lambda) - EA$ which starts with a population of μ image variants. In each iteration the algorithm produces λ new variants, which are checked for validity and added to the new population. Then the entire population is scanned to remove the variants that contribute the least to feature diversity in the population. Once the size of the population is reduced back to μ again the algorithm

FIGURE 8.1: Church benchmark starting image.

proceeds to the next iteration. In all of our experiments we used $\mu = 20$ and $\lambda = 10$ which gives a reasonable compromise between the potential for population diversity and evolutionary speed.

There are several elements of the above algorithm that are specialised to our application domain. We discuss these in turn.

The starting value S is a colour image. In our experiments we used two starting images. The first is a uniform grey square where each colour channel is initialised to the middle of its range. The second benchmark is the square colour image of a church shown in Figure 8.1. In our experiments our image sizes range from 50×50 for our preliminary experiments to 150×150 for later experiments.

The *mutate*(I) operator perturbs all three colour channels of one or more pixels in the image I. For our experiments *mutate*(I) mutates a single pixel of I a random amount uniformly distributed in the range $[-20, +20]$ intensity levels. Note the intensity levels of all channels are integers in the range $[0, 255]$. Individually, these mutation operations have a very small impact, which facilitates a gradual and smooth evolutionary process at the cost of requiring many iterations to substantially change an image.

During the evolutionary process all images are constrained using the *valid* function. The *valid*(I) function checks to see if the variant image I is within a certain feature *distance* of the starting image S. Images that fail the constraint are excluded from the population. In deriving a definition for *valid* we experimented with a number of pixel and smoothness constraints. The constraint that gave the most visually pleasing results across the range of features used was $valid_{RMSE_{10}}$ which, given an image I of N pixels with 3 colour channels is defined:

$$valid_{RMSE_{10}}(I) = \sqrt{\sum_{i=1}^{N} \sum_{c=1}^{3} (S_{ic} - I_{ic})^2 / 3N} < 10.$$

The $valid_{RMSE_{10}}$ is a global constraint limiting each color channel to an average deviation of 10 from the original image.

8.2.1 The Feature Diversity Measure

The key component of $(\mu + \lambda) - EA_D$ is the feature diversity contribution measure: $d(I, P)$ which returns the contribution of image I with respect to the population P. For the case of a single feature function f which maps an image I to a scalar feature value $f(I)$ the diversity measure $d(I, P)$ is derived by the following steps. First, we sort the individuals I_i in P according to the feature function f to produce a range of values: $f(I_1) \leq f(I_2) \leq \ldots \leq f(I_k)$. Now, where $f(I_i) \neq f(I_1) \neq f(I_k)$ we calculate $d(I_i, P) = (f(I_i) - f(Ii - 1)) \times (f(I_i) - f(I_{i+1}))$. This means $d(I_i, P)$ will be large when I_i is far from its neighbours in the feature dimension represented by f. For the extremes: $f(I_i) = f(I_1)$ or $f(I_i) = f(I_k)$ we set $d(I_i, P) = R^2$ where $R = f(I_1) - f(I_k)$. Note that, for convenience, our definition of R differs from [74] in being relative to the range of feature values in that dimension rather than an absolute upper limit on the range of f.

To extend $d(I, P)$ to multiple features $\{f_1, \ldots, f_k\}$ we use population weighted diversity measure d' from [74]:

$$d'(I, P) = \sum_{i=1}^{k} (w_i \times d_{f_i}(I, P)), \tag{8.1}$$

where w_i is a weighting assigned to each feature dimension bias search toward or away from selected features. In our experiments we use $k \in \{1, 2\}$. In our experiments we set $w_i = 1$ for our features except where noted otherwise. The art produced depends strongly on our choice of features we use. We describe these next.

8.2.2 Features

A feature f maps an image I to a scalar value$f(I)$ measuring that feature's value for I. The features we use here fall into two categories. The first category is aesthetic features [99]. High aesthetic feature values correspond to images that are pleasing or natural with the exception of the Ross-Ralph-Zong Bell-Curve feature, where low values correspond to pleasing images. These features, taken from the literature, have been empirically derived from surveys or from databases of popular images. The second category is general features. These features are neutral measures of the properties of an image. In Chapter 3 we described each feature category in turn.

The final setting for $(\mu+\lambda)-EA_D$ is the termination condition. Here, we terminate at either 5 million iterations, or when the extreme feature values in our population change by less than 2% in $20\,000$ iterations, or when 72 hours of time has elapsed, whichever comes first. This completes our outline of our setup for the $(\mu+\lambda)-EA_D$ algorithm. Next, we describe our experimental results.

8.3 Experimental Investigations

We implemented the $(\mu+\lambda)-EA_D$ algorithm and all of its code in MATLAB (R2016a) and ran all of our experiments on single nodes of a Lenovo NeXtScale M5 Cluster with Two Intel® Xeon® E5-2600 v4 series 16 core processors, each with 64 GB of RAM. We ran our experiments in three phases. First, we experimented with single features and constraints on low resolution 50 x 50 images to quickly identify useful feature metrics. We ran a second set of experiments with these selected features at a higher resolution $(150$ x $150)$. We then ran a third set of experiments on these images using two feature dimensions.

8.3.1 Experiments

The preliminary experiments ran all of our features using different validity constraints on both the grey and church images at low resolution to eliminate the configurations producing the least interesting results. From these experiments we were able to establish that global constraints such as $RMSE_{10}$, in most cases, produced more varied images than equivalent constraints that limited the range of luminosity or colour channel in each pixel.

We also identified two features: *Ben*, and *Bell* – that failed to produce images that departed strongly from the original. The results of these runs on the church image is shown in Figure 8.2. It can be seen that amount of variation for both of these features is quite low - with the main differences between images being the amount of noise present. It should be noted that both of these features are based on the *shape* of intensity distributions in the image. As such, large variations in these features may sometimes be obtained with only small absolute changes in pixel intensities. As a preliminary observation, this level of sensitivity to change makes these features less useful for generating art in this context than those that are sensitive to absolute rather than relative pixel values. We excluded these features from further study in this work.

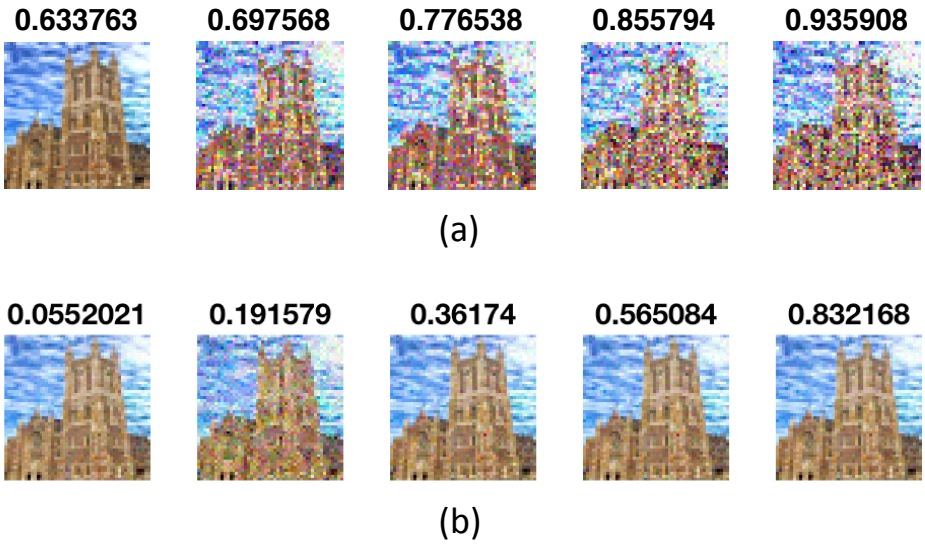


FIGURE 8.2: Individuals 1, 5, 10, 15, and 20 from each of the populations for runs for the *Ben* (a). And, the *Bell* feature (b). Feature values are printed above the images. For *Bell* a low feature value is considered to be more aesthetically pleasing.

In our preliminary experiments we ran the $(\mu + \lambda) - EA_D$ algorithm on both the grey and church images. We quickly observed that the results were much more interesting on the church image. The prior colour variations in the church image interacted with the features in interesting ways. In contrast the lack of colour variations in the grey images tended to produce noisy images – even for features with a geometric component like *GCF*. Figure 8.3 shows the most visually interesting outcomes for the single feature runs on the grey images. As can be seen the lowest scoring individual in each case is the grey starting image. For the *Info* feature (part (a)) this is because a plain image has minimal information content while a noisy image will maximise this feature. For the *Hue* feature (part (b)) a plain grey image with zero saturation has a hue value of zero. For the Global-Contrast-Factor feature (part (c)) a grey image has zero-contast at any scale. The image that scores most highly in this feature exhibits a semi-random high-contrast rippled structure with strong variations at the resolution most favoured by GCF.

In all cases, when we started with a grey image, we saw relatively limited structure in the resulting images. A similar result has been observed in art generated from maximising neuron activations in deep learning neural networks without the use of prior images [200]. To focus on the more interesting outcomes we limited our later, higher-resolution, experiments to the church image from Figure 8.1.

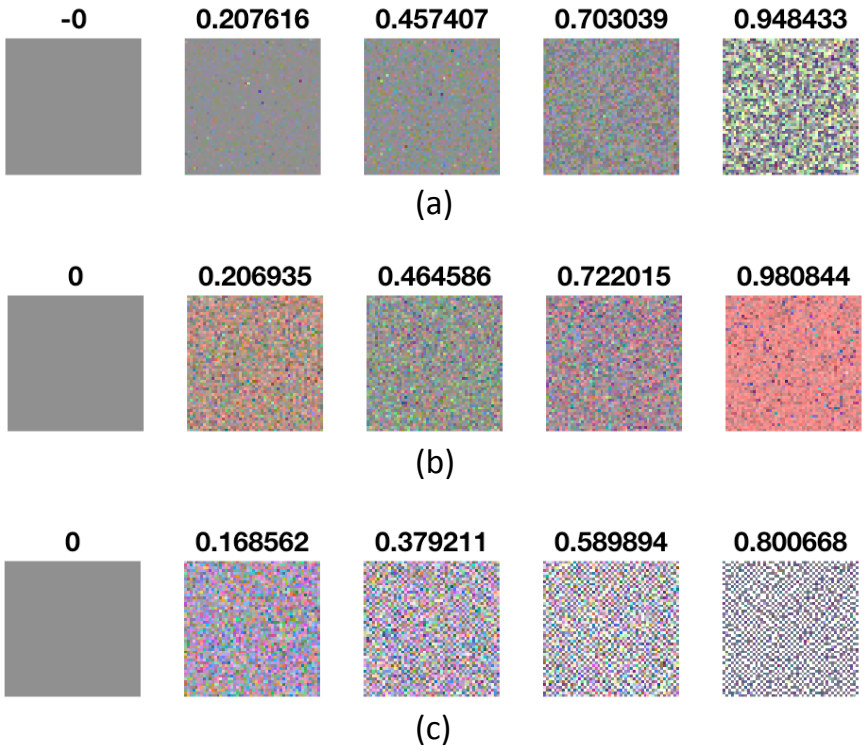


FIGURE 8.3: Individuals 1, 5, 10, 15, and 20 from the populations for the *Info* (a), *Hue* (b) and *GCF* (c) features run against grey images. The $RMSE_{10}$ constraint was used in all cases. The feature values for *GCF* were scaled by $1/20\,000$.

8.3.2 Single Dimensional Feature Experiments

We ran single dimensional experiments at 150×150 resolution for the, *Hue*, *SDHue*, *Saturation*, *GCF*, *Info*, *Smoothness*, and *Symmetry* features. These experiments were all run with the $RMSE_{10}$ constraint. The visual results of these experiments are shown in Figure 8.4 shows images sampled from the population of these single-feature runs. The first three rows in Figure 8.4 correspond to colour features. Row (a) is produced by the *Hue* feature. Individuals in this population will be spread across the colour spectrum, which is red at both ends. Row (b) is produced by the *SDHue* feature. Images that score low in this feature will be monochromatic and in the middle of the spectrum. High-scoring images will appear red because it samples from both extremes. Row (c) is produced by the *Saturation* feature. Images that score low in this feature are monochromatic and individuals that score high are nearly fully saturated. All of the colour features produce populations of images that follow an interesting progression of colour combinations.

The last four rows of Figure 8.4 correspond to features that are affected by relative

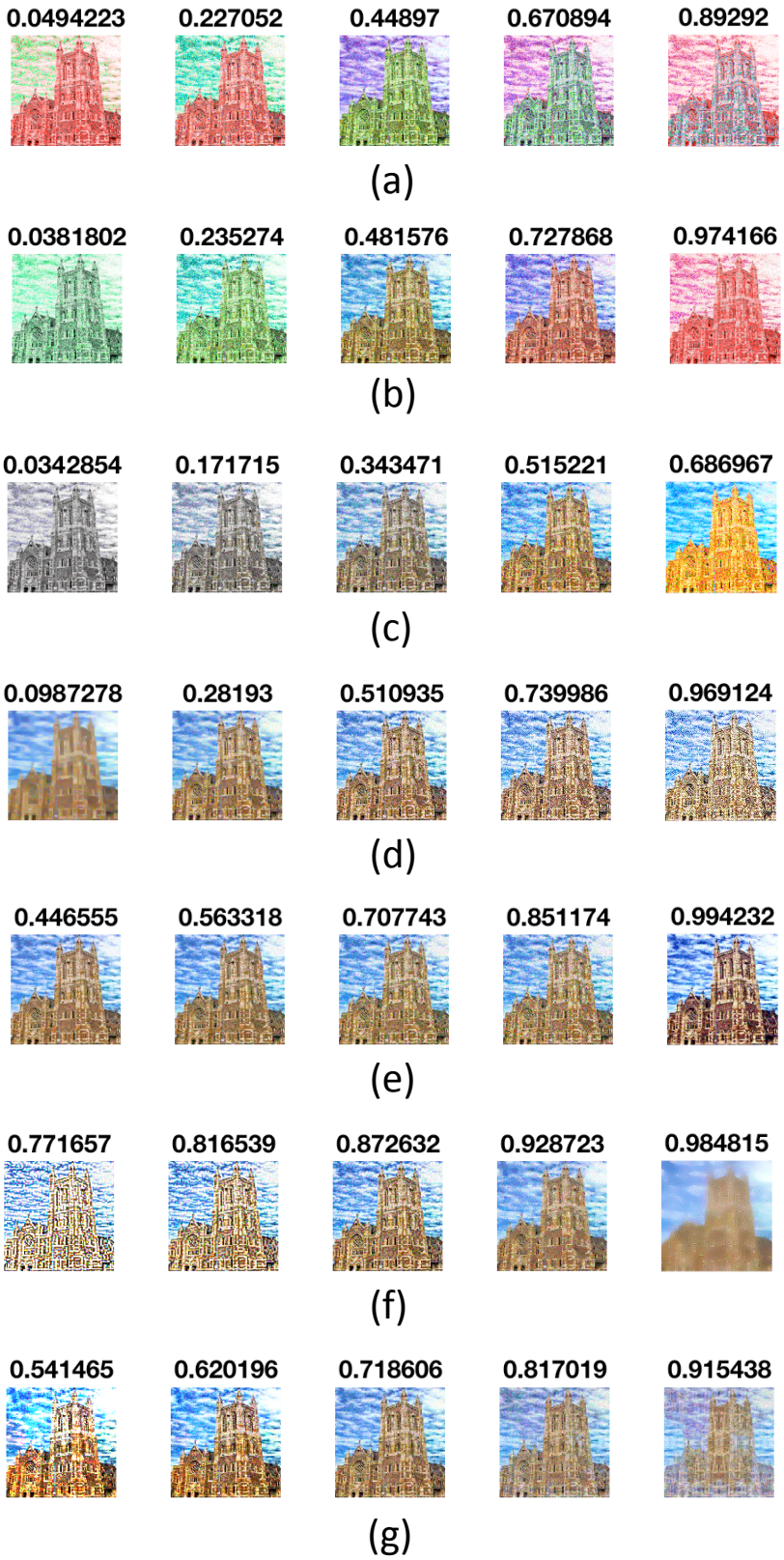


FIGURE 8.4: Individuals 1, 5, 10, 15, and 20 from the populations for the *Hue* (a), *SDHue* (b), *Saturation* (c), *GCF* (d), *Info* (e), *Smoothness* (f), and *Symmetry* (g) features. Each experiment was run with the $RMSE_{10}$ constraint. Here we scale *GCF* by $1/100\,000$ to account for the larger image size.

pixel luminosities. Row (d) is *GCF* which is scores high for images with high contrasts at medium and low resolutions. The pixelated appearance of the highest scoring individual and the low contrast evident in the lowest scoring individual are indicative of *GCF*'s response. Row (e) is the *Info* feature which is an approximation of the entropy of the image. The images that score high in this feature have sharply contrasting areas and the low scoring images have relatively uniform contrast. Row (f) is produced by the *Smoothness* feature. The low scoring individuals have sharp edges and the high scoring individuals have a de-focused appearance. Finally, the *Symmetry* feature produces higher levels of asymmetry in the low scoring individuals. In the highest scoring images, the evolutionary process enhances existing image features to produce highly symmetrical patterns cantered around the details in the church tower.

The feature values that correspond to the individual images that develop during the $(\mu + \lambda) - EA_D$ run can be traced over time. Figure 8.5 shows the trace of feature values for the populations sampled in Figure 8.4. As can be seen for every feature the $(\mu + \lambda) - EA_D$ algorithm steadily pushes the feature values apart. For the *Hue*, *SDHue*, *Saturation* and *Info* features the algorithm was able to spread the population close to its maximum theoretical extent. For the other features the population was still diverging at the end of the run. The *GCF* run reached its 72 hour job time limit before the $5,000,000$ iteration limit was reached.

In order to check if evolution was slowed by a high constraint violation rate, we traced the proportion of invalid individuals generated as the evolutionary run progressed. The trace of constraint violation rates for each feature is shown in Figure 8.6. As can be seen the violation rates tend to converge to approximately 10% of evaluations. Note that for our experiments 10% of individuals are at the population extremes and the violations are more likely to occur for these individuals. This means that the violation rates seen are likely to be having an impact on the speed of evolution for these extreme individuals and thus the population as a whole. There are differences between features – the *Info* feature in particular seemed to be able progress without the need to produce large cumulative colour deviations from the original image. In contrast, violations for *GCF* seem to rise quite quickly with time – this is likely to because images with high values for *GCF* have high local contrast.

8.3.3 Two-Dimensional Feature Experiments

Our next set of experiments attempts to evolve images that are diverse in two dimensions according to Equation 8.1. One aim of these experiments is to make preliminary observations of the effect of running the $(\mu+\lambda)-EA_D$ algorithm on multiple features.

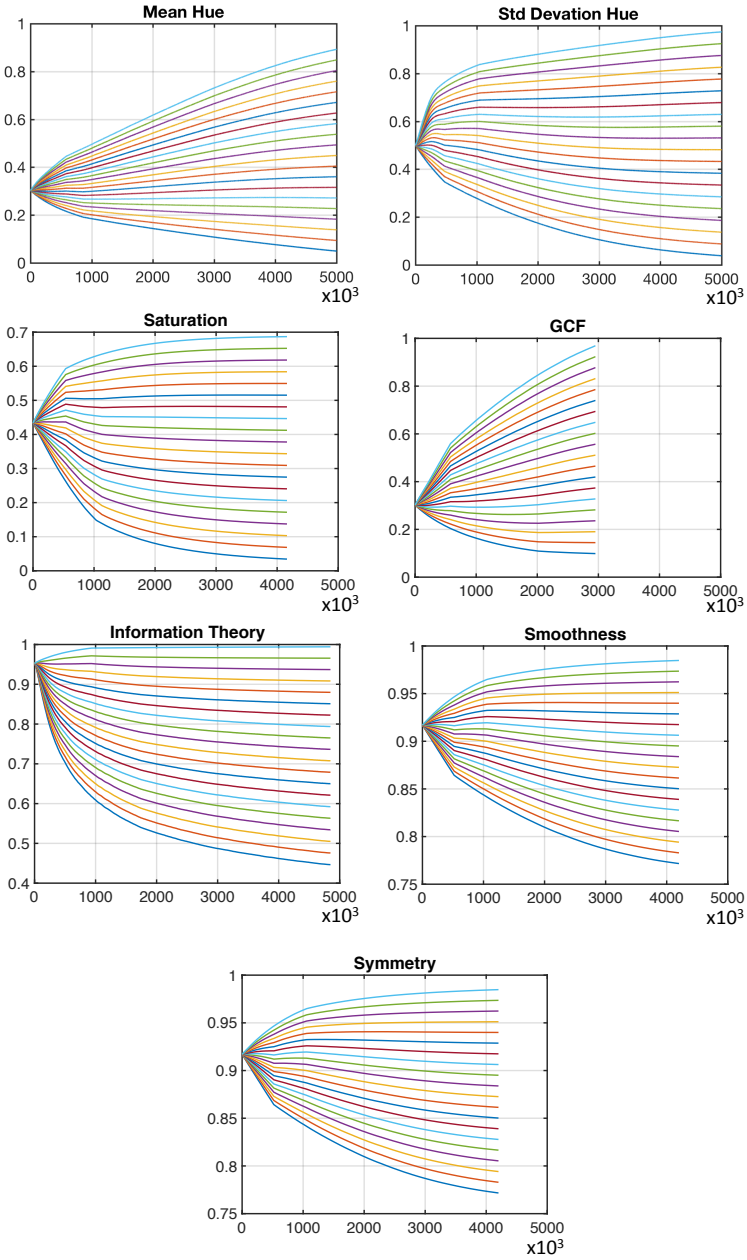


FIGURE 8.5: Traces of feature values versus number of iterations for the runs terminating in the images sampled in Figure 8.4.

A second aim is to see how strongly different sets of features correlate at the end of $(\mu + \lambda) - EA_D$ runs. A-priori, a strong correlation between two features is indicative of features that interact strongly through the target image whereas a weak correlation indicates the features can vary orthogonally in the same image.

In this work we ran experiments for the following pairs of features: (*Hue*, *GCF*), (*Hue*, *Saturation*), (*Hue*, *Smoothness*), (*Hue*, *SDHue*), (*GCF*, *Saturation*), (*GCF*,

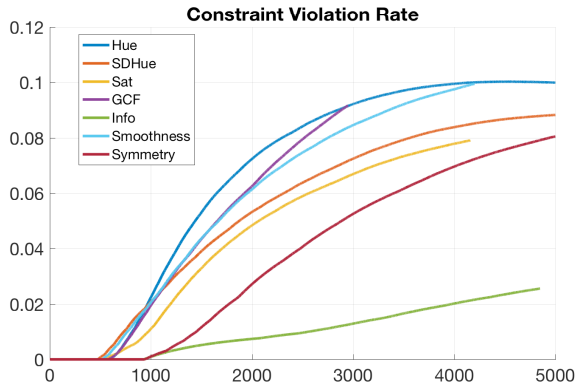


FIGURE 8.6: Constraint violation rates for the features.

Smoothness), (*Smoothness*, *Saturation*), (*SDHue*, *GCF*), (*SDHue*, *Saturation*), (*SDHue*, *Smoothness*), (*Symmetry*, *Hue*), and (*Symmetry*, *SDHue*).

Prior to experimentation, we would expect more constrained, interactions to occur between pairs of structural features such as (*GCF*, *Smoothness*) and between pairs of colour features such as (*Hue*, *SDHue*). We would expect less constrained interactions between pairs of colour and structural features such as (*Symmetry*, *Hue*). We would expect to see more diverse artistic results from evolutionary runs that use these latter, less constrained pairs.

As an initial test of these ideas, we show the results of runs for the feature-pairs (*GCF*, *Smoothness*) and (*Symmetry*, *Hue*) in Figures 8.7 and 8.8. In the figures the rows are ordered by first feature values: *GCF* and *Symmetry*. Within each row the columns are ordered by the second feature values *Smoothness* and *Hue*, respectively. In Figure 8.7 the first row contains some smooth images, in contrast the last row contains no smooth images. This indicates that it is difficult, at least under our current setup, to evolve images that are both smooth and score highly in *GCF*. This apparent conflict is unsurprising because *GCF* scores highly if there is strong contrast between neighbouring pixels, however *Smoothness* scores highly if there is low contrast between adjacent pixels.

Figure 8.8 presents a different scenario, all rows exhibit at least some variation in the dimension of *Hue* which indicates that it is possible for a picture to achieve some diversity across both the *Hue* and *Symmetry* dimensions.

If we plot the individuals in the populations for these experiments across both feature dimensions, as we do in Figures 8.9 and 8.10 we can visualise how strongly these features are bound. In each of these figures we show the feature values at the end of each evolutionary run. The diameters of each point in these figures is determined by

FIGURE 8.7: Population of images resulting from the evolution for diversity of images for both *GCF* and *Smoothness*. The rows represent increasing values for *GCF*. The values for *GCF* and *Smoothness*, respectively, are shown above each image. Note how the values for *GCF* and *Smoothness* are contra-variant.

the size of the contribution of that individual to the diversity of the population. It can be seen the members of the population in Figure 8.9 are almost co-linear and negatively related. Note that in this experiment we scaled the *Smoothness* feature so that its range was similar to that of *GCF* so the search is not biased by the large values that *GCF* can assume. This result indicates that it is difficult to evolve images that score high or low on both feature measures.

In contrast, the population in Figure 8.10 exhibits a good spread of values in both dimensions indicating that it is possible for images to move in both feature dimensions with relative freedom. As an additional note, the population in Figure 8.10 appears to cling to the perimeter of a diamond. This is at least in part due to the multi-dimensional contribution metric in Equation 8.1. This metric is based on a weighted sum, which is an L_1 distance measure which encourages individuals to spread out maximally in each dimension independently.

To see how different pairs of dimensions relate we ran correlations on different pairs of features. The results are shown in Table 8.1. As can be seen, most metrics are quite weakly related, which indicates reasonable freedom to evolve individuals in both dimensions. (*GCF*, *Saturation*) exhibits a broad correlation. This is partly due

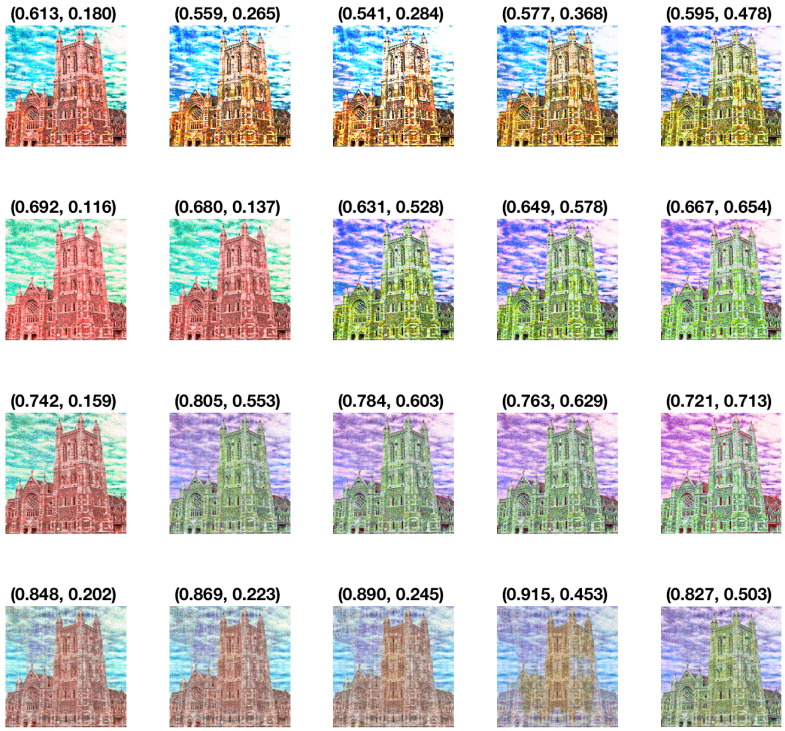


FIGURE 8.8: Population of images resulting from the evolution for diversity of images for both *Symmetry* and *Hue*. The rows represent increasing values for *Symmetry*. In each row there are increasing values of the *Hue* feature. The values for *Symmetry* and *Hue*, respectively, are shown above each image. Note how the values of these features vary more freely.

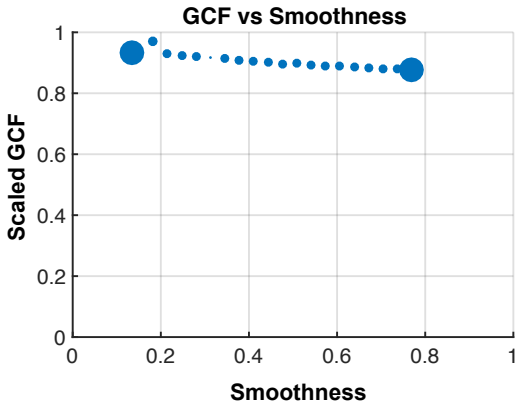


FIGURE 8.9: Plot of feature and contribution values at the end of the $GCF - Smoothness$ run. The *GCF* values are scaled to fit the range $[0, 1]$. It can be seen that the feature values are very highly correlated with coeff = -0.92.

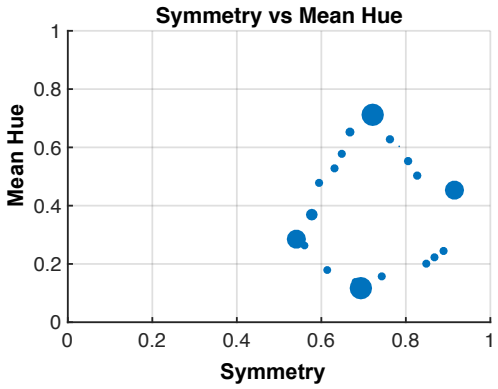


FIGURE 8.10: Plot of feature and contribution values at the end of the *Symmetry – Hue* run. It can be seen that the feature values have a very low correlation with coeff = 0.04.

Feature pair	Correlation Coeff
Hue – GCF	0.3125
Hue – Saturation	0.3911
Hue – Smoothness	-0.0861
Hue – SDHue	0.3599
GCF – Saturation	0.4835
GCF – Smoothness	-0.9154
Smoothness – Saturation	0.1454
SDHue – GCF	0.3769
SDHue – Saturation	0.2656
SDHue – Smoothness	0.3155
Symmetry – Hue	0.0366
Symmetry – SDHue	0.0426

TABLE 8.1: Correlations between feature values in populations for evolutionary runs for two features.

to the fact that, due to limits on contrast in saturated images, it is difficult to evolve an image is both highly saturated and scores high for *GCF*. (*Hue*, *SDHue*) are also moderately related. This is partly because images with a high *SDHue* are restricted in their choice of colours. Such interactions are potentially complex and warrant further investigation.

8.4 Conclusions

In this chapter, we have demonstrated that is possible to generate interesting sets of picture variants by evolving populations of images to maximise diversity in one or more feature dimensions. This evolutionary process succeeds in spreading individuals in the population along the spectrum of each image feature which allows the viewer to better visualise the aspects of the image that impact directly on the feature(s). In the experiments done so far, the features combinations of most interest have been those that combined colour metrics such as *Hue* with structural metrics such as *Smoothness* and *Symmetry*. In a single dimension, the colour features most obviously illustrate the impact of feature diversity. However, interesting image sequences are obtained from structural measures such as *GCF*, *Info* and *Smoothness*. In each of these latter cases, the spread of images in the population highlight what image features are enhanced and suppressed as we move across the feature dimension.

We have also shown that by aiming to maximise diversity in multiple feature dimensions we can start to see how the features interrelate for high, medium and low values of each feature. By plotting feature values after evolution, we can see the extent to which individuals were free to move in the space of feature dimensions. The methodology outlined in this work forms a basis for exploring which features are most compatible with each other in the context of evolutionary generative art. At a more specific level this methodology can also indicate the parts of the feature space where interactions between features are flexible or tightly constrained.

The work presented here is a proof-of-concept and the results we have seen so far are indicative rather than definitive. To gain more certainty of the relationships between features it will be necessary to conduct more evolutionary runs on more images. There is scope to generalise these results by exploring the impact of changing evolutionary settings such as the mutation rate and the relative weightings between features. Such experiments will tell us if the relationships observed between features are robust.

We can also extend this work to a wider variety of features and image validity constraints. Finally, we can also redefine our multi-dimensional diversity measure, so it is framed in terms of L_2 distance. This should lead to more regular coverage of the feature space and provide an interesting basis for comparison to other means of providing diversity.

A conference version containing the results of this chapter has been published in the Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2017 (see B. Alexander, J. Kortman and A. Neumann (2017) [3]).

Chapter 9

Discrepancy-Based Evolutionary Diversity Optimization

9.1 Introduction

In this chapter, we introduce a diversity optimization approach that utilizes the discrepancy measure. This approach allows one to evolve diverse sets without having any assumption on the preferred weightening of the different diversity criteria.

Diversity plays a crucial role in evolutionary computation. Traditionally, diversity is used to avoid premature convergence and it is generally assumed that crossover-based evolutionary algorithms need a diverse population in order to produce good results. During the last 10 years, using evolutionary algorithms to produce a diverse set of solutions has gained increasing attention. In Chapter 2 we introduced in more detail the notion of discrepancy. Ulrich and Thiele [253] introduced evolutionary computation approaches that are able to produce diverse sets of solutions by evolving a population according to quality criteria and diversity measures.

Recently, this approach has been adapted to evolve diverse sets of Traveling Salesperson Problem (TSP) instances [74] as well as diverse sets of images [3]. In the case of images, the population of an evolutionary algorithm has been used to evolve images that are close to a given one (in terms of an error measure) and that are diverse with respect to different artistic features. Furthermore, an evolutionary image composition approach based on a feature-based covariance error function has been introduced in [196]. Diversity optimization approach builds on a simple diversity measure that measures diversity according to a given feature. In order to extend this approach to more than one feature, a diversity measure weightening the different features has been used.

We investigate the use of the star discrepancy measure in evolutionary diversity

optimization for one setting previously studied in the literature, namely diversity optimization for images [3].In terms of images, we also introduce a new and more effective mutation operator based on random walks than the one introduced in [3]. This self-adaptive random walk operator allows to reduce the number of iterations needed to construct good diverse set of solutions from $1-4$ million [3] to 2000 and therefore reduces the number of required generations by three orders of magnitude.

Our experiments are carried out for diversity optimization tasks using two and three features. We show that the previously used approach for images [3] computing a weighted diversity contribution in terms of the considered features constructs solution sets with a very high discrepancy compared to our approach using the discrepancy measure. Furthermore, we show that the weighted diversity contribution approach can be used in an effective way for doing tie-breaking between sets of solutions having the same discrepancy value.

This chapter is structured as follows. In Section 9.2, we introduce our discrepancy-based diversity optimization approach. In Section 9.3, we introduce the new mutation operator for diversity optimization of images. Here, we describe our approach for evolving images of low discrepancy in respect to the given features. In Section 9.4, we evaluate the discrepancy optimization approach for images. Finally, in Section 9.5 we finish with some concluding remarks.

9.2 Discrepancy-based Diversity Optimization

We consider evolutionary diversity optimization. Given a search space S, our aim is to construct a diverse set of solutions $P = \{X_1, \ldots, X_\mu\}$ where each solution $X_i \in S$ fulfills a given quality criteria, i.e., we have $q(X_i) \geq \alpha$ for a given quality threshold α.

Properties of our potential solutions X_i are characterized by features $f_1, \ldots, f_d$ which are problem specific. Let $I \in S$ be an individual in a population P. We associate with I its feature vector $f(I) = (f_1(I), \ldots, f_d(I))$.

Traditionally, the goal of constructing a set of points with a low discrepancy is defined in $[0,1]^d$. Therefore, the feature values are scaled before the calculation of discrepancy. Let $f_i^{\max}$ and $f_i^{\min}$ be the maximum and minimum value of feature f_i. We evaluate our set of points in terms of discrepancy using the scaled feature values

$$f_i'(I) = (f_i(I) - f_i^{\min})/(f_i^{\max} - f_i^{\min}).$$

We have $f'(I) \in [0,1]^d$ for all scaled feature vectors $f'(I)$ if $f_i^{\min} \leq f_i(I) \leq f_i^{\max}$, $1 \leq i \leq d$. $f_{\max}$ and $f_{\min}$ are set based on initial experiments. Feature values outside

Algorithm 13 $(\mu + \lambda)$-EA_D

Initialize the population P with μ instances of quality at least α.
Let $C \subseteq P$ where $|C| = \lambda$.
For each $I \in C$, produce an offspring I' of I by mutation. If $q(I') \geqslant \alpha$, add I' to P.
While $|P| > \mu$, remove an individual $I = \arg\min_{J \in P} D^*(P \setminus J)$.

Repeat step 2 to 4 until termination criterion is reached.

that range will be scaled to 0 and 1, respectively, to allow the algorithm to work with non anticipated features values.

Let $f'(P) = \cup_{I \in P} f'(I)$ be the set of scaled features vectors in P. We denote by $D^*(P)$ the discrepancy of $f'(P)$ in $[0,1]^d$. Throughout this section, we use the star-discrepancy. Given $P = \{I_1, \ldots, I_k)$ with feature vectors $f'(I_1), \ldots, f'(I_k)$, we define

$$D^*(P) := \sup_{J \in Y} D(J, P),$$

where

$$D(J, P) = \frac{|I \in P \mid f'(I) \in J|}{k} - \mathrm{Vol}(J),$$

$\mathrm{Vol}(J)$ denotes the volume of the box J, and Y is the class of all axis-parallel boxes of the form $J = \prod_{i=1}^{d} [0, u_i)$ with $0 \leq u_i \leq 1$ for $1 \leq i \leq d$.

A key difficulty to overcome in the optimization for low star discrepancy values is the computational hardness of its evaluation [80]. The best known algorithm for the star discrepancy computation has a running time of order $n^{1+d/2}$ [38], which is exponential in the dimension d. As we are interested in dimension $d = 2, 3$, we can use this algorithm, and make use of the implementation that is available on [264]. The reader interested in a discussion of computational aspects of geometric discrepancies is referred to [44].

We use the $(\mu + \lambda)$-EA_D given in Algorithm 15 to compute a diverse population where each individual meets a given quality criteria q according to a given threshold α, i.e., we have $q(I) \geq \alpha$ for all individuals in the population P. The population P is a multi-set, i.e., it may contain an instance more than once. The algorithm is initialized with a population where each individual meets the given criteria. In each iteration λ offspring are produced. Offspring that do not meet the quality criteria are directly discarded. Offspring that meet the criteria are added to the population and survival selection is performed afterwards to obtain a population of size μ. To do this, individuals are removed iteratively. Having a population of size $k > \mu$, in each

FIGURE 9.1: Image S.

iteration an individual I is removed that leads to a population $P \setminus I$ of size $k-1$ having the smallest discrepancy among all populations that can be constructed by removing exactly one individual from P.

The discrepancy minimization algorithm is compared to the evolutionary diversity optimization approach in [74], which aims at maximizing the feature-based population diversity using a weighted contribution measure for each individual. The weighted diversity contribution of an individual I with feature vector $f(I)$ is defined as $c(I,P) = \sum_{i=1}^{k}(w_i \cdot d_{f_i}(I,P))$, where $d_{f_i}(I,P)$ represents the normalised contribution of individual I to the population diversity over feature f_i and w_i represents the weight for feature f_i.

The resulting algorithm $(\mu+\lambda)$-EA$_C$ differs from $(\mu+\lambda)$-EA$_D$ only in step 4, and removes in each of these steps an individual I with the smallest weighting contribution $c(I,P)$ to the population diversity. Furthermore, we consider the algorithm $(\mu+\lambda)$-EA$_T$ that uses both the discrepancy measure and the weighted contribution measure. It is the same as $(\mu+\lambda)$-EA$_D$ but uses the weighted contribution measure as tie-breaking in step 4 of the algorithm; i.e. if there is more than one individual whose removal leads to the minimum discrepancy value, then the one among these with the smallest contribution to the weighted diversity contribution is removed.

In the following, we evaluate our discrepancy-based diversity optimization approaches for evolving diverse sets of images. We also introduce a new mutation operator for images based on random walks which significantly speeds up the diversity optimization process when constructing a diverse set of images.

9.3 Self-Adjusting Offset Random Walk Mutation

We consider the task of evolving a diverse set of images as previously investigated in [3]. Given an image S, the task is to compute a diverse set of images $P = \{I_1, \ldots, I_\mu\}$

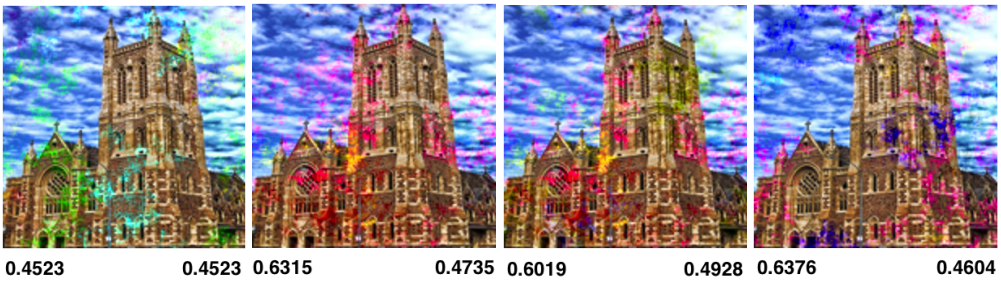


FIGURE 9.2: Selected images from the population after discrepancy minimization for the Hue and Saturation features.

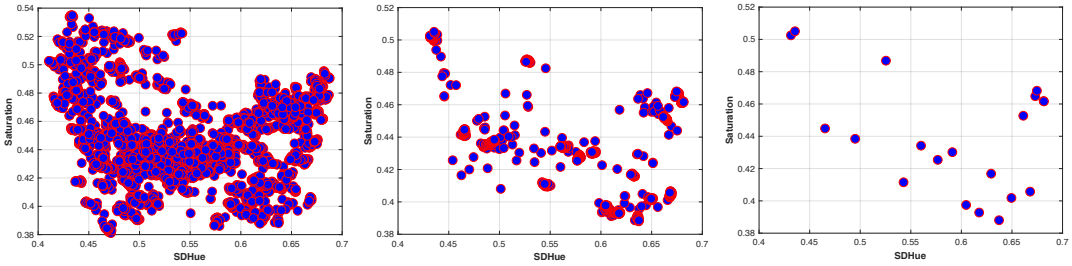


FIGURE 9.3: All feature vectors generated in 10 runs of $(\mu+\lambda)$-EA_T with 1000 iterations each (left), one run with 1000 iterations (middle), the final population after 1000 iteration with discrepancy 0.22637 (right).

that meets a given quality criterion $q(I)$ for each $I \in P$. For our experimental investigations, an image I meets the quality criterion if the *mean-squared error* in terms of the RGB-value of the pixels of the image I with respect to the input image S (shown in Figure 10.3) is less than 500.

In our investigations, we focus on a selected set of features used in [3, 99]. In Chapter 3 we present additional details on the chosen feature's characterisation. We are working with the scaled feature values when computing the discrepancy of a given set of points. It should be pointed out that not all feature vector combinations within the given feature intervals are usually possible. To illustrate this, we consider the features SD-hue and Saturation and run the EA (using the mutation operator described in Section 9.3) for 1000 iterations. Figure 9.3 shows all feature vectors produced during 10 runs of the $(20+1)$-EA_T (left), all feature vectors produced during one run (middle), and the feature vectors of the final population (right). It can be observed that the area where both feature values are high does not contain any points (similarly if both feature values are very low). The seems to indicate that the problem is constrained to a subspace of the unit square. If this is true, then this has a direct consequence on the best possible discrepancy value that can be obtained, as discrepancy is a measure in $[0,1]^d$.

Algorithm 14 OFFSRANDOMWALKMUTATION $(X, t_{\max})$

Let X is a image with pixels $X_{ij} \in X$.
$Y \leftarrow X$.
Choose starting pixel $Y_{ij} \in Y$ uniformly at random.
Choose offset $o \in [-r, r]^3$ uniformly at random.
$t \leftarrow 1$.
while $t \leq t_{\max}$ **do**
$Y_{ij} = Y_{ij} + o$.
Choose $Y_{kl} \in N(Y_{ij})$ uniformly at random.
$i \leftarrow k$, $j \leftarrow l$.
$t = t + 1$. Return Y.

The algorithm uses a variant of the random walk mutation introduced in [196] for evolutionary image composition. This speeds up the process of diversity optimization by three orders of magnitude compared to [3] where for a mutation operator changing in each step a single pixel $1 - 4$ million iterations where required to construct a diverse set of images. Our new mutation operator enables us to construct diverse sets of images for all three algorithms (including the $(\mu + \lambda)$-EA$_C$ investigated in [3]) within just 2000 generations.

The random walk in this work differs from the one for image composition given in [196] by changing the RGB values by an offset vector $o \in [-r, r]^3$ chosen in each mutation step uniformly at random. The mutation operator is shown in the Algorithm 14.

The random walk causes movement from the current pixel X_{ij} to the next pixel by moving either right, left, down or up. We define the neighborhood $N(X_{ij})$ of pixel X_{ij} as

$$N(X_{ij}) = \left\{X_{(i-1)j}, X_{(i+1)j}, X_{i(j-1)}, X_{i(j+1)}\right\}.$$

The random walk chooses an element of $N(X_{ij})$ uniformly at random in every step. Furthermore, the random walk is wrapped around the boundaries of the image. We produce an offspring Y from X by setting each visited pixel X_{ij} to the value of $X_{ij} + o$. Given a current image X, our $(\mu+\lambda) - EA_D$ algorithm uses the random walk mutation to alter all visited pixels. Note that pixels may be visited more than once, and the offset may be applied several times in this case. The random walk paints all the visited pixels by adding the chosen offset vector o. Each random walk mutation is run for $t_{\max}$ steps, where $t_{\max}$ is chosen in an adaptive way.

Self-Adjustment We decrease the length of random walks through decreasing $t_{\max}$ when the discrepancy value does not decrease as a result of an unsuccessful mutation. We increase $t_{\max}$ if the discrepancy decreases as a result of a successful mutation. This builds on the assumption that mutations doing less change to the image are needed to obtain an improvement if it is hard to make progress with the current choice of $t_{\max}$. On the other hand, a better progress may be achievable if the current setting of $t_{\max}$ is already able to decrease the discrepancy. Our adaptive approach makes use of the parameter adjusting scheme recently used in [39]. This method, originally proposed in [128], applies the classical $1/5$-success rule from evolution strategies to a discrete setting.

Our approach increases $t_{\max}$ for a successful outcome or decreases $t_{\max}$ in the case that the new offspring is not accepted. In our algorithm, $t_{\max}$ can take on values in $t_{\mathrm{LB}} \leq t_{\max} \leq t_{\mathrm{UB}}$, where t_{LB} is a lower bound on $t_{\max}$ and t_{UB} is an upper bound on $t_{\max}$. For a successful mutation, we set $t_{\max} := \min\{F \cdot t_{\max}, t_{\mathrm{UB}}\}$ and for an unsuccessful mutation, we set $t_{\max} := \max\left\{F^{-1/k} \cdot t_{\max}, t_{\mathrm{LB}}\right\}$, where $F > 1$ is a real value and $k \geq 1$ an integer which determines the adaptation speed. For our experimental investigations, we set $t_{\mathrm{LB}} = 1000$, $t_{\mathrm{UB}} = 20000$, $F = 2$, $k = 8$, and $t_{\max} = 1000$ at initialization based on preliminary experimental investigations.

9.4 Experimental Investigations

All algorithms were implemented in $Matlab$ $(R2017b)$. We ran all of our experiments on single nodes of a Lenovo NeXtScale M5 Cluster with two Intel Xeon E5−2600 v4 series 16 core processors, each with 64GB of RAM.

Firstly, we consider the discrepancy-based diversity optimization for two features. We select features in order to combine different aesthetic and general features based on our initial experimental investigations and previous investigations in [188]. Furthermore, we set $f^{\min}$ and $f^{\max}$ as follows. The $f^{\min}$ values used for $SD\text{-}hue$, Hue, $Saturation$, $Smoothness$, GCF, $Symmetry$ are 0.42, 0.25, 0.42, 0.42, 0.906, 0.0245, and 0.715, respectively. The corresponding $f^{\max}$ values are 0.7, 0.4, 0.5, 0.5, 0.918, 0.0275, and 0.74, respectively.

After considering the combination of two features, we investigate sets of three features. Here, we select different features combining aesthetic and general features together used in the previous experiment. In order to obtain a clear comparison between experiments, we apply the same range of feature values as before.

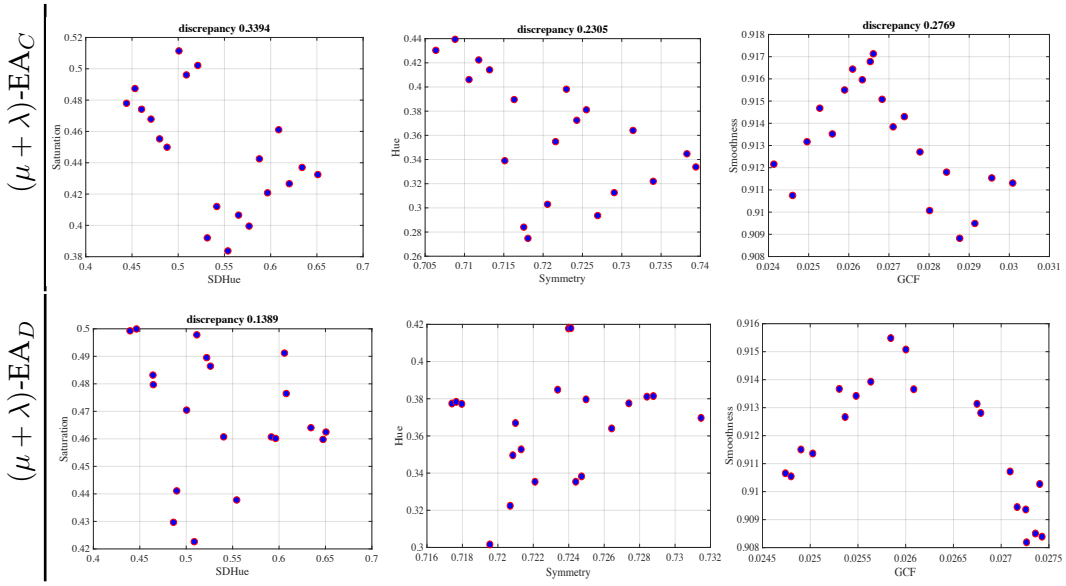


FIGURE 9.4: Feature vectors for final population of $(\mu + \lambda)$-EA$_C$ (top) and $(\mu + \lambda)$-EA$_D$ (bottom) for images based on two features from left to right: (SDHue, Saturation), (Symmetry, Hue), (GCF, Smoothness).

Furthermore, we run the $(\mu+\lambda)$-EA_C diversity algorithm from [188] using the self-adjusting random walk mutation operator in order to compare the two approaches for diversity optimization. We use the same settings for the $(\mu + \lambda)$-EA_C as for our discrepancy-based diversity algorithm, the $(\mu + \lambda)$-EA_D. Finally, we consider the $(\mu + \lambda)$-EA_T, which uses discrepancy-based diversity optimization plus tie-breaking according to weighted feature contributions when more than one individual exists whose removal would result in the same minimal discrepancy value.

We run each algorithm for 2000 generations with a population size of $\mu = 20$ and $\lambda = 1$. In order to evaluate our results using statistical tests, each algorithm is run 30 times with the same setting applied to each considered pair and triple of features.

We perform a series of experiments to evaluate the performance of our discrepancy-based diversity evolutionary algorithm. Our experiments establish that global constraints like *mean-squared error* can be used to produce more diverse images than equivalent constraints which are limited to the range of the color or luminosity channel for each pixel.

The images displayed in Figure 9.2 match the color features. In particular, images are produced using the *SDHue* and *Saturation* feature. The values for *SDHue* and *Saturation* are displayed above each image for features, respectively. The color spectrum is red at each end with individuals of the population spread along it. Images which have a low score for this feature will be monochromatic and will appear in the middle of the spectrum. Images with a high score will be red as it is a sample from

	$(\mu+\lambda)$-EA_C (1)				$(\mu+\lambda)$-EA_D (2)				$(\mu+\lambda)$-EA_T (3)			
	min	mean	std	stat	min	mean	std	stat	min	mean	std	stat
(f1, f2)	0.2014	0.3234	0.0595	$2^{(-)}$,$3^{(-)}$	0.1272	0.2038	0.1157	$1^{(+)}$	0.1119	0.1530	0.0269	$1^{(+)}$
(f3, f4)	0.1964	0.2945	0.0497	$2^{(-)}$,$3^{(-)}$	0.1574	0.2280	0.0592	$1^{(+)}$,$3^{(-)}$	0.1051	0.1417	0.0179	$1^{(+)}$,$2^{(+)}$
(f5, f6)	0.1997	0.2769	0.0344	$2^{(-)}$,$3^{(-)}$	0.1363	0.2025	0.0538	$1^{(+)}$	0.1457	0.1800	0.0234	$1^{(+)}$
(f1, f2, f3)	0.3389	0.4327	0.0613	$2^{(-)}$,$3^{(-)}$	0.1513	0.3335	0.1062	$1^{(+)}$	0.2253	0.2814	0.0422	$1^{(+)}$
(f1, f4, f3)	0.2754	0.3395	0.0483	$2^{(-)}$,$3^{(-)}$	0.2100	0.3118	0.1309	$1^{(+)}$	0.2224	0.2600	0.0123	$1^{(+)}$
(f5, f4, f2)	0.4775	0.6488	0.0841	$2^{(-)}$,$3^{(-)}$	0.2021	0.3007	0.1467	$1^{(+)}$	0.1983	0.2229	0.0125	$1^{(+)}$

TABLE 9.1: Statistics of discrepancy values for images. f1, f2, f3, f4, f5, f6 denote features SD-hue, Saturation, Symmetry, Hue, GCF and Smoothness, respectively.

both of the extremes. Low-scoring images in the *Saturation* feature are monochromatic whilst high-scoring individuals are almost entirely saturated.

Figure 9.4 shows feature plots of the final populations of the $(\mu+\lambda)$-EA_C (top) and the $(\mu+\lambda)$-EA_D (bottom) for 3 pairs of feature combinations. It can be observed that the discrepancy value for the $(\mu+\lambda)$-EA_D and the combination $(SD-hue, \textit{Saturation})$ is 0.1389. This is significantly smaller than the one for the $(\mu+\lambda)$-EA_C at 0.3394. The middle row shows the combination of *Hue* and *Symmetry*. The discrepancy value of *Symmetry* and *Hue* for $(\mu+\lambda)$-EA_D is 0.1544 whereas it is 0.2305 for $(\mu+\lambda)$-EA_C. In Figure 9.4 the right column shows the final populations of the diversity optimization when considering GCF and *Smoothness*. The discrepancy value for *GCF* and *Smoothness* is 0.1366 for $(\mu+\lambda)$-EA_D and 0.2769 for $(\mu+\lambda)$-EA_C. This is an indication of the difficulty of evolving images which are smooth as well as scoring high in *GCF* in our current setup. However, this conflict is expected as the *GCF* highly scores in the case of strong contrast between adjacent pixels. The *Smoothness* scores have a high value for low contrast between neighboring pixels.

We now consider the results of the $(\mu+\lambda)$-EA_C from [188] using the self-adjusting random walk mutation operator in greater detail. Looking at Figure 9.4 (top) which shows the population of instances for (*SDHue*, *Saturation*), (*Symmetry*, *Hue*), and (*GCF*, *Smoothness*), respectively, we observe that the distribution of the points for the features vectors for final population follows a linear pattern. This is due to the chosen weights which favor lines of feature vectors orthogonal to the used weight vector $(1,1)$.

We use the Kruskal-Wallis test with 95% confidence in order to measure the statistical validity of our results. We apply the Bonferroni post-hoc statistical procedure that is used for multiple comparison of a control algorithm to two or more algorithms. For more detailed description on the statistical tests we refer the reader to [27]. In Table 9.1 we provide statistics on the discrepancy values for the final populations of $(\mu+\lambda)$-EA_C, $(\mu+\lambda)$-EA_D and $(\mu+\lambda)$-EA_T, respectively. For each algorithm and feature combination the minimum, mean, and standard deviation of the discrepancy value of the final population of 30 runs is shown. $X^{(+)}$ is equivalent to the statement that algorithm in

the column outperformed algorithm X, and $X^{(-)}$ is equivalent to the statement that X outperformed the algorithm given in the column. In the case if the algorithm X not appears this means that no significant difference was determined between algorithms. $(\mu+\lambda)$-EA_D clearly outperforms the $(\mu+\lambda)$-EA_C for all feature combinations. Furthermore, $(\mu+\lambda)$-EA_T which uses tie-breaking according to weighted feature contribution leads to a further improvement of $(\mu+\lambda)$-EA_D for almost all feature combinations in terms of the mean and minimal discrepancy value achieved within 30 runs.

9.5 Conclusions

Constructing point sets of low discrepancy has a prominent role in mathematics and a set of low discrepancy can be seen as being one that is covering the considered space $[0,1]^d$ in a good way as they aim for a good balance of points in every hyper-box with respect to their volume. We have introduced a discrepancy-based evolutionary diversity optimization approach that constructs sets of solutions meeting a given quality criteria and having a low discrepancy with respect to the considered features. Our experimental results for evolving diverse sets of images show that this approach constructs sets of solutions with a much lower discrepancy that the previously used weighted contribution approach according to the given features. Our discrepancy-based diversity optimization process for images makes use of a new random walk mutation operator which reduces the number of required generations to obtain a good diverse set of images by 3 orders of magnitude The best results across all our experimental investigations are obtained by the $(\mu+\lambda)$-EA$_T$, which uses discrepancy-based diversity optimization in conjunction with a tie-breaking rule based on the weighted contribution diversity measure.

A conference version has been published in the Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2018 (see A. Neumann, W. Gao, C. Doerr, F. Neumann, and M. Wagner (2018) [194]).

Chapter 10

Evolutionary Diversity Optimization Using Multi-Objective Indicators

10.1 Introduction

In this chapter, we focus on evolutionary diversity optimization. We are interested in computing a diverse set of high quality solutions that can be presented to a decision maker. Arguably, the most prominent area of evolutionary computation where a diverse set of solutions is sought is evolutionary multi-objective optimization [31]. Given a set of usually conflicting objective functions, the goal is to compute a set of solutions representing the different trade-offs of the considered functions.

Evolutionary algorithms have been widely applied to multi-objective optimization problems and it is one of the key success areas for applying evolutionary algorithms. Over the years, many evolutionary multi-objective algorithms have been developed. Popular algorithms, among many others, are NSGA-II [32], NSGA-III[155], MOEA/D [273], and IBEA [277]. Making them applicable to the area of evolutionary diversity optimization provides huge potential for high performing evolutionary diversity optimization approaches. With this chapter, we bridge the areas of evolutionary diversity optimization and evolutionary multi-objective optimization. We consider popular indicators from the area of evolutionary multi-objective optimization and show how to make them applicable in the area of evolutionary diversity optimization.

Evolutionary algorithms have been used for a wide range of optimization problems and to discover novel designs for various engineering problems. Diversity plays a crucial role when designing evolutionary algorithms as it often prevents the algorithms from premature convergence. In recent years, evolutionary diversity optimization has

gained increasing attention [3, 74, 194, 252, 253]. Evolutionary diversity optimization uses an evolutionary algorithm in order to compute a diverse set of solutions that all fulfill given quality criteria. Presenting decision makers with such alternative designs that are all of good quality gives them a variety of design choices and helps to better understand the space of good solutions for the problem at hand. Related to evolutionary diversity optimization is the concept of novelty search [220, 238]. Here evolutionary algorithms are used to discover new designs without focusing on an objective. The goal of novelty search is to explore designs that are different to the ones previously obtained.

In the previous chapter, we have shown the framework for evolutionary diversity optimization based on works of Ulrich and Thiele [253]. They studied how to evolve diverse sets of instances for single-objective problems to the underlying search space. Furthermore, this diversity optimization approach has been introduced into multi-objective search [252]. This approach has been adapted in order to create variations of a given image that are close to it but differ in terms of the chosen image features [3]. We presented this approach in more detail in Chapter 8.

An important question that arises when using evolutionary diversity optimization for more than one criterion or feature is how to measure the diversity of a given set of solutions. The weighted contribution approach used in [3, 73] has the disadvantage that it heavily depends on the chosen weightening of the features and does not distribute that well for two or three dimensions. In [194], an evolutionary diversity optimization approach has been introduced that aims to minimize the discrepancy of the solution set in the feature space. It has been shown that using the star discrepancy as a diversity measure achieves sets of higher diversity than the previous approaches using weighted contributions.

In this chapter, we show how to use popular indicators from the area of evolutionary multi-objective optimization for evolutionary diversity optimization. Indicators play a prominent role in the area of evolutionary multi-objective optimization and are frequently used to assess the quality of solution sets produced by evolutionary multi-objective algorithms [277, 278]. Based on the evaluation of this indicator the selection for survival is carried out. We show how to adapt popular indicators in the area of evolutionary multi-objective optimization to evolutionary diversity optimization. We study important indicators such as the hypervolume indicator (HYP), the inverted generational distance (IGD), and the additive epsilon approximation (EPS), and compare them in terms of their ability to lead to high quality and diverse sets of solutions.

We investigate these indicators for the problem of constructing diverse sets of images as already studied in the literature. Our results show that HYP and IGD are well suited for evolutionary diversity optimization. They obtain the best results for the their respective indicator and also obtain sets of solutions of a better discrepancy when comparing them to the discrepancy-based approach given in [194].

The outline of the chapter is as follows. First, we describe our approach in Section 10.2. Then, in Section 10.3, we introduce evolutionary diversity optimization using multi-objective indicators for our problem: evolving diverse sets of images. Finally, in Section 10.4 we draw some conclusions.

10.2 Indicator-based Diversity Optimization

Let $I \in X$ be a search point in a given search space X, $f\colon X \to \mathbb{R}^d$ a function that assigns to each search point a feature vector and $q\colon X \to \mathbb{R}$ be a function assigning a quality score to each $x \in X$ [13]. Diversity is defined in terms of a function $D\colon 2^X \to \mathbb{R}$ which measures the diversity of a given set of search points. Considering evolutionary diversity optimization, the goal is to find a set $P = \{I_1, \ldots, I_\mu\}$ of μ solutions maximizing D among all sets of μ solutions under the condition that $q(I) \geq \alpha$ holds for all $I \in P$, where α is a given quality threshold. Here μ is the size of the set that we are aiming for, which determines the parent population size in our evolutionary diversity optimization approach.

As already outlined in Chapter 2, diversity has been optimized in a few different ways over the years. Of particular interest to us is the optimization of diversity in a given set of problem instances. We will use this domain as an application area to demonstrate that the general goal of diversity optimization with respect to multiple features is achievable. If diversity is sought with respect to a single feature, then the generation of instances can focus on covering the range of values in some fashion. If two or more features are of interest, then covering this space evenly is not straightforward, as a metric is needed to assess the coverage.

Recently, [194] have used the mathematical concept of "discrepancy" to measure the irregularities of distributions and used this measure for evolutionary diversity optimization. The used star-discrepancy uses axis-parallel boxes: ideally, the number of points inside the box is proportional to the size of the box. The computation of this metric is time consuming ($n^{1+d/2}$ [38]) and the resulting distributions are counter-intuitive.

Here, we propose to use a very well-established concept, i.e., the use of indicators from multi-objective optimization. In multi-objective optimization, a function $g\colon X \to$

$\mathbb{R}^d$ containing d objectives is given and all objectives should be optimized at the same time. As the given objectives are usually conflicting, one is interested in the trade-offs with respect to the given objective functions.

Indicators in the area of multi-objective optimization have been used for many years to compare sets of solutions in the objective space, either for the purposes of comparing algorithm performance, or for use within an algorithm to drive a diversified search. Similarly to the diversity measure D in evolutionary diversity optimization, an indicator $\mathcal{I}\colon 2^X \to \mathbb{R}$ measures the quality of a set of solutions according to some indicator function $\mathcal{I}$. The immediate problem with applying multi-objective optimization indicators is that diversity does not have a notion of dominance. In the context of multi-objective optimization, the optimal solutions are also referred to as non-dominated solutions. A solution x is called non-dominated (or Pareto optimal) if there is no other solution that is at least as good as x with respect to every objective and better in at least one objective. As multi-objective approaches aim to compute a set of non-dominated solutions, they reject dominated solutions over time. In evolutionary diversity optimization, every solution meeting the quality criteria is eligible and only the diversity among such solutions matters. Hence, we have to adapt the multi-objective indicators in a way that makes all solutions meeting the quality criterion non-dominated. We do this by ensuring that all solutions are incomparable when applying these indicators. For a more comprehensive introduction to dominance we refer the interested reader to [14], which is present in a large number of multi-objective optimization indicators.

In the following, we will first intoduce our evolutionary algorithm for optimizing diveristy in Section 10.2.2. Then, we introduce the generic $(\mu + \lambda)$-EA_D and the concrete variants that will form the basis for our subsequent experimental studies on diversity optimization. In Chapter 2 we presented existing multi-objective optimization indicators for diversity optimization.

10.2.1 Multi-objective Optimization Indicators for Diversity Optimization

In this work, we use three quality indicators that evaluate the quality of a given set of objective vectors S. For a given set of search points P which we call the population, and a function $g\colon X \to \mathbb{R}^d$, we define $S = \{g(x) \mid x \in P\}$ as the set of objective vectors of P. The indicators are as follows:

- Hypervolume (HYP): HYP is the volume covered by the set of objective vectors S with respect to a given reference point r.

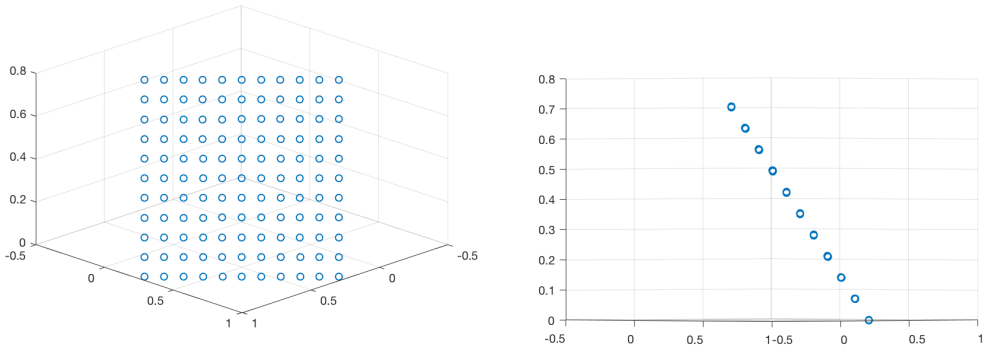


FIGURE 10.1: Reference set in 3D using 11^2 objective vectors. The normal vector that goes through the centre of the square goes through the origin. We use 101^2 feature vectors in our experiments. Note that all solution vectors are by definition on the Pareto front, i.e., in the unit square.

- Inverted generational distance (IGD): IGD measures S with respect to a given reference set R
- Additive epsilon approximation (EPS): EPS measures the approximation quality of the worst approximated point in R that S achieves.

In Chapter 2.5.7, we give a detailed description on these three indicators. It have to be mention that indicators cannot be applied immediately, as there is no reference set (which some indicators require) and one has to deal with the issue of dominance as there is no preference of one feature value over the other. For example, let us consider two scaled features and visualize the combinations as points in a two-dimensional unit square. In this case, we would like to cover the entire square evenly, without preferring one region over the other, and in particular we cannot say that one area is preferred over another – a naive multi-objective optimization setup for this two-dimensional problem might focus, for example, only on the area near the origin.

We propose two approaches to deal with this challenge: (1) transformation of the two-dimensional problem into a three-dimensional problem, (2) doubling the number of dimensions.

10.2.1.1 Problem Transformation

When we are interested in covering a two-dimensional feature space, we can mitigate the problem of EPS-/HYP-preferred regions by transforming the two-dimensional problem into a three-dimensional one. We do so as follows:

1. We place the unit square with its original x/y-coordinates in the three-dimensional space using $z = 0$.

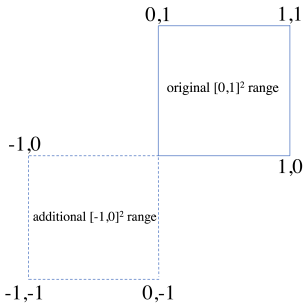


FIGURE 10.2: Visualisation of the 2D-dimensional space.

2. We rotate it around the x and y axis by 45 degrees each time.

3. We translate it such that the center point of the transformed unit square is at $(sqrt(2)/4)^3$ (see Figure 10.1).

After these steps, the normal vector that goes through the center of the unit square also goes through the origin.Note that tor the rotation, we use Java 1.8's method java.awt.geom.AffineTransform.getRotateInstance(...). This orientation allows us to use the wide spectrum of well-established quality indicators from the field of multi-objective optimization, designed for assessing various aspects of solutions sets, such as convergence and distribution – and no modifications are needed at all. Especially for the volume- and dominance-based indicators our transformation has the important benefit that all features are of equal importance.

As we perform the same transformation with the instance set (i.e., our population) as well as the reference set (after rescaling it into the unit square based on known lower and upper values for the features), this means that the population is always on the Pareto front; this is a situation that is not that common in multi-objective optimization. Our goal is now to cover the reference set "evenly", as defined by the respective indicators.

10.2.1.2 Dimension Doubling

To avoid the dominance issue, we propose the following transformation. Given a feature vector $p = (p_1, p_2, \ldots, p_d)$ in the d-dimensional space, we project it into the $2D$-dimensional space by copying the original feature values and negating their copy, resulting in

$$p' = (p_1, p_2, \ldots, p_d, -p_1, -p_2, \ldots, -p_d),$$

see Figure 10.2. With this, dominance between solution vectors vanishes, and we can employ the hypervolume indicator without the need for any modifications.

Algorithm 15 $(\mu + \lambda)$-EA_D

Initialize the population P with μ instances of quality at least α.
Let $C \subseteq P$ where $|C| = \lambda$.
For each $I \in C$, produce an offspring I' of I by mutation. If $q(I') \geqslant \alpha$, add I' to P.
While $|P| > \mu$, remove an individual with the smallest loss to the diversity indicator D.
Repeat step 2 to 4 until termination criterion is reached.

Because we work with rescaled value ranges in $[0, 1]^d$, the necessary hypervolume reference point r has to be adequately chosen in the 2D-dimensional space. For example. (1^d,0^d) would be based on the ranges' extreme values, and (2^d,1^d) would put an increased focus on maintaining extreme points in the population.

While this transformation mitigates the dominance issue, it remains an open problem how this can be made to work with the epsilon indicator as well. The challenge here is to define an evenly spread out reference set in the 2D-dimensional space given our dimension doubling.

10.2.2 Evolutionary Algorithm for Optimizing Diversity

The algorithm used to optimize the feature-based population diversity follows the setting in [74] with modifications. Algorithm 15 shows the evolutionary algorithm used for optimizing diversity. Let $I \in P$ be an individual in a population P. A problem specific feature vector $f(I) = (f_1(I), \ldots, f_d(I))$ is used to describe a potential solution. The indicators are calculated based on the feature vector.

Since the indicators introduced are defined in the space of $[0, 1]^d$, the feature values are scaled before the calculation of indicators. Let $f_i^{\max}$ and $f_i^{\min}$ be the maximum and minimum value of a certain feature f_i obtained from some initial experiments. The feature values are normalized based on the formula

$$f_i'(I) = (f_i(I) - f_i^{\min})/(f_i^{\max} - f_i^{\min}).$$

Feature values outside the range $[f_i^{\min}, f_i^{\max}]$ are set to 0 or 1, to allow the algorithm to work with non-anticipated features values.

Based on this, we investigate the following diversity-optimizing algorithms in this study:

- $EA_{HYP\text{-}2D}$ and EA_{EPS} use the idea of transforming the two-dimensional problem into a three-dimensional one.

FIGURE 10.3: Image I^*.

- EA_{HYP} uses the idea of doubling the dimensions.
- EA_{IGD} uses IGD, which can be used without the need to transform the feature vectors, as it does not consider concepts like dominance or volume like HYP and EPS.

In addition, we use EA_{DIS} with discrepancy minimization, as used in [194]. As IGD and EPS require a reference set (e.g. solutions situated on the Pareto front), we use regular grids in the unit square and unit cube with a resolution of 101^2 solutions and 11^3 solutions. The necessary hypervolume reference point r for $\text{EA}_{\text{HYP-2D}}$ is set based on the extreme values of the reference set after the described rotations; for EA_{HYP} it is set to $(2^d,1^d)$ to increase the focus on extreme points. Note that these indicators are a major differentiator from the work in [184]. There, the approach was able of only evolving one instance at a time with the goal of reaching a particular target vector. In our case, we evolve an entire population of diverse instances and do not require explicitly set targets.

10.3 Experimental Investigations

In this section, we aim to evolve a diverse set of images as described in [3]. Given an image I^*, we want to compute a diverse set of images $P = \{I_1, \ldots, I_\mu\}$ that agree on a given quality criteria $q(I)$ for each $I \in P$. We will use the image I^* given in Figure 10.3 for our investigations. An image I fulfills the quality criteria $q(I)$ if the *mean-squared error* in terms of the RGB-values of I with respect to I^* is less than 500.

Many different features have been widely applied to measurements of the properties of images. They often provide a good characterization of images. We select the set of features identified in [3] and described in more detail in Chapter 3. We carry out the

	Notation	f^{min}	f^{max}	Description
f_1	SDHue	0.420	0.700	standard deviation hue
f_2	saturation	0.420	0.500	mean saturation
f_3	symmetry	0.715	0.740	reflectional symmetry
f_4	hue	0.250	0.400	color descriptor
f_5	GCF	0.024	0.027	Global Contrast Factor
f_6	smoothness	0.906	0.918	smoothness

TABLE 10.1: Description of features for images.

indicator-based evolutionary optimization approach with respect to different multi-objective indicators and different sets of features. Our evolutionary algorithm evolves diverse populations of images for each indicator and for each feature combination.

In our experiments we used the following features: standard-deviation-hue, mean-saturation, reflectional symmetry [99], hue [108], Global Contrast Factor [172], and smoothness [203] (see Chapter 3). Instead of applying the star discrepancy [246] to measure diversity we use the multi-objective indicators as previously introduced. Otherwise, the configuration of Algorithm 15 is the same as in [194]. In order to produce a new solution the algorithm uses a self-adaptive offset random walk mutation introduced in [194]. Based on a random walk on the image this operator alters the RGB-values of the pixels visited in a slight way such that a new but similar image is obtained. Random walk lengths are increased in the case of a successful mutation and decreased in the case of unsuccessful ones. For details, we refer the reader to [194, 196].

10.3.1 Experimental Settings

Now, we consider the indicator-based diversity optimization for combinations of two and three features. We select features in order to combine different aesthetic and general features based on our initial experimental investigations and previous investigations in [188]. In this section, we explore several features and features ranges described in Table 10.1. We use scaled feature values while we calculate the different indicators values of a given set of points. After having consider the combination of two features, we investigate sets of three features. Here, we select different features combining aesthetic and general features together used in the previous experiment.

In order to obtain a clear comparison between our present experiments and experiments based on the discrepancy-based evolutionary algorithm introduced in [194] we

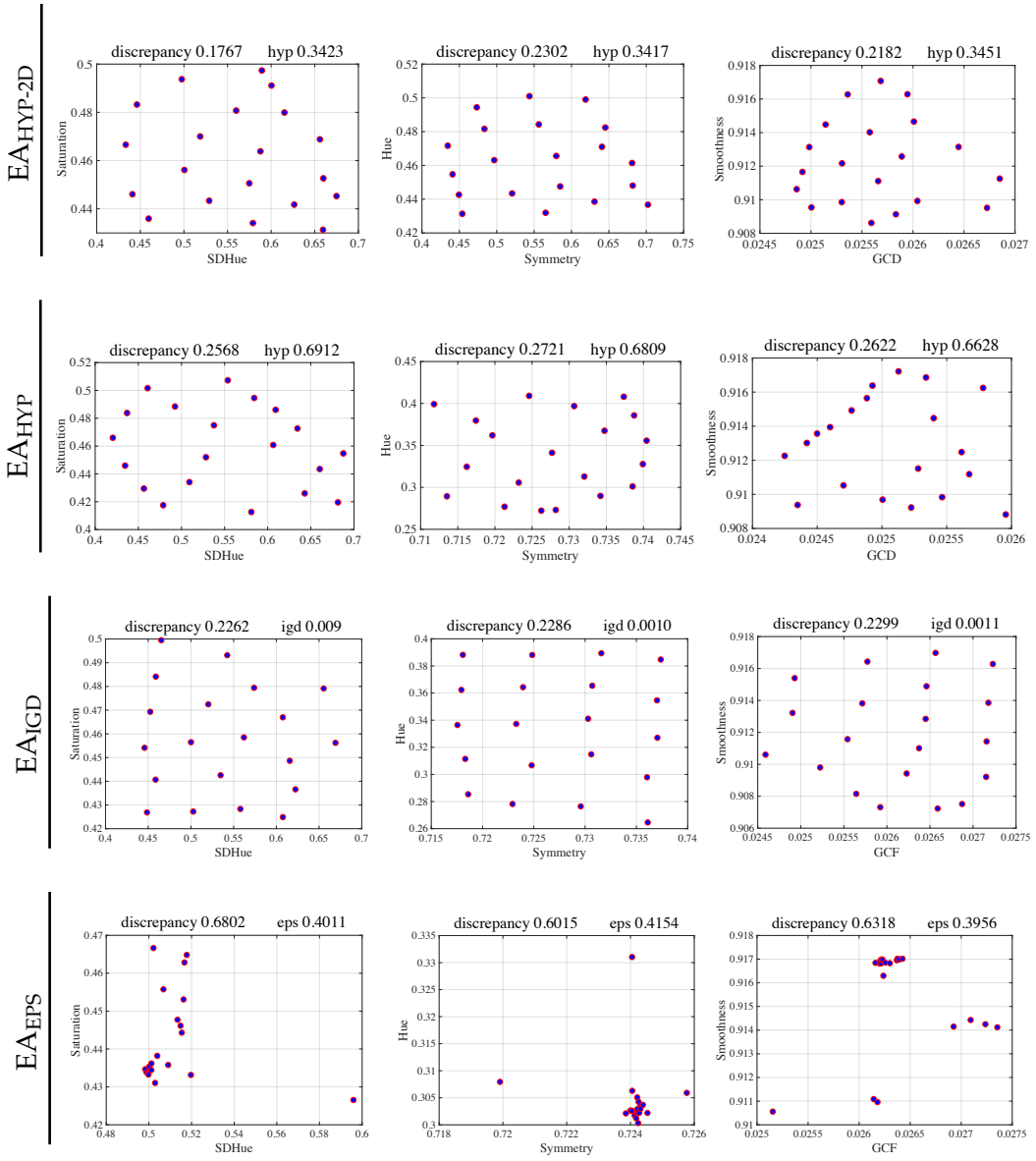


FIGURE 10.4: Feature vectors for final population of $EA_{HYP\text{-}2D}$ (top), EA_{HYP}, EA_{IGD} and EA_{EPS} (bottom) for images based on pair of features from left to right: (f_1, f_2), (f_3, f_4), (f_5, f_6).

work with the same range of feature values.

We run each configuration for 2000 generations with a population size of $\mu = 20$ and $\lambda = 1$. To assess our results using statistical tests, we run each combination of feature-pair and indicator 30 times with the same setting applied to each considered pairs of features. All algorithms were implemented in $Matlab$ ($R2017b$) and run on 48-core compute nodes with AMD 2.80 GHz CPUs and 128 GB of RAM.

		$EA_{HYP\text{-}2D}$ (1)			EA_{HYP} (2)			EA_{IGD} (3)			EA_{EPS} (4)			EA_{DIS} (5)		
		mean	st	stat	mean	st	stat	mean	st	stat	mean	st	stat	mean	st	stat
HYP-2D	f_1,f_2	0.347	0.004	$4^{(+)},5^{(+)}$	0.382	0.007	$3^{(+)},4^{(+)},5^{(+)}$	0.335	0.003	$2^{(-)},5^{(+)}$	0.198	0.019	$1^{(-)},2^{(-)}$	0.112	0.030	$1^{(-)},2^{(-)},3^{(-)}$
	f_3,f_4	0.344	0.004	$2^{(+)},4^{(+)},5^{(+)}$	0.268	0.014	$1^{(-)},3^{(-)},4^{(+)},5^{(+)}$	0.339	0.004	$2^{(+)},4^{(+)},5^{(+)}$	0.221	0.015	$1^{(-)},2^{(-)},3^{(-)}$	0.105	0.025	$1^{(-)},2^{(-)},3^{(-)}$
	f_5,f_6	0.350	0.007	$2^{(+)},3^{(+)},4^{(+)},5^{(+)}$	0.342	0.004	$1^{(-)},4^{(+)},5^{(+)}$	0.332	0.004	$1^{(-)},4^{(+)},5^{(+)}$	0.220	0.045	$1^{(-)},2^{(-)},3^{(-)}$	0.134	0.016	$1^{(-)},2^{(-)},3^{(-)}$
HYP	f_1,f_2	0.525	0.012	$3^{(+)},4^{(+)},5^{(+)}$	0.693	0.013	$3^{(+)},4^{(+)},5^{(+)}$	0.374	0.006	$1^{(-)},2^{(-)},4^{(+)}$	0.344	0.003	$1^{(-)},2^{(-)},3^{(-)}$	0.363	0.014	$1^{(-)},2^{(-)}$
	f_3,f_4	0.500	0.007	$3^{(+)},4^{(+)},5^{(+)}$	0.681	0.010	$3^{(+)},4^{(+)},5^{(+)}$	0.268	0.072	$1^{(-)},2^{(-)},4^{(+)},5^{(+)}$	0.280	0.010	$1^{(-)},2^{(-)},3^{(-)}$	0.267	0.014	$1^{(-)},2^{(-)},3^{(-)}$
	f_5,f_6	0.518	0.012	$2^{(-)},4^{(+)},5^{(+)}$	0.663	0.010	$1^{(+)},3^{(+)},4^{(+)},5^{(+)}$	0.335	0.004	$2^{(-)},4^{(+)}$	0.317	0.006	$1^{(-)},2^{(-)},3^{(-)}$	0.327	0.008	$1^{(-)},2^{(-)}$
IGD	f_1,f_2	0.001	0.335	$2^{(+)},4^{(+)},5^{(+)}$	0.003	0.000	$1^{(-)},3^{(-)}$	0.001	0.000	$2^{(+)},4^{(+)},5^{(+)}$	0.003	0.000	$1^{(-)},3^{(-)},5^{(+)}$	0.005	0.001	$1^{(-)},3^{(-)},4^{(-)}$
	f_3,f_4	0.001	0.339	$2^{(+)},4^{(+)},5^{(+)}$	0.004	0.000	$1^{(-)},3^{(-)},5^{(+)}$	0.001	0.000	$2^{(+)},4^{(+)},5^{(+)}$	0.003	0.000	$1^{(-)},3^{(-)},5^{(+)}$	0.005	0.001	$1^{(-)},2^{(-)},3^{(-)},4^{(-)}$
	f_5,f_6	0.002	0.332	$2^{(+)},5^{(+)}$	0.007	0.000	$1^{(-)},3^{(-)},4^{(-)},5^{(-)}$	0.001	0.000	$2^{(+)},4^{(+)},5^{(+)}$	0.003	0.001	$2^{(+)},3^{(-)}$	0.004	0.001	$1^{(-)},2^{(+)},3^{(-)}$
EPS	f_1,f_2	0.190	0.198	$2^{(+)},4^{(+)},5^{(+)}$	0.498	0.011	$1^{(-)},3^{(-)}$	0.194	0.032	$2^{(+)},4^{(+)},5^{(+)}$	0.402	0.039	$1^{(-)},3^{(-)},5^{(+)}$	0.600	0.106	$1^{(-)},3^{(-)},4^{(-)}$
	f_3,f_4	0.198	0.221	$2^{(+)},4^{(+)},5^{(+)}$	0.569	0.016	$1^{(-)},3^{(-)}$	0.208	0.035	$2^{(+)},4^{(+)},5^{(+)}$	0.418	0.036	$1^{(-)},3^{(-)},5^{(+)}$	0.615	0.069	$1^{(-)},3^{(-)},4^{(-)}$
	f_5,f_6	0.125	0.220	$2^{(+)},4^{(+)},5^{(+)}$	0.946	0.001	$1^{(-)},3^{(-)},4^{(-)}$	0.225	0.064	$2^{(+)},4^{(+)},5^{(+)}$	0.397	0.110	$1^{(-)},2^{(+)},3^{(-)}$	0.587	0.063	$1^{(-)},3^{(-)}$
DIS	f_1,f_2	0.171	0.018	$2^{(+)},4^{(+)},5^{(+)}$	0.257	0.010	$1^{(-)},4^{(+)}$	0.201	0.031	$4^{(+)},5^{(+)}$	0.686	0.064	$1^{(-)},2^{(-)},3^{(-)},5^{(-)}$	0.204	0.116	$1^{(-)},3^{(-)},4^{(+)}$
	f_3,f_4	0.234	0.031	$4^{(+)}$	0.273	0.041	$3^{(-)},4^{(+)},5^{(-)}$	0.198	0.017	$2^{(+)},4^{(+)}$	0.606	0.054	$1^{(-)},2^{(-)},3^{(-)},5^{(-)}$	0.228	0.059	$2^{(+)},4^{(+)}$
	f_5,f_6	0.221	0.026	$4^{(+)}$	0.263	0.070	$3^{(-)},4^{(+)},5^{(-)}$	0.205	0.055	$2^{(+)},4^{(+)}$	0.633	0.158	$1^{(-)},2^{(-)},3^{(-)},5^{(-)}$	0.203	0.054	$2^{(+)},4^{(+)}$

TABLE 10.2: Investigations for images with two features. Comparison in terms of mean, standard deviation and statistical test for considered indicators.

10.3.2 Experimental Results and Analysis

We present a series of experiments for two- and three-feature combinations in order to evaluate our evolutionary diversity algorithms based on the use of indicators from multi-objective optimization described in Section 10.2.

10.3.2.1 Two-feature Combinations

Our results are summarized in Table 10.2 and Table 10.3. The columns represent the algorithms with the corresponding mean value and standard deviation. The rows represent the indicators HYP-2D, HYP, IGD, EPS and discrepancy (DIS). For each indicator, we obtained results for all sets of features.

Additionally, we use the Kruskal-Wallis test for statistical validation with 95% confidence and subsequently apply the Bonferroni post-hoc statistical procedure. For a detailed description of the statistical tests we refer the reader to [27]. Our experimental analysis characterizes the behavior of the four examined indicator-based evolutionary algorithms and discrepancy-based evolutionary algorithm. In the statistical tests shown in Table 10.2 and Table 10.3, $A^{(+)}$ is equivalent to the statement that the algorithm in this column outperformed algorithm A, and $A^{(-)}$ is equivalent to the statement that A outperformed the algorithm given in the column. If the algorithm A does not appear, this means that no significant difference was determined.

Figure 10.4 illustrates feature plots of (randomly selected) final populations of $EA_{HYP\text{-}2D}$ (top), EA_{HYP}, EA_{IGD} and EA_{EPS} (bottom) for three pairs of feature combinations. In the first column, we see the feature vectors for the final population of the

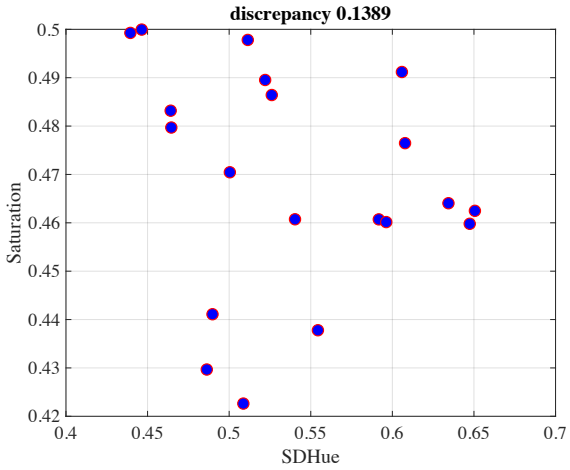


FIGURE 10.5: Feature vectors for final population of EA_{DIS} [194] for images based on (f_1, f_2).

four algorithms for image based on pairs of features (f_1,f_2). It can be observed that the discrepancy value for $EA_{HYP\text{-}2D}$ is 0.1767. This is significantly smaller than the one for EA_{EPS} at 0.6802. Note that smaller discrepancy values are considered to be better. The middle column shows the combination of the feature pair (f_3,f_4). The discrepancy value for feature pair (f_3,f_4) for EA_{IGD} is 0.2286 whereas it is 0.6015 for EA_{EPS}. The last column shows the final populations of the diversity optimization when considering feature pair (f_5,f_6). The discrepancy value for feature pair (f_5,f_6) is the smallest among all algorithms for $EA_{HYP\text{-}2D}$ at 0.2182 and the highest for EA_{EPS} at 0.6318.

In summary, we observe that $EA_{HYP\text{-}2D}$, EA_{HYP} and EA_{IGD} achieve a good and even coverage of the feature space, especially in comparison with the discrepancy-based diversification (see Figure 10.5 for an example from [194]). Interestingly, EA_{EPS} appears to experience difficulties, and it achieves the worst coverage in the search space in all scenarios.

Moreover, in Table 10.2, we observe that the EA_{HYP} algorithm has the best performance among all algorithms. It has the highest hypervolume values for all features combinations, and this is also statistically significant. Also, due to the statistical tests we can say that $EA_{HYP\text{-}2D}$ outperforms EA_{EPS} and EA_{DIS} with respect to the inverted generational distance and additive epsilon approximation indicator measurements values for all sets of features. We observe that $EA_{HYP\text{-}2D}$ considering IGD and EPS values has no significant differences to EA_{IGD}. In terms of discrepancy, the $EA_{HYP\text{-}2D}$ has a following characteristic: for set of features (f_1,f_2) the $EA_{HYP\text{-}2D}$ outperforms EA_{HYP}, EA_{EPS} and EA_{DIS}, however, it only outperforms the EA_{EPS} for the set of features (f_3,f_4) and (f_5,f_6).

Furthermore, EA_{IGD} outperforms the EA_{HYP}, EA_{EPS} and the EA_{DIS} with respect to

IGD, EPS and DIS indicators measurements in most of the cases and achieves the lowest values for IGD measurements among all others algorithms for all sets of features. The best performance achieves EA_{IGD} for discrepancy measurements for the combinations of features (f_3,f_4) and (f_5,f_6) with values 0.198 and 0.205. The hypervolume-based approaches $EA_{HYP\text{-}2D}$ and EA_{HYP} outperform EA_{IGD} for all sets of features.

Among all others algorithms EA_{EPS} shows the worst performance. Especially, according to all indicators measurements and all sets of features, the EA_{EPS} is dominated by EA_{HYP} and EA_{IGD}, and this difference is statistically significant.

Finally, the EA_{DIS} is dominated by $EA_{HYP\text{-}2D}$ and EA_{HYP}, EA_{IGD} and EA_{EPS} with respect to the HYP-2D, HYP, IGD and EPS indicator values. Also, most results are significantly different with respect to the HYP, IGD, EPS indicators. EA_{DIS} achieves the best performance with respect to the DIS indicator for the combinations of features (f_3,f_4) and (f_5,f_6). The EA_{DIS} outperforms the EA_{HYP} and EA_{EPS} in this case. For the combinations (f_1,f_2) with respect to the DIS indicator, the EA_{DIS} is dominated by $EA_{HYP\text{-}2D}$ and EA_{IGD}.

10.3.2.2 Three-feature Combinations

The triplets of features are described in Table 10.1 and the results are summarized in Table 10.3. As before, the columns represent the algorithms with the corresponding mean value and standard deviation, and the rows represent the indicators.

Figure 10.6 shows feature plots of (randomly selected) final populations of EA_{HYP} (top), EA_{IGD} and EA_{DIS} (bottom) for all sets of features. We can observe that the HYP value for EA_{HYP} is 0.5249, which is significantly higher than the ones for EA_{IGD} at 0.2092 and EA_{DIS} at 0.2193. The IGD value for EA_{IGD} is the lowest (and best) at 0.0065. The EA_{DIS} achieves the lowest (and best) discrepancy value 0.3352. The situation is similar for the other two triplets. The HYP values for EA_{HYP} 0.4993 and 0.5177 are significantly higher than the ones for EA_{IGD} at 0.2139 and 0.1784, and accordantly for EA_{DIS} at 0.2284 and 0.1957. In contrast, EA_{IGD} obtains the smallest discrepancy values at 0.2858 for the second set of features.

In Table 10.3, we compare EA_{HYP} and EA_{IGD} with EA_{DIS} algorithm with respect to two multi-objective indicators and the discrepancy measurement. Table 10.3 shows that EA_{HYP} outperforms EA_{IGD} and EA_{DIS} for all three sets of features with respect to the HYP indicator. In particular, for the first set of features (f_1,f_2,f_3) the EA_{HYP} algorithm obtains the value 0.5251, and only 0.2096 for IGD, and 0.2196 for discrepancy.

Comparing EA_{IGD} to EA_{HYP} and EA_{DIS} with respect to the IGD indicator, we find a similar picture as for the EA_{HYP} algorithm. EA_{IGD} clearly outperforms the EA_{HYP}

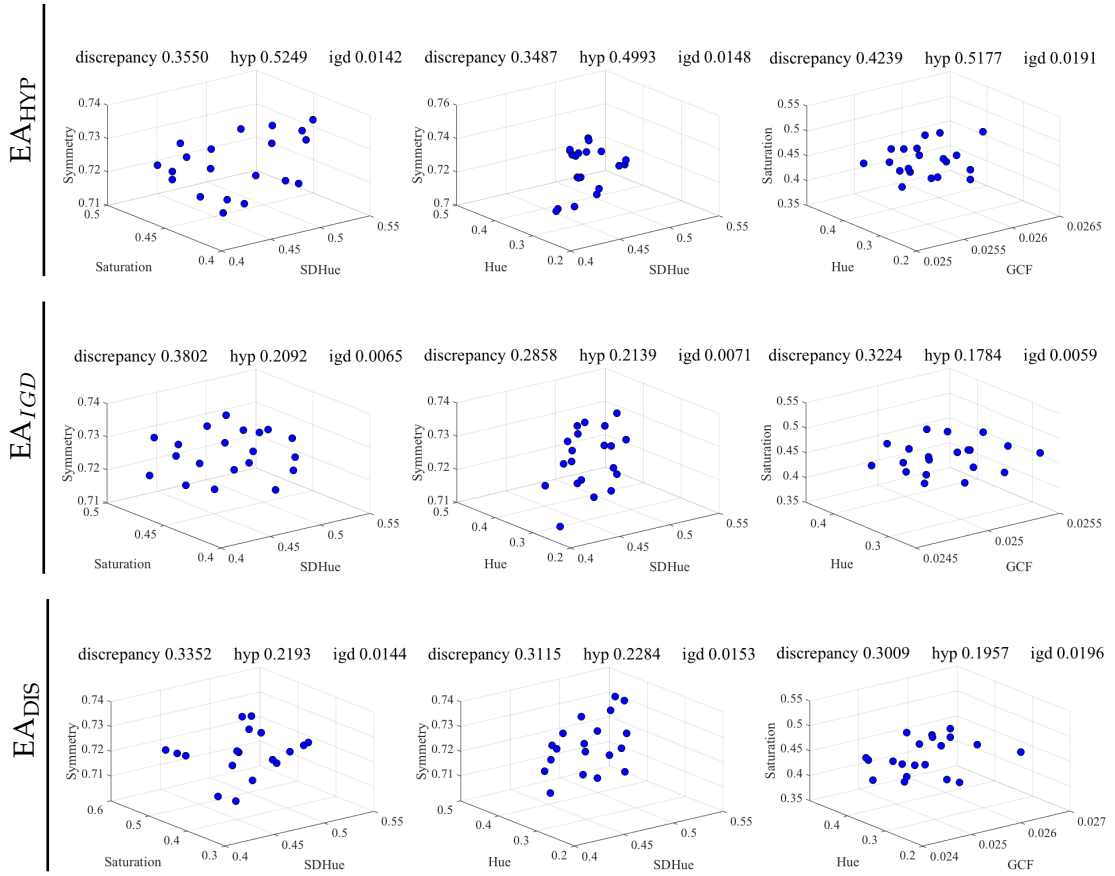


FIGURE 10.6: Feature vectors for final population of EA_{HYP} (top), EA_{IGD} (middle) and EA_{DIS} (bottom) for images based on three features from left to right: (f_1, f_2, f_3), (f_1, f_4, f_3), (f_5, f_4, f_2).

and EA_{DIS} for all three sets of features. The EA_{DIS} algorithm also clearly outperforms EA_{HYP} and EA_{IGD} with respect to discrepancy. Overall, EA_{DIS} achieves improvements in terms of discrepancy value among another two algorithms for all sets of features apart from one exception. It can be observed that for the set of feature (f_1,f_4,f_3) EA_{DIS} does not have a major advantage over the EA_{IGD}.

In a nutshell, according to our statistical tests the EA_{HYP} outperforms all examined algorithms with respect to the HYP indicator values for all sets of features in case of two-feature combination. Moreover, EA_{IGD} outperforms EA_{HYP}, EA_{EPS} and EA_{DIS} with respect to the IGD indicator, which was expected, but it shows no significant difference to $EA_{HYP\text{-}2D}$ for the first two sets of features. The EA_{EPS} algorithm has the worst performance, no matter the indicator considered. Similarly, considering our experiments for three-feature combinations, EA_{HYP} and EA_{IGD} achieve the best results, which are also statistically significant.

		EA_{HYP} (1)			EA_{IGD} (2)			EA_{DIS} (3)		
		mean	st	stat	mean	st	stat	mean	st	stat
HYP	f_1,f_2,f_3	0.5251	0.0122	$2^{(+)},3^{(+)}$	0.2096	0.0018	$1^{(-)},3^{(-)}$	0.2196	0.0110	$1^{(-)},2^{(+)}$
	f_1,f_4,f_3	0.4998	0.0071	$2^{(+)},3^{(+)}$	0.2142	0.0036	$1^{(-)},3^{(-)}$	0.2286	0.0034	$1^{(-)},2^{(+)}$
	f_5,f_4,f_2	0.5181	0.0122	$2^{(+)},3^{(+)}$	0.1785	0.0017	$1^{(-)},3^{(-)}$	0.1961	0.0023	$1^{(-)},2^{(+)}$
IGD	f_1,f_2,f_3	0.0146	0.0001	$2^{(-)},3^{(+)}$	0.0067	0.0003	$1^{(+)},3^{(+)}$	0.0148	0.0003	$1^{(-)},2^{(-)}$
	f_1,f_4,f_3	0.0150	0.0001	$2^{(-)}$	0.0074	0.0002	$1^{(+)},3^{(+)}$	0.0151	0.0001	$2^{(-)}$
	f_5,f_4,f_2	0.0193	0.0001	$2^{(-)},3^{(+)}$	0.0062	0.0002	$1^{(+)},3^{(+)}$	0.0199	0.0007	$1^{(-)},2^{(-)}$
DIS	f_1,f_2,f_3	0.3554	0.0458	$2^{(+)},3^{(-)}$	0.3809	0.0522	$1^{(-)},3^{(-)}$	0.3350	0.1002	$1^{(+)},2^{(+)}$
	f_1,f_4,f_3	0.3493	0.0532	$2^{(-)}$	0.2860	0.0342	$1^{(+)},3^{(+)}$	0.3118	0.1309	$2^{(-)}$
	f_5,f_4,f_2	0.4237	0.0643	$2^{(-)},3^{(-)}$	0.3227	0.0557	$1^{(+)},3^{(-)}$	0.3007	0.1467	$1^{(+)},2^{(+)}$

TABLE 10.3: Investigations for images with tree features. Comparison in terms of mean, standard deviation and statistical test for considered indicators.

10.4 Conclusions

In this chapter, we proposed a new approach for evolutionary diversity optimization. It bridges the areas of evolutionary diversity optimization and evolutionary multi-objective optimization and shows how techniques developed in evolutionary multi-objective optimization can be used to come up with diverse sets of solutions of high quality for a given single-objective problem.

Our investigations demonstrated that well-established multi-objective performance indicators can be used to achieve a good diversity of sets of solutions according to a given set of features. The advantages of our approaches are (i) their simplicity and (ii) the quality of diversity achieved as measured by the respective indicators. The best performing approaches use HYP or IGD as indicators. We have shown that they achieve excellent results in terms of all indicators and often even outperform the discrepancy-based approach [194] when measuring quality in terms of discrepancy. This is surprising as they are not tailored towards this measure.

In this chapter, we concentrated on using popular multi-objective indicators in existing diversity optimization approaches. For future work, it would be interesting to use popular evolutionary multi-objective approaches such as MOEA/D, IBEA or NSGA-II/III for evolutionary diversity optimization.

A conference version that contains the results of this chapter has been accepted for publication in the Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2019 (see A. Neumann, W. Gao, M. Wagner, F. Neumann [195]).

Chapter 11

Evolution of Images Using Generative Adversarial Networks

11.1 Introduction

In this chapter, we investigate the use of Generative Adversarial Networks (GANs) in order to generate novel images in the lower dimensional space. There has been remarkable progress made in the direction of image synthesis with the emergence of the generative modelling approach based on a generator network, that is an important problem in computer vision. In contrast to previously works, we create images by evolving the latent vector to maximise and minimise single and two-dimensional image feature values for two datasets, namely faces and butterflies. In our system, images are created by optimising the latent space of the GAN in order to create images that score high or low on feature measures. We show that the generation of images in this space requires the use of carefully constructed constraints regarding image realism. We also show that GANs trained on different image sets appear to impose different bounds on the feature values that can be evolved. We show that GANs and evolutionary computation methods can be successful in generating of photorealistic images. Our approach can be applied for the discovery of new images that can be used in field of designing video games.

In recent years, Generative Adversarial Networks [84] have been used to map low dimensional real-valued latent vector into images. GANs are a machine learning approach that have the ability to generate novel images. Preliminary description of GANs we describe in Chapter 2. There has been work in using GANs to generate and mix novel images [201] and perform style transfer [75], along with other applications. However, to date there has no work using evolutionary to explore the latent space of a GAN to generate images according to feature measures. Evolutionary search has frequently

been used to generate artistic images [99, 138, 188]. In previous work [28, 175], have either reduced the dimensionality of the search space through programmatic encodings or have constrained the images with priors [3, 188].

Aesthetic feature measures have been often applied to the creation of new artistic images using evolutionary search [28, 99, 138, 190]. There has also been work in the evolution of existing images [189, 196]. This work differs from previous work in the use of a GAN as a mapper from latent search vector to the image feature space and also in the use of the discriminator network and feature metrics to constrain these images.

In terms of deep learning, Gatys [75, 76, 78] used a convolutional network to transfer artistic style into existing image. These new approaches in network architectures and training methods enabled the generation of realistic images [53, 212]. Recently Dosovitskiy and Brox [52] trained networks of generating images from feature vector and combining an auto-encoder-style approach with Deep Convolutional Generative Adversarial Networks training. Furthermore, Nguyen [201] used priors from Deep Generative Network to generate image variants that look close to natural within a preferred inputs for neurons.

The evolutionary algorithm for diversity optimisation was introduced in [74], based on optimisation for Traveling Salesman Problem (TSP) which is a NP-hard combinatorial optimisation problem with real world applications. For our investigation we consider recent work [3] on feature-based diversity optimisation for images.

The chapter is structured as follows. Section 11.2 presents the methodology used for evolving images. Sections 11.3 presents our results on single dimensional feature experiments with constraints. Here, we describe the main approach on single and two-dimensional feature experiments with the Cut-off function. Finally, Section 11.4 discusses results and future work.

11.2 Evolution of Diverse Images Using GANs

In this section, we discuss the methods used to evolve images. Follows, the descriptions of technical system, features, and features optimization.

11.2.1 Our System

Now, we describe our system that is based on Generative Adversarial Networks (GANs) [84]. In the nutshell, GANs are based on a two-player game in which the generator network produces sample, competes against the discriminator network, that has to distinguish between the training data and generator samples [83]. Figure 11.1

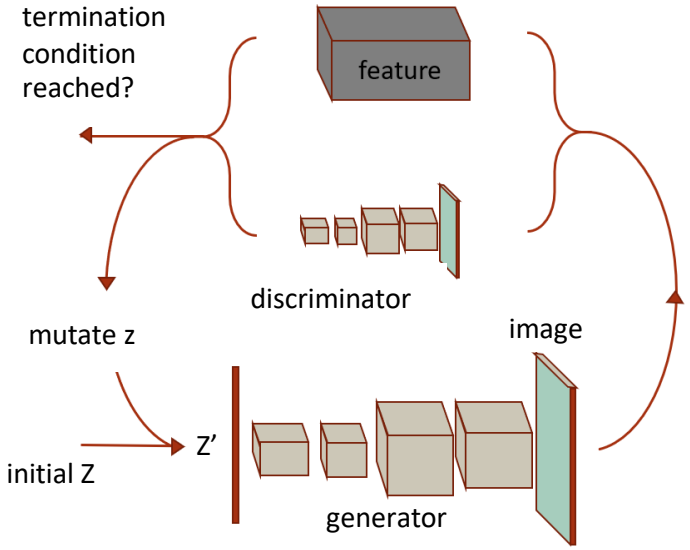


FIGURE 11.1: The final setup of the system. The latent vector Z is randomly seeded and sent through the system, mutating until an optimal solution or the termination condition is reached.

shows the structure of our system. We train two networks: (1) a generator, to generate images from a latent vector Z of 100 real numbers, and (2) a discriminator which scores the images in the generator for realness as both networks are trained. We make two copies of the GAN formed by these networks and train one with two different image datasets: the Celebfaces attributes (*CelebA*) dataset containing more than 150k celebrity faces [158] and the other with ImageNet [34] class of butterflies containing over 45k butterfly images. In this section, we present our experiments as a proof-of-concept.

Our GAN implementation is build on a modified version of the PyTorch GAN code [23]. The generator component of the network consists of five deconvolutional layers. The activation functions for the first four layers are a ReLU activation [185]. The last deconvolutional layer uses Tanh. The discriminator uses LeakyReLU activation in all layers. The generator takes 100 elements of vector Z as input and generates a 128 x 128 pixels image. The discriminator takes an image and generates a normally distributed *realness* score with the most real images scoring close to zero. As a final step, Figure 11.1 shows, we applied the feature function, as described in [3].

The GAN and the necessary feature functions are linked together to drive evolution. The combined system works as follows. A randomly initialised latent feature vector is sent into the generator, which outputs an image. This image is run through both the chosen feature function and the discriminator, and both contribute to a score.

The evolutionary process, guided by the score, mutates Z with the goal of optimising both the realness and the desired feature of the output image.

11.2.2 Features

This section describes in more detail the features used in our experiments. We denote a function f for an image I, representing feature. This function maps an image I to a scalar value $f(I)$. The features are neutral measures of the properties of an image. We intend to use features that taken from the literature [203], and have been empirically derived from surveys studies or from databases of popular images. For our experiments we use the following features: hue, mean-saturation, smoothness, Reflectional Symmetry [99] and Global Contrast Factor [172] (see Chapter 3).

11.2.3 Features Optimisation

In this section, we investigated the use of single and two-dimensional features optimization. We explore this optimisation space with respect to feature values. For a single feature our system optimise particular feature. We define the minimization process *feature* and maximization process $(1.0-\mathit{feature})$[1]. For two-dimensional features (f, g) we have four optimisation targets representing the combinations of minimizing and maximizing f and g.

To maintain image realness, we penalise the combined score with a measure for realness from the GAN *discriminator*. Thus, our fitness functions for single features are shown in equations (11.1) and for two-dimensional features in (11.2).

$$\begin{aligned} \mathit{feature} \times \mathit{discriminator} \\ (1.0 - \mathit{feature}) \times \mathit{discriminator} \end{aligned} \tag{11.1}$$

$$\begin{aligned} \mathit{feature\ 1} \times \mathit{feature\ 2} \times \mathit{discriminator} \\ \mathit{feature\ 1} \times (1.0 - \mathit{feature\ 2}) \times \mathit{discriminator} \\ (1.0 - \mathit{feature\ 1}) \times \mathit{feature\ 2} \times \mathit{discriminator} \\ (1.0 - \mathit{feature\ 1}) \times (1.0 - \mathit{feature\ 2}) \times \mathit{discriminator} \end{aligned} \tag{11.2}$$

In two-dimensional feature experiments we use six feature combinations: hue-saturation, hue-symmetry, saturation-symmetry, smoothness-saturation, GCF-smoothness and GCF-saturation. These combinations were chosen to produce potentially interesting outputs. GCF-smoothness and GCF-saturation were selected due to

[1]Note that for feature GCF maximisation is achieved through $1/\mathrm{GCF}$ and scaling in the range $[0, 1]$.

related work indicating GCF-smoothness would constrain each other [3], resulting in lower image diversity.

11.3 Experimental Investigations

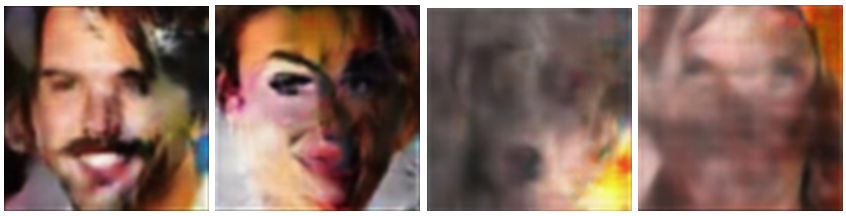
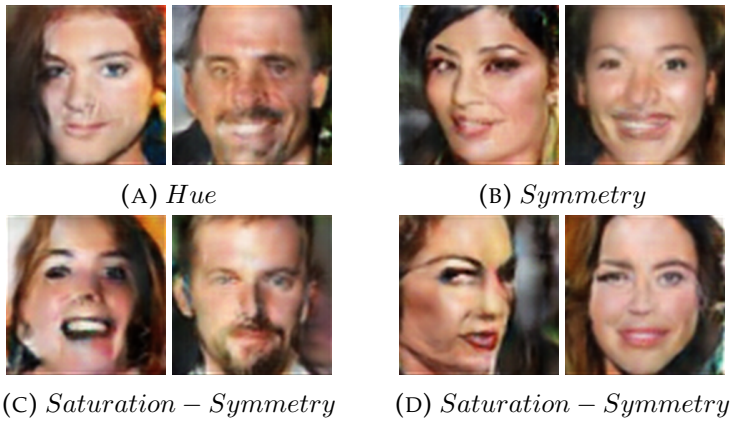


FIGURE 11.2: The first two images obtained by using evolutionary algorithms: *(1+1) EA*. Two following images obtained without realness constraints by minimizing saturation and hue.

We refined the methodology through an experimental process. Initial experimentation used *(1+1) EA* and Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) [95] frameworks to determine the performance of both algorithms. CMA-ES is well-known for evolving vectors of real numbers. Because of this useful property for optimizing non-linear, non convect problem in the continuous space we applied CMA-ES for 2000 mutations (the equivalent of 80000 iterations), and *(1+1) EA* for 80000 iterations. As illustrated in Figure 11.2, CMA-ES was able to achieve more extreme feature value. This superiority of optimisation applied to all feature metrics. We ran all of our experiments on single nodes Intel® Core^{TM} i7-6700 series with 4 core processors. CMA-ES ran for 80 minutes and *(1+1) EA* ran 240 minutes for an optimisation run.

11.3.1 Feature Experiments without Realness Constraint

It was initially assumed that the GAN would be able to create face-like images with some input vector. The tests were performed which did not incorporate a constraint on realness as part of the optimisation process. As Figure 11.2 demonstrates optimizing features without the discriminator produces abstract images. To constrain images to be more realistic three constraining methods were tested: (1) reducing the degrees of freedom in the covariance matrix, (2) discarding images that failed, according to the discriminator, a certain given realness threshold, and (3) incorporating the discriminator's return value into the optimisation function. It was found that the third option of integrating realness as a variable *discriminator* gave the most visually interesting results. The option of discarding images resulted in CMA-ES failing to progress.



(A) Hue (B) $Symmetry$

(C) $Saturation - Symmetry$ (D) $Saturation - Symmetry$

FIGURE 11.3: Images (A) and (B) obtained by single features. The first image corresponds to minimizing features hue (0.0841) and symmetry (0.7985), respectively. The second image corresponds to maximizing features hue (0.1275) and symmetry (0.9198), respectively. Images (C) and (D) are obtained by two-dimensional features. The first image corresponds to minimizing and the second image corresponds to maximizing features saturation and symmetry.

Feature	MIN-F Const	MAX-F Const	Min-Face	Max-Face	Min-Butter	Max-Butter
Hue	0.0841	0.1275	**0.0337**	**0.4886**	**0.1083**	**0.5282**
Saturation	0.3306	0.3543	0.2176	0.4954	0.1205	0.5918
Smoothness	0.9737	0.9843	0.9582	0.9843	0.9462	0.9887
Symmetry	**0.7985**	**0.9198**	0.5904	0.9198	0.5568	0.9428
GCF	0.0276	0.0286	0.0106	0.0348	0.0090	0.0417

TABLE 11.1: Single features value with cut-off of 0.2 for faces and butterfly datasets. Single dimensional feature values obtained from experiments with constraint for butterfly datasets.

11.3.2 Single Dimensional Feature Experiments with Constraint

Single feature experiments require the fewest variables to optimise and as such, could be expected to evolve the image with the least difficulty. The results of each feature values are shown in Table 11.1 (col. 1-2). As can be seen the ranges of features above are very small, with the exception of symmetry. In these runs the *discriminator* term has strong effect on constraining the feature values. For symmetry, the larger range might be explained by presence of both symmetric and asymmetric faces in the training dataset. In line with the small feature ranges, the constrained images only showed small variations as seen in Figure 11.3 (a)-(b) corresponding to the hue and symmetry measures in Table 11.1.

FIGURE 11.4: The results of minimizing hue with cut-off at 0.2 (left), 0.05 (middle) and 0.02 (right).

11.3.3 Two-Dimensional Feature Experiments with Constraint

Running the experiments on two-dimensional features gave similarly constrained images to the single features. Figure 11.3 (c) and (d) shows images evolved to minimise and maximise saturation and symmetry. As can be seen, there is some success in evolving different amounts of symmetry but not a particularly strong difference in saturation. It appears, for all feature combinations, the realness constraint is preventing strong exploration of the feature space.

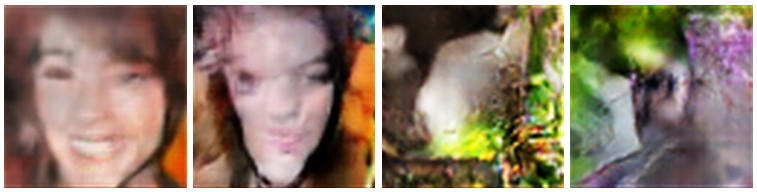
11.3.4 Impact of Cut-off Function

In order to maintain balance between image realness and exploration we adjusted the discriminator term by passing the raw result of *discriminator* $= x$ for an image. The cut-off function f are defined as follows:

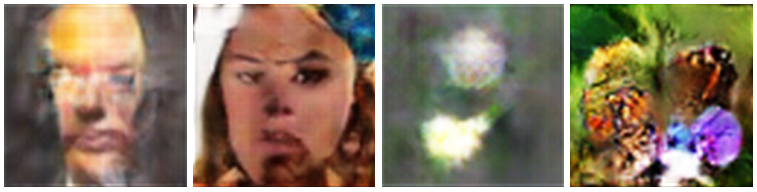
$$f(x) = \begin{cases} x & \text{if } x \geq c \\ s & \text{if } x < c \end{cases} \tag{11.3}$$

with a cutoff c and stable value s. In the experiments that follow we set s to the value close to zero, returning maximum realness, and adapted c to test its effect. With the cut-off function, the search is unaffected until the image reaches a certain threshold of realness. When the threshold is reached the system has no variation in respect to the realness value, thus giving priority to aesthetic features over realness values. A subjective analysis of possible cut-off values needed to be performed in order to determine the optimal value for future experiments. Figure 11.4 demonstrates the effect of different cut-off values on both image realness and aesthetic feature value.

A relationship can be observed from the above images. As the cut-off decreases, so does the degree we are able to evolve the feature value. However, it can be noted that even the 0.02 cut-off was able to create a far lower hue compared to the constrained results – while still being realistic enough to be called a face. The 0.02 cut-off was used in the remaining single feature experiments. This section presents the results of



(A) Hue



(B) $Saturation$

(C) $Smoothness$

(D) $Symmetry$

(E) GCF

FIGURE 11.5: All single feature optimisations with 0.02 cut-off.

evolution in single and two-dimensional image features. The previous experiments were all carried on faces. In the following, GANs generated from both the faces and the butterfly datasets were used.

11.3.5 Single Dimensional Feature Experiments with Cut-off Function

We conducted single feature dimension experiments using faces and butterfly datasets and optimised the following features: hue, saturation, smoothness, symmetry and GCF (see Chapter 3). For these experiments, we use a cut-off of 0.02 on the discriminator output for both GANs. The results were obtained for the minimum and maximum

feature values from each experiment. Figure 11.5 shows the results of the experiments for the single dimensional feature with corresponding to an image minimising (left) and maximising (right) the feature for faces and butterflies respectively.

Table 11.1 (col. $3-6$) shows the minimum and maximum value for each feature for the faces (Min-Face, Max-Face) and butterfly datasets (Min-Butter, Max-Butter), respectively. We observe that hue has the highest range with respect to feature value. The use of the butterfly dataset provides good way to see how evolution with aesthetic measure responds to the priors embedded in GAN. For the single dimensional experiments we observe in Figure 11.5 that images generated with the faces dataset appear more real than the images generated with butterfly dataset. This is likely to be due to the more diverse nature of the butterfly dataset. The images shown in Figure 11.5 (a) have the most variance in the hue dimension. Images with the lowest value for hue appear most realistic. In contrast the image with higher value for hue appears less realistic. We observe that image generated from the butterfly dataset achieves a higher feature range. Figure 11.5 (b) shows that in spite of the saturation feature for faces extending over a narrow range the resulting faces are not very realistic. The butterfly dataset is able to produce higher values of saturation, resulting in a realistic and colorful image. In Figure 11.5 (c) we observe that minimization produces realistic images with superimposed darker shadows. In contrast maximizing smoothness produces less realistic images. The images shown in Figure 11.5 (d) produce high values for symmetry for both datasets. These images appear symmetrical and less real. Images with lower symmetry value are more real. Finally, the images shown in Figure 11.5 (e) exhibit less realness for faces and butterfly dataset in minimization and more realism in maximization.

11.3.6 Two-Dimensional Feature Experiments with Cut-off Function

In our next experiment, we evolve images using the GAN to minimise and maximise in two feature dimensions. These experiments aim to give us insight into how features interact with each other and also the impact of the image priors as embedded in the GAN on the extent to which features can be optimised. After training our GAN models on the faces and butterfly dataset, we run experiments with the following feature combinations: GCF-saturation; GCF-smoothness; hue-saturation; hue-symmetry; saturation-symmetry; and smoothness-saturation.

The feature pair values resulting from these experiments are shown in Tables 11.2 (for faces) and 11.3 (for butterflies). The images corresponding to these values are shown in Figure 11.7 and 11.8. The first column of Figures 11.7 and 11.8, show images,

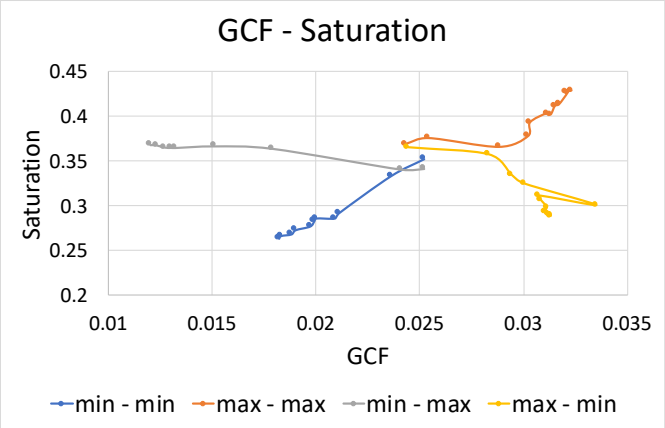


FIGURE 11.6: The trace of GCF and Saturation for the four face images shown in Figure 11.7 (a).

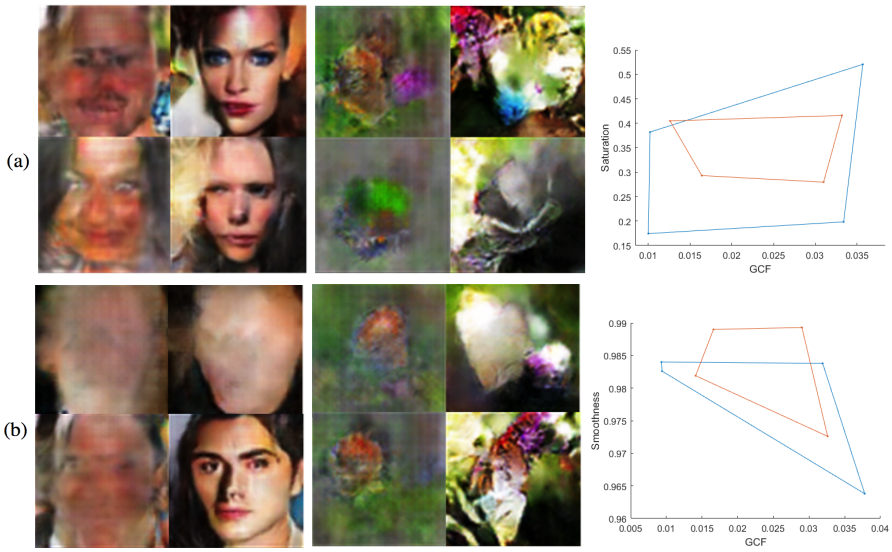


FIGURE 11.7: Images obtained by multi-features with a 0.008 cut-off constraint. Images correspond to 2 feature pairs, GCF, saturation and GCF, smoothness for faces and butterfly dataset, from top left, respectively. Note the images follow their positions on the graph.

in clockwise order from top-left, for min-max, max-max, max-min, min-min combinations of features for faces. The corresponding pictures butterflies dataset is shown in the second column. The last column, plots the positions in feature-space, of the four face images from the first column (in red) and the four butterfly images from the second column (in blue). The shape of the quadrilateral in these plots provides an indication of how feature values are constrained with respect to each other and by the GAN used to generate them. Figure 11.6 shows the trace of GCF and Saturation for faces as the evolutionary run proceeds.

Based on our findings from the previous experiments with single dimension diversity, we reduced the cut-off to 0.008 for the two-dimensional feature experiments

to try to maintain the realism of the images. The impact of this smaller cut-off can be observed in Figures 11.7 and 11.8 in terms of the relatively small areas of feature space contained by the plots. Looking at two-dimensional features in turn. The images shown in Figure 11.7 (a) have the highest GCF values and for max-min and max-max optimisation appear most realistic. We observe in Figure 11.7 (a) that image generated on the butterflies dataset achieve a higher score for GCF and permit a higher range of saturation. The feature plot in Figure 11.7 shows that GCF and Saturation can vary independently. In contrast, Figure 11.7 (b)'s plot indicates some difficulty in minimising both smoothness and GCF. From the plot it also appears to be relatively difficult to simultaneously maximise GCF and smoothness. This result is in concordance with the observations in [3] which found that GCF and smoothness, being spatial features, appeared to be in conflict with each other. More specifically the high contrast required for high GCF scores is in direct conflict with the low contrast required for high smoothness scores. Also notable from Figure 11.7 (b) is a relative lack of realism in the faces as compared to those in Figure 11.7 (a). The images shown in Figure 11.8 (a) illustrate the relationship between hue and saturation. The butterfly pictures show the most variance in the saturation dimension and the face pictures show marginally more variance in hue. Images high in saturation seem to appear sharper – with the face image that maximises both features having quite harsh colour, more contrast, and a mask-like appearance. For Figure 11.8 (b) both sets of images have similar ranges of symmetry but faces have a much narrower range of hue. Highly symmetric images seem to be less realistic, tending to ovoid shapes, with detail seemly sacrificed in order to maximise symmetry. In contrast asymmetric images appear to have more realistic textures and more intense colours.

Figure 11.8 (c) combines saturation and symmetry. As before, highly symmetric images appear less realistic. The evolutionary process seems to have difficulty maximising both saturation and symmetry for both GANs. Clearly it is possible to create artificial images that score highly on both feature dimensions so this difficulty may be reflective of the rarity of this feature combination in the training sets for these GANs. As a final observation, the butterfly picture maximising both features resemble an insect's face, perhaps an interesting consequence of having diverse images in the training set.

Finally, the images in Figure 11.8 (d) show difficulty in minimising both smoothness and saturation. There is a smaller corresponding problem in maximising both features. In both data sets the most realistic images are produced by the minimisation of smoothness and the maximisation of saturation, perhaps indicating that the priors

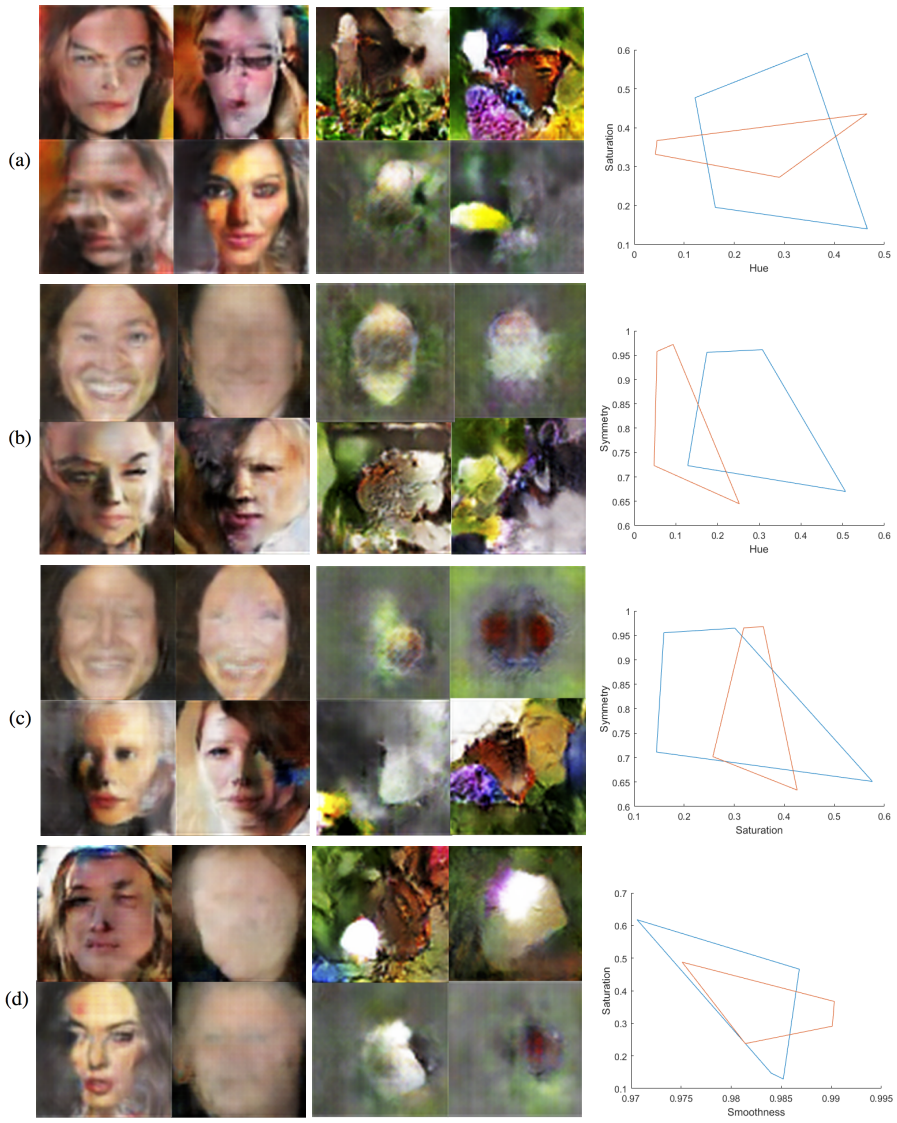


FIGURE 11.8: Images obtained by multi-features with a 0.008 cut-off constraint. The first four images correspond to features hue and saturation, hue and symmetry, saturation and symmetry, smoothness and saturation, from top, respectively. The images follow their positions on the graph.

in the dataset are biased toward rougher and more colourful images.

11.4 Conclusions

In this chapter, we have shown how to apply GAN in order to generate images scoring high or low for given feature values. We used evolutionary search to maximise and minimise single features and pairs of features for two datasets, faces and butterflies.

Feature pairs	Min.f1–Min.f2	Min.f1–Max.f2	Max.f1–Min.f2	Max.f1–Max.f2
$Hue - Saturation$	0.0426 - 0.3318	0.0457 - 0.3671	0.2900 - 0.2727	0.4651 - 0.4257
$Hue - Symmetry$	0.0480 - 0.7233	0.0549 - 0.9577	0.2523 - 0.6448	0.0935 - 0.9722
$Saturation - Symmetry$	0.2573 - 0.7020	0.3190 - 0.9654	0.4256 - 0.6336	0.3582 - 0.9679
$Smoothness - Saturation$	0.9814 - 0.2378	0.9751 - 0.4876	0.9901 - 0.2912	0.9903 - 0.3670
$GCF - Saturation$	0.0164 - 0.2930	0.0126 - 0.4048	0.0310 - 0.2796	0.0332 - 0.4160
$GCF - Smoothness$	0.0166 - 0.9890	0.0166 - 0.9890	0.0326 - 0.9762	0.0290 - 0.9893

TABLE 11.2: Two-dimensional features with cut-off for faces dataset.

Feature pairs	Min.f1–Min.f2	Min.f1–Max.f2	Max.f1–Min.f2	Max.f1–Max.f2
$Hue - Saturation$	0.1622 - 0.1956	0.1218 - 0.4772	0.4659 - 0.1402	0.3458 - 0.5912
$Hue - Symmetry$	0.1286 - 0.7232	0.1744 - 0.9558	0.5067 - 0.6701	0.3075 - 0.9614
$Saturation - Symmetry$	0.1447 - 0.7133	0.1594 - 0.9554	0.5760 - 0.6512	0.3014 - 0.9647
$Smoothness - Saturation$	0.9840 - 0.1469	0.9706 - 0.6177	0.9852 - 0.1291	0.9868 - 0.4661
$GCF - Saturation$	0.0100 - 0.1743	0.0102 - 0.3820	0.0334 - 0.1983	0.0357 - 0.5205
$GCF - Smootness$	0.0094 - 0.9826	0.0093 - 0.9840	0.0378 - 0.9638	0.0319 - 0.9838

TABLE 11.3: Two-dimensional features with cut-off for butterfly dataset

We have shown that GANs known for their successful generation of photorealistic images with combination of evolutionary search can be a powerful technique for creating novel images. We have presented a novel latent variable approach based on nature-inspired methods and have shown how to explore the latent space of a GAN to create semi-realistic images that sample different regions of feature spaces. Additionally, we studied the effects of different values of the Cut-off function on the appearance of the images. Finally, our experimental results on two datasets demonstrated that proposed approach is promising image transformation in regards to maximizing diversity in single- and two-dimensional spaces applications.

For future research, it would be interesting to explore intermediate points in the feature space to gain more insight into the relationships between features and to explore additional constraints and their effect on the process of generating novel images. The potential are of the future work is to use Multi-Objective Optimization Algorithms to evolve the latent vector in terms of choosing multiple-criteria. The work can be extended by exploring different aesthetic features and can be use in fields of industrial design, entertainment (video games) or architecture that can beneficial to the exploration of variant solutions.

A conference version containing the results of this chapter has been published in the Proceedings of the Neural Information Processing - 25rd International Conference (ICONIP) 2018 (see A. Neumann, C. Pyromallis, B. Alexander (2018) [193]).

Chapter 12

Conclusions and Future Work

In this thesis, we investigated how algorithms and machine learning methods assist the processes of evolving images. Evolutionary algorithms have been widely and successful applied to combinatorial optimization problems including the areas of music and art. The significant advantage of evolutionary computation methods is that they do not require exact knowledge about the optimization problem. The use of evolutionary algorithms and machine learning methods for the generation of images has attracted an increasing interest from various research areas. The main aim in the research area of evolutionary algorithms and digital art is to evolve artistic images using evolutionary processes.

We have shown that evolutionary image transition is able to transfer a starting image into a target image by utilizing an evolutionary algorithm. We investigated how random walk algorithms can be used in the evolutionary image transition process. In respect to the utilisation of mutation operators we explored different ways of incorporating uniform and biased random walks that lead to different effects during the transition process. Furthermore, we studied the impact of the different approaches in respect to different artistic features.

We provide new insights into the evolutionary image transition process for evolutionary image painting. The key idea is to make use of the biased random walk and its behaviour of favouring similar colors to paint an image.

Additionally, we investigated quasi-random methods that allow one to create interesting random behaviors by determining a few parameters within a quasi-random algorithm. We presented a new approach to carry out image transition and animations. This concept generalizes the use of multiple agents by performing quasi-random walks for image animation. This approach allows us to create different forms of image transition and animation by determining a sequence for each agent to be used in the quasi-random walk. Furthermore, we have proposed a new approach of image composition based on feature covariance matrices by considering different pairs of images.

This approach facilitates the composition of new images. We have introduced a genetic algorithm evolving a set of images by crossover and mutation operators based on random walk. For our experimental investigation, we have presented the final results with respect to different parameters.

Nevertheless, it is important to obtain a good weighting of interesting regions of the two images when using evolutionary image composition. We have applied colour-based segmentation based on K-Means clustering to come up with this weighting of images. Our results show that this preserves the chosen colour regions of the images and leads to composed images that preserve colours better than our previous approach that was based on saliency masks.

Diversity plays a crucial role in evolutionary computation. Traditionally, diversity is used to avoid premature convergence and it is generally assumed that crossover-based evolutionary algorithms need a diverse population in order to produce good results. We applied a methodology for evolving image variants to maximise diversity in image feature metrics and demonstrated how those image variants change across the spectrum of feature values. We investigated how correlations between feature metrics can be obtained mapping populations evolved for multiple features. The correlations revealed the combinations of features which permit the evolution of diverse images across each feature dimension.

Furthermore, we introduced a discrepancy-based evolutionary diversity optimization approach that constructs sets of solutions that meet a given quality criteria and have a low discrepancy in respect to the considered features. We investigated the use of the star discrepancy measure in evolutionary diversity optimization diversity optimization for images. We introduced a new and effective mutation operator based on random walks. Our experimental results for evolving diverse sets of images showed that using discrepancy-based diversity optimization in conjunction with a tie-breaking rule based on the weighted contribution diversity measure obtained the best results.

We proposed a new approach for evolutionary diversity optimization based on popular indicators in the area of evolutionary multi-objective optimization. In this case, we bridged the areas of evolutionary diversity optimization and evolutionary multi-objective optimization and showed how techniques developed in evolutionary multi-objective optimization can be used to produce diverse sets of solutions of high quality for a given single-objective problem. We investigated how to adapt those indicators to evolutionary diversity optimization and compared them in terms of their ability to lead to diverse sets of solutions.

Another important class of machine learning systems are Generative Adversarial

Networks that are vigorously studied as image-to-image translation by classifying the output image, while simultaneously learning a loss that adapts to the data. We have shown how to employ GANs in order to generate images that score for particular feature values. We used evolutionary algorithm to maximise diversity in single- and two-feature spaces according to aesthetic feature measures. Finally, our experimental results demonstrated that the proposed approach is successful for the generation of semi-realistic images utilising evolutionary computation methods. For future research, it would be interesting to explore intermediate points in the feature space to gain more insight into the relationships between features and to explore additional constraints and their effect on the process of generating images.

In the future, it would be interesting to examine the behavior of other bio-inspierd algorithms on the image transition and animation in greater detail as well as to employ other mutation operators. Second interesting point is to investigate multi-objective optimization algorithms with additional constraints, and to explore their effect on the process of generating novel images. Another approach for future research directions that would be beneficial is to bring together/connect diversity optimization, evolutionary image transition, multi-objective optimization and popular machine methods as generative adversarial networks. The synergy of this areas would have a potential to generate compiling images and therefore profound promote the establishment of novel algorithms and machine learning methods.

References

[1] Esteve Del Acebo and Mateu Sbert. "Benford's law for natural and synthetic images". In: *Computational Aesthetics in Graphics, Visualization and Imaging*. The Eurographics Association, 2005.

[2] Mohammad Majid al-Rifaie and John Mark Bishop. "Swarmic Paintings and Colour Attention". In: *Evolutionary and Biologically Inspired Music, Sound, Art and Design, EvoMUSART 2013*. Vol. 7834. Lecture Notes in Computer Science. Springer, 2013, pp. 97–108.

[3] Bradley Alexander, James Kortman, and Aneta Neumann. "Evolution of artistic image variants through feature based diversity optimisation". In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2017*. 2017, pp. 171–178.

[4] Benjamin Allen, Gabor Lippner, Yu-Ting Chen, Babak Fotouhi, Naghmeh Momeni, Shing-Tung Yau, and Martin A Nowak. "Evolutionary dynamics on any population structure". In: *Nature* 544.7649 (2017), p. 227.

[5] Rui Filipe Antunes, Frederic Fol Leymarie, and William H. Latham. "On Writing and Reading Artistic Computational Ecosystems". In: *Artificial Life* 21.3 (2015), pp. 320–331.

[6] Vincent Arsigny, Pierre Fillard, Xavier Pennec, and Nicholas Ayache. "Log-Euclidean metrics for fast and simple calculus on diffusion tensors". In: *Magnetic Resonance in Medicine* 56.2 (2006), pp. 411–421.

[7] David Arthur and Sergei Vassilvitskii. "K-means++: the advantages of careful seeding". In: *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*. 2007.

[8] Anne Auger. "Benchmarking the (1+1)-ES with one-fifth success rule on the BBOB-2009 noisy testbed". In: *Proceedings of Genetic and Evolutionary Computation*. GECCO 2009, ACM, 2009, pp. 2453–2458.

[9] Anne Auger, Mohamed Jebalia, and Olivier Teytaud. "Algorithms (X, sigma, eta): Quasi-random mutations for evolution strategies". In: *Proceedings of Artificial Evolution EA 2005*. 2005, pp. 296–307.

[10] Shumeet Baluja, Dean Pomerleau, and Todd Jochem. "Towards Automated Artificial Evolution for Computer-generated Images". In: *Connect. Sci.* 6.2-3 (1994), pp. 325–354.

[11] Perry Barile, Vic Ciesielski, and Karen Trist. "Non-photorealistic rendering using genetic programming". In: *Asia-Pacific Conference on Simulated Evolution and Learning*. Springer. 2008, pp. 299–308.

[12] Frank Benford. *The law of anomalous numbers*. Vol. 78. 4. American Philosophical Society, 1938, pp. 551–572.

[13] Rudolf Berghammer, Tobias Friedrich, and Frank Neumann. "Convergence of set-based multi-objective optimization, indicators and deteriorative cycles". In: *Theor. Comput. Sci.* 456 (2012), pp. 2–17.

[14] Leonardo C. T. Bezerra, Manuel López-Ibáñez, and Thomas Stützle. "An empirical assessment of the properties of inverted generational distance on multi- and many-objective optimization". In: *Evolutionary Multi-Criterion Optimization*. Cham: Springer International Publishing, 2017, pp. 31–45.

[15] Christian Blum. "Ant colony optimization: Introduction and recent trends". In: *Physics of Life reviews* 2.4 (2005), pp. 353–373.

[16] Christian Blum and Daniel Merkle. "Swarm intelligence". In: *Swarm Intelligence in Optimization* (2008), pp. 43–85.

[17] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm intelligence - From natural to artificial systems*. Studies in the sciences of complexity. Oxford University Press, 1999.

[18] Jeffrey E. Boyd, Gerald Hushlak, and Christian J. Jacob. "SwarmArt: Interactive Art from Swarm Intelligence". In: *Proceedings of the 12th Annual ACM International Conference on Multimedia*. MULTIMEDIA 2004. ACM, 2004, pp. 628–635.

[19] Karl Bringmann and Tobias Friedrich. "Parameterized average-case complexity of the hypervolume indicator". In: *Proceedings of the Genetic and Evolutionary Computation*. ACM, 2013, pp. 575–582.

[20] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. "Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data?" In: *Perspectives on psychological science* 6.1 (2011), pp. 3–5.

[21] Sung-Hyuk Cha. "Comprehensive survey on distance/similarity measures between probability density functions". In: *City* 1.2 (2007), pp. 1–8.

[22] Nachol Chaiyaratana, Theera Piroonratana, and Nuntapon Sangkawelert. "Effects of diversity control in single-objective and multi-objective genetic algorithms". In: *Journal of Heuristics* 13.1 (2007), pp. 1–34.

[23] Yunjey Choi. "PyTorch tutorial: Deep Convolutional Generative Adversarial Networks DCGAN". In: (2017). `https://github.com/yunjey/pytorch-tutorial`.

[24] Simon Colton, Jakob Halskov, Dan Ventura, Ian Gouldstone, Michael Cook, and Blanca Pérez Ferrer. "The painting fool sees! New projects with the automated painter". In: *Proceedings of the Sixth International Conference on Computational Creativity, ICCC*. 2015, pp. 189–196.

[25] Joshua Cooper, Benjamin Doerr, Joel Spencer, and Gábor Tardos. "Deterministic random walks on the integers". In: *European Journal of Combinatorics* 28.8 (2007), pp. 2072–2090.

[26] Joshua N. Cooper and Joel Spencer. "Simulating a random walk with constant error". In: *Combinatorics, Probability and Computing* 15.06 (2006), pp. 815–822.

[27] Gregory W. Corder and Dale I. Foreman. *Nonparametric statistics for non-statisticians: A step-by-step approach*. Wiley, 2009.

[28] Joao Correia, Penousal Machado, Juan Romero, and Adrian Carballal. "Evolving figurative images using expression-based evolutionary art". In: *Proceedings of the International Conference on Computational Creativity, ICCC*. 2013, pp. 24–31.

[29] Charles Darwin. *On the Origin of Species by Means of Natural Selection*. or the Preservation of Favored Races in the Struggle for Life. London: Murray, 1859.

[30] R. Dawkins. *The Blind Watchmaker: Why the evidence of evolution reveals a universe without design*. National bestseller. Science. Norton, 1986.

[31] Kalyanmoy Deb. *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: John Wiley & Sons, 2001.

[32] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. "A fast and elitist multiobjective genetic algorithm: NSGA-II". In: *IEEE Trans. Evolutionary Computation* 6.2 (2002), pp. 182–197.

[33] Amir Dembo, Yuval Peres, Jay Rosen, and Ofer Zeitouni. "Cover Times for Brownian Motion and Random Walks in Two Dimensions". In: *Annals of Mathematics* 160.2 (2004), pp. 433–464.

[34] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database". In: *CVPR*. IEEE Computer Society, 2009, pp. 248–255.

[35] Emily L. Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. "Deep generative image models using a Laplacian pyramid of adversarial networks". In: *Advances in Neural Information Processing Systems 28*. Curran Associates, Inc., 2015, pp. 1486–1494.

[36] Josef Dick and Friedrich Pillichshammer. *Digital nets and sequences: discrepancy theory and quasi–Monte Carlo integration*. Cambridge University Press, 2010.

[37] Steve R. DiPaola and Liane Gabora. "Incorporating characteristics of human creativity into an evolutionary art algorithm". In: *Genetic Programming and Evolvable Machines* 10.2 (2009), pp. 97–110.

[38] David P. Dobkin, David Eppstein, and Don P. Mitchell. "Computing the discrepancy with applications to supersampling patterns". In: *ACM Trans. Graph.* 15 (1996), pp. 354–376.

[39] Benjamin Doerr and Carola Doerr. "Optimal Parameter Choices Through Self-Adjustment: Applying the 1/5-th Rule in Discrete Settings". In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015*, pp. 1335–1342.

[40] Benjamin Doerr, Mahmoud Fouz, and Tobias Friedrich. "Experimental analysis of rumor spreading in social networks". In: *Design and Analysis of algorithms: First Mediterranean Conference on Algorithms, MedAlg 2012, Proceedings*. Springer Berlin Heidelberg, 2012, pp. 159–173.

[41] Benjamin Doerr and Tobias Friedrich. "Deterministic Random Walks on the Two-Dimensional Grid". In: *Combinatorics, Probability & Computing* 18.1-2 (2009), pp. 123–144.

[42] Benjamin Doerr, Tobias Friedrich, and Thomas Sauerwald. "Quasirandom rumor spreading". In: *ACM Trans. Algorithms* 11.2 (2014), 9:1–9:35.

[43] Benjamin Doerr, Tobias Friedrich, and Thomas Sauerwald. "Quasirandom rumor spreading: expanders, push vs. pull, and robustness". In: *Automata, Languages and Programming: 36th International Colloquium, ICALP 2009, Proceedings, Part I*. Springer Berlin Heidelberg, 2009, pp. 366–377.

[44] Carola Doerr, Michael Gnewuch, and Magnus Wahlström. "Calculation of Discrepancy Measures and Applications". In: *A Panorama of Discrepancy Theory*. Vol. 2107. LNCS. 2014, pp. 621–678.

[45] Carola Doerr and François-Michel De Rainville. "Constructing low star discrepancy point sets with genetic algorithms". In: *Proc. of Genetic and Evolutionary Computation Conference, GECCO 2013*. ACM, 2013, pp. 789–796.

[46] Marco Dorigo and Mauro Birattari. *Ant colony optimization*. Springer, 2010.

[47] Marco Dorigo and Christian Blum. "Ant colony optimization theory: A survey". In: *Theoretical computer science* 344.2-3 (2005), pp. 243–278.

[48] Marco Dorigo and Gianni Di Caro. "Ant colony optimization: a new meta-heuristic". In: *Proceedings of the Congress on Evolutionary Eomputation, CEC*. Vol. 2. IEEE. 1999, pp. 1470–1477.

[49] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. "The ant system: An autocatalytic optimizing process". In: Technical Report 91-016 Revised, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1991.

[50] Marco Dorigo and Thomas Stützle. *Ant colony optimization: Overview and recent advances*. Cham: Springer International Publishing, 2019, pp. 311–351.

[51] Marco Dorigo and Thomas Stützle. *The ant colony optimization metaheuristic: Algorithms, applications, and advances*. Springer, 2003, pp. 250–285.

[52] Alexey Dosovitskiy and Thomas Brox. "Generating images with perceptual similarity metrics based on deep networks". In: *Proceedings of the Neural Information Processing Systems, NIPS 2016*. 2016, pp. 658–666.

[53] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. "Learning to generate chairs with convolutional neural networks". In: *CVPR*. IEEE Computer Society, 2015, pp. 1538–1546.

[54] Scott Draves. "The Electric Sheep screen-saver: A case study in aesthetic evolution". In: *Workshops on Applications of Evolutionary Computation*. Springer. 2005, pp. 458–467.

[55] Michael Drmota and Robert F. Tichy. *Sequences, discrepancies and applications*. Springer, 2006.

[56] Magdalena Droste. *Bauhaus, 1919-1933*. Taschen, 2002.

[57] Rajdeep Dua and Manpreet Singh Ghotra. *Keras deep learning cookbook*. Packt Publishing, 2018.

[58] Richard O. Duda and Peter E. Hart. *Pattern classification and scene analysis*. A Wiley-Interscience publication. Wiley, 1973.

[59] Ioana Dumitriu, Prasad Tetali, and Peter Winkler. "On Playing Golf with Two Balls". In: *SIAM J. Discrete Math.* 16.4 (2003), pp. 604–615.

[60] Ágoston E. Eiben and James E. Smith. *Introduction to evolutionary computing*. (2. ed.) Natural Computing Series. Springer, Berlin, Heidelberg, 2015.

[61] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. "From data mining to knowledge discovery: An overview". In: *Advances in Knowledge Discovery and Data Mining*. 1996, pp. 1–34.

[62] Michael Fenton, James McDermott, David Fagan, Stefan Forstenlechner, Erik Hemberg, and Michael O'Neill. "PonyGE2: Grammatical Evolution in Python". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO 2017*. ACM, 2017, pp. 1194–1201.

[63] David B. Fogel. *Evolutionary computation - toward a new philosophy of machine intelligence (3. ed.)* Wiley-VCH, 2006.

[64] Gary B. Fogel, David B. Fogel, and Lawrence J. Fogel. "Evolutionary programming". In: *Scholarpedia* 6.4 (2011), p. 1818.

[65] Lawrence J. Fogel. "Autonomous automata". In: *Industrial Research* 4 (1962), pp. 14–19.

[66] Lawrence J. Fogel, Alvin J. Owens, and Michael J. Walsh. *Artificial intelligence through simulated evolution*. John Wiley, 1966.

[67] David A Forsyth and Jean Ponce. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002.

[68] Tobias Friedrich, Christian Horoba, and Frank Neumann. "Multiplicative approximations and the hypervolume indicator". In: *Proceedings of Genetic and Evolutionary Computation Conference, GECCO 2009*. ACM, 2009, pp. 571–578.

[69] Tobias Friedrich, Maximilian Katzmann, and Anton Krohmer. "Unbounded Discrepancy of Deterministic Random Walks on Grids". In: *Proceedings of International Symposium on Algorithms and Computation, ISAAC*. 2015, pp. 212–222.

[70] Tobias Friedrich and Lionel Levine. "Fast simulation of large-scale growth models". In: *Random Structures and Algorithms* 42.2 (2013), pp. 185–213.

[71] Tobias Friedrich and Thomas Sauerwald. "The Cover Time of Deterministic Random Walks". In: *Electr. J. Comb.* 17.1 (2010).

[72] John Gage. *Colour in Art. World of Art*. Thames and Hudson, 2007.

[73] Wanru Gao, Samadhi Nallaperuma, and Frank Neumann. "Feature-Based Diversity Optimization for Problem Instance Classification". In: *CoRR* abs/1510.08568 (2015).

[74] Wanru Gao, Samadhi Nallaperuma, and Frank Neumann. "Feature-Based Diversity Optimization for Problem Instance Classification". In: *Parallel Problem Solving from Nature - PPSN XIV 2016, Proceedings*. Vol. 9921. Lecture Notes in Computer Science. Springer, 2016, pp. 869–879.

[75] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. "Image style transfer using convolutional neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*. IEEE Computer Society, 2016, pp. 2414–2423.

[76] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Texture Synthesis Using Convolutional Neural Networks". In: *Proceedings of the Neural Information Processing Systems, NIPS*. 2015, pp. 262–270.

[77] Leon A Gatys, Alexander S Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. "Controlling perceptual factors in neural style transfer". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*. IEEE Computer Society, 2017, pp. 3730–3738.

[78] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. "Controlling Perceptual Factors in Neural Style Transfer". In: *CVPR*. IEEE Computer Society, 2017, pp. 3730–3738.

[79] Leon A Gatys, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. "Preserving color in neural artistic style transfer". In: *arXiv:1606.05897* (2016).

[80] Michael Gnewuch, Anand Srivastav, and Carola Winzen. "Finding optimal volume subintervals with k points and calculating the star discrepancy are NP-hard problems". In: *J. Complexity* 25 (2009), pp. 115–127.

[81] Michael Gnewuch, Magnus Wahlström, and Carola Winzen. "A new randomized algorithm to approximate the star discrepancy based on threshold accepting". In: *SIAM J. Numer. Anal.* 50 (2012), pp. 781–807.

[82] David E. Goldberg and Kalyanmoy Deb. "A Comparative Analysis of Selection Schemes Used in Genetic Algorithms". In: *Proceedings of the First Workshop on Foundations of Genetic Algorithms, 1990*. Morgan Kaufmann, 1990, pp. 69–93.

[83] Ian Goodfellow, Yoshua Bengio, and Aaron C. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016.

[84] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets". In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.

[85] Leo Grady. "Random Walks for Image Segmentation". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 28.11 (2006), pp. 1768–1783.

[86] Jeanine Graf and Wolfgang Banzhaf. "Interactive Evolution of Images". In: *Evolutionary Programming*. 1995, pp. 53–65.

[87] Gary Greenfield. "Avoidance drawings evolved using virtual drawing robots". In: *Proceedings of Evolutionary and Biologically Inspired Music, Sound, Art and Design - 4th International Conference, EvoMUSART 2015*. 2015, pp. 78–88.

[88] Gary Greenfield. "Robot paintings evolved using simulated robots". In: *Workshops on Applications of Evolutionary Computation*. Springer. 2006, pp. 611–621.

[89] Gary Greenfield and Penousal Machado. "Ant- and Ant-Colony-Inspired ALife Visual Art". In: *Artificial Life* 21.3 (2015), pp. 293–306.

[90] Gary R. Greenfield. "Evolutionary Methods for Ant Colony Paintings". In: *Applications of Evolutionary Computing, EvoWorkshops 2005, Proceedings*. Vol. 3449. Lecture Notes in Computer Science. Springer, 2005, pp. 478–487.

[91] Gary R. Greenfield. "On the Co-Evolution of Evolving Expressions". In: *International Journal of Computational Intelligence and Applications* 2.1 (2002), pp. 17–31.

[92] Kai Guo, Prakash Ishwar, and Janusz Konrad. "Action recognition using sparse representation on covariance manifolds of optical flow". In: *Proceedings of IEEE Conf. Advanced Video and Signal Based Surveillance*. AVSS 2010. IEEE, 2010, pp. 188–195.

[93] John H. Halton. "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals". In: *Numerische mathematik* 2.1 (1960), pp. 84–90.

[94] Greg Hamerly and Charles Elkan. "Learning the k in K-means". In: *Proceedings of the 17th International Conference on Neural Information Processing Systems, NIPS 2003*. MIT Press, 2003, pp. 281–288.

[95] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)". In: *Evolutionary computation* 11.1 (2003), pp. 1–18.

[96] David A. Hart. "Toward Greater Artistic Control for Interactive Evolution of Images and Animation". In: *Applications of Evolutionary Computing, EvoWorkshops 2007*. Vol. 4448. Lecture Notes in Computer Science. Springer, 2007, pp. 527–536.

[97] David Hasler and Sabine E Suesstrunk. "Measuring colorfulness in natural images". In: *Electronic Imaging 2003*. International Society for Optics and Photonics. 2003, pp. 87–95.

[98] Eelco den Heijer and Ágoston E. Eiben. "Using aesthetic measures to evolve art". In: *Proceeding IEEE Congress Evolutionary Computation*. CEC 2010. IEEE, 2010, pp. 1–8.

[99] Eelco den Heijer and A. E. Eiben. "Investigating aesthetic measures for unsupervised evolutionary art". In: *Swarm and Evolutionary Computation* 16 (2014), pp. 52–68.

[100] Erma Hermens. "Technical art history: The synergy of art, conservation and science". In: *Art History and Visual Studies in Europe*. Leiden, The Netherlands: Brill, 2012, pp. 151 –165.

[101] Aaron Hertzmann. "Painterly Rendering with Curved Brush Strokes of Multiple Sizes". In: *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1998*. ACM, 1998, pp. 453–460.

[102] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. "GANs trained by a two time-scale update rule converge to a Local Nash Equilibrium". In: *Proceedings of the Neural Information Processing Systems, NIPS*. 2017, pp. 6629–6640.

[103] Nick. J. Higham. *Functions of matrices: theory and computation*. Society for Industrial and Applied Mathematics (SIAM), 2008.

[104] Philip F Hingston, Luigi C Barone, and Zbigniew Michalewicz. *Design by evolution: advances in evolutionary design*. Springer Science & Business Media, 2008.

[105] John H. Holland. *Adaptation in natural and artificial aystems: An introductory analysis with applications to biology, control and artificial intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[106] Alexander E. Holroyd and James G. Propp. "Rotor walks and Markov chains". In: *Algorithmic Probability and Combinatorics*. Contemporary Mathematics 520 (2010), pp. 105–126.

[107] Xiaodi Hou, Jonathan Harel, and Christof Koch. "Image signature: highlighting sparse salient regions". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 34.1 (2012), pp. 194–201.

[108] John F Hughes, Andries Van Dam, James D Foley, Morgan McGuire, Steven K Feiner, David F Sklar, and Kurt Akeley. *Computer graphics: principles and practice*. Pearson Education, 2014.

[109] Richard S Hunter. "Photoelectric color-difference meter". In: *J. Opt. Soc. Am.* 38.7 (1948), pp. 651–651.

[110] The MathWorks Inc. `https://au.mathworks.com/help/images`.

[111] Ruth E. Iskin. *Re-envisioning the Contemporary Art Canon: Perspectives in a Global World*. London and New York: Routledge, 2017, p. 294.

[112] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[113] Ashkan Izadi, Victor Ciesielski, and Marsha Berry. "Evolutionary Non Photo-Realistic Animations with Triangular Brushstrokes". In: *Advances in Artificial Intelligence - 23rd Australasian Joint Conference, AI 2010*. Vol. 6464. Lecture Notes in Computer Science. Springer, 2010, pp. 283–292.

[114] János Izsák and László Papp. "A link between ecological diversity indices and measures of biodiversity". In: *Ecological Modelling* 130.1-3 (2000), 151—156.

[115] Thomas Jansen and Dirk Sudholt. "Analysis of an Asymmetric Mutation Operator". In: *Evolutionary Computation* 18.1 (2010), pp. 1–26.

[116] Sadeep Jayasumana, Richard I. Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Tafazzoli Harandi. "Kernel methods on Riemannian manifolds with Gaussian RBF kernels". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 37.12 (2015), pp. 2464–2477.

[117] Sadeep Jayasumana, Richard I. Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Tafazzoli Harandi. "Kernel methods on the Riemannian manifold of symmetric positive definite matrices". In: *Proceedings IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2013*. IEEE Press, 2013, pp. 73–80.

[118] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution". In: *European Conference on Computer Vision, ECCV*. Springer. 2016, pp. 694–711.

[119] Jean-Michel Jolion. "Images and Benford's law". In: *Journal of Mathematical Imaging and Vision* 14.1 (2001), pp. 73–81.

[120] W. Kandinsky and H. Rebay. *Point and Line to Plane*. Dover Fine Art, History of Art. Dover Publications, 1926.

[121] Hyung W Kang, Uday K Chakraborty, Charles K Chui, and Wenjie He. "Multi-scale stroke-based rendering by evolutionary algorithm". In: *Proceedings of the International Workshop on Frontiers of Evolutionary Algorithms, JCIS 2005*. 2005, pp. 546–549.

[122] Omid Kardan, Emre Demiralp, Michael C. Hout, MaryCarol R. Hunter, Hossein Karimi, Taylor Hanayik, Grigori Yourganov, John Jonides, and Marc G. Berman. "Is the preference of natural versus man-made scenes driven by bottom–up processing of the visual features of nature?" In: *Frontiers in Psychology* 6 (2015), p. 471.

[123] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. "Progressive growing of GANs for improved quality, stability, and variation". In: *Proceedings of the International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[124] Stephen Karungaru, Minoru Fukumi, Norio Akamatsu, and A Takuya. "Automatic human faces morphing using genetic algorithms based control points selection". In: *International Journal of Innovative Computing, Information and Control* 3.2 (2007), pp. 1–6.

[125] James Kennedy. "Swarm intelligence". In: *Handbook of nature-inspired and innovative computing*. Springer, 2006, pp. 187–219.

[126] James Kennedy and Russel C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann, 2001.

[127] James Kennedy and Russell C. Eberhart. "Particle swarm optimization". In: *Proceedings of the IEEE International Conference on Neural Networks*. 1995, pp. 1942–1948.

[128] Stefan Kern, Sibylle D. Müller, Nikolaus Hansen, Dirk Büche, Jiri Ocenasek, and Petros Koumoutsakos. "Learning probability distributions in continuous evolutionary algorithms - a comparative review". In: *Natural Computing* 3 (2004), pp. 77–112.

[129] Adnan Mujahid Khan, Korsuk Sirinukunwattana, and Nasir M. Rajpoot. "Geodesic geometric mean of regional covariance descriptors as an image-level descriptor for nuclear atypia grading in breast histology images". In: *Machine Learning in Medical Imaging - 5th International Workshop, MLMI 2014, Held in Conjunction with MICCAI 2014*. Springer, Cham, 2014, pp. 101–108.

[130] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. "Learning to discover cross-domain relations with generative adversarial networks". In: Proceedings of Machine Learning Research (2017), pp. 1857–1865.

[131] Shuhei Kimura and Koki Matsumura. "Genetic Algorithms Using Low-discrepancy Sequences". In: *Proc. of Genetic and Evolutionary Computation Conference, GECCO 2005*. ACM, 2005, pp. 1341–1346.

[132] Shuhei Kimura and Koki Matsumura. "Improvement of the performances of genetic algorithms by using low-discrepancy sequences". In: *Transactions of the Society of Instrument and Control Engineers* 42 (2006), pp. 659–667.

[133] Michael Kleber. "Goldbug variations". In: *The Mathematical Intelligencer* 27.1 (2005), pp. 55–63.

[134] Paul Klee and Jürgen Glaesemer. "Beiträge zur bildnerischen Formlehre". In: v. 2. Schwabe, 1979, p. 3900.

[135] John E. Kobza, Sheldon H. Jacobson, and Diane E. Vaughan. "A Survey of the Coupon Collector's Problem with Random Sample Sizes". In: *Methodology and Computing in Applied Probability* 9.4 (2007), pp. 573–584.

[136] Alexander Kolesnikov and Christoph H. Lampert. "PixelCNN models with auxiliary variables for natural image modeling". In: *Proceedings of the 34th International Conference on Machine Learning, ICML*. 2017, pp. 1905–1914.

[137] Taras Kowaliw, Alan Dorin, and Jon McCormack. "An empirical exploration of a definition of creative novelty for generative art". In: *Australian Conference on Artificial Life*. Springer. 2009, pp. 1–10.

[138] Taras Kowaliw, Alan Dorin, and Jon McCormack. "Promoting Creative Design in Interactive Evolutionary Computation". In: *IEEE Trans. Evolutionary Computation* 16.4 (2012), pp. 523–536.

[139] John R. Koza. *Genetic programming - on the programming of computers by means of natural selection*. Complex adaptive systems. MIT Press, 1993.

[140] John R. Koza. *Genetic programming 2 - automatic discovery of reusable programs*. Complex adaptive systems. MIT Press, 1994.

[141] John R. Koza. *Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems*. Tech. rep. 1990.

[142] John R. Koza. "Human-competitive results produced by genetic programming". In: *Genetic Programming and Evolvable Machines* 11.3-4 (2010), pp. 251–284.

[143] John R Koza, David Andre, Martin A Keane, and Forrest H Bennett III. *Genetic programming III: Darwinian invention and problem solving*. Vol. 3. Morgan Kaufmann, 1999.

[144] John R Koza, Martin A Keane, Matthew J Streeter, William Mydlowec, Jessen Yu, and Guido Lanza. *Genetic programming IV: Routine human-competitive machine intelligence*. Vol. 5. Springer Science & Business Media, 2006.

[145] Paul Kubelka. "New Contributions to the Optics of Intensely Light-Scattering Materials. Part I". In: *J. Opt. Soc. Am.* 38.5 (1948), pp. 448–457.

[146] Trevor Lamb and Janine Bourriau. *Colour: Art and Science*. Cambridge University Press, 1995.

[147] Nicholas Lambert, William H. Latham, and Frederic Fol Leymarie. "The emergence and growth of evolutionary art: 1980-1993". In: *Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2013*. ACM, 2013, pp. 367–375.

[148] Serge Lang. *Fundamentals of differential geometry*. New York: Springer, 1999.

[149] William Latham. "Black Form Synth". In: (1985). `http://collections.vam.ac.uk`.

[150] William Latham. "Form Synth: The Rule-based Evolution of Complex Forms from Geometric Primitives". In: *Computers in Art, Design, and Animation, John Lansdown and Rae Earnshaw* (1989).

[151] Christian Ledig et al. "Photo-realistic single image super-resolution using a generative adversarial network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, pages=4681–4690, year=2017*.

[152] Yee Leung, Yong Gao, and Zongben Xu. "Degree of population diversity - a perspective on premature convergence in genetic algorithms and its Markov chain analysis". In: *IEEE Trans. Neural Networks* 8.5 (1997), pp. 1165–1176.

[153] Lionel Levine. "The rotor-router model". In: (2002). http://arxiv.org/abs/math/0409407.

[154] Matthew Lewis. "Evolutionary visual art and design". In: *The art of artificial evolution*. Springer, 2008, pp. 3–37.

[155] Ke Li, Kalyanmoy Deb, Qingfu Zhang, and Sam Kwong. "An Evolutionary Many-Objective Optimization Algorithm Based on Dominance and Decomposition". In: *IEEE Trans. Evolutionary Computation* 19.5 (2015), pp. 694–716.

[156] Peter Litwinowicz. "Processing images and video for an impressionist effect". In: *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH*. ACM, 1997, pp. 407–414.

[157] Ming-Yu Liu and Oncel Tuzel. "Coupled Generative Adversarial Networks". In: *Proceedings of the Neural Information Processing Systems, NIPS* (2016), pp. 469–477.

[158] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Deep learning face attributes in the wild". In: *Proceedings of International Conference on Computer Vision, ICCV 2015*. IEEE Computer Society, 2015, pp. 3730–3738.

[159] Stuart P. Lloyd. "Least squares quantization in pcm". In: *IEEE Transactions on Information Theory* 28 (1982), pp. 129–137.

[160] L. Lovász. "Random Walks on Graphs: A Survey". In: *Combinatorics, Paul Erdős is Eighty*. Ed. by D. Miklós, V. T. Sós, and T. Szőnyi. Vol. 2. Budapest: János Bolyai Mathematical Society, 1996, pp. 353–398.

[161] John Lyons. *Language and Linguistics: An Introduction*. Cambridge: Cambridge University Press, 1981.

[162] Jiayi Ma, Wei Yu, Pengwei Liang, Chang Li, and Junjun Jiang. "FusionGAN: A generative adversarial network for infrared and visible image fusion". In: *Information Fusion* 48 (2019), pp. 11–26.

[163] Catarina Maçãs, Pedro Cruz, Pedro Martins, and Penousal Machado. "Swarm Systems in the Visualization of Consumption Patterns". In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*. 2015, pp. 2466–2472.

[164] Penousal Machado and Amílcar Cardoso. "All the Truth About NEvAr". In: *Appl. Intell.* 16.2 (2002), pp. 101–118.

[165] Penousal Machado and João Correia. "Semantic aware methods for evolutionary art". In: *Genetic and Evolutionary Computation Conference, GECCO 2014*. ACM, 2014, pp. 301–308.

[166] Penousal Machado and Luís Pereira. "Photogrowth: non-photorealistic renderings through ant paintings". In: *Genetic and Evolutionary Computation Conference, GECCO 2012*. ACM, 2012, pp. 233–240.

[167] Penousal Machado, Juan Romero, and Bill Manaris. "Experiments in computational aesthetics". In: *The art of artificial evolution*. Springer, 2008, pp. 381–415.

[168] Penousal Machado, Juan Romero, María Luisa Santos, Amílcar Cardoso, and Bill Manaris. "Adaptive critics for evolutionary artists". In: *Workshops on Applications of Evolutionary Computation*. Springer. 2004, pp. 437–446.

[169] Penousal Machado, Adriano Vinhas, João Correia, and Anikó Ekárt. "Evolving ambiguous images". In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*. 2015, pp. 2473–2479.

[170] Samir W Mahfoud. "Crowding and preselection revisited". In: *Urbana* 51 (1992), p. 61801.

[171] Samir W Mahfoud. "Niching methods for genetic algorithms". In: *Urbana* 51.95001 (1995), pp. 62–94.

[172] Kresimir Matkovic, László Neumann, Attila Neumann, Thomas Psik, and Werner Purgathofer. "Global Contrast Factor - a New Approach to Image Contrast." In: *Computational Aesthetics* 2005 (2005), pp. 159–168.

[173] Jiri Matoušek. *Geometric discrepancy, volume 18 of Algorithms and Combinatorics*. 2nd. Springer-Verlag, Berlin, 2010.

[174] John McCharty. *The LISP programmer's manual*. 1960.

[175] Jon McCormack and Mark d'Inverno, eds. *Computers and Creativity*. Computers and Creativity. Berlin; Heidelberg: Springer, 2012.

[176] Olaf Mersmann, Mike Preuss, and Heike Trautmann. "Benchmarking Evolutionary Algorithms: Towards Exploratory Landscape Analysis". In: *Parallel Problem Solving from Nature - PPSN 2010, Proceedings, Part I*. Vol. 6238. Lecture Notes in Computer Science. Springer, 2010, pp. 73–82.

[177] Olaf Mersmann, Bernd Bischl, Heike Trautmann, Markus Wagner, Jakob Bossek, and Frank Neumann. "A novel feature-based approach to characterize algorithm performance for the traveling salesperson problem". In: *Ann. Math. Artif. Intell.* 69.2 (2013), pp. 151–182.

[178] Zbigniew Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs (3rd Ed.)* Berlin, Heidelberg: Springer-Verlag, 1996.

[179] Brad L. Miller and Michael J. Shaw. "Genetic Algorithms with Dynamic Niche Sharing for Multimodal Function Optimization". In: *Proceedings of 1996 IEEE International Conference on Evolutionary Computation, 1996*. IEEE, 1996, pp. 786–791.

[180] Michael Mitzenmacher and Eli Upfal. *Probability and computing : randomized algorithms and probabilistic analysis*. New York: Cambridge University Press, 2005.

[181] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. "Spectral normalization for generative adversarial networks". In: *Proceedings of the International Conference on Learning Representations, ICLR* (2018).

[182] Nicolas Monmarché, Mohamed Slimane, and Gilles Venturini. "On Improving Clustering in Numerical Databases with Artificial Ants". In: *Advances in Artificial Life, 5th European Conference, ECAL 1999, Proceedings*. Vol. 1674. Lecture Notes in Computer Science. Springer, 1999, pp. 626–635.

[183] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. "Inceptionism: Going deeper into neural networks". In: *Google Research Blog.* 20 (2015).

[184] Mario A. Muñoz, Laura Villanova, Davaatseren Baatar, and Kate Smith-Miles. "Instance spaces for machine learning classification". In: *Machine Learning* 107.1 (2018), pp. 109–147.

[185] Vinod Nair and Geoffrey E Hinton. "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the International Conference on Machine Learning, ICML*. 2010, pp. 807–814.

[186] Samadhi Nallaperuma, Markus Wagner, and Frank Neumann. "Analyzing the Effects of Instance Features and Algorithm Parameters for Max-Min Ant System and the Traveling Salesperson Problem". In: *Front. Robotics and AI* 2015 (2015).

[187] Samadhi Nallaperuma, Markus Wagner, Frank Neumann, Bernd Bischl, Olaf Mersmann, and Heike Trautmann. "A feature-based comparison of local search and the christofides algorithm for the travelling salesperson problem". In: *Foundations of Genetic Algorithms XII, FOGA 2013*. ACM, 2013, pp. 147–160.

[188] Aneta Neumann, Bradley Alexander, and Frank Neumann. "Evolutionary image transition using random walks". In: *Proceedings of Computational Intelligence in Music, Sound, Art and Design - 6th International Conference, EvoMUSART 2017*. Vol. 10198. Lecture Notes in Computer Science. 2017, pp. 230–245.

[189] Aneta Neumann, Bradley Alexander, and Frank Neumann. "The Evolutionary Process of Image Transition in Conjunction with Box and Strip Mutation". In: *Neural Information Processing , ICONIP 2016, Proceedings, Part III*. 2016, pp. 261–268.

[190] Aneta Neumann and Frank Neumann. "Evolutionary computation for digital art". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO 2018*. ACM, 2018, pp. 937–955.

[191] Aneta Neumann and Frank Neumann. "On the Use of Colour-Based Segmentation in Evolutionary Image Composition". In: *2018 IEEE Congress on Evolutionary Computation, CEC 2018*. IEEE, 2018, pp. 1–8.

[192] Aneta Neumann, Frank Neumann, and Tobias Friedrich. "Quasi-random Agents for Image Transition and Animation". In: *CoRR* abs/1710.07421 (2017).

[193] Aneta Neumann, Christo Pyromallis, and Bradley Alexander. "Evolution of images with diversity and constraints using a generator network". In: *Proceedings of Neural Information Processing - 25rd International Conference, ICONIP 2018*. Vol. 9949. Lecture Notes in Computer Science. 2018, pp. 261–268.

[194] Aneta Neumann, Wanru Gao, Carola Doerr, Frank Neumann, and Markus Wagner. "Discrepancy-based evolutionary diversity optimization". In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2018*. ACM, 2018, pp. 991–998.

[195] Aneta Neumann, Wanru Gao, Markus Wagner, and Frank Neumann. "Evolutionary diversity optimization using multi-objective indicators". In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2019*. ACM, 2019, pp. 837–845.

[196] Aneta Neumann, Zygmunt L. Szpak, Wojciech Chojnacki, and Frank Neumann. "Evolutionary image composition using feature covariance matrices". In: *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2017*. ACM, 2017, pp. 817–824.

[197] Frank Neumann and Shayan Poursoltan. "Feature-based algorithm selection for constrained continuous optimisation". In: *IEEE Congress on Evolutionary Computation, CEC 2016*. IEEE, 2016, pp. 1461–1468.

[198] Frank Neumann and Ingo Wegener. "Randomized local search, evolutionary algorithms, and the minimum spanning tree problem". In: *Theor. Comput. Sci.* 378.1 (2007), pp. 32–40.

[199] John von Neumann. *Theory of self-reproducing automata*. University of Illinois Press, 1966.

[200] Anh Nguyen, Jason Yosinski, and Jeff Clune. "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*. 2015, pp. 427–436.

[201] Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks". In: *Proceedings of the Neural Information Processing Systems, NIPS 2016*. 2016, pp. 3387–3395.

[202] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Vol. 35. 4. 1993, pp. 680–681.

[203] Mark Nixon and Alberto S. Aguado. *Feature extraction & image processing, Second Edition*. 2nd. Academic Press, 2008.

[204] Augustus Odena, Christopher Olah, and Jonathon Shlens. "Conditional image synthesis with auxiliary classifier gans". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. 2017, pp. 2642–2651.

[205] Augustus Odena, Jacob Buckman, Catherine Olsson, Tom B. Brown, Christopher Olah, Colin Raffel, and Ian J. Goodfellow. "Is generator conditioning causally related to GAN performance?" In: *Proceedings of Machine Learning Research, ICML 2018*. Vol. 80. PMLR, 2018, pp. 3846–3855.

[206] Michael O'Neill, Leonardo Vanneschi, Steven M. Gustafson, and Wolfgang Banzhaf. "Open issues in genetic programming". In: *Genetic Programming and Evolvable Machines* 11.3-4 (2010), pp. 339–363.

[207] Erdal Paksoy, Wai-Yip Chan, and Allen Gersho. "Vector quantization of speech LSF parameters with generalized product codes". In: *The Second International Conference on Spoken Language Processing, ICSLP 1992*. ISCA, 1992.

[208] Karl Pearson. "The problem of the random walk". In: *Nature* 72.1867 (1905), p. 342.

[209] Fatih Porikli, Oncel Tuzel, and Peter Meer. "Covariance tracking using model update based on Lie algebra". In: *Proceedings of IEEE Conf. Computer Vision and Pattern Recognition*. CVPR 2006. IEEE, 2006, pp. 728–735.

[210] Vyatcheslav B. Priezzhev, Deepak Dhar, Abhishek Dhar, and Supriya Krishnamurthy. "Eulerian walkers as a model of self-organized criticality". In: *Physical Review Letters* 77.25 (1996), p. 5079.

[211] James Propp. "Three lectures on quasirandomness". In: (2004). http://faculty.uml.edu/jpropp/berkeley.html.

[212] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *4th International Conference on Learning Representations, ICLR 2016, Conference Track Proceedings*. 2016.

[213] Ibrahim M. H. Rahman, Christopher Hollitt, and Mengjie Zhang. "Contextual-based top-down saliency feature weighting for target detection". In: *Mach. Vis. Appl.* 27.6 (2016), pp. 893–914.

[214] Günther R. Raidl and Bryant A. Julstrom. "Edge sets: an effective evolutionary coding of spanning trees". In: *IEEE Trans. Evolutionary Computation* 7.3 (2003), pp. 225–239.

[215] François-Michel De Rainville, Christian Gagné, Olivier Teytaud, and Denis Laurendeau. "Evolutionary optimization of low-discrepancy sequences". In: *ACM Trans. Model. Comput. Simul.* 22.2 (2012), 9:1–9:25.

[216] Ingo Rechenberg. *Evolutionsstrategie–Optimierung technisher Systeme nach Prinzipien der biologischen Evolution*. Problemata. Stuttgart-Bad Cannstatt: Frommann-Holzboog, 1973.

[217] Ingo Rechenberg. "Evolutionsstrategien". In: *Simulationsmethoden in der Medizin und Biologie*. Springer, 1978, pp. 83–114.

[218] Katharina Reinecke, Tom Yeh, Luke Miratrix, Rahmatri Mardiko, Yuechen Zhao, Jenny Liu, and Krzysztof Z. Gajos. "Predicting users' first impressions of website aesthetics with a quantification of perceived visual complexity and colorfulness". In: *Conference on Human Factors in Computing Systems, CHI 2013*. ACM, 2013, pp. 2049–2058.

[219] Jaume Rigau, Miquel Feixas, and Mateu Sbert. "Informational aesthetics measures". In: *IEEE Computer Graphics and Applications* 28.2 (2008).

[220] Sebastian Risi, Sandy D. Vanderbleek, Charles E. Hughes, and Kenneth O. Stanley. "How novelty search escapes the deceptive trap of learning to learn". In: *Genetic and Evolutionary Computation Conference, GECCO 2009*. ACM, 2009, pp. 153–160.

[221] Juan Romero and Penousal Machado, eds. *The art of artificial evolution: a handbook on evolutionary art and music*. Natural Computing Series. Berlin - Heidelberg: Springer, 2008.

[222] Juan Romero and Penousal Machado, eds. *The Art of Artificial Evolution: A Handbook on Evolutionary Art and Music*. Natural Computing Series. Springer, 2008.

[223] Steven Rooke. "The Evolutionary Art of Steven Rooke". In: (2006). `http://srooke.com`.

[224] Brian J. Ross, William Ralph, and Hai Zong. "Evolutionary image synthesis using a model of aesthetics". In: *IEEE International Conference on Evolutionary Computation, CEC 2006, part of WCCI 2006*. IEEE, 2006, pp. 1087–1094.

[225] Franz Rothlauf and David E. Goldberg. *Representations for genetic and evolutionary algorithms*. Physica-Verlag, 2002.

[226] Manuel Ruder, Alexey Dosovitskiy, and Thomas Brox. "Artistic style transfer for videos". In: *German Conference on Pattern Recognition*. Springer. 2016, pp. 26–36.

[227] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. "Improved techniques for training gans". In: *Advances in neural information processing systems*. 2016, pp. 2234–2242.

[228] Bruno Sareni and Laurent Krahenbuhl. "Fitness sharing and niching methods revisited". In: *IEEE transactions on Evolutionary Computation* 2.3 (1998), pp. 97–106.

[229] Marc Schoenauer, Fabien Teytaud, and Olivier Teytaud. "A Rigorous Runtime Analysis for Quasi-Random Restarts and Decreasing Stepsize". In: *Proc. of Artificial Evolution EA 2011*. 2011, pp. 37–48.

[230] Hans-Paul Schwefel. *Evolution and optimum seeking*. Vol. 1515. Wiley New York, 1995.

[231] Hans-Paul Schwefel. *Numerical optimization of computer models*. John Wiley & Sons, Inc., 1981.

[232] Karl Sims. "Artificial evolution for computer graphics". In: *Proceedings of the Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1991*. Ed. by James J. Thomas. ACM, 1991, pp. 319–328.

[233] Karl Sims. "Galápagos". In: (1997). `http://www.karlsims.com/galapagos/`.

[234] John Maynard Smith. *Evolution and the Theory of Games*. Cambridge University Press, 1982.

[235] Ilya M. Sobol. "On the distribution of points in a cube and the approximate evaluation of integrals". In: *Zh. Vychisl. Mat. Mat. Fiz.* 7.4 (1967), pp. 784–802.

[236] Andrew R Solow and Stephen Polasky. "Measuring biological diversity". In: *Environmental and Ecological Statistics* 1.2 (1994), 95—103.

[237] Yvonne Spielmann. "Aesthetic features in digital imaging: collage and morph". In: *Wide Angle* 21.1 (1999), pp. 131–148.

[238] Kenneth O. Stanley and Joel Lehman. *Why greatness cannot be planned - The myth of the objective*. Springer, 2015.

[239] Guy Steele. *Common LISP: the language*. Elsevier, 1990.

[240] Thomas Strothotte and Stefan Schlechtweg. *Non-photorealistic Computer Graphics: Modeling, Rendering, and Animation*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002.

[241] Dirk Sudholt. "A New Method for Lower Bounds on the Running Time of Evolutionary Algorithms". In: *IEEE Trans. Evolutionary Computation* 17.3 (2013), pp. 418–435.

[242] Hideyuki Takagi. "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation". In: *Proceedings of the IEEE* 89.9 (2001), pp. 1275–1296.

[243] Olivier Teytaud. "Quasi-random numbers improve the CMA-ES on the BBOB testbed". In: *Artificial Evolution - 12th International Conference, Evolution Artificielle, EA*. 2015, pp. 58–70.

[244] Olivier Teytaud and Hervé Fournier. "When does quasi-random work?" In: *Parallel Problem Solving from Nature, PPSN*. 2008, pp. 325–336.

[245] Olivier Teytaud and Sylvain Gelly. "DCMA: yet another derandomization in covariance-matrix-adaptation". In: *Genetic and Evolutionary Computation Conference, GECCO 2007*. ACM, 2007, pp. 955–963.

[246] Eric Thiémard. "An Algorithm to Compute Bounds for the Star Discrepancy". In: *J. Complexity* 17 (2001), pp. 850–880.

[247] Stephen Todd and William Latham. *Evolutionary art and computers*. Academic Press, Inc., 1992.

[248] Karen Trist, Vic Ciesielski, and Perry Barile. "An artist's experience in using an evolutionary algorithm to produce an animated artwork". In: *International Journal of Arts and Technology, IJART* 4.2 (2011), pp. 155–167.

[249] Oncel Tuzel, Fatih Porikli, and Peter Meer. "Pedestrian detection via classification on RIemannian manifolds". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 30.10 (2008), pp. 1713–1727.

[250] Oncel Tuzel, Fatih Porikli, and Peter Meer. "Region covariance: a fast descriptor for detection and classification". In: *Proc. 9th European Conf. Computer Vision (ECCV '06)*. Vol. 3952. Lecture Notes in Computer Science. Berlin - Heidelberg: Springer, 2006, pp. 589–600.

[251] Tamara Ulrich, Johannes Bader, and Lothar Thiele. "Defining and Optimizing Indicator-Based Diversity Measures in Multiobjective Search". In: *Parallel Problem Solving from Nature - PPSN XI, 11th International Conference*. 2010, pp. 707–717.

[252] Tamara Ulrich, Johannes Bader, and Eckart Zitzler. "Integrating decision space diversity into hypervolume-based multiobjective search". In: *Genetic and Evolutionary Computation Conference, GECCO 2010*. ACM, 2010, pp. 455–462.

[253] Tamara Ulrich and Lothar Thiele. "Maximizing population diversity in single-objective optimization". In: *Genetic and Evolutionary Computation Conference, GECCO 2011*. ACM, 2011, pp. 641–648.

[254] Tatsuo Unemi. "Embedding movie into SBART - breeding deformed movies". In: *Proceedings of the IEEE International Conference on Systems, Man & Cybernetics, 2004*. IEEE, 2004, pp. 5760–5764.

[255] Tatsuo Unemi. "SBART 2.4: breeding 2D CG images and movies and creating a type of collage". In: *Conference on Knowledge-Based Intelligent Information Engineering Systems, KES 1999*. IEEE, 1999, pp. 288–291.

[256] Tatsuo Unemi. "SBArt4 for an automatic evolutionary art". In: *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2012*. IEEE, 2012, pp. 1–8.

[257] Paulo Urbano. "Consensual Paintings". In: *Applications of Evolutionary Computing, EvoWorkshops 2006, Proceedings*. Vol. 3907. Lecture Notes in Computer Science. Springer, 2006, pp. 622–632.

[258] Paulo Urbano. "Playing in the Pheromone Playground: Experiences in Swarm Painting". In: *Applications of Evolutionary Computing, EvoWorkshops 2005, Proceedings*. Vol. 3449. Lecture Notes in Computer Science. Springer, 2005, pp. 527–532.

[259] Rasmus K Ursem. "Diversity-guided evolutionary algorithms". In: *International Conference on Parallel Problem Solving from Nature*. Springer. 2002, pp. 462–471.

[260] David Allen Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. Air Force Institute of Technology, 1999.

[261] Adriano Vinhas, Filipe Assunção, João Correia, Aniko Ekárt, and Penousal Machado. "Fitness and novelty in evolutionary art". In: *Proceedings of International Conference on Evolutionary and Biologically Inspired Music and Art, EvoMUSART 2016*. Springer. 2016, pp. 225–240.

[262] Israel A. Wagner, Michael Lindenbaum, and Alfred M. Bruckstein. "Distributed covering by ant-robots using evaporating traces". In: *IEEE Transactions on Robotics and Automation* 15.5 (1999), pp. 918–933.

[263] Markus Wagner, Karl Bringmann, Tobias Friedrich, and Frank Neumann. "Efficient optimization of many objectives by approximation-guided evolution". In: *European Journal of Operational Research* 243.2 (2015), pp. 465 –479.

[264] Magnus Wahlström. Implementations of the DEM- and the TA-algorithm. Available at `http://www.mpi-inf.mpg.de/~wahl/`.

[265] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. "Locality-constrained linear coding for image classification". In: *Computer Vision and Pattern Recognition CVPR*. IEEE. 2010, pp. 3360–3367.

[266] John Willats and Frédo Durand. "Defining pictorial style: Lessons from linguistics and computer graphics". In: *Axiomathes* 15.2 (2005).

[267] Stephen Wilson. *Information arts: intersections of art, science, and technology*. MIT press, 2002.

[268] Carsten Witt. "Tight Bounds on the Optimization Time of a Randomized Search Heuristic on Linear Functions". In: *Combinatorics, Probability & Computing* 22.2 (2013), pp. 294–318.

[269] Brian JF Wong, Koohyar Karimi, Zlatko Devcic, Christine E McLaren, and Wen-Pin Chen. "Evolving attractive faces using morphing technology and a genetic algorithm: A new approach to determining ideal facial aesthetics". In: *The Laryngoscope* 118.6 (2008), pp. 962–974.

[270] Bing Xue, Mengjie Zhang, Will N. Browne, and Xin Yao. "A survey on evolutionary computation approaches to feature selection". In: *IEEE Trans. Evolutionary Computation* 20.4 (2016), pp. 606–626.

[271] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. "Self-Attention Generative Adversarial Networks". In: *Proceedings of the 36th International Conference on Machine Learning*. Vol. 97. Proceedings of Machine Learning Research. 2019, pp. 7354–7363.

[272] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks". In: *Proceedings of the IEEE International Conference on Computer Vision, ICCV*. 2017, pp. 5907–5915.

[273] Qingfu Zhang and Hui Li. "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition". In: *IEEE Trans. Evolutionary Computation* 11.6 (2007), pp. 712–731.

[274] Junbo Jake Zhao, Michaël Mathieu, and Yann LeCun. "Energy-based Generative Adversarial Networks". In: *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[275] Xi Zhou, Kai Yu, Tong Zhang, and Thomas S Huang. "Image classification using super-vector coding of local image descriptors". In: *European Conference on Computer Vision, ECCV 2010*. Springer. 2010, pp. 141–154.

[276] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE International Conference on Computer Vision, ICCV*. 2017, pp. 2223–2232.

[277] Eckart Zitzler and Simon Künzli. "Indicator-Based selection in multiobjective search". In: *Parallel Problem Solving from Nature, PPSN 2004*. Vol. 3242. LNCS. Springer, 2004, pp. 832–842.

[278] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert da Fonseca. "Performance assessment of multiobjective optimizers: an analysis and review". In: *IEEE Trans. Evolutionary Computation* 7.2 (2003), pp. 117–132.