



Norwegian University of
Science and Technology

Computer generated art based on musical input

Viktor Zoric

Master of Science in Computer Science

Submission date: June 2017

Supervisor: Björn Gambäck, IDI

Norwegian University of Science and Technology
Department of Computer Science

Abstract

This thesis explores computer-generated art based on musical input. The computer-generated art gives an end-user support in various life domains, which impose various aesthetic, artificial intelligence and software engineering goals and requirements.

This thesis combines theoretical and practical approaches. It is a design, implementation and experiment-driven exercise and focuses on the use of evolutionary algorithms (EA) in image creation. End-user involvement and survey results contributed to the EA functionality, e.g. seeding techniques, fitness functions, mappings between genotype and phenotype information, etc.

The EA for music-to-image transformation in this work operates on music and image metadata, rather than raw data. The metadata (for music and image features) is utilized by mapping tables, which work as recipes for images. This thesis uses EA to generate images by evolving the mapping tables without interactive involvement. Usage of metadata and mapping tables in EA introduces an alternative approach to computer generated image art, compared to the research body.

Design tests and experiments focused on: (a) using mapping tables to match music and image features, and (b) experiments with various fitness functions that combined user preferences with art novelty aspects. The obtained experimental results and the end user interaction tests and evaluations, are promising and motivate for further development. Fitness functions based on a distance measure between user preferred mapping tables and the computer-generated mapping tables, can efficiently drive the EA towards the user's preference. Novel and interesting mappings between the image and music features can be achieved with fitness functions that selectively ignore some user preferences. Evolution of color palettes and figure parameters can match user expectations with reasonable time-usage without knowing the structure of the optimal mapping table.

The developed software framework enables both future evolutionary algorithm improvements, and use of alternative AI approaches.

Sammendrag

Denne masteroppgaven tar for seg datamaskin-generert bildekunst basert på musikk. Kunsten støtter sluttbrukeren i ulike domener som stiller ulike mål og krav innen estetikk, kunstig intelligens og programvaredesign.

Arbeidet kombinerer teori og praksis. Det er et design-, implementasjon- og eksperimentbasert verk som fokuserer på bruk av evolusjonære algoritmer (EA) for å skape bilder. Engasjement fra sluttbrukere og resultater av spørreundersøkelsen bidro til funksjonaliteten av EA, f.eks. så-teknikker, fitnessfunksjoner, assosiasjon mellom genotyper og phenotyper, osv.

EA for musikk-til-bilde assosiasjonen i dette arbeidet operer på musikk- og bildemetadata i stedet for rådata. Assosiasjonstabeller bruker metadata (for musikk- og bildeegenskaper) for å lage oppskrifter for bildegenerering. Dette arbeidet bruker EA for å generere bilder ved hjelp av evolusjon av assosiasjonstabeller, uten direkte brukerinvolvering i evolusjonen. Bruk av metadata og assosiasjonstabeller i EA introduserer en alternativ metode for datamaskin-generering av bilder, sammenliknet med eksisterende forskning.

Design tester og eksperimenter fokuserte på: (a) bruke assosiasjonstabeller for å jamføre musikk- og bildeegenskaper, og (b) eksperimenter med ulike fitnessfunksjoner som kombinerer brukerpreferanser med nye aspekter i kunst. Oppnådde eksperimentresultater og interaksjoner og tester med sluttbrukere, er lovende og motiverer for videre utvikling. Fitnessfunksjoner basert på et avstandsmål mellom sluttbrukerprefererte assosiasjonstabeller og datamaskin-genererte assosiasjonstabeller, kan effektivt drive evolusjonen i retning av brukerpreferansen. Nye og interessante assosiasjoner mellom musikk- og bildeegenskaper kan oppnås ved bruk av fitnessfunksjoner som selektivt velger bort ulike brukerpreferanser. Evolusjon av fargepaletter og figurparametere kan stemme overens med brukerforventninger innen akseptabelt tidsforbruk, uten å vite hvordan den optimale assosiasjonstabellen er.

Det utviklede programvarerammeverket fasiliterer både forbedringer av EA og bruk av alternative metoder i kunstig intelligens.

Preface

This Master's thesis focuses on computer generated image art based on musical input. The whole system is implemented from scratch as a software framework for future work and research. Bjørnar Walle Alvestad, Endre Larsen and I developed together, from scratch with equal contributions, the core for a general evolutionary algorithms library, that we use for our separate master's theses.

I am truly grateful to my supervisor Björn Gambäck, for his helpful guidance and feedback.

Viktor Zoric

Trondheim, June 1, 2017

CONTENTS

1	Introduction	1
1.1	Motivation	1
1.2	Problem Definition and Research Questions	4
1.3	Structure of This Thesis	5
2	Background	7
2.1	Creativity	7
2.2	Computer Art, Music and Colors	7
2.3	Computational Creativity	8
2.4	Evolutionary Algorithms	9
2.5	Software Framework Design Approach	11
3	Related Work	13
3.1	Computational Art and Aesthetics	13
3.1.1	Interactive Evolutionary Computation	14
3.1.2	Aesthetics and Fitness Functions	17
3.1.3	Aesthetics and Information Theory	18
3.1.4	Future Possibilities for Fitness Computation	19
3.1.5	Evolving and Seeding Techniques	19
3.1.6	Novelty in Evolutionary Art	21
3.2	Systems, Frameworks, Projects and Tools	23
4	Architecture and Design	29
4.1	Mapping of Music and Images	29
4.1.1	Music Features and Related Parameters	29
4.1.2	Image Features and Related Parameters	30
4.2	Evolutionary Algorithms — Numerical Definition and Setup	30
4.2.1	Genotype	34
4.2.2	Phenotype	34
4.2.3	Fitness Functions	34
4.2.4	Selection Strategies	37
4.2.5	Crossover Techniques	38

4.2.6	Mutation Techniques	38
5	Implementation	39
5.1	Implementation of the Software Framework	39
5.2	External Functionality	40
5.3	Evolutionary Algorithms.....	41
5.3.1	Implementation of the Genotype and Phenotype.....	41
5.3.2	Implementation of Fitness Functions.....	43
5.4	Implementation of the Painting Algorithm	45
6	Experiments and Results.....	49
6.1	Experimental Setup	49
6.2	Software Implementation Tests.....	50
6.2.1	T.1 – Sound analysis unit tests.....	51
6.2.2	T.2 – Image creation tests	51
6.2.3	T.3 – Evolutionary Algorithms tests.....	52
6.3	Research Experiments with Evolutionary Algorithms.....	54
6.3.1	E.1 – Experiments with fitness functions	55
6.3.2	E.2 – Experiments with seeding techniques	61
6.3.3	E.3 – Experiments with mutation pressure	64
6.3.4	E.4 – Experiments with crossover techniques	69
6.3.5	E.5 – Experiments with selection strategies	71
6.3.6	E.6 – Performance evaluations	73
6.3.7	E.7 – Experiments with user involvement	77
7	Discussion.....	85
7.1	The Project from the High-Level Perspective.....	85
7.2	Project Goals and Research Questions.....	86
7.3	Creativity	90
7.4	Learnings and Experiences	91
7.5	Future Prospects	91
8	Conclusion	93
	Bibliography	95

Appendix A: Spotify Audio Features Table	I
Appendix B: Phenotype & Genotype	II
Appendix C: Spotify Analysis Results.....	III
Appendix D: User Survey	IV

List of Figures

Figure 1. Typical Use Cases of Computer Generated Image Art in Human Life.....	3
Figure 2. Evolutionary loop flowchart.....	10
Figure 3. Illustration of the Painting Algorithm's work.....	31
Figure 4. Pseudocode of the fitness function for optimizing towards user specified mapping table.....	43
Figure 5. Pseudocode of the fitness function novelty combined with user specified mapping table.....	44
Figure 6. Random rectangles	48
Figure 7. Random ovals	48
Figure 8. Random polygons.....	48
Figure 9. Random lines	48
Figure 10. Random curves	48
Figure 11. No effects.....	48
Figure 12. Slight oil effect	48
Figure 13. Strong oil effect.....	48
Figure 14. Slight blur	48
Figure 15. Strong blur.....	48
Figure 16. Illustration of images used in the image creation tests, performed by random-generating a small set of images.....	51
Figure 17. Excerpt from the logfile after generating image 1 from the above set of four images. Each debug line describes what painting tools is being used and with what parameter values.	52
Figure 18. Hamming Distance EA optimizing towards zero. SD is the standard deviation between average (AVG) and best value in each generation.	53
Figure 19. OneMax EA optimizing towards one. SD is the standard deviation between average (AVG) and best value in each generation.....	53
Figure 20. Result images after evolution optimizing towards user preference.....	55
Figure 21. Result images after evolution optimizing towards novelty search freely selecting variable for colors.....	56
Figure 22. Result images after evolution optimizing towards novelty search ignoring some user defined keys.....	57
Figure 23. Result images after evolution optimizing towards user preferred colors for each song without an optimal mapping table.	60

Figure 24. Result images after evolution optimizing towards user preferred base color, color spread in palette and number of figures.....	60
Figure 25. Another set of result images after evolution optimizing towards user preferred base color, color spread in palette and number of figures.	60
Figure 26. Result images after seeded evolution towards user preferred base color, color spread in palette and number of figures. Seeding genotypes taken from earlier experiments (Figure 20).	60
Figure 27. Graph showing mean fitness values of the evolutionary run for EA without seeding. Marked point shows the last best fitness value.....	62
Figure 28. Result images after evolution optimizing towards user preference without seeding. .	62
Figure 29. Graph showing mean fitness values of the evolutionary run for EA with seeding. Marked point shows the last best fitness value.....	63
Figure 30. Result images after evolution optimizing towards user preference with seeding.	63
Figure 31. Graph showing mean fitness values of the evolutionary run for EA with low mutation pressure. Marked point shows the last best fitness value.	65
Figure 32. Result images after evolution optimizing towards user preference with low mutation pressure.	65
Figure 33. Graph showing mean fitness values of the evolutionary run for EA with medium mutation pressure. Marked point shows the last best fitness value.	66
Figure 34. Result images after evolution optimizing towards user preference with medium mutation pressure.	66
Figure 35. Graph showing mean fitness values of the evolutionary run for EA with medium mutation pressure. Marked point shows the last best fitness value.	67
Figure 36. Result images after evolution optimizing towards user preference with medium mutation pressure.	68
Figure 37. Graph showing mean fitness values of the evolutionary run for EA with One Point crossover. Marked point shows the last best fitness value.	69
Figure 38. Result images after evolution optimizing towards user preference with One Point crossover.	70
Figure 39. Graph showing mean fitness values of the evolutionary run for EA with Uniform crossover. Marked point shows the last best fitness value.	70
Figure 40. Result images after evolution optimizing towards user preference with Uniform crossover.	70
Figure 41. Graph showing mean fitness values of the evolutionary run for EA with Tournament selection. Marked point shows the last best fitness value.	71
Figure 42. Result images after evolution optimizing towards user preference with Tournament selection.	72

Figure 43. Graph showing mean fitness values of the evolutionary run for EA with Proportionate selection. Marked point shows the last best fitness value. 72

Figure 44. Result images after evolution optimizing towards user preference with Proportionate selection. 72

Figure 45. Mean duration to reach desired fitness value 74

List of Tables

Table 1. EA pseudo code for Music-to-Image creation.....	33
Table 2. Example of mapping between tool parameters and music feature variables.....	34
Table 3. Hash table example of mapping between tool parameters and music feature variables.	42
Table 4. List of the implementation tests and EA research experiments.....	49
Table 5. Describing what tests and research experiments are used to address each research question.....	50
Table 6. Basis EA configuration for the evolutionary runs. Some values are modified in specific experiments.....	54
Table 7. Base color preference of the user that provided input for experiments in Figure 23 - Figure 26.....	58
Table 8. Table showing color palette differences between songs A and B in Figure 20 and Figure 34	67
Table 9. Results after a number of generations during the evolution. Optimizing towards user specified mapping table.	76

List of Abbreviations

ANN	Artificial Neural Network
API	Application Programming Interface
BPM	Beats Per Minute
CA	Cellular Automata
CAE	Computationally Aesthetic Evaluation
CPPN	Compositional Pattern-Producing Networks
CSP	Constraint Satisfaction Problem
EA	Evolutionary Algorithms
EC	Evolutionary Computation
EiT	Experts in Teamwork ¹
FFT	Fast Fourier Transform
FW	Framework
GP	Genetic Programming
ICT	Information and Communication Technology
IE	Interactive Evolution
IEA	Interactive Evolutionary Algorithms
IEC	Interactive Evolutionary Computing
R&D	Research and Development
RGB	Red Green Blue
RYB	Red Yellow Blue
SW	Software
TDD	Test Driven Development
URI	Uniform Resource Identifier
URL	Uniform Resource Locator

¹ <http://www.ntnu.edu/eit>

1 INTRODUCTION

The topic of this work is to create a system that can generate image art based on musical input with use of evolutionary algorithms. This means that the system should create images that share features with the corresponding music. Such features could be emotional or artistic, where the aim is to create a correlation between image art and music that the end user agrees on. The system should be able to take an arbitrary musical piece as input and create a corresponding image, considering both end user specifications and novelty.

This thesis combines a theoretical and a practical approach. It is a design, implementation and experiments-driven exercise and focuses on the use of evolutionary algorithms (EA) in image creation. End-user involvement, survey results and user interaction tests, contributed to the EA functionality, e.g. seeding techniques, feedback functions, and mappings between genotype and phenotype information, etc.

The EA for music-to-image transformation in this work operates on music and image metadata, rather than raw data. The metadata (for music and image features) is utilized by mapping tables, which work as recipes for images. This thesis uses EA to generate images by evolving the mapping tables without interactive involvement. Using of metadata and mapping tables in EA introduces an alternative approach to computer generated image art, compared to the research body.

1.1 Motivation

Music and art are everyday parts of many people's lives. People can find inner peace, joy or energy while listening to a special melody or admiring an image. Humans can find or make a correlation between music and art, often by visualizing the music in their mind. A combination of several sensory impressions, here both sound and vision, may trigger a deeper feeling/mood than just one sensory impression would. Visual art can enrich various aspects, phases and situations in human life, as illustrated by Figure 1.

The four use cases have different requirements, application areas, and interaction workflows, and the support role for the computing art is situation dependent:

1. ***Dynamic Ambient Decoration*** focuses on interior decoration, e.g. domestic spaces (houses, flats), public facilities (schools, kindergartens), public transportation facilities, etc. Each ambient might have quite different requirements and uses of the technology.
2. ***Therapeutic Art*** addresses the decoration of spaces (walls, canvases, etc.) that should have a positive psychological effect on the person (e.g. patient).
3. ***Artist's Work tool*** provides tools for artist-guided generation of computer art, where the artist is the guide and corrective factor, and the tool follows, suggests and realizes. Here the focus is on the collaboration and interplay between two creativities (artist's and computer's).
4. ***AI Art generator*** – a self-sustained generator of computer art is the use case with probably the highest art-quality requirement and the novelty factor more expressed than in the other Use Cases.

These use cases illustrate not just a variety of possible usages of computer generated art in human life, but suggest that quite different requirements, maybe also aesthetic criteria (fitness and quality functions) might govern. E.g. the therapeutic art use case might need a fitness function dominated by personal preferences (non-disturbing aesthetics), the use cases artist's work tool and AI art generator might emphasize the novelty function and effect of artistic surprise (even when disturbing the user). Art is often created to challenge the human mind by making it study images and look for deeper meanings or patterns. This process can have multiple effects on humans. Some people enjoy art as decoration in their homes where they add a mood to specific areas, while others find art to work as psychological therapy by using art to aid meditation or to induce calmness while searching for deeper meanings within the image. Using image and sound therapeutically can contribute to healing psychosomatic disorders such as stress, depression, anxiety, or others where the state of mind plays a major role.

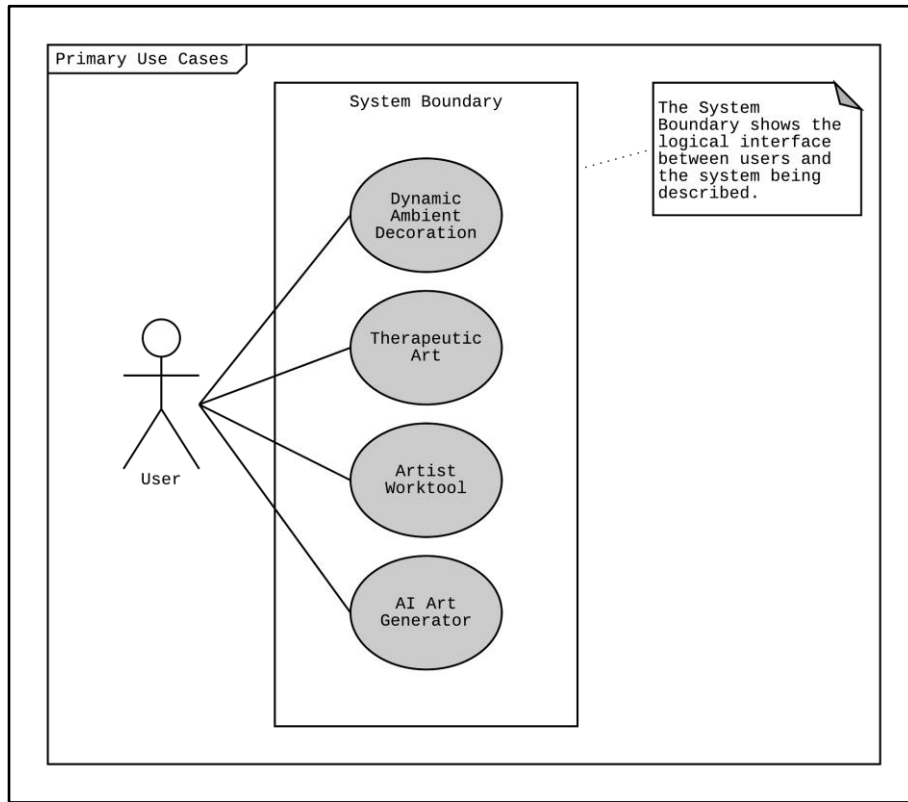


Figure 1. Typical Use Cases of Computer Generated Image Art in Human Life.

1.2 Problem Definition and Research Questions

The major goals of this project are:

1. Design the computer artist system

This project aims to create a system that generates image art, limited to Music-to-Image creation. The overall system should be modular and flexible such that different AI methods and generation tools can easily be implemented and experimented upon. It could be organized as a software framework.

2. Implement evolutionary algorithms in the system to create images based on musical input

This project will focus on the use of evolutionary algorithms (EA) and their use in image generation. However, the EA will not be used to directly evolve image art, but rather to evolve a mapping between music and images. This evolution requires custom phenotypes, genotypes, genetic operators and fitness functions, to function properly.

The research questions this project is addressing are:

1. How can the computer artist system be analyzed and evaluated?

To evaluate the performance of the system as to whether the system generates accurate results based on an input, an evaluation procedure is necessary. Can the system itself evaluate the performance sufficiently through a fitness function or is it necessary to involve humans to aid the evaluation process?

2. Can the system learn about user preferences and user notions of aesthetics, and how?

If the system can learn about user preferences, it can generate tailored results to each user and gain higher probability of generating accurate results for new users. Can the evolutionary algorithms be guided by earlier experiences and the use of seeding?

3. How can user feedback be collected and used in the system?

User feedback is necessary to train the system to produce more accurate results, evaluated by the end user. How can this feedback be structured and collected, and later reused to increase the performance of the system?

4. Can the system create aesthetically pleasing and meaningful images without user involvement during evolution?

This question aims to guide a search for a trivial fitness function that can be used to generate aesthetically pleasing images that have a meaningful correlation between music and image without involving humans in the evolutionary process.

1.3 Structure of This Thesis

This thesis is structured in the following way:

- **Chapter 1** introduces the thesis, motivation for the work, communicates the problem definition and the research questions.
- **Chapter 2** provides a background, with focus on (1) creativity, (2) computer art, music and colors, and (3) evolutionary algorithms (EA). It narrows the above-mentioned general perspective to the field of computational creativity, and further down to EA. It concludes with the software engineering aspects of evolutionary algorithms and their practical applications.
- **Chapter 3** deals with the existing knowledge body, limited to Computational Art and Aesthetics and related systems, frameworks, projects and tools. The study of computational art and aesthetics is limited to the practical aspects of evolutionary algorithms and similar AI technologies. Analysis of many research publications provided an interesting overview of the field and gave good advices and ideas for the research work of this thesis.
- **Chapter 4** communicates the architecture and design approach taken in this work, in particular the methodology for mapping of music to images, techniques for design of genotype and phenotype, as well as the numerical definition and setup of Evolutionary Algorithm systems developed in this work. This theoretical design analysis gives a good foundation for the next chapter.

- **Chapter 5** details the implementation of the concepts discussed in the previous chapter. The solution implemented in this work is a combination of a custom approach to EA with some of the research ideas taken over from the intensive state-of-the-art analysis provided in Chapter 3.
- **Chapter 6** focuses on experiments and results. It starts with an explanation of experimental setups and user-focused tests and feedbacks. The chapter continues with the presentation of numerous software tests and research experiments. The experiment-based software design approach gave the possibility to improve the system, and also to test some of the research findings and claims from the previous publications. Gradual and experiments-focused evaluation and design of the EA was particularly useful, because the software system was implemented from scratch. The reasons for this are discussed in chapters 5 and 7.
- **Chapter 7** evaluates and discusses the theoretical and practical results produced in this work and gives some ideas for future work. The software framework produced in this work gives a flexible and customizable foundation for various improvements and upgrades of both regarding AI aspects and the services and applications which might use them. In other words, it gives a good basis for future research and development activities.
- **Chapter 8** concludes this work.

2 BACKGROUND

This chapter explains some of the necessary background theory of this project. First a brief explanation of creativity is given, then a discussion of computational creativity. Following is a short background theory of evolutionary algorithms, and other artificial intelligence topics, relevant to the fields of computational creativity and computational art. The chapter concludes with a short discussion of the need for a SW Framework approach as a means of interacting with already existing ICT system architectures.

2.1 Creativity

Creative ideas must be novel, surprising, and valuable (Boden, 1998). Novel ideas can be novel for the individual that came up with the idea or novel for the whole world. P-creativity (psychological creativity) is the category where an idea already exists, but is new for an individual. H-creativity (historical creativity) is the category for creative ideas that are new in the history of creativity (Boden, 1998). There are three main types of creativity describing generation of novel ideas: combinatorial, exploratory, and transformational (Boden, 1998). Combinatorial creativity is the generation of novel ideas through combination of familiar ideas. Examples are poetic imagery and analogy. Exploratory creativity involves exploration of structured conceptual spaces, that often results in structures or ideas that are novel and unexpected. Musical creativity often belongs to this category. Another example, that can be considered exploratory creativity, is the introduction of new musical styles, such as jazz in the 1930s, rock 'n' roll in the 1950s, and rap in the 1980s. Transformational creativity involves transformation of some dimensions of the domain space, such that new structures can be created that could not be created in the previous space. An example of transformational creativity can be artefact generation that takes inspiration from a two-dimensional artefact and creates a new artefact in three dimensions.

2.2 Computer Art, Music and Colors

One of the widest definitions of art includes anything that is human-made (Britannica, 2010). This can be divided into kinds of art, for example fine art and useful art, thus excluding man-made objects that do not carry aesthetic implications (for example storage units, or garbage dumps which

are solely functional). With the ongoing development of artificial intelligence, it becomes relevant to include computer generated art as a new kind of art which is more and more independent from human creation and creativity. In this thesis, the term "art" is encompassing all intentional and artificial aesthetic expression. Therefore, including both amateur and professional art, fine and useful art, man-made and computer generated art.

Through centuries art has utilized colors to express emotions and feelings in harmony with drawn shapes or lines. The main difference between colors in the computer world and art world are the three primary colors, and how they are mixed to obtain new colors. Computer monitors emit colors through light, while handmade paintings reflect colors (absorb some wavelengths of light) through the paint.

Absorption of wavelengths to obtain new colors belongs to the subtractive color model. It describes how mixing colors results in a reduction of reflected colors from white light. Mixing increasing amounts of color dye yields darker colors. On the other hand, mixing different wavelengths of light belongs to the additive color model. This model yields lighter colors through mixing of high numbers of wavelengths of light.

The RGB (Red Green Blue) color model belongs to additive color mixing and is mostly used to represent colors in computers. RYB (Red Yellow Blue) belongs to subtractive color mixing and has its roots in the art world. RYB mixed colors tend to be gentler to the eye, while RGB can represent a larger set of colors.

Using colors to express emotions or feelings is a key aspect in art. The links between a color and an emotion or a feeling have been mapped, through a comparative study, by 98 students at Georgia University (Kaya & Epps, 2004). Red color variations were mapped to an angry feeling, while blue was correlated to a calm feeling. Shades of yellow described a comfortable or happy feeling. Color meaning can also differ from culture to culture, and vary based on subjective interpretation and connotation.

2.3 Computational Creativity

Computational creativity is a subfield of Artificial Intelligence where the aim is to build a system that creates artefacts and ideas. Colton and Wiggins gave a working definition of what

computational creativity is: “The philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviors that unbiased observers would deem to be creative.” (Colton & Wiggins, 2012, p.1). Computer models of creativity include examples of all three types of creativity: combinatorial, explorational, and transformational. Models that focus on exploratory creativity are the most successful (Boden, 1998). Exploratory creativity can be correlated to search and optimization in algorithms, of which EA is an example.

2.4 Evolutionary Algorithms

Over the last two decades there have been many different approaches to generation of art using computers. A recurring method of generating art is using evolutionary algorithms. This method is highly dependent on a fitness function which accurately describes, in mathematical terms, how good a solution is. Lacking this feature the algorithm will struggle to generate a good solution set. As it is very difficult to make a general mathematical function which accurately evaluates aesthetics in an image, several approaches to generative art programs use interactive evolutionary algorithms, where humans take part in the evaluation of aesthetics/quality.

Evolutionary algorithms (EA) are algorithms within artificial evolutions, inspired by natural evolution, used to solve various optimization problems, discover novel solutions, or to explore areas that are usually addressed by a human design. EA rely on three key features; the genotype, the phenotype, and the fitness function (Floreano & Mattiussi, 2008a). The genotype defines the genes/building blocks of a solution, and the phenotype is generally a solution built using the genotype. Every discovered solution (phenotype) must be evaluated to determine how it will continue its evolution. Solutions having high fitness values also have higher probabilities of producing offspring with advanced fitness, while low fitness valued solutions tend to be exterminated from the evolution (Floreano & Mattiussi, 2008a).

Figure 2 shows a flowchart diagram of the evolutionary loop and is further described here. EA start with an initial population of N phenotypes generated from N genotypes. Each phenotype is evaluated using a fitness function. If a stopping criterion is not reached, the evolution continues. A stopping criterion can be to obtain a maximum fitness value in the population, or maximum number of generations reached, where a generation has passed every time the population is altered and re-evaluated. The population passes through a selection algorithm to determine which

phenotypes are to be selected for mating. Several selection algorithms with different benefits exist in EA (Floreano & Mattiussi, 2008a, pp.23-26). Pairs of selected phenotypes are set up for mating where their genotypes are extracted, and crossover and mutation algorithms are performed to generate offspring. The newly created population is evaluated with the same initial fitness function. If still no stopping criterion is met, the evolutionary loop continues as previously described.

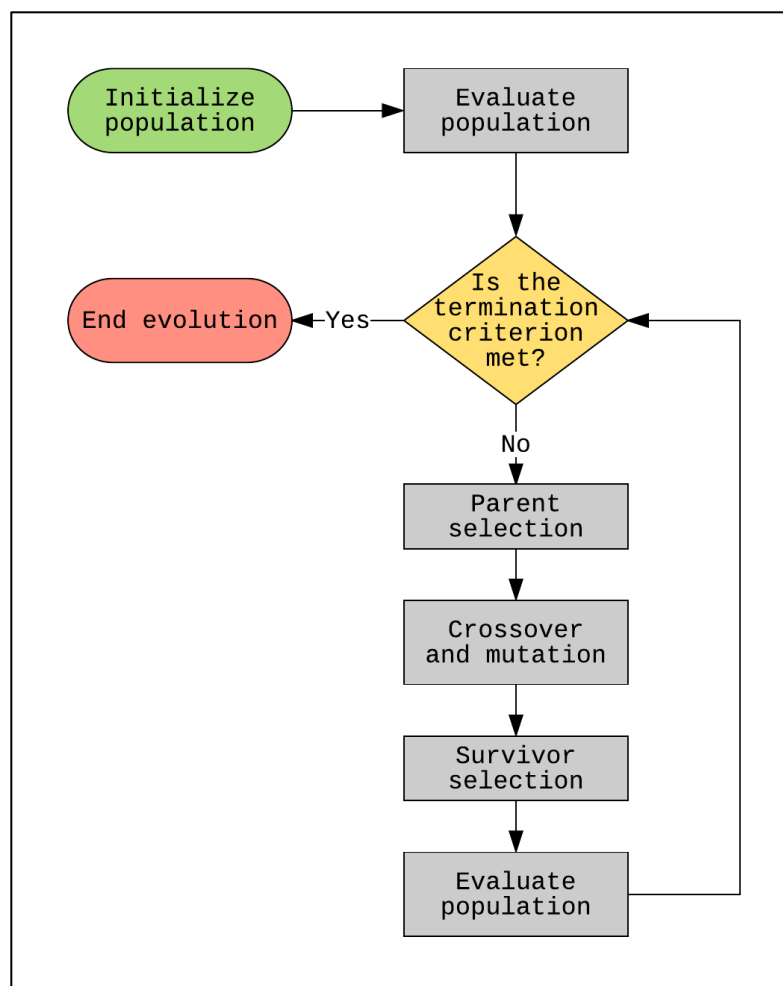


Figure 2. Evolutionary loop flowchart

There are several types of evolutionary algorithms. The main difference is the choice of genetic representation and operators. The choice of the EA type depends on the properties of the problem to be solved. Some of the most common types are genetic programming, genetic algorithms, evolutionary programming, evolutionary strategies, and island models. Genetic programming uses

a tree representation of the genotype. Genetic algorithms operate on binary representations of the genotype such as bit-strings. Evolutionary programming operates directly on parameters that define the phenotype. Evolutionary strategies are very similar to evolutionary programming, but use additional genetic encoding for mutation parameters that is evolved along with the phenotypes. Island models evolve separate populations in parallel and exchange genetic information among the populations in-between generations. Information on several other types of evolutionary algorithms among a deeper explanation of said types can be found in *Bio-Inspired Artificial Intelligence* by Floreano and Mattiussi (2008a).

Designing the fitness function can be the greatest challenge in some EA problems. Evaluating aesthetics and art is a subjective process, hence a well performing general fitness function for art is absent. Instead of writing functions that find subjectively good looking patterns in image art, often interactive search methods are used. Interactive evolutionary algorithms are used to address subjective evaluation of solutions. Humans must evaluate solutions through their subjective judgement (Eiben & Smith, 2015), that the algorithm can use to generate offspring, by setting the fitness value of each solution, or by selecting phenotypes to mate. Since a human must interact with the evolutionary loop, the process of evolution can be substantially slower compared to an automatic fitness evaluation.

Lehman and Stanley (2010) claim that though based on abstractions of nature, current evolutionary algorithms and artificial life models lack the drive to complexity characteristic of natural evolution. However, a consensus in biology is building that the pressure to maximize fitness may not be responsible for complexity growth in natural evolution (Gould, 1998; Lynch, 2007; Miconi, 2008).

2.5 Software Framework Design Approach

It is important that new R&D (Research and Development) results contribute to existing information and communication technology (ICT) systems (both commercial and R&D). Contributing means interfacing and interacting with existing systems such as reuse of code, interface standards and APIs, and exchange of data and information. In some cases, it is also important to include the produced R&D functionality in existing ICT systems. It requires well-organized and structured software (SW) engineering processes, as e.g. the SW Framework Design Approach that is based on the principles of modular and flexible SW architectures, standard

structure of workflows and work processes, standard design and implementation methodology, and well-defined data and information structures, to mention just a few. By respecting these important SW engineering principles, the relevance and the value of the R&D contribution can significantly increase. This is shortly discussed in this section and clearly reflects the whole design and implementation approach of this whole work.

A SW framework is a reusable design and implementation, that presents a model of an application domain and how this model can be executed consisting of abstract classes or interfaces (Riehle, 2000). SW frameworks are built to increase productivity in a domain by reusing software. A good framework has well-defined boundaries, along which it interacts with clients, and an implementation that is hidden from the outside (Riehle, 2000). SW frameworks help developers avoid costly re-invention of standard software components by implementing common design patterns and factoring out common implementation roles (Schmidt, 1997). SW frameworks provide reusable software components for applications by integrating sets of abstract classes and defining standard ways that instances of these classes collaborate (Johnson, 1997). The recommended approach gives important software engineering benefits:

1. Software code developed in a project can:
 - Use already existing software libraries, tools and other standardized software functional entities.
 - Interface to existing industrial APIs and protocols.
 - Interact with various database and media management solutions.
 - Implement various functionality (modularized) in a flexible way, at relatively low implementation costs.
2. Various functional entities can be tested and evaluated under the same conditions (testing, logging, evaluation functionality, etc.).
3. Software modules can be subject to similar usage workflows, giving the possibility to compare and improve them, etc.

3 RELATED WORK

This chapter continues the theory part by presenting the existing knowledge body relevant for our approach to design and implementation of evolutionary art. The first part deals with individual ideas, theories, techniques and approaches, while the second presents integrated solutions, existing systems, frameworks, tools and platforms. Reading existing research is done to identify candidate approaches, aspects, and results applicable to the research topic of this thesis.

3.1 Computational Art and Aesthetics

Ashmore and Miller (2004) suggest that the main difference between evolutionary art and other search problems is that the fitness of an image is based on something that is very hard to describe or maybe even to understand. The attractiveness of an image is personal and differs among people. There can never be a strict fitness function when assessing the attractiveness of an image. It is the user who provides the fitness of an image in the majority of evolutionary art. With evolutionary art, the search is more exploratory, with divergence and diversity being the key factors (Ashmore & Miller, 2004).

Computational aesthetics is concerned with the development of computational methods to make human-like aesthetic judgements. The main focus is on developing aesthetic measures as functions which compute the aesthetic value of an object (den Heijer & Eiben, 2010). There are numerous computational approaches to aesthetics (Galanter, 2012; den Heijer, 2013).

Cohen (1988) argues that in computational art relatively rich representations may result from sparse information. It requires a knowledge of representation that is part of the expertise of the artist, just as the representation of a body of knowledge within an expert system requires an analogous expertise of the knowledge engineer. Understanding the nature of visual representation requires asking what artists need to know in order to make representational objects: what they need to know, not only about the world, but also about the nature and the strategies of representation itself (Cohen, 1988).

The rest of the Section 3.1 suggests some approaches for handling above-mentioned challenges and complexity, starting with the research community using interactive evolutionary computation,

and complementing them by various technical solutions (as fitness functions, seeding and evolving techniques) in order to reach relevant and convergent evolutionary art.

3.1.1 Interactive Evolutionary Computation

Karl Sims (1991) pioneered the approach to fitness assignment that became known as interactive evolutionary computation (IEC). IEC allows users to guide evolution toward regions of the solution space that match their preferences. Unfortunately, it also puts a significant burden on the user, who is “forced” to evaluate thousands of artifacts and compelled to reason based on a local perspective. Sims (1991) used a rich function set, containing image-processing functions (blur, convolution and gradient functions, color-noise, warped-color-noise) as well as simple mathematical functions.

Interactive evolutionary computing (IEC) dominated in the 1990’s the evolutionary art field (Todd & Latham, 1992). Lewis (2008) has cataloged a large number of projects with nearly 200 citations in evolutionary visual art. The vast majority of the systems noted are interactive using some form of case-by-case human judgment.

Galanter (2010) suggests aesthetic judgments made by people as alternatives to an automated fitness function for EA. Interactive evolutionary computation (IEC) uses human feedback to evaluate content; essentially, IEC human users decide which individuals breed and which ones die. As surveyed by Takagi (2001), IEC has been applied to several domains including games, industrial design and data mining.

McCormack (2005) mentions some important moments and challenges when dealing with creativity and evolutionary algorithms (EA), in particular the problem of evaluating the quality of individuals within the population when aesthetic concepts are considered. Creativity and aesthetic principles are not clearly understood, which keeps from accurately defining a function that can properly measure it. Researchers soon rely on humans to perform that specific task, thus defining Interactive Evolutionary Algorithms (IEA): standard EA whose fitness evaluations are performed by human beings in an interactive fashion (McCormack (2005)).

Frade et al. (2010) mentions human fatigue caused by repeated interaction and tries to address it. Several steps have been taken to lessen user fatigue by IEC methods, such as limiting the rating levels of fitness, predicting fitness from user rankings (Liapis et al., 2013), or showing a subset of

the evolving population to the user. Another potential solution to the problem of user fatigue is seeding the initial population of IEC with meaningful, high-quality individuals.

Interactive evolution (IE) incorporates human preferences into the selective pressure that guides the search (Takagi, 2001; Semet, 2002). Specifically, IE poses an open-ended evolutionary search, where the objective function of traditional EA is replaced by a subjective evaluation carried out by a human user of the system.

In fact, some of the earliest EA were open-ended interactive systems, such as the well-known Biomorphs program (Dawkins, 1996). For instance, IE has been combined with the recently proposed Novelty Search algorithm (Woolley & Stanley, 2012), to promote and exploit these properties.

Secretan et al. (2011) and Clune and Lipson (2011) use web-based IEA to evolve artistic artifacts using a generative encoding, compositional pattern producing networks. In both cases, user (connected clients) collaboration is encouraged. Both works offer webpages (see Picbreeder.org² and EndlessForms.com³), where users can select or create random individuals and evolve lineages of artifacts based on their preferences. In these systems, a user can take a previously evolved artifact and continue the search process himself, building upon a previous evolutionary design in a sequential manner. Therefore, evolved artifacts can be the product of a collaborative search process. Another feature is that evolved artifacts can be rated by users, and since users can create individual accounts, the ratings provide a way to rank users, or to select previously evolved artifacts based on the particular style of each user.

Several authors discuss challenges related to the interactive evolutionary art, e.g.:

- Having an artist or other judge “in the loop” becomes the rate-limiting step of the iterative process. This is sometimes called “the fitness bottleneck”. Human judgment is also limited by fatigue (Galanter, 2010).
- Todd & Werner (1999) emphasize that EA systems can produce new populations orders of magnitude faster than a human can score them. The practical result is that Interactive

² <http://picbreeder.org/> (accessed 23.02.2017)

³ <http://endlessforms.com/> (accessed 23.02.2017)

Evolutionary Computing (IEC) systems typically suffer from relatively small populations and few generations.

- Over time user choices will become less consistent, and skew towards novelty for its own sake rather than quality (Galanter, 2010).
- Takagi, and Yuan and Gong (2001; Yuan & Gong, 2008), suggest “crowdsourcing” based evaluation to fight the fitness bottleneck and fatigue problems.

Galanter (2010) stresses that computational aesthetic evaluation (CAE) is a non-trivial unsolved problem, and gives numerous examples of attempts to solve it, e.g.: Saunders and Gero (2004), and Greenfield (2006), Jaskowski et al. (2007), Fornari et al. (2007), and Ciesielski et al. (2007), McDermott et al. (2005), (Birkhoff, 1933; Machado & Cardoso, 1998), Machado et al. (2008), and Gedeon (2007).

Galanter (2010) and Whitelaw (2004) discuss the notion of meta-creation, referring to the role of the artist shifting from the creation of artifacts to the creation of processes that in turn create artifacts.

Machado et al. (2007), Baluja et al. (1994) and Kowaliw et al. (2009) report on efforts to overcome the limitations of IEC, by developing automated fitness assignment schemes, including the use of hardwired fitness functions, machine-learning approaches, the pursuit of novelty etc. suggest automated fitness assignment as an effective way to fight user fatigue.

Machado et al. (2016) explores the impact of these changes on the sense of authorship, highlighting the range of imagery that can be produced by the system. They claim that to some extent, the solution defeats the purpose, because users are no longer able to express themselves. They act only as curators of the works produced by the automated system.

Significant for this thesis are the analyses of Machado et al. (2007), Baluja et al. (1994) and Kowaliw et al. (2009) that present various computational techniques to overcome the limitations of the IEC. This work will try to make a compromise by using the results of both "worlds" (IEC and automated computing based on fine-tuned fitness functions and other techniques to influence and steer evolutionary algorithms). This complementarity can possibly offer both promising artistic results and convergent algorithms.

3.1.2 Aesthetics and Fitness Functions

Many researchers have addressed the relation between aesthetics and fitness functions, experimented with different fitness functions and combined them with various implementation approaches. Some of those presented below will be tested and evaluated also in this work.

Galanter (2010) pointed out the lack of robust fitness functions for the problems related to computational aesthetic evaluation and the need for art theory around evolutionary and generative art. Ashmore and Miller (2004) explain that choosing a fitness function set is very time consuming. It is important that there are enough functions to create complex images but not so many that there are functions that have a negative effect on image fitness. Such functions make traversal through the search space very laborious. Picking functions is difficult as it is very hard to determine what effect any given function will have on an image. Ashmore and Miller (2004) tested and omitted many functions from the final set because they either had no noticeable effect or created too much “noise” in the search space.

Den Heijer & Eiben (2010) investigated and compared four aesthetic measures within the context of evolutionary art:

- Machado and Cardoso (1998) — relation between image complexity and processing complexity.
- Ross and Ralph (2006) — observation that many fine art paintings exhibit functions over color gradients that conform to a normal or bell curve distribution.
- Fractal dimension (Spehar et al., 2003) — the aesthetic preference of people for various fractal types. Images with a higher dimension were considered complex, and images with a lower dimension were considered uninteresting.
- Combined weighted sum by aforementioned three aesthetic measures.

Den Heijer & Eiben (2010) concluded that the use of different aesthetic measures clearly results in different “styles” of evolutionary art. They found differences in variety of the output of the four aesthetic measures. Some aesthetic measures have little flexibility. They suggest investigating some other measures, e.g. the pattern measure (Klinger & Salingaros, 2000), or aesthetic measure based on information theory (Rigau Vilalta et al., 2008).

Johnson (2016) presents a taxonomy of the ways in which fitness is used in EA art and music systems, with two dimensions: what the fitness function is applied to, and the basis by which the function is constructed. Johnson (2016) classifies a large collection of research using this taxonomy. Some papers mentioned in this reference fell into more than one category. As an example, the work by Li and Hu (2010) uses notions of aesthetic measures, which are tuned by interaction with a user, which are in turn proposed as artificial critics for future works.

This thesis will combine some of the IEC techniques (with focus on handling the user fatigue and performance bottlenecks) with fitness function techniques based on direct and indirect measures of user preferences. Direct techniques require the users to specify their preferences, while the indirect techniques extract the preferences from the chosen/preferred art samples. This will be detailed in Chapters 4 and 5.

3.1.3 Aesthetics and Information Theory

Informational theories of aesthetics have applied several information theory measures, with varying success and limitations:

- Entropy in measuring order and complexity of stimuli in relation to their aesthetic value (Majidal-Rifaie, 2016; Javid et al., 2015).
- Mathematical aesthetic measure of aesthetic quality as indirect relation to the degree of order and in reverse relation to the complexity of an object (Birkhoff, 1933). The validity of Birkhoff's model, and his definition of order and complexity, have been challenged by empirical studies (Wilson, 1939).
- Information gain (Andrienko et al. (2000), Bates and Shepard (1993), and Wackerbauer et al. (1994)).
- The application of entropy to quantify order and complexity in Birkhoff's (1933) formula by adapting statistical measure of information in aesthetic objects (Moles (1968), Bense (1969) and Arnheim (1969)).
- A model based on Birkhoff's (1933) approach as the ratio of image complexity to processing complexity (Machado and Cardoso (1998)).

- Adapting Bense’s (1969) informational aesthetics to different approaches of the concepts of order and complexity in an image (Rigau Vilalta et al., 2008; Rigau et al., 2007). Three measures have been applied: Kolmogorov complexity (Vitányi, 2008), Shannon entropy (for RGB channels), Zurek’s physical entropy (Zurek, 1989).

These approaches are considered as interesting but not the highest priority of this research. However, information-based measures might participate in some hybrid fitness functions, offering the information perspective on the image aesthetic quality.

3.1.4 Future Possibilities for Fitness Computation

This section discusses the features of artworks, or the art creating process, which have not been significantly considered, in the existing research work on the evolutionary art. Johnson (2012) suggests a number of topics that could provide a meaningful basis for fitness computation in evolutionary art:

- **Memory** — The members of previous generations should have influence on the current population; or using the memory of the entire process-so-far to influence decisions in the current stage of the artistic/design process. For example, a component that does not fit with the current prototypes might be revisited at a later stage when an opportunity to fit it in with a later version of the overall design occurs (Johnson, 2012).
- **Scaffolding** — This concept of giving a fitness value to components by virtue of their role in a network rather than by their value in isolation (Nagel et al., 2011; Kötter & Berthold, 2012), and the idea of a fitness measure based on what a population member connotes as well as what it denotes has been explored (Johnson, 2012)).
- **Engagement with the outside world** — Connotation of Web search — a coherence regarded as being artistically valuable — the piece is about something, but without this “something” being represented directly (Johnson, 2012).

3.1.5 Evolving and Seeding Techniques

Some research groups have tried complementing the fitness function techniques with other ways of influencing the evolutionary algorithms. Two efforts are interesting: (1) improving the image evolving techniques, (2) using the evolutionary seeding approaches. They are discussed in this section.

Image evolving techniques

Re et al. (2016) claim that the production of interesting images that represent precise, recognizable patterns, is still lacking. They explain that one of the main concerns when developing these applications, is how to represent a solution, that is, in this case, an image. Woolley and Stanley (2011) have shown that evolving a target image can be very hard. Contrarily to the case of abstract images, in which the objective is to evolve “something” that has to satisfy some global criteria of “beauty”, but does not have to necessarily have precise characteristics (like for instance particular figures or shapes), with figurative images, the goal is to reproduce a precise object or pattern (not necessarily representing a real object). Woolley and Stanley (2011) claim that the used representation has a major impact on the evolution of images (including performance). Using representations with different properties could make the search easier or harder. Given the hardness of this kind of application, it would be desirable to have representations that have high locality (Galván-López et al., 2011; Galvan et al., 2013). Re et al. (2016) state that a representation has high locality if small changes to genotype correspond to small changes to phenotype — and thus representations that can help us to define smooth fitness landscapes (Stadler, 2002).

Image seeding techniques

Despite the skepticism about the effect of the seeding on the evolutionary algorithms and strategies, e.g. Eiben and Smith (2015), several research groups have attempted to use them, and some ideas are worth trying:

- Mills (2016) mutated progressively the genotypic functions from a priori aesthetically evolved images and temporally sequenced their phenotypes to produce animated versions.
- Several attempts at seeding interactive evolution have been made, e.g.:
 - Seeding IEC, by starting from previous users’ creations, e.g. in saved, previously evolved images of Picbreeder (Secretan et al., 2011).
 - MaestroGenesis indirectly seeds evolution by using a human-authored sound file as a scaffold to generate musical accompaniments through IEC (Hoover et al., 2011).
 - SentientWorld seeds evolution by using back-propagation to generate game terrain that matches a low-level sketch drawn by the user (Liapis et al., 2013).

Seeding techniques can improve the performance of the evolutionary algorithms if used correctly, and therefore reduce both the search space and execution time. This thesis will experiment with seeding algorithms to reduce search time and to point the evolution in a given user requested direction within the search space.

3.1.6 Novelty in Evolutionary Art

One of the key aspects of the evolutionary art is the novelty. One might discuss the role and use of the image novelty in different application areas (illustrated in Figure 1), and the required level of novelty in those use cases, however, the novelty plays an important role and needs both qualitative and quantitative analysis and evaluation.

The main idea of novelty search is to reward behavioral novelty as a proxy for the stepping stones that a fixed fitness function may fail to reward (Lehman & Stanley, 2008).

In the work of Saunders et al. (2001), the novelty concept is mathematically modelled according to an interestingness definition based on two factors: (i) the ability to create artefacts out of the box (unexpectedness) and (ii) the feasibility of taking an action as a consequence of the discovery (actionability) (Silberschatz & Tuzhilin, 1996).

Saunders and Gero (2001), and Silberschatz and Tuzhilin (1996) suggest computing the novelty as the classification error of an image being associated to the best category. The second step consists in applying the interestingness function, computing the interest of each image given its novelty degree.

Vinhas et al. (2016) explore the effects of introducing novelty search in evolutionary art (they define novelty as phenotypic diversity). Their algorithm combines fitness and novelty metrics to frame image evolution as a multi-objective optimization problem, promoting the creation of images that are both suitable and diverse.

Lehman and Stanley's work (2010) proposes a novelty search algorithm for valuing each image's uniqueness. For each individual, a novelty score is computed, taking into account its neighbors and an archive containing the most novel individuals. Each novelty score computation requires a phenotype comparison, using a dissimilarity metric, between the individual being evaluated and a set of neighbors chosen from the population and the archive. Then, the novelty score for the

individual being evaluated (ind_{eval}) is defined as the average of the dissimilarity scores of the k most similar neighbors, as in:

$$nov(ind_{eval}) = \frac{1}{k} \sum_{j=1}^k dissim(ind_{eval}, ind_j) \quad (1)$$

where *dissim* denotes the chosen dissimilarity metric, and j the j -th most similar individual (ind_j) when compared to ind_{eval} .

The novelty of a newly generated individual is computed with respect to the behaviors (i.e. not the genotypes) of an archive of past individuals and the current population. The aim is to characterize how far away the new individual is from the rest of the population and its predecessors in behavior space, i.e. the space of unique behaviors. A good metric should thus compute the sparseness at any point in the behavior space. Areas with denser clusters of visited points are less novel and therefore rewarded less; candidates from more sparse regions of this behavioral search space then receive higher novelty scores (Lehman & Stanley, 2010).

In part to address this problem, unlike typical evolutionary computation (EC) models, an approach in neuro-evolution (i.e. evolving artificial neural networks) called novelty search does not reward progress towards an ultimate objective; instead evolution in novelty search is open-ended, like open-ended evolutionary models in artificial life (Adami et al., 2000; Channon & Damper, 2000; Kampis & Gulyás, 2008; Standish, 2003).

Silberschatz and Tuzhilin (1996) suggest that a definition of interestingness can be either objective or subjective: objective interestingness uses relationships found entirely within the object considered interesting, while subjective interestingness compares properties of the object with beliefs of a user to determine interest.

The subjective measure of interestingness uses a measure of unexpectedness, or novelty, to select which of the artworks available at an instance in time should be used to continue its search for further novelty and in its attempts to get itself recognized as creative (Silberschatz & Tuzhilin, 1996).

Interestingness in an artwork is calculated using an approximation to a well-known arousal response curve from studies of animals and humans to various forms of arousal. The arousal response curve is called the Wundt curve (Saunders & Gero, 2001). An approximation to the Wundt curve is described by Berlyne (1971). Berlyne also refers to the Wundt curve as a “hedonic function”, in reference to the pleasure/pain response that is often associated with responses to arousing stimuli.

Saunders and Gero (2001) present a computational model of creativity that attempts to capture the search for novelty. Their computational model consists of multiple novelty seeking agents that can assess the interestingness of artworks. The agents can communicate to particularly interesting artworks to others. Agents can also communicate to reward other agents for finding interesting artworks.

This research is planning to combine the use of:

- the distance functions, which measure the difference among the aesthetic properties of computer generated and user-preferred images, and
- the dissimilarity-based novelty estimators.

Both these measures play an important role in aesthetic image evaluation.

3.2 Systems, Frameworks, Projects and Tools

This section is dedicated to implemented systems, frameworks, project and tools. These works are implementation oriented and their experience and practical approaches influence our software framework. Several projects attempted to produce frameworks and tools for Evolutionary Art, and some of them will be mentioned.

Trujillo et al. (2013) and García-Valdez et al. (2013) present a collaborative-interactive evolutionary algorithm (C-IEA) that evolves artistic animations and is executed on the Web. They have developed an open source framework for Web and Cloud-based C-IEA systems, using current web standards and libraries for mobile devices, called EvoSpace-Interactive. EvoSpace provides a central repository for the evolving population and remote clients (EvoWorkers) that interact with the system to perform fitness evaluation using an interactive approach. Fitness evaluation and genetic variations are carried out asynchronously over a distributed evolutionary process.

Zhang et al. (2015) present DrawCompileEvolve, a web-based drawing tool which allows users to draw simple primitive shapes, group them together or define patterns in their groupings (e.g. symmetry, repetition). A user's vector drawing is then compiled into an indirectly encoded genetic representation, which can be evolved interactively, allowing the user to change the image's colors, patterns and ultimately transform it. The human artist has direct control while drawing the initial seed of an evolutionary run and indirect control while interactively evolving it, thus making DrawCompileEvolve a mixed-initiative art tool. Zhang et al. (2015) reason that in evolutionary art and music, interactive evolutionary computation is often the method of choice for exploring the search spaces of such highly subjective domains. The particular imagination, preference and real-world experience of a human user can guide search in interesting, unexpected and meaningful ways (Zhang et al., 2015).

Dumoulin et al. (2016) investigate the construction of a single, scalable deep network that can capture the artistic style of a diversity of paintings. They claim that their model permits a user to explore new painting styles by arbitrarily combining the styles learned from individual paintings.

Ashmore and Miller (2004) have also used Evolutionary Computation Techniques to evolve a wide variety of aesthetically pleasing images using Cartesian Genetic Programming (CGP). They (Ashmore & Miller, 2004) discussed the challenges that arise from employing a fitness function based on aesthetics, and the benefits that CGP can provide.

Ashmore and Miller demonstrated that by the SW that places a focus on providing the user with efficient control over the evolutionary process. Several 'non-user' fitness functions that assess the phenotypes and genotypes of the chromosomes were also employed with varying success. To improve these results, methods of maintaining diversity within the population that take advantage of the neutrality of CGP were implemented and tested.

Machado et al. (2016) report on Photogrowth, a creativity support tool for the creation of non-photorealistic renderings of images. They propose a meta-level interactive art system in which users express their artistic intentions through the design of a fitness function. Their (Machado et al., 2016) idea is to take users out of the evolutionary loop and allow them to explicitly specify their goals, by allowing users to design a fitness function that guides evolution, freeing them from the need to perform individual assessments and allowing them to express aesthetic and artistic

goals by specifying the characteristics they desire among the ones considered by the system (for more information see Machado et al. (2016)).

IMAGENE (Xu et al., 2007) is an interactive image generation tool. It is based on genetic programming (GP) where humans take the part of evaluating which images should remain in evolution and which should be removed from the genetic pool. The genome is represented as a mathematical expression operating on two parameters: the x and y coordinates of the output image. A similar project that also uses interactive evolution is NEvAr (Neural Evolutionary Art) (Machado & Cardoso, 2002). It uses GP with a set of very simple functions. The idea of Machado and Cardoso (2002) is that using complex functions incorporates a bias into the search and makes it more confined. Using only simple functions opens up the search space, but unfortunately it also increases the time taken to breed 'good' images. The training of artificial neural networks required large amounts of training data to perform well. Nevertheless, the system could generate some appealing images without user involvement.

Painting Ants is a project where Monmarché et al. (2008) utilized ant-like agents with individual behavior to generate images, by depositing color where each ant has passed. The Painting Ants project involved interactive evolution to optimize towards user desired images. The resulting images tend to have the same art style, namely straight lines and clusters of colors.

The Painting Fool (Colton, 2012) creates art by simulating natural painting strokes of different types through parameterization. This feature allows for discovery of novel painting styles. Colton experimented with emotions within The Painting Fool as he believes human emotions play an enormous role in visual arts (Colton, 2012, p.20). Having a database with mappings between emotions and different painting styles, and information of what facial features are needed to express a specific emotion, The Painting Fool could alter some of the styles to try to enhance the given emotion. The Painting Fool team also implemented evolutionary algorithms to expand the abilities to create novel results. Human generated fitness functions worked great for their attempt on creating a city skyline scene, but it made The Painting Fool lack imagination. To avoid human generated fitness functions, they included Colton's HR mathematical discovery system to try to computationally invent a fitness function for scene generation (Colton, 2008). This solution did, however, not work for scenes that required satisfaction of some constraints. To address this, they implemented the scene generation as a constraint satisfaction problem (CSP) in combination of

EA to optimize towards the number of satisfied constraints. Colton states that The Painting Fool needs to construct scenes for a purpose to be considered creative (Colton, 2012, p.29). To create an artefact that contains a message to the audience, The Painting Fool uses the Internet to fetch source materials.

Sågawave (Bredesen et al., 2016) was a project created by a group of students during the Experts in Teamwork (EiT) 2016 village “Computational Creativity” at NTNU, which focused on creating images from songs. The project was run through a web interface and creates images on the fly. Bredesen, et al. used Spotify API to fetch songs, Web Audio API to analyze songs, and React front end library to draw images. Song analysis was performed using the Web Audio API utilizing FFT. This API returns an array where indices correspond to specific frequencies and the value at a given index is the amplitude of the given frequency. In addition, number of beats per minute (BPM) was fetched. These values are mapped to different colors and shapes which are drawn onto a canvas. Frequency values are to determine where on the canvas to draw different shapes and how many shapes to draw. As the image is generated while the music is playing, objects are drawn from left to right as the song progresses. Amplitude values are used to select colors for shapes. Following a color spectrum that goes from purple to red (similar to the rainbow), low amplitude values correspond to purple and blue colors, midrange values to green and yellow, and high values to orange and red. Lastly Bredesen, et al. map BPM to weighting of colors and whether to draw sharp edged objects or not. Weighting of colors is available for the system to favor some colors depending on the rhythm, such that slow songs mostly use calm colors like blue and green, while fast songs move towards reddish colors. Due to some randomness in the drawing the same input creates unique images with similar features.

Another project from a different EiT group generated graphical plants with music as input. Growdio (Barkhall et al., 2016) generated graphical trees as plants using custom algorithms, where music decided how the tree should grow. On the opposite side, another group generated music with basis in visual input (Sandve et al., 2016). Sandve et al. used clustering algorithms to extract emotions from a digital image, and then used a probability-driven algorithm, guided by common music theory to generate compositions that reflect the extracted emotion.

“WordPainter” (Heggedal et al., 2016) is a project that generates images based on some input text. Heggedal et al. used an artificial neural network (ANN) to generate images, using similar methods

as Mordvintsev et al. (2015). The input text is processed using text analysis and sentiment analysis, then the ANN is trained to create new models based on the analysis. This ANN can then be used to generate new images.

“Pixel-ate.it” (Østdahl et al., 2016) is a project that focused on creating artwork based on a selected image, where the goal was to create a tool that would be useful and inspirational for the user, aiding human creativity. The images created using “Pixel-ate.it” are similar to results from Painting Ants (Monmarché et al., 2008), where the input image is altered to create something new.

This thesis will build upon several ideas and experiences the research overview presented above. Multiple projects utilize interactive evolution to surpass problems involved with automatic fitness functions. On the other hand, some projects also experiment with automatic fitness functions to remove implications of humans thus reducing search time and user fatigue. This project will experiment with hybrid solutions to utilize positive sides of each solution. A combination of the support for user involvement seen in these projects can be combined with some of the implementations of automatic fitness functions. Involvement of neural networks to either learn a specific artistic style or to try to capture the essence of beauty is not favorable in this project, but the idea to learn user preferences can be incorporated into the evolutionary algorithms. Using gained experience within the evolutionary search can improve performance and target a specific part of the search space. This is closely related to seeding. Colton’s work on The Painting Fool (2012) is great inspiration on design of painting tools and the effect of non-photorealistic images, among his view of emotions in images and people’s reactions towards it. The Painting Fool has several great solutions on computer generated images that can be useful for this project. Sågawave (Bredesen et al., 2016) elegantly extracts information on musical pieces from Spotify API, which is very useful for this project. This project will build upon this idea and extract more information from Spotify in addition to having other libraries for music and image analysis. Several other projects contribute in the brainstorming of this project by giving insight of the performance of different solutions. This work learned from the great experiences of the above-mentioned projects. The following software design and implementation principles deserve particular attention: standardization of interfaces, software framework approach, flexibility and modularity, reconfigurability and interaction with existing solutions. They will be further elaborated in Chapter 5.

4 ARCHITECTURE AND DESIGN

This chapter discusses the architecture and design of the system for Music-to-Image transformation. Evolutionary Algorithms (EA) will be applied to this transformation, which implies interpreting the numerical definition and the setup of the EA to "the music-to-image problem", and applying the EA techniques and processing phases to this art-oriented application type. Section 4.1 presents mapping of music and images to features and related parameters, while Section 4.2 elaborates on how these mapping techniques are employed in EA algorithms. In addition, this section gives a high-level design perspective on the crucial EA elements, such as genotype and phenotype structure, fitness function design, selection strategies, and crossover and mutation techniques.

4.1 Mapping of Music and Images

This section describes how the framework can operate on music and image parameters. Multiple analysis modules exist that can operate on both music files and music metadata. Modules operating on metadata are preferred in this work.

4.1.1 Music Features and Related Parameters

The evolutionary algorithms use the music indirectly, i.e. via various music parameters / descriptors, also denoted as metadata. For this, various already existing application programming interfaces can be used, e.g. Spotify API. Spotify audio features objects, obtained using the API, contain several variables that describe how the song in question is related to them (full description of parameters in Appendix A). In addition, various custom extracted features can be used (obtained by analyzing raw music files), as:

1. Amplitude variances over frequencies, and over time.
2. Mean amplitude variance over frequencies, and over time.
3. Mean amplitudes over frequencies, and over time.
4. Mean frequencies weighted by amplitudes.
5. Standard deviation over frequencies weighted by amplitudes.

This project will mainly focus on music features obtained from the Spotify API. Small experiments with Spotify API and audio file analysis showed that audio file analysis can be time consuming both at the implementation stage and execution stage. Rather than running audio analysis on raw data, which requires the client to possess the music file, Spotify API can be used to obtain audio metadata that contains more information than the local functions currently can return.

4.1.2 Image Features and Related Parameters

Various image parameters, obtained either in the preprocessing phase (i.e. while generating the image) or at the postprocessing phase (image analysis of the finished image), can be used to characterize the images. Preprocessed image parameters can be obtained through log files, where each painting tool has a full description of its parameters. This set of parametrized tools gives a direct description of how the resulting image will look. However, for another type of image analysis, such as pattern recognition or search for other hidden image features, postprocessing is required. This project has one postprocessing function implemented, that extracts a color palette from the image using the k-means clustering algorithm. The returned color palette can be closer to the perceived colors in the image, than the original palette used due to color mixing during painting. Some evolutionary art projects use image analysis for evolution (Machado & Cardoso, 1998; Klinger & Salinger, 2000). This framework contains functionality to evolve raw images. This means that the phenotypes in the EA are images, and that the fitness functions directly analyze the images. A small experiment was conducted to investigate the performance impact of evolution of images. The EA optimized images towards filling the canvas with color, such that the ratio of the original canvas background color should decrease. A population of 10 and using a maximum of 10 generations, the evolution used around 20 minutes to complete. This project uses evolution on metadata, and analyzes parameters used to create the end result rather than analyzing the end result itself. The experiments performed in this project do not rely on postprocessing during evolution as it is a highly time-consuming process even for a small population size.

4.2 Evolutionary Algorithms — Numerical Definition and Setup

In this section, the numerical definition, setup and main design elements will be presented.

Figure 3 illustrates the workflow of the painting algorithm. The phenotype in this work will be called "the Image Artist" and it produces a set of images, based on the genotype (represented by

the feature mapping table). For each input song, a corresponding image is generated by using this feature mapping table. It is a metadata set (with key-value pairs) used to create painting instructions.

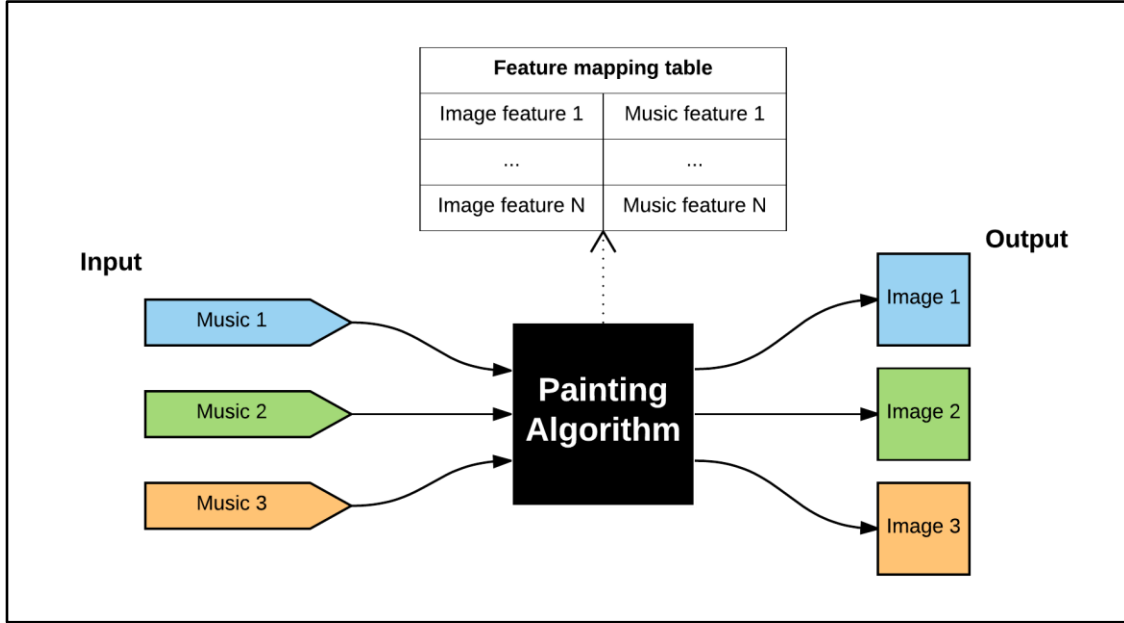


Figure 3. Illustration of the Painting Algorithm's work

The music file m is a member of a set of music files M , on which the painting algorithm, “the Image Artist”, will operate. It is described (represented) by a set of the music parameters p_i^m , with indices i from $1 \dots N$.

$$m = \{p_1^m, \dots, p_i^m, \dots, p_N^m\} \in M \quad (2)$$

An image r is a member of a set of images R and is the result of a painting process. An image r is described by a set of image parameters p_j^r , with indices j from $1 \dots M$.

$$r = \{p_1^r, \dots, p_j^r, \dots, p_M^r\} \in R \quad (3)$$

To emphasize, both the input music file m and the resulting image r are represented (approximated) by corresponding parameter sets (stored in mapping tables t).

A mapping table t is a member of a set of mapping tables T , that the painting algorithm utilizes. A mapping table t is a function used for feature mapping, by using image parameters as keys (K) and music parameters as values (V).

$$t = f : K \rightarrow V \quad (4)$$

An image r is created by adding functions of the music parameters. Such a function of the music parameters can be denoted as a painting tool. "The Image Artist" uses several tools ($f_1 \dots f_k$) to create an image, each of them being a function of the mentioned music parameters, i.e.

$$r = f_1(p_1^m, \dots, p_i^m, \dots, p_N^m) + \dots + f_k(p_1^m, \dots, p_i^m, \dots, p_N^m) \quad (5)$$

Images are evaluated by a fitness function Φ , which uses its sub functions $\phi_1, \dots, \phi_k, \dots, \phi_K$ that estimate various image criteria, with indices k from $1 \dots K$.

$$\begin{aligned} \Phi(r) = & \phi_1(p_1^r, \dots, p_j^r, \dots, p_M^r) + \phi_k(p_1^r, \dots, p_j^r, \dots, p_M^r) \\ & + \phi_K(p_1^r, \dots, p_j^r, \dots, p_M^r) \end{aligned} \quad (6)$$

Fitness functions can have many sub functions, depending on the evaluation criteria and given goals of the image creation process. Examples of criteria can be user-defined aesthetical fitness, novelty function, their combination, etc.

EA pseudo code for Music-to-Image creation might be as outlined in Table 1:

OPERATION	COMMENT
BEGIN EVOLUTION	
Initialize population with a set of mapping tables T	Set $T(s)$ is the set of mapping functions, $t = f : K \rightarrow V$ where s is current generation
Create initial image set R using T and M	Set $R(s)$ is the set of images a , represented by the parameter sets $r = r(p_1^r, \dots, p_j^r, \dots, p_M^r)$
Evaluate music-to-image set	$\Phi(r) = \phi_1(p_1^r, \dots, p_j^r, \dots, p_M^r)$ $+ \phi_k(p_1^r, \dots, p_j^r, \dots, p_M^r)$ $+ \phi_K(p_1^r, \dots, p_j^r, \dots, p_M^r)$
WHILE termination criterion is not met DO	
1. Select parents	$T(s) \rightarrow U(s)$, where $U(s) \subseteq T(s)$ Create a new set (parents) which a subset of the population
2. Reproduce	$U(s) \rightarrow V(s)$ Create a new set (children) which is the result of reproduction
3. Mutate the resulting offspring	$V(s) \rightarrow W(s)$ Create a new set which is the result of mutation
4. Evaluate all $a \in W(s)$	Evaluate the new set using the fitness functions $\Phi(r) = \phi_1(p_1^r, \dots, p_j^r, \dots, p_M^r)$ $+ \phi_k(p_1^r, \dots, p_j^r, \dots, p_M^r)$ $+ \phi_K(p_1^r, \dots, p_j^r, \dots, p_M^r)$
5. Select individuals for the next generation	$T(s + 1) \subseteq W(s)$ The new population for the next generation is a subset of the mutated set
END WHILE	
END EVOLUTION	

Table 1. EA pseudo code for Music-to-Image creation.

4.2.1 Genotype

The genotype will be a mapping table operating on meta data from both images and music. Image meta data consists of parameters for painting tools that are used to construct an image recipe, while the music metadata is obtained/extracted through Spotify API and consists of music parameters as tempo, energy, loudness, etc. (The full metadata set is given in Appendix A).

Following is an example of a mapping table with example metadata:

Image Feature (parameter for painting tool)	Music Feature and numerical intervals (music metadata variable and scaling intervals)
Brush strokes	Tempo, [20, 300]
Base color	Energy, [270, 0]

Table 2. Example of mapping between tool parameters and music feature variables.

More details will be provided in Section 5.3.

4.2.2 Phenotype

The phenotype is an artist object (the so called "The Image Artist") that utilizes the mapping table representing the genotype (exemplified by Table 2). The task of the phenotype is to create an image recipe that can be used to paint the final image. This is done by utilizing the mapping table in the genotype. For each tool parameter (key in the mapping table), the associated value is fetched. This value object contains a music parameter and an output interval. The numerical value of the music parameter is used to calculate an output tool parameter value by linearly scaling the music value to the output interval. In this way, an image recipe is created (a set of instructions for the painting tools), which the "The Image Artist" uses to paint the image.

4.2.3 Fitness Functions

The phenotype uses a fitness function to calculate its fitness. Fitness functions in this system are software modules. The fitness functions operate on distance measures towards an optimum, making this evolution a minimization problem. Optimum fitness is reached when the distance between the current genotype and an optimum mapping table is zero. A detailed description and implementation of the mentioned fitness functions is given in Section 5.3.2. This work experiments

with several fitness functions by using the fitness function structure explained in Section 4.2.3.1 as a basis. In total, three fitness functions have been used:

1. Optimizing towards user specified mapping table
2. Novelty combined with user specified mapping table
3. Optimizing towards user preference without knowing the mapping table

They are explained in the following sections.

4.2.3.1 Optimizing towards a user specified mapping table

This fitness function guides the evolution to find a mapping table that is “close” to what the user has specified. The distance between any two mapping tables is the sum of distances between key-value pairs in the mapping tables. Each tool parameter (key) should map to the correct music feature variable (value) and have the correct output interval. The Image Artist uses the output interval to calculate a value for a painting tool parameter.

Given the target interval $T = [t_1, t_2]$ and the current interval $C = [c_1, c_2]$ the distance between these intervals is

$$d(T, C) = |t_1 - c_1| + |t_2 - c_2| \quad (7)$$

For mismatching music variables for a given tool parameter, a penalty factor is calculated. The penalty factor is determined by

$$k(m_1, m_2) = \begin{cases} 1, & m_1 = m_2 \\ K, & m_1 \neq m_2 \end{cases} \quad (8)$$

where m_1 is the target music variable, m_2 the current music variable, and K the is the penalty factor. This penalty is multiplied by the distance between intervals in values, i.e. calculating the distance between map entries is done by multiplying the two sub-functions (7) and (8). The fitness function for the current genotype G is therefore calculated as a sum of contributions from each tool parameter

$$f(G) = \sum_{i=TP_1}^{N_{TP}} (d(T, C)k(m_1, m_2))_i \quad (9)$$

where G is a current genotype and N_{TP} is the number of tool parameters.

To summarize, for each tool parameter the associated value is extracted and used to calculate the distance function d and the penalty function k . They are multiplied and added to the global sum (9). The resulting value of this global sum is the fitness value of a given genotype.

This fitness function is used to optimize to specific requirements provided by the user. With small modifications, some map entries can be evaluated differently. The distance function designed in this way is quite generic, and can be used as a basis for various fitness functions, discussed below.

4.2.3.2 Novelty combined with user specified mapping table

This fitness function is created to introduce a notion of novelty in the evolution. Novelty is introduced by optimizing towards a user suggested mapping table, but ignoring some of the map entries. By ignoring some user suggested map entries, the system can stochastically select how tool parameters are mapped to music parameters and output intervals. It uses the same structure as in Section 4.2.3.1 as a basis, using the same distance and penalty functions. However, the fitness function is not iterating over all possible tool parameters. Some arbitrary tool parameters are not included in the fitness calculation, thus some differences between the user selected mapping table and the mapping table in question will not be visible in the fitness value. This fitness function can:

- Ignore whole entries based on a key (tool parameter)
- Ignore only a parameter mismatch between tool parameter and music parameter
- Ignore differences between output intervals

In this way, both the user suggestions (representing user's aesthetic criteria) and the novelty can be combined, and hopefully provide certain aesthetic qualities.

4.2.3.3 Optimizing towards user preference without knowing the mapping table

This fitness function operates somewhat differently compared to the above-mentioned fitness functions. Instead of having an optimum mapping table to optimize towards, this fitness function uses user description of how the final result should be as a guidance. For instance, the user can provide the system how the final color palette should be, but not provide how the palette should be generated. Thus, the system is missing information of critical parameters, and must find a mapping table that can generate the requested final result. This suits well for the use-case "Artist's Work tool" in Figure 1.

This fitness function uses a different distance measure than the above-mentioned fitness functions. Equation (10) demonstrates a distance between two variables A and B , where A is the user requested result and B is the currently generated result. A and B can have different meanings varying from color palettes to the total number of brush strokes. It depends on the opinion of the user.

$$dist(A, B) = |A - B| \quad (10)$$

The final fitness function is therefore

$$f(G) = \sum_{i=R_1}^{N_R} (dist(A, B))_i \quad (11)$$

where G is a current genotype and N_R is the number of user-specified requests.

This fitness function guides the evolution in search for mapping tables that match requested end results rather than predetermined mapping tables. This way, the evolutionary algorithms can introduce interesting and unexpected mapping tables.

4.2.4 Selection Strategies

The choice of selection strategies is another way to influence the evolution. Poor performing selection strategies can reduce the efficiency of the evolution. According to overview of Floreano and Mattiussi (2008a) there are multiple available techniques used in different implementations of EA, e.g.:

- Proportionate selection
- Tournament selection
- Rank selection
- Sigma Scaling selection

These selection algorithms expect fitness maximization. However, this project uses EA for fitness minimization. Two selection strategies, proportionate selection and tournament selection, have been converted for minimization. These two selection techniques will be tested and analyzed to see how they influence the EA in this application.

4.2.5 Crossover Techniques

Crossover techniques can also change how the EA works, by controlling how two genotypes merge and create a new genotype. Multiple recombination techniques are available according to Floreano and Mattiussi (2008a), and focus will lay on those applicable to this application domain (music-to-image transformations). Two popular techniques such as One-Point crossover and Uniform crossover will be implemented. Some modifications of the algorithm proposal of Floreano and Mattiussi (2008a) must be implemented to support mapping tables as genotypes, rather than binary strings.

4.2.6 Mutation Techniques

Mutation techniques are an essential part of the evolution and are necessary to introduce diversity among the population, and ensure a more complete search in the domain space. Mutation techniques can be modelled as stochastic processes that influence offspring. It is possible to implement them in various ways. Having a mapping table as genotype, a new mutation technique must be implemented such that all parts of the table are mutable. This means that key-value pairs can be altered, and the information within the values can be modified. Below is an illustration of a mutation process:

Given a simple mapping:

Image feature	Music feature
Palette base color	Music Tempo
Number of brush strokes	Music Valence [0, 100]

Could mutate to:

Image feature	Music feature
Palette base color	Music valence
Number of brush strokes	Music Tempo [0, 200]

This concludes the high-level overview of the used EA methods and technologies in this work. It hopefully gives a basis for the discussion of the implementation approach. Further design and implementation details are provided in Chapter 5.

5 IMPLEMENTATION

Considering the software design and implementation strategies this work is using a software framework design and implementation approach. This means that the produced software functionality should be flexible and modular, with well-defined interfaces, algorithms, data and metadata, to mention just a few software engineering principles. This way, the resulting software increases the industrial relevance. It can relatively easily collaborate with existing commercial and open source systems. More details on implementation guidelines and principles are provided in Section 5.1.

Section 5.2 contains a brief explanation of used tools or methods in this project. The Spotify API is the main used external tool. This tool is used to fetch and extract musical features of audio from Spotify song URIs.

Section 5.3 discusses the implementation of the EA functionality, presented in Section 4.2. Section 5.4 focuses on the implementation of the painting algorithm.

5.1 Implementation of the Software Framework

The software architecture produced in this work had requirements addressing modularity and flexibility, well-defined interfaces (also towards external tools (music and image)), structured workflows and algorithms that are easy to change and experiment with. The whole software system was based on good software engineering practices, including logging, testing and experimenting facilities. In addition to that the work had to produce the generic evolutionary algorithm functionality that can be used for various use cases, shown in Figure 1.

The notion of SW framework in this work is wider than the above-listed arguments:

1. It does not include just the produced SW, but also various SW development tools and libraries. In addition to well-designed and tested SW code, they give the SW engineering functionality to edit, create, compile, debug and test the code.
2. This work also tries to standardize the design of algorithms and processing workflows, in order to easily change the functionality (e.g. methods, algorithms, modules), and test/evaluate the differences under relatively similar conditions.

3. Use of standardized logging and testing facilities.
4. Use of standardized SW design patterns, design and implementation rules and styles,
5. Use of standard SW modelling methodologies, e.g. Unified Modelling Language (UML)⁴ (Rumbaugh et al., 2004).

This whole framework is written in Java from scratch with high use of inheritance and polymorphism to enhance the modularity of the software. A custom written interface is used to fetch music features using Spotify API. Graphical parts of the software are done using Java's built in graphical library. Evolutionary Algorithms are implemented from scratch to have full control of the evolution and its strategies.

5.2 External Functionality

External libraries, interfaces, APIs are important, because one of the main requirements of this work was to enable interacting with mainstream commercial and open source interfaces, some of which are discussed below.

Spotify API

Spotify is a Swedish music streaming service that provides access to millions of songs. The provided songs have available previews that are downloadable, and contain audio analysis data that can be obtained through their API. Spotify Web API is an interface which supports many useful functions that provide a lot of information about a set of tracks (Appendix A and B). There are three specific endpoints that are useful for music analysis; tracks, audio-features, and audio-analysis. Every endpoint operates on a Spotify track ID:

Tracks: “Get Spotify catalog information for a single track identified by its unique Spotify ID.”⁵

Audio-features: “Get audio feature information for a single track identified by its unique Spotify ID.”⁶

Audio-analysis: “Get a detailed audio analysis for a single track identified by its unique Spotify ID.”⁷

⁴ <http://www.uml.org/>, retrieved 14.04.2017

⁵ <https://developer.spotify.com/web-api/get-track>, retrieved 14.04.2017

⁶ <https://developer.spotify.com/web-api/get-audio-features>, retrieved 14.04.2017

⁷ <https://developer.spotify.com/web-api/get-audio-analysis>, retrieved 14.04.2017

A Spotify track ID can be obtained through the Spotify client. Spotify requires the user to have an account and register an application which is used to obtain authorization tokens and perform API calls.

5.3 Evolutionary Algorithms

Evolutionary Algorithms (EA) depend on three main parts: the genotype, the phenotype, and the evolutionary loop. The evolutionary loop is further guided by a fitness function. Creating images from music requires a form of mapping between music features and image features. Music features are obtained through Spotify API, while image features need to be created in the act of painting using the available painting tools.

5.3.1 Implementation of the Genotype and Phenotype

Image metadata is implemented as an enumeration of tool parameters. This enumeration tells the painting algorithm how to parameterize each painting tool. For instance, the enumeration `ToolParameter` contains a value called “`StraightBrushStrokes`”, which tells the painting algorithm the amount of straight line brush strokes to paint. Music metadata is an enumeration of feature variables obtained through the Spotify API. A detailed description of this data can be found in Appendix A. These two enumerations are the key elements of the mapping table in the genotype. This relation is indicated in Appendix B and an UML Diagram.

The purpose of the genotype is to create a recipe for the painting algorithm that describes how values from music features are mapped into painting tool parameter values. For instance, musical tempo can be mapped onto amount of brush strokes to paint, such that slow melodies create calm images, while high tempo melodies create chaotic images using lots of paint strokes. The genotype contains a hash table that is responsible for the mapping between features. Enumeration `ToolParameter` is used as the key and the class `GenotypeEntry` is used as the value. `GenotypeEntry` consists of three main variables: `FeatureVariable`, input value range, and output value range. `FeatureVariable` is an enumeration containing all music feature variables available. The input range is the minimum and maximum value the selected feature variable can have, and the output range is the minimum and maximum value the tool parameter can receive. A detailed UML class diagram of this relation is provided in Appendix B.

Key : ToolParameter	Value : GenotypeEntry (FeatureVariable, input value range, output value range)
StraightBrushStrokes	Tempo, [0, 200], [20, 300]
ColorDegree	Energy, [0, 1], [270, 0]

Table 3. Hash table example of mapping between tool parameters and music feature variables.

The hash table is used by the painting algorithm where it looks up the values (GenotypeEntry) mapped for each tool parameter and calculates what the tool parameter will receive as input. Calculation is done by scaling the feature variable value from the input range to the output range using (12).

$$f(x) = \begin{cases} m_1, & x < m_0 \\ M_1, & x > M_0 \\ \frac{x - m_0}{(M_0 - m_0)(M_1 - m_1) + m_1}, & \text{otherwise} \end{cases} \quad (12)$$

where m is min, M is max, subscript 0 denotes input and 1 output

Looking at Table 3, the amount of straight brush strokes is to be calculated based on the tempo value. Tempo has the domain [0, 200], and the output [20, 300]. If tempo has a value of 90, the output is 146, meaning the painting algorithm must draw 146 brush strokes. This calculation process is repeated for each tool parameter that rely on a music variable.

Genetic operators such as crossover and mutation are implemented within the genotype object. Genotype crossover is performed by merging two hash tables (the mapping tables). One-point crossover slices the two tables at an arbitrary index and combines the two parts from each genotype into a new genotype. Uniform crossover iterates over all keys and stochastically selects which value from the two hash tables to duplicate into the new genotype. Mutation is performed by altering the hash table. A key is stochastically selected and the mutation is applied on the associated value (GenotypeEntry). With a given probability the feature variable is altered, such that selected tool parameter (key) is mapped to a different feature variable, or the output range is altered using a given mutation pressure. The mutation pressure in an interval $[-t, t]$, from which a random value in this interval is selected and added to a numeric variable selected for mutation.

5.3.2 Implementation of Fitness Functions

Following are three fitness functions implementations, where two implementations operate on user specified mapping tables, while the last also operate on user descriptions of the expected end result.

5.3.2.1 Optimizing towards user specified mapping table

This implementation is based on the algorithm proposed in Section 4.2.3.1. The fitness value is the difference (arithmetic distance) between a target mapping table (user specified) and the mapping table in the genotype. Figure 4 shows the pseudocode for this fitness function. For each tool parameter (key in the map, see Table 3), map values are extracted from both the target and the genotype. The difference in input and output intervals are added up. If the feature variables mismatch, the sum is multiplied by 2.0, thus giving an extra penalty.

```
function mappingTableDistance(targetTable T, genotypeTable G)
  d = 0 // the accumulated distance
  for each key tp in all available ToolParameters
    // An object in the map contains
    // a variable and an output interval
    d = d + abs(T[tp].outputmin - G[tp].outputmin)
    d = d + abs(T[tp].outputmax - G[tp].outputmax)
    if T[tp].variable = G[tp].variable
      m = 1
    else
      m = 2
    d = d × m
  return d
```

Figure 4. Pseudocode of the fitness function for optimizing towards user specified mapping table

5.3.2.2 Novelty combined with user specified mapping table

This fitness function is implemented based on the description in Section 4.2.3.2. It is a slight modification of the basis fitness function (5.3.2.1) such that EA can explore the search space more freely and find potential novel solutions. The fitness function is essentially the same as in optimizing towards user specified mapping table, but ignores some user specifications. It can ignore values in the intervals, feature variables used, or complete map entries. What gets ignored is a stochastic or deterministic process, based on user input. If the user leaves some parts of the

target mapping table unspecified, the fitness function will let the evolution select these values stochastically. The user can flag parts of the mapping table that the system can explore within. Ignoring user specifications leaves the system to arbitrary select variables and values to use, thus giving it a possibility to introduce novelty in the results. Pseudocode for this fitness function is given in Figure 5.

```

function mappingTableDistance(targetTable T, genotypeTable G)
  sum = 0 // the accumulated distance
  for each key tp in all available ToolParameters
    d = 0
    // An object in the map contains
    // a variable and an output interval
    if T[tp] = null
      continue
    if not ignore_output_min
      d = d + abs(T[tp].outputmin - G[tp].outputmin)
    if not ignore_output_max
      d = d + abs(T[tp].outputmax - G[tp].outputmax)
    if T[tp].variable = null
      sum = sum + d
      continue
    if T[tp].variable = G[tp].variable
      m = 1
    else
      m = 2
    sum = sum + d × m
  return sum

```

Figure 5. Pseudocode of the fitness function novelty combined with user specified mapping table

5.3.2.3 Optimizing towards user preference without knowing the mapping table

This fitness function is measuring how well the user preference is presented in the resulting image by analyzing how the mapping table affects the recipe of the image. Instead of providing user specified mapping table, abstract information is provided along each song, such as base color, number of brush strokes, color deviation, etc. In this case the fitness functions analyzes how the mapping table affects the image in each genotype and compares these results with the provided information for each song. For instance, base color distance is the Euclidean distance (using RGB values as vectors) between user provided base color and generated base color.

This fitness function can introduce unexpected mapping tables that match user preferences but have interesting effects with other music. This fitness function can also be combined with one of the previous mentioned fitness functions, such that concrete user preferences can be combined with abstract preferences.

5.4 Implementation of the Painting Algorithm

This framework is mainly for creating abstract art by using different shapes and simulated brush strokes. It is, however, not limited to abstract art. Having simulated brush strokes allow for creation many art styles. The software itself is not able to draw/paint reproductions of nature, interior and exterior details, reproductions of living creatures, portraits or similar, but this feature can be implemented in the future. The painting tools can be combined using layers, where each tool creates a layer on top of a digital canvas. The tools are also highly parameterized to utilize each tool to its full potential.

Each created tool is designed in such a way that it can be used together with the rest of the SW framework by utilizing inheritance from object oriented programming, and is tested by creating separate images composed of only one tool, and images composed of multiple creation tools.

Geometric Shapes

There are three main geometric shapes implemented: rectangles, ovals, and polygons.

Rectangles (Figure 6) and ovals (Figure 7) both have position and dimension parameters, along with the desired color. In addition, a border can be drawn with desired color and thickness. Colors used in the two images are random with a bias towards white (Figure 6) and blue (Figure 7). Rectangles as shapes in images can be used to segment the image into sections, or to represent rectangle shaped objects such as a building or a box. A combination of multiple rectangles in specific positions on the canvas can be used to create representations of more complex structures such as tables, chairs, stairs, and so on. Rectangles are also useful to represent flat surfaces. Ovals are also useful for the representation of real objects such as wheels, eyes, planets, clouds, and so on. Since ovals have no edges, they can be used to enhance a calm emotion as they have no sudden points or edges. Ovals can be related to a smooth motion or a soft object.

Polygons (Figure 8) are painted using points (coordinates) on the canvas and random colors biased towards red. The number of points to be used, the positions of them, thickness and color of the border, are all parameters which can be set. Polygons can be used to create triangle shapes to circlelike shapes (high number of points following the perimeter of a circle). Having many random points often yields pointy objects that can be related to aggressiveness and anger. Points in a specific order with specific positions can create complex polygons that can remind of many everyday objects.

Brush strokes

Two types of brush strokes are implemented: curved and straight. Curved brush strokes (Figure 10) try to simulate brushes with a circular head, while the straight brush strokes (Figure 9) try to simulate brushes with a flat rectangular head. Simulating brush strokes can help the created images to be deemed as creative since humans can relate the images to results from human artists. Having only these two types of brushes can create endless amounts of images. Using enough strokes and the right parameters can yield reproductions of many famous artworks. This, however, is practically hard to achieve.

Both types of brush strokes are implemented using a high number of regular straight lines tightly packed that all follow the same direction (from start of the stroke, to the end). Every line within the stroke is altered differently as the stroke is painted to give the effect of paint smudging and fading. In the straight brush strokes the color intensity degrades as the stroke is painted, and fades out at the end. In the curved strokes is effect is slightly reduced as it naturally occurs due to the layout of lines. The curved brush has all its painting lines in a 2D normal distribution, while the straight brush has all its lines evenly spread out among its width.

Image Effects

Currently there are three types of image effects implemented: cloud effect, blur effect, and oil effect. The cloud effect creates a monochrome layer of noise that can induce some diversity among equal shapes. Cloud effect affects mostly the colors of an object. Figure 11 shows a sample image with no effects applied. This image is the base for the effect demonstration images that follows. Figure 14 shows a slight blur effect that softly smooths out sharp corners in the image. Figure 15 has a much stronger blur effect where the result is clearly a blurred image. Figure 12 shows a slight oil effect. This effect removes some of the clearly artificial lines resulting from the brush painting

algorithm. Figure 13 creates a much stronger effect where the result image reminds of an artistic effect involving water or oil, rather than computer generated curved lines.

The blur effect is implemented using the Gaussian blur function (Nixon & Aguado, 2008). The Oil effect is implemented based on a guide⁸ for implementation of an image oil effect, and is modified to be compatible with the software framework.

⁸ <http://supercomputingblog.com/graphics/oil-painting-algorithm>

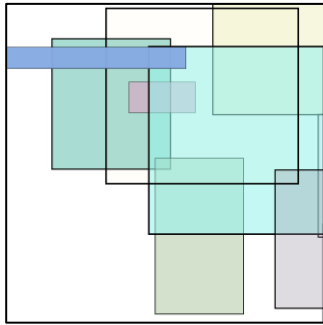


Figure 6. Random rectangles

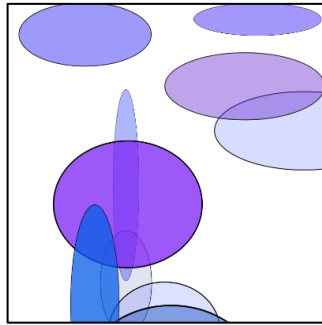


Figure 7. Random ovals

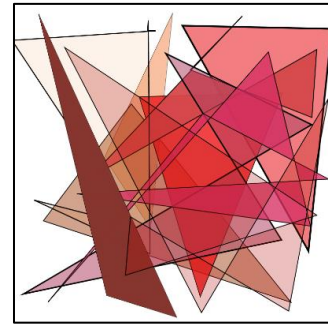


Figure 8. Random polygons

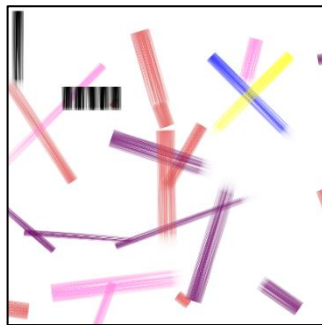


Figure 9. Random lines

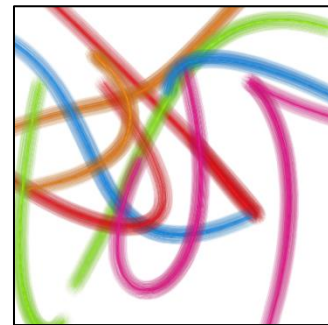


Figure 10. Random curves

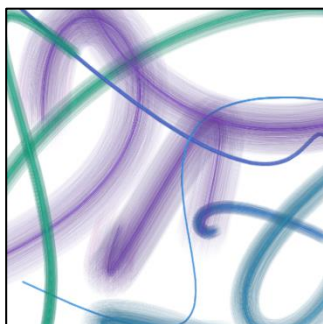


Figure 11. No effects



Figure 12. Slight oil effect

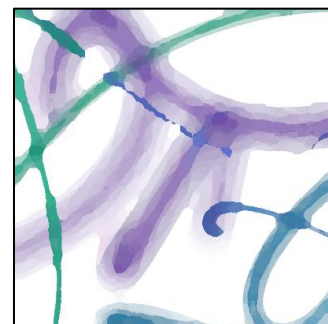


Figure 13. Strong oil effect

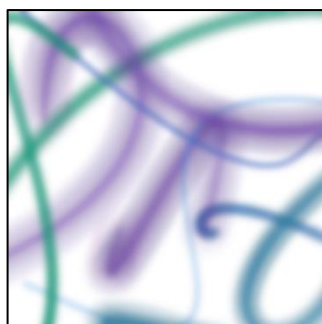


Figure 14. Slight blur

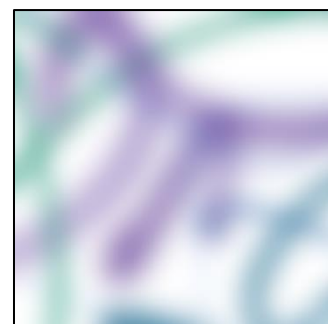


Figure 15. Strong blur

6 EXPERIMENTS AND RESULTS

This chapter presents the experimental setup and results. Section 6.1 presents the experimental setup, which consists of two main parts: system tests and experiments, and trials with user involvement (collection of user preferences by surveys and using them in the EA). Section 6.2 discusses the software implementation tests, while Section 6.3 presents the result of experiments and user involvement trials.

6.1 Experimental Setup

Table 5 and Table 5 list the tests (Tests T.1 – T.3) and experiments (Experiments E.1 – E.7), which are used during the implementation of the system, finetuning of the algorithms, evaluation of the results and finally used to answer the project's research questions. It should be noted that tests and experiments are just a part of the methodology to address and answer the project's research questions. This is thoroughly discussed in Section 6.2, 6.3 and Chapter 7.

List of the implementation tests and EA research experiments	Name of tests (T.x) and experiments (E.x)
SW implementation tests	T.1 – Sound analysis unit tests T.2 – Image creation tests T.3 – Evolutionary Algorithms tests
Research EA Experiments	E.1 – Experiments with fitness functions E.2 – Experiments with seeding techniques E.3 – Experiments with mutation pressure E.4 – Experiments with crossover techniques E.5 – Experiments with selection strategies E.6 – Performance evaluations E.7 – Experiments with user involvement

Table 4. List of the implementation tests and EA research experiments

Main research question	Examples of tests (T.x) and research experiments (E.x) that contributed answering the research questions
1. How can the computer artist system be analyzed and evaluated?	<ul style="list-style-type: none"> - As a SW system (qualitative analysis and SW tests T.1 – T.3) - As an AI system (Research experiments E.1-E.7).
2. Can the system learn about user preferences and user notion of aesthetics, and how?	<ul style="list-style-type: none"> - E.1 – Experiments with fitness function - E.2 – Experiments with seeding techniques - E.7 – Experiments with user involvement
3. How can user feedback be collected and used in the system?	<ul style="list-style-type: none"> - E.1 – Experiments with fitness function - E.2 – Experiments with seeding techniques - E.7 – Experiments with user involvement
4. Can the system create aesthetically pleasing and meaningful images without user involvement during evolution?	<ul style="list-style-type: none"> - Without any user involvement, only the fitness function can guide the search towards possibly aesthetically pleasing and meaningful images (E.1, E.7). - It is however possible to collect information for the system to use during the evolution without involving humans (E.2, E.7).

Table 5. Describing what tests and research experiments are used to address each research question

6.2 Software Implementation Tests

Various software tests are performed to ensure that the software works as intended and that is stable enough to perform valid experiments. In addition, the software must meet given requirements for a software framework.

Software tests have been executed as a combination of individual unit tests (focusing on the module functionality), and integrative framework tests, where the execution of the whole system workflow has been followed. Both test types have been confirmed by the log-file outputs, the example of which are shown in Figure 17. For all the changes in the algorithm and further experiments a series of regression tests have been carried out, ensuring that the changed functionality did not negatively affect the already stable algorithm parts and SW framework modules.

6.2.1 T.1 – Sound analysis unit tests

Sound analysis is performed by Spotify functionality, accessed through their API, in addition to some custom implemented analysis functions. It is necessary to verify that Spotify API delivers correct data for given music ID's, and that the custom functions perform as intended. These tests will be performed by some unit tests to ensure data integrity of Spotify analysis, and by analyzing software workflow through log files. Spotify API integration is tested using unit tests that compare expected results to actual results. From a set of sample songs each test passes as expected. The Spotify test suite verifies data integrity by comparing the expected results to the obtained values. Log files are used to validate that the software worked as intended.

6.2.2 T.2 – Image creation tests

Image creation tests are performed to ensure that parametrized image tools paint as intended, such that EA can evolve the system towards expected results. As the system is creating image art, it is essential that all painting algorithms perform well, and are able to provide images that meet user requirements. Image creation will be tested through comparison between parameterized painting tools (viewed from the log files) and the returned image.

Image creation tests are performed by random-generating a small set of images. Each image has to match the parameterized set of tools used to generate that image. Four such images are displayed below.

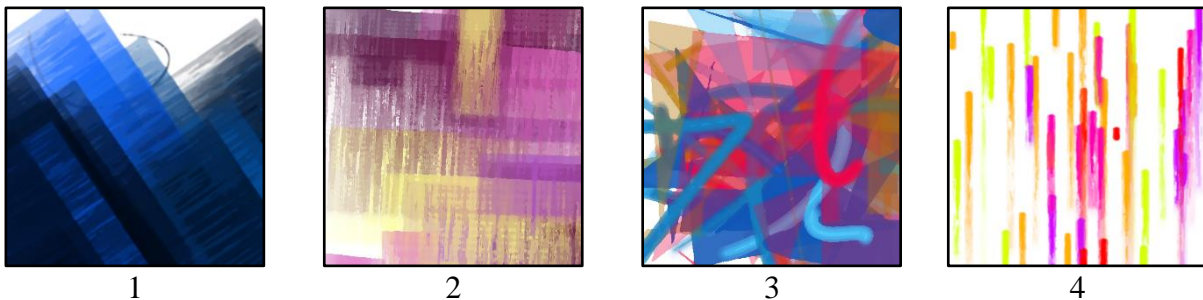


Figure 16. Illustration of images used in the image creation tests, performed by random-generating a small set of images.

The following excerpt (Figure 17) from the log file shows some parameterized tools used to generate image 1.

```

[DEBUG MappingArtist::paint] 2017.05.01 12:54:02:721 - ColorPalette:
[( 7, 89,243), ( 0, 12, 31), ( 3, 25, 56), ( 0, 4, 9), ( 5, 73,146), ]
[DEBUG MappingArtist::paintCrueBrushStroke] 2017.05.01 12:54:03:864 - CurveBrushPainter
[ 6, ( 0, 12, 31), 352, 79, 808, 95, 115, 600, 67, 67]
[DEBUG MappingArtist::paintStraightBrushStroke] 2017.05.01 12:54:08:020 - StraightBrushPainter
[ 139, 348, 715, 711, ( 7, 89,243), 55.88]
[DEBUG MappingArtist::paintStraightBrushStroke] 2017.05.01 12:54:08:182 - StraightBrushPainter
[ 17, 374, 642, 460, ( 0, 12, 31), 55.88]
[DEBUG MappingArtist::paintStraightBrushStroke] 2017.05.01 12:54:08:275 - StraightBrushPainter
[ 166, 671, 668, 515, ( 0, 4, 9), 55.88]
[DEBUG MappingArtist::paintStraightBrushStroke] 2017.05.01 12:54:08:361 - StraightBrushPainter
[ 211, 219, 710, 732, ( 0, 12, 31), 55.87]
[DEBUG MappingArtist::paint] 2017.05.01 12:54:08:557 - Oil [5, 15]
[DEBUG MappingArtist::paint] 2017.05.01 12:54:10:356 - Blur [6]

```

Figure 17. Excerpt from the logfile after generating image 1 from the above set of four images. Each debug line describes what painting tools is being used and with what parameter values.

Comparing the whole set of parametrized tools to each image does indeed confirm that the image creation algorithms work as intended.

6.2.3 T.3 – Evolutionary Algorithms tests

Evolutionary Algorithms are used to search for fit solutions within a search space. These solutions should converge towards global maxima given that the EA is configured correctly. These tests will be performed by utilizing the EA on small known problems and looking for convergence in the results. This EA implementation supports both maximization and minimization of fitness values. Maximization is tested on the “OneMax” problem that searches for a binary string containing only ones, while minimization is tested on a Hamming distance problem that searches for given string.

EA modules are tested through two problems, where the objective of EA is to converge to desired fitness values, targeting both minimization and maximization problems. Following charts display convergence in EA. Hamming Distance EA optimizes towards the target string “HELLOWORLD” and uses Hamming distance to calculate the fitness value. Hamming Distance EA is a minimization problem where target fitness is 0. OneMax EA optimizes towards a string of length 50 containing only ones, from a binary alphabet, where the fitness function is the ratio of ones in the string. OneMax EA is a maximization problem where target fitness is 1.0.

The charts in Figure 18 and Figure 19 show that EA do converge towards desired fitness values, and thus confirm that the EA work as expected. In both test cases the EA reached global maximum. In the Hamming Distance scenario, it reached a local minimum and stayed there for a while, but eventually mutated out of the local minimum and reached global minimum. The Hamming Distance scenario has a low probability of making the necessary mutation to get from distance 1 to distance 0. This is due to the high number of possibilities in the mutation, as each character in the string can mutate to 26 possible characters. This low probability of making the correct last mutation often results in the EA getting stuck in local minimums for a while and thus slow evolution towards the end of the scenario.

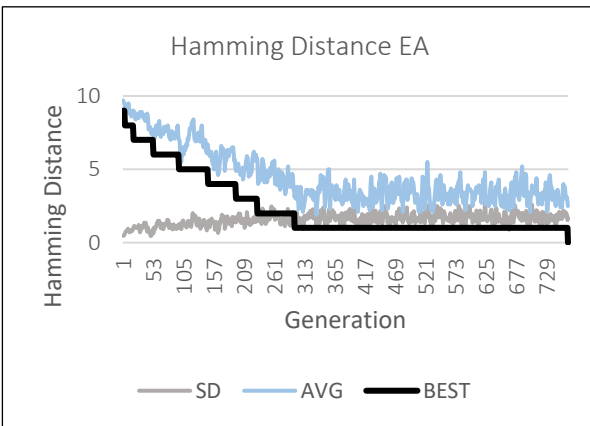


Figure 18. Hamming Distance EA optimizing towards zero. SD is the standard deviation between average (AVG) and best value in each generation.

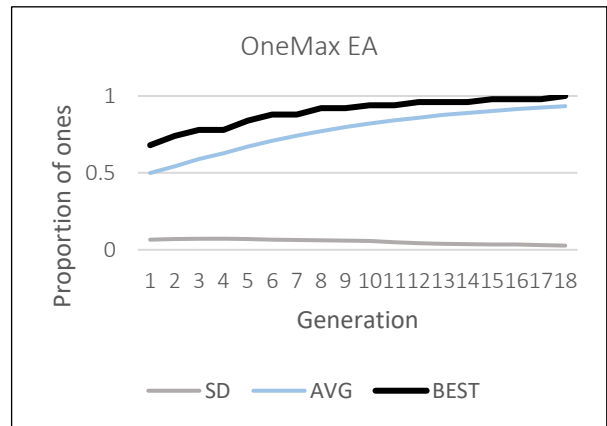


Figure 19. OneMax EA optimizing towards one. SD is the standard deviation between average (AVG) and best value in each generation.

6.3 Research Experiments with Evolutionary Algorithms

Various research experiments (E.1-E.7) are performed to gain knowledge about the system behavior, importance and sensitivity of various methods techniques and approaches, and to obtain data for later system analysis and discussion. A set of six songs is selected for these experiments. These songs differ from each other on several musical features, but are also equal in some musical features. The following letters represent each song throughout the experiments and are used to denote which image is generated from which song:

- A. Billie Jean – Michael Jackson
- B. Chained to The Rhythm – Katy Perry
- C. I Promise – Alex Kozobolis
- D. Kaleidoscope – Kasbo
- E. No Time for Caution – Hans Zimmer
- F. The Imperial March (Darth Vader's Theme) – John Williams

Experiments E.2 – E6. use the same fitness function as described in Section 6.3.1 (Optimizing towards a user specified mapping table) with the same user specified mapping table. These experiments focus on the operations of EA. Table 6 shows a basis EA configuration that is used throughout the experiments. Some experiments use different values and will be discussed in the relevant experiment sections.

EA option	Value
Population size	20
Max generations	2000
Elites	1
Crossover rate	0.7
Mutation rate	0.7
Parent selection	Tournament
Crossover type	One Point
Mutation pressure	20

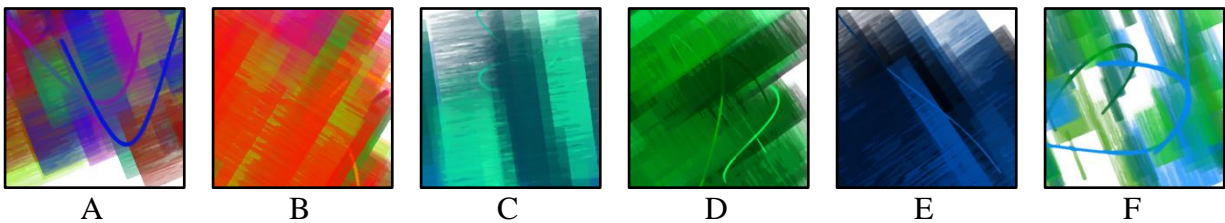
Table 6. Basis EA configuration for the evolutionary runs. Some values are modified in specific experiments.

6.3.1 E.1 – Experiments with fitness functions

Experiments with different fitness functions are performed to investigate how they influence the end results and whether some parts of the function can guide the EA to fulfill some objectives. A small change in the fitness function can introduce novelty in the results or guide the EA directly towards user preference. The fitness functions can analyze the mapping table or the painted images to evaluate how fit a solution is.

Evolution towards a user preferred mapping table

This experiment uses the fitness function “Optimizing towards user specified mapping table” (Section 5.3.2.1) to evolve a mapping table that is close to what the user has suggested. Figure 20 shows the generated images after an evolution.



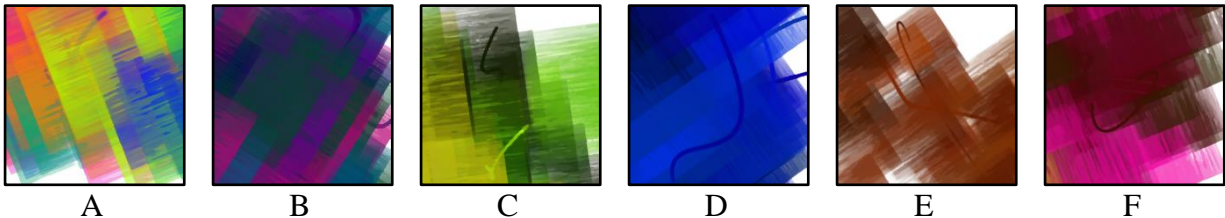
Legend: A – Billie Jean, B – Chained to The Rhythm, C – I Promise, D – Kaleidoscope, E – No time for Caution, F – The Imperial March.

Figure 20. Result images after evolution optimizing towards user preference.

Figure 20 shows images that correlate well with the mapping the user has specified. For instance, the color palette is generated using music feature variables energy and valence. Appendix C shows the numeric values for each music feature variable for each song. “Chained to The Rhythm” has high energy value and yields a color palette based on the color red, while “I Promise” has low energy value and therefore generates a color palette in the blue specter. This fits well to what the user has specified.

Evolution towards novelty search by freely selecting variable for colors

This experiment evolves towards user preferred mapping table, but ignores user specified parameters for color. Figure 21 shows the resulting images.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

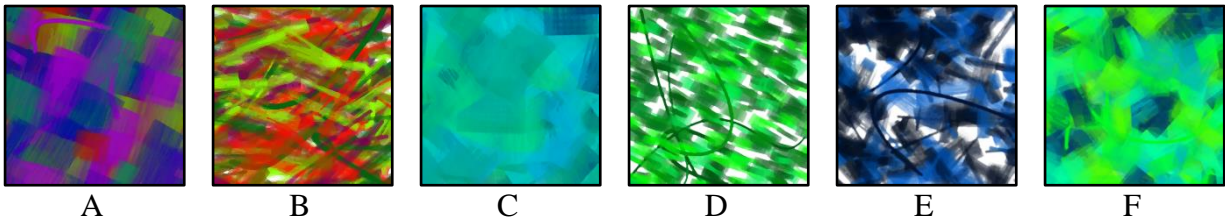
Figure 21. Result images after evolution optimizing towards novelty search freely selecting variable for colors.

Comparing Figure 20 and Figure 21, the major difference is in the color palette in each image. The user argues that both color palettes fit to the music. Image A in Figure 21 has multiple bright colors, while in Figure 20 the color palette is darker, however the user claimed that both color palettes fit with the music “Billie Jean”. This also follows the other images. The user also claimed that image F Figure 21 fits better with the music “The Imperial March (Darth Vader's Theme)” than in Figure 20, due to the presence of dark and red-pink colors. This result was a surprising and appreciated by the user.

In order to avoid subjectivity, the future work can include image analysis (image specter, color distribution, etc.) and various statistical analyses to quantify the differences between the images chosen by the user and images produced by the system.

Novelty search ignoring some user defined keys

This experiment evolves towards a user preferred mapping table, but ignores some arbitrary parameters in the mapping table. It is conducted to see whether the resulting images can surprise the user, but also match most of the user's preferences. Another purpose was to see how sensitive the system is to the parameters. Figure 22 shows the generated images in this evolution.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 22. Result images after evolution optimizing towards novelty search ignoring some user defined keys.

Comparing the generated images in Figure 20 and Figure 22, they mostly differ in shape construction, where Figure 20 uses few big shapes, while Figure 22 uses a high number of small shapes. This introduces style differences between the two result sets. It is visible that the system is sensitive to changes in parameter values. In order to fully understand how sensitive the system is to parameter changes, it is necessary to conduct parameter sensitivity analysis. The user that specified the initial mapping table agreed that the resulting images in Figure 22 fit to the music, but also introduced a positive element of surprise.

These experiments (see Figure 20 - Figure 22) confirm that the parameters used for the fitness function influence the style of the results, as pointed out by den Heijer & Eiben (2010) and den Heijer (2013). They point out that it might not be beneficial for the application. However, our analysis show that for some use cases it still might be beneficial, e.g. Use Cases "Dynamic Ambient Decoration" and "Therapeutic Art" (see Figure 1), while some other Use Case might require more novelty and artistic freedom, e.g. Use Cases "Artist's Work Tool", or "AI Art Generator" (illustrated in Figure 1).

Optimizing towards user preference without knowing the mapping table

The following set of experiments (see Figure 23 - Figure 26) optimize towards abstract user preferences, meaning the user specified how parts of the end result should be rather than how to generate them. This evolution did therefore not have a known mapping table to optimize towards, but had to search for a mapping table that fits the user preferences. These preferences are extracted from the user survey, where for each song a base color is selected, image aggressiveness, and amount of brush strokes to use. Following experiments are based on the preferences from one arbitrary user.

Figure 23 shows the results after evolution optimizing towards user preferred base color for each song, and the color spread in the palette. As there is no optimization towards the amount of brush strokes, the final amount happened to be high therefore filling the whole painting canvas. Table 7 shows the preferred base colors the evolution optimized towards. Comparing the resulting colors in Figure 23 to Table 7 there are some differences in shades, but there is agreement between the base colors.





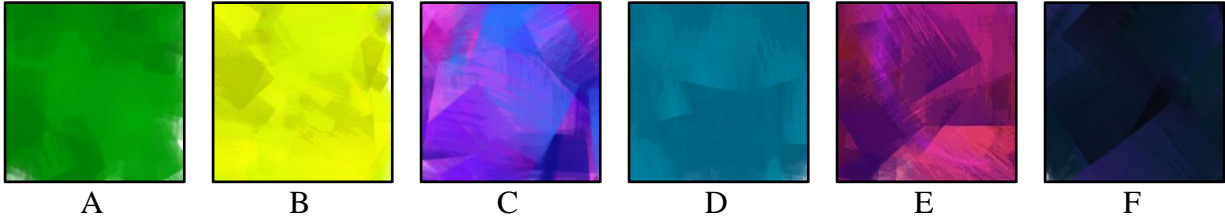
Music	A	B	C	D	E	F
Base color						

Table 7. Base color preference of the user that provided input for experiments in Figure 23 - Figure 26

Figure 24 shows the results after evolution optimizing towards user preferred base color for each song, and the color spread in the palette. However, this set was generated using all the available painting tools, to generate a set of images that differ from other experiments. Comparing Figure 23 and Figure 24 there are two different painting styles in the images. Images in Figure 24 are more dynamic with the use of several painting tools. The polygons provide some aggressiveness to the images, while the small ovals give small elements of surprise that contribute to novelty. Figure 25 is another set generated the same way as Figure 24, but with different parameters. This image set is slightly more dynamic with the use of rotation in some brush strokes.

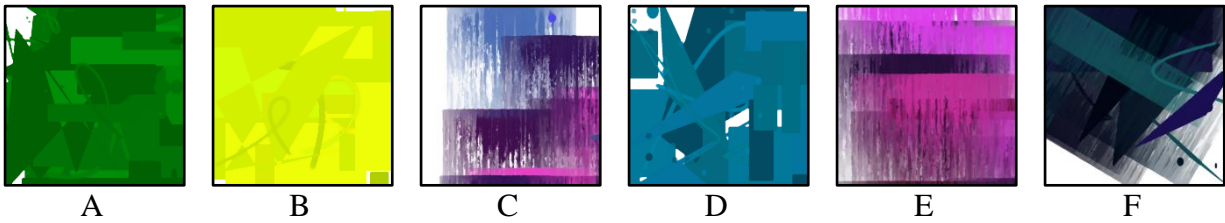
Figure 26 shows a result set after seeded evolution. The fitness function operates on preferences from the same user as in Figure 23, but the evolution is seeded with genotypes from the experiment that produced Figure 20. These two users have different preferences. Comparing Figure 20, Figure 23 and Figure 26 shows similarities in all results, where Figure 26 shares image features from both experiments. This shows that seeding does affect the images, and can also introduce novelty. It is important to make additional future image analyses to measure the strengths of the influence of seeding, and the sensitivity of seeding parameters.

The three earlier experiments (Figure 20 - Figure 22) use data from the same user, while the current set of experiments (Figure 23 - Figure 26) use data from another user with different preferences. Therefore, the final results cannot be directly compared. However, some general remarks can be given. All sets of experiments did match user preference either through direct mapping tables or through specific requirements within the end results such as colors.



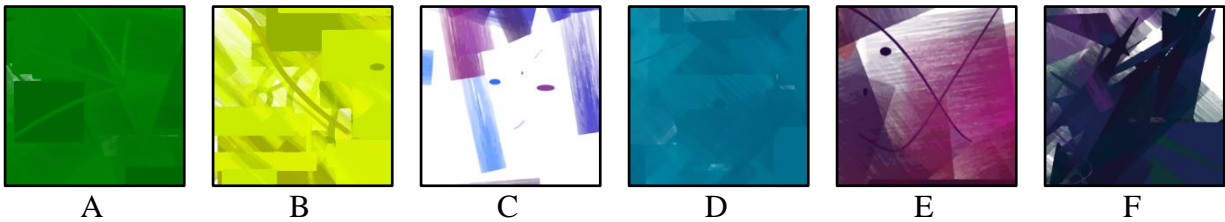
Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 23. Result images after evolution optimizing towards user preferred colors for each song without an optimal mapping table.



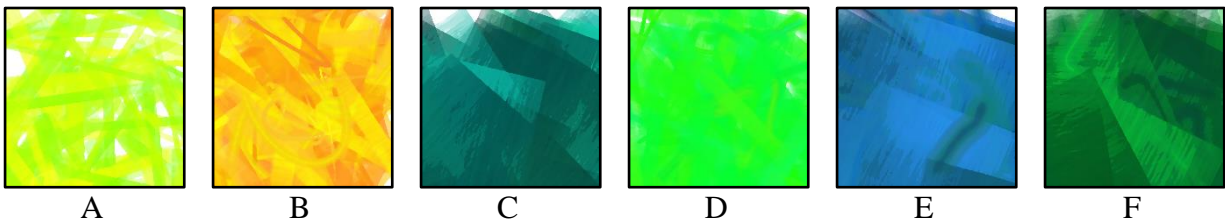
Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 24. Result images after evolution optimizing towards user preferred base color, color spread in palette and number of figures.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 25. Another set of result images after evolution optimizing towards user preferred base color, color spread in palette and number of figures.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 26. Result images after seeded evolution towards user preferred base color, color spread in palette and number of figures. Seeding genotypes taken from earlier experiments (Figure 20).

6.3.2 E.2 – Experiments with seeding techniques

Seeding techniques are implemented to influence the initial population of the EA, giving the opportunity to affect where in the search space the EA should start. Seeding has been met by skepticism, (e.g. Eiben and Smith (2015) claiming that it might be unnecessary), but can still be an efficient push in the right direction for the EA. Seeding techniques can also be used to incorporate previously generated results that meet user criteria, such that the EA can explore this local search space. This experiments aims to investigate how seeding influences the results and performance of the EA, by:

1. Seeding initial population.

Generate the initial EA population based on given genotypes, such that the search starts in a predetermined place in search space.

2. Without seeding of initial population.

Initialize the EA with fully stochastic population.

Comparing the results from both variants gave a good understanding of seeding in this project. These experiments are performed to observe how seeding affects both the evolution of mapping tables and the result images. Both experiments are performed five times each, where the results are averaged. The best genotype of all five runs from each experiment is used to draw the resulting images. The evolutionary runs in these experiments optimize towards a user specified mapping tables, thus knowing how the “optimal” table is.

Without seeding

Figure 27 shows the mean best fitness values from all five runs. The steepest decline in fitness values happen through the first 1000 generations. The last 1000 generations generate only slight improvements on the population.

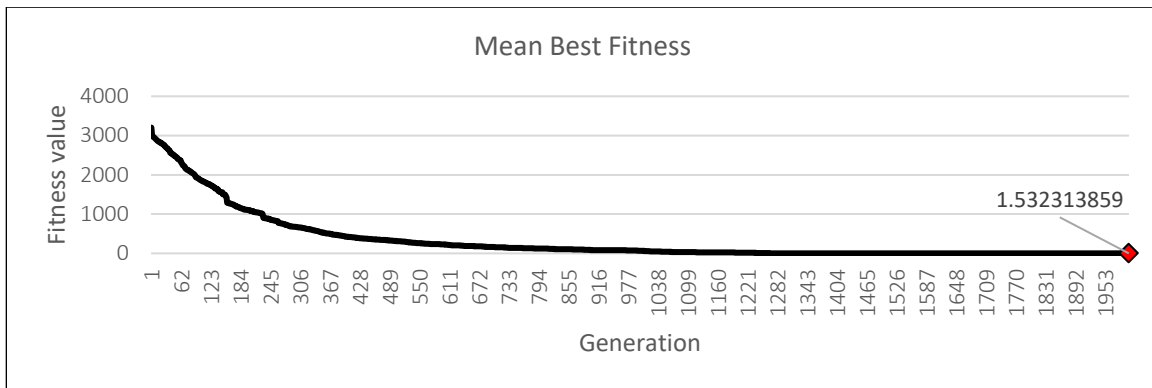
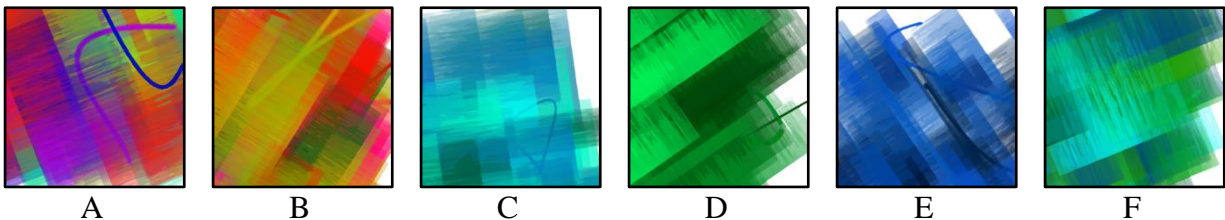


Figure 27. Graph showing mean fitness values of the evolutionary run for EA without seeding. Marked point shows the last best fitness value.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 28. Result images after evolution optimizing towards user preference without seeding.

This experiment optimizes towards a user preferred mapping table, therefore it is expected to get similar results as in Figure 20, as they both use the same configuration. This is indeed the case, where the colors used are similar, rotation of figures and amount of brush strokes all match. The visual differences are due to stochastic painting order and color selection in the figures.

With seeding

Seeding is implemented by allowing the EA to receive genotypes for the initial population. The best genotypes from the previous five runs (Without seeding) are used to seed a portion of the initial population.

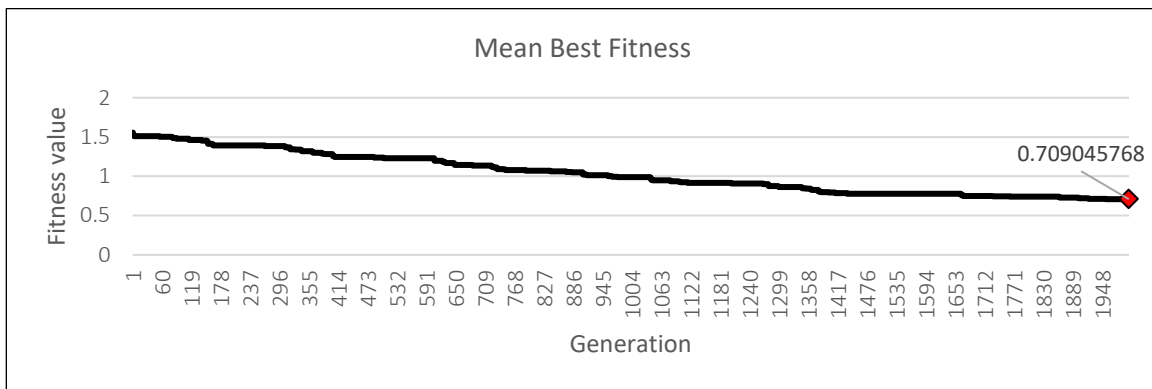
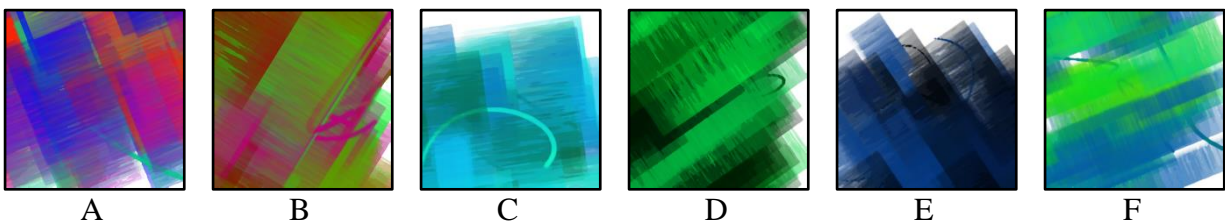


Figure 29. Graph showing mean fitness values of the evolutionary run for EA with seeding. Marked point shows the last best fitness value.

Figure 29 indicates that seeding drastically improves the performance of the EA in the first generations, as rediscovery of previous genotypes with low fitness values is avoided. However, during 2000 generations only slight improvements are discovered.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 30. Result images after evolution optimizing towards user preference with seeding.

Comparing the resulting images between seeding (Figure 28) and non-seeding (Figure 30), it is visible that they share the same image features. Considering the seeding experiment got 2000 more generations, it has not improved the fitness values greatly, but the search difficulty increases as the fitness values converge towards zero. Figure 27 and Figure 29 highlight the last best fitness

value in each experiment, showing a mean improvement of only 0.82 across the two experiments. An evolutionary run with or without seeding, that optimizes towards the same known mapping table, generates very similar images, meaning the only major contribution of seeding is towards runtime. As Eiben & Smith (2015) stated, seeding is not necessary. EA will eventually reach its target fitness value if configured correctly and manages to escape local minimums. However, if the main objective of the evolutionary run is to reduce execution time, seeding can be an efficient option.

To conclude this analysis, this work agrees that the seed is not really necessary as (Eiben & Smith, 2015) claimed. Experiments in this work indicated that seeding influences neither the EA behavior nor the quality of the resulting images. On the other side, seeding does improve the search speed drastically.

6.3.3 E.3 – Experiments with mutation pressure

These experiments are performed to evaluate if the mutation pressure influences the resulting pictures, and to what extent. Some research (den Heijer & Eiben (2010) and den Heijer (2013)) stresses that the evolutionary algorithm setup can introduce various artistic styles, i.e. that the algorithm gets dependent on the user-defined parameters and commands to the algorithm. For some use cases (see Figure 1) it might be beneficial (e.g. for the use cases: Therapeutic Art and Dynamic Ambient Decoration), while for some it might have deteriorating effect (e.g. minimizing the novelty of the use case: AI Art Generator). Three different mutation pressures are experimented with:

1. Low mutation pressure:
Numerical values in the genotype are added with a value in $[-5,5]$
2. Medium mutation pressure:
Numerical values in the genotype are added with a value in $[-20,20]$
3. High mutation pressure:
Numerical values in the genotype are added with a value in $[-50,50]$

Mutation pressure is the boundary value of changes within the intervals of the mapping table in the genotype. When mutation should occur, there is a probability for the music feature variable to change, or for the output interval to change. The minimum and maximum values in the interval

mutate separately. If the minimum value is selected for mutation, the system selects a random value in $[-mp, mp]$, where mp is mutation pressure, and adds this value to the current value of the minimum. Having a low boundary limits the system to small mutations, while a high boundary gives the system ability to change intervals drastically in one mutation.

Low mutation pressure

Low mutation pressure boundary is in $[-5,5]$.

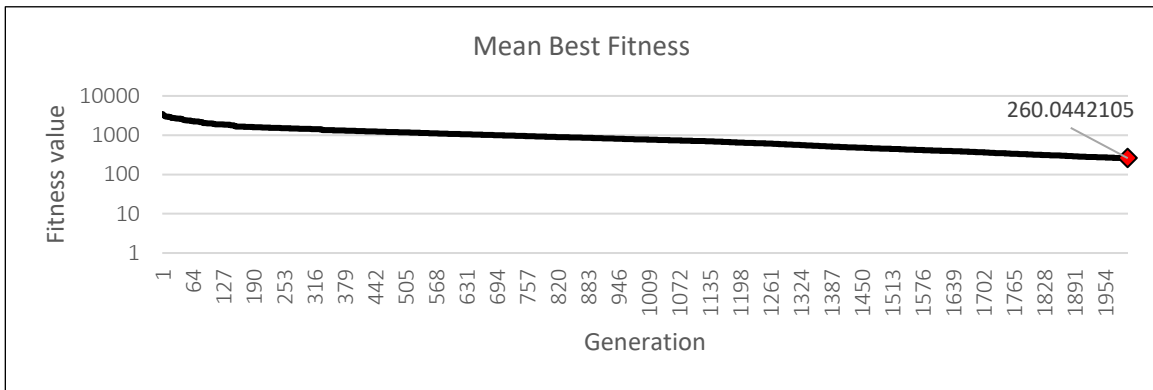
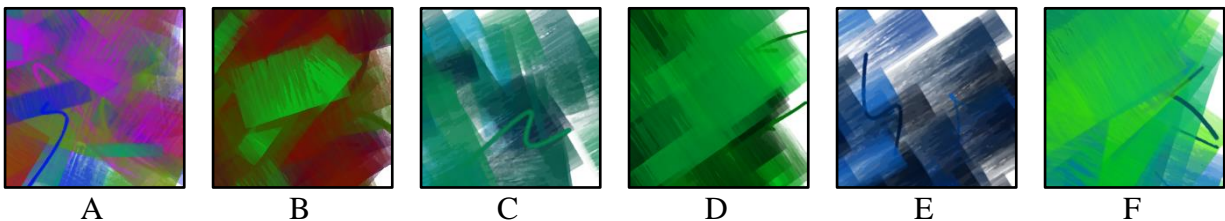


Figure 31. Graph showing mean fitness values of the evolutionary run for EA with low mutation pressure. Marked point shows the last best fitness value.

Figure 31 shows a convergence in fitness values towards zero. However, 2000 generations were not enough for the evolution to reach fitness values below 10. Low mutation pressure tends to give a slow evolution, and therefore a requirement for more generations.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 32. Result images after evolution optimizing towards user preference with low mutation pressure.

The resulting images (Figure 32) differ from images in Figure 20 on several image features. Colors are darker, figure dimensions are smaller and figure rotation is not uniform. Even though the images differ, there are also some equalities that place the two image sets close to each other. Color palettes are indeed darker, but share the same base colors. Figures are also similar enough in dimensions and rotation. It is visible that Figure 32 contains images that are less fit than Figure 20 based on this users' preference.

Medium mutation pressure

Medium mutation pressure boundary is in $[-20,20]$.

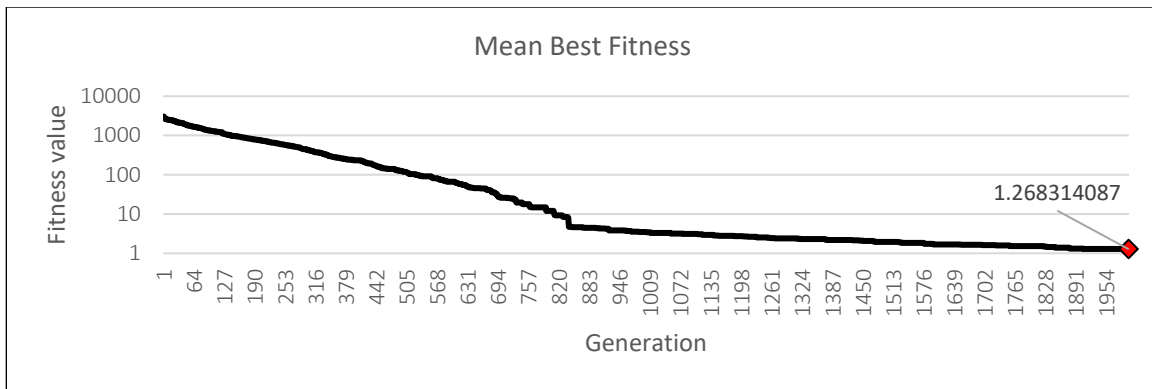
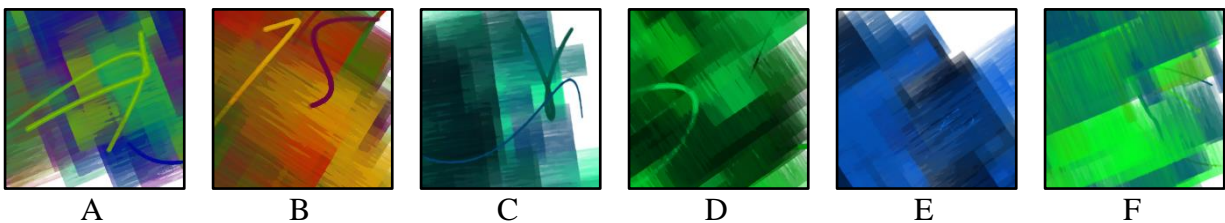


Figure 33. Graph showing mean fitness values of the evolutionary run for EA with medium mutation pressure. Marked point shows the last best fitness value.

Using medium mutation pressure increases the speed of the evolution to reach fitness values below 10. During the last 1000 generations, only small improvements are discovered.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 34. Result images after evolution optimizing towards user preference with medium mutation pressure.

The resulting images are very similar to Figure 20, where the most visible differences are the colors. While generating color palettes both a base color is given, and a set of intervals. These intervals give the ranges of possible color brightness and saturation values. The table below compares two color palettes between Figure 20 and Figure 34 for songs A and B.

	A – Billie Jean	B – Chained to The Rhythm
Figure 20		
Figure 34		

Table 8. Table showing color palette differences between songs A and B in Figure 20 and Figure 34

The genotype in these two sets allows the color palette generation algorithm to stochastically choose a brightness value for the palettes in the interval $[0.5, 1.0]$. Due to randomness, the set in Figure 34 got lower brightness values, thus resulted in darker images.

High mutation pressure

Medium mutation pressure boundary is in $[-50,50]$.

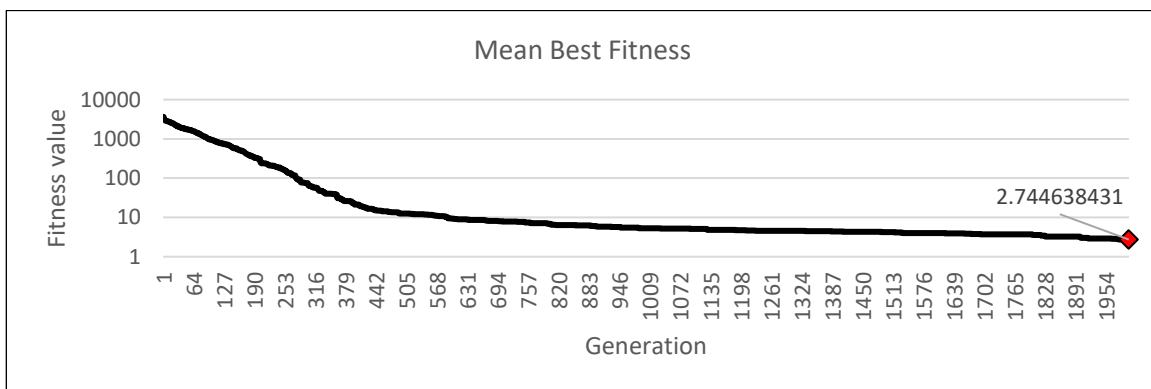
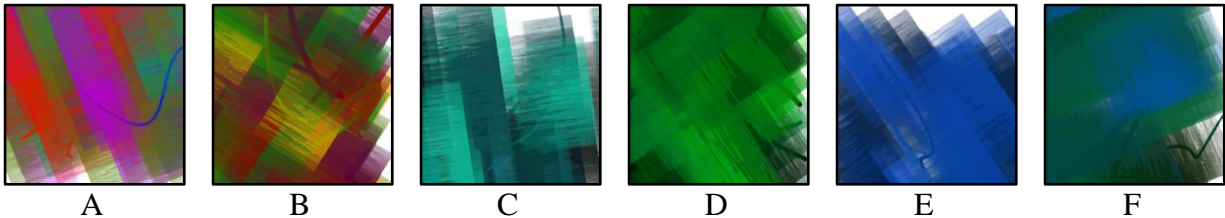


Figure 35. Graph showing mean fitness values of the evolutionary run for EA with medium mutation pressure. Marked point shows the last best fitness value.

Using a high mutation pressure interval allowed the evolution to find solutions with fitness values lower than 10 in around 500 generations. During the last 1500 generations, little progress was made as the mutation is too intense.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 36. Result images after evolution optimizing towards user preference with medium mutation pressure.

Comparing the three different intervals for mutation pressure, it is visible that each of them have their strengths and weaknesses. Low mutation pressure yields good results when the fitness improvements start to decrease. High mutation pressure can quickly improve the fitness values in the early generations. Medium mutation pressure performs well between the early generations and the late generations when fitness improvement start to flatten out. An adaptive mutation pressure that utilizes the strengths of each part, may be a better approach, than selecting either low, medium or high mutation pressure. For instance, using high mutation pressure during the first generations and steadily decreasing mutation pressure as the population gets more fit.

6.3.4 E.4 – Experiments with crossover techniques

As another parameter of the EA, crossover techniques can alter how EA "walks" through the search space. This test is performed to investigate the impact of different crossover techniques on a mapping table. Two types of crossover techniques will be tested, namely one point crossover and uniform crossover.

1. *One point crossover:*

Implementation of one point crossover on a mapping table, where the slicing point is in the list of mapping keys.

2. *Uniform crossover:*

Implementation of uniform crossover on a mapping table where the mapping keys determine what values are recombined.

The main difference between one point crossover and uniform crossover is the gene contribution on the segment level and individual level. One point crossover selects segments from parents that are brought over to children, while uniform crossover mixes up individual genes from the genotype, possibly giving each gene a higher level of importance.

These two experiments use the same basic configuration (Table 6) with two different crossover types. Both crossover techniques use the keys in the mapping table to arrange the mixing of values.

One Point Crossover

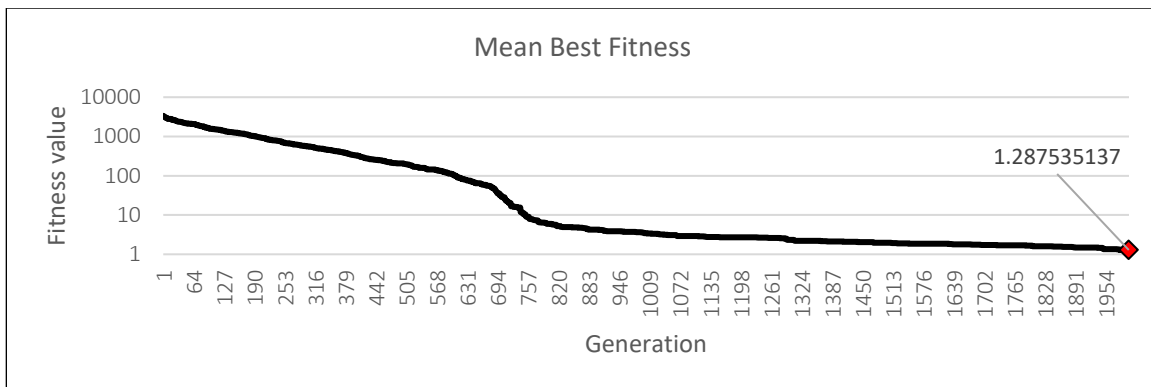
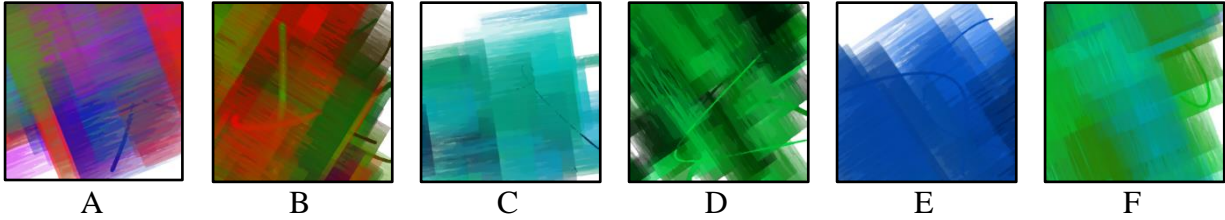


Figure 37. Graph showing mean fitness values of the evolutionary run for EA with One Point crossover. Marked point shows the last best fitness value.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 38. Result images after evolution optimizing towards user preference with One Point crossover.

Uniform Crossover

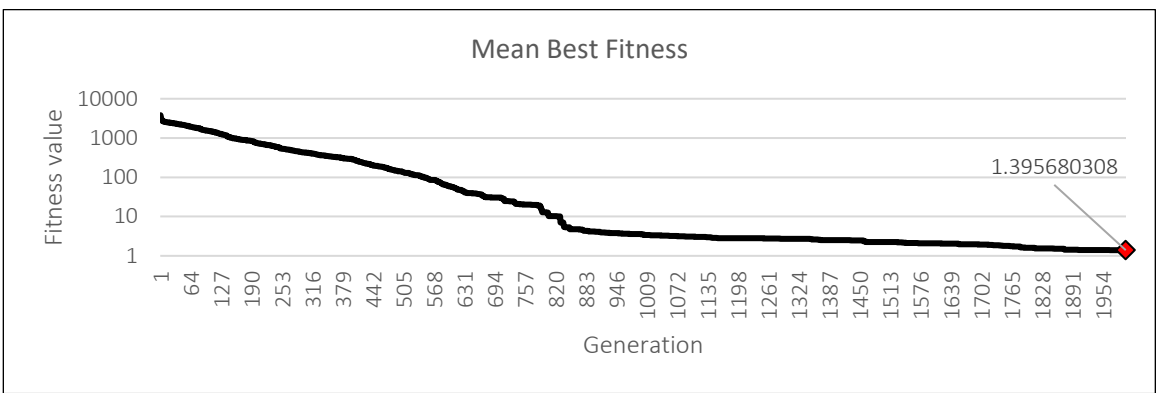
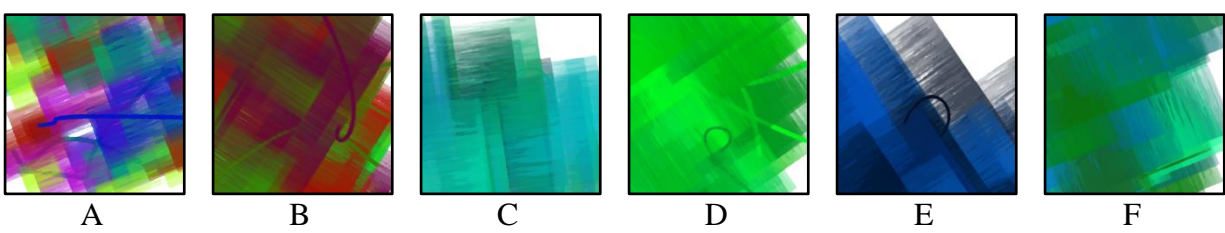


Figure 39. Graph showing mean fitness values of the evolutionary run for EA with Uniform crossover. Marked point shows the last best fitness value.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 40. Result images after evolution optimizing towards user preference with Uniform crossover.

Comparing the graphs for Uniform and One Point crossover evolutions, there is no evidence of larger differences that make one of them favorable. The One Point crossover reached fitness values below 10 slightly faster than the Uniform. One out of the five evolutionary runs using the Uniform

crossover used many generations to reach fitness values below 10, therefore pushing the mean fitness value higher compared to the One Point crossover. Taking aside this anomaly there are no significant differences between the two crossover types.

6.3.5 E.5 – Experiments with selection strategies

Two main selection strategies are tested to see how each strategy affects the performance in the manner of time used to reach desired fitness values:

1. *Tournament selection*

Divides the population into several pools, where the phenotype with the best fitness value within each pool is selected for mating.

2. *Proportionate selection*

Each phenotype has a proportionate probability to its fitness value to be selected for mating.

Different selection mechanisms may impact how fast the EA reaches desired fitness, and how diverse the population is.

Tournament selection

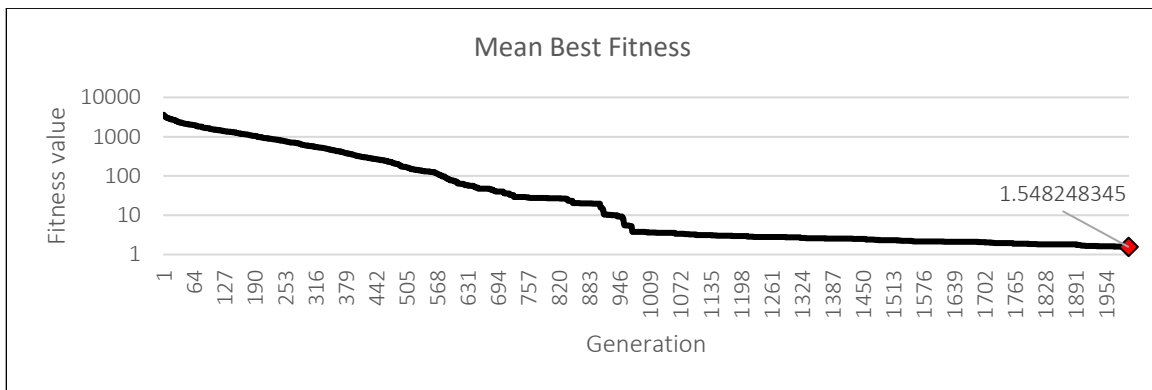
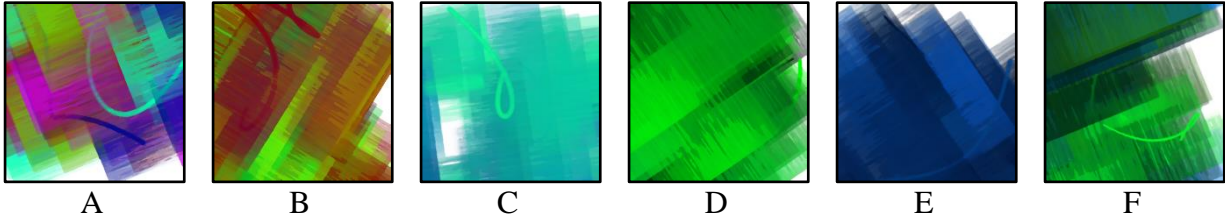


Figure 41. Graph showing mean fitness values of the evolutionary run for EA with Tournament selection. Marked point shows the last best fitness value.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.
 Figure 42. Result images after evolution optimizing towards user preference with Tournament selection.

Proportionate selection

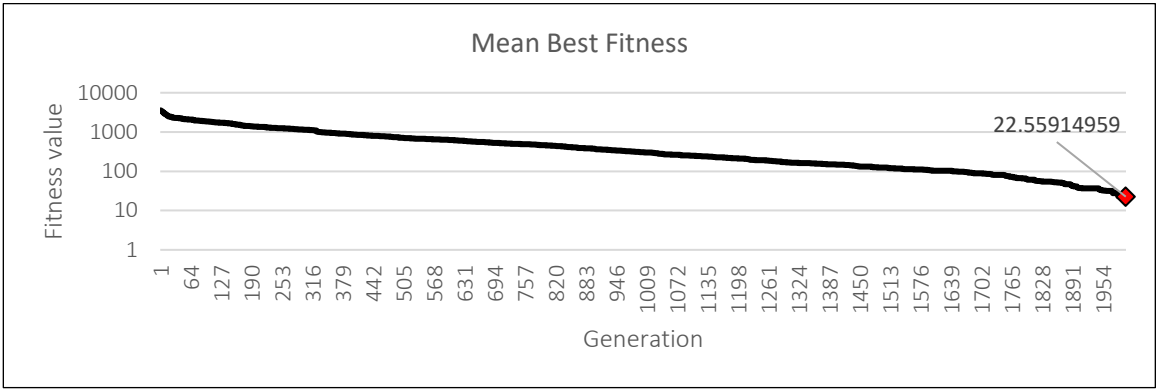
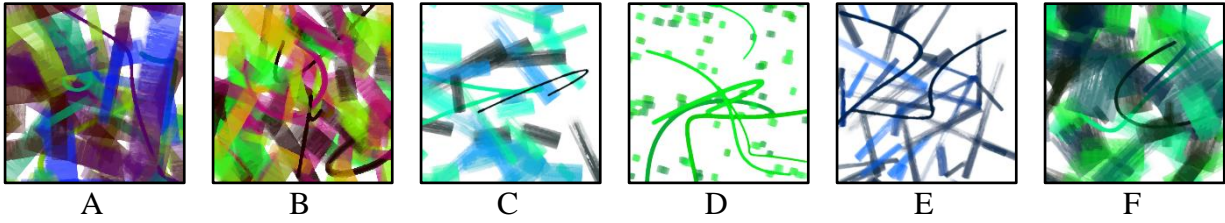


Figure 43. Graph showing mean fitness values of the evolutionary run for EA with Proportionate selection. Marked point shows the last best fitness value.



Legend: A – *Billie Jean*, B – *Chained to The Rhythm*, C – *I Promise*, D – *Kaleidoscope*, E – *No time for Caution*, F – *The Imperial March*.

Figure 44. Result images after evolution optimizing towards user preference with Proportionate selection.

The results from Tournament selection and Proportionate selection indicate that there are differences between the selection strategies that affect the evolution and therefore the end results. Tournament selection can quickly select fit solutions to pass to the next generations, while proportionate selection uses more time to filter out poor performing solutions. In the end, the

proportionate selection strategy limits the evolution and results in a less fit genotype when the generation limit is reached. Due to the less fit genotype in Proportionate selection, the painting algorithm fails to generate an image that meets the user's requirements as seen in Figure 44.

6.3.6 E.6 – Performance evaluations

These experiments are done to evaluate how the EA performs in this project. It is necessary to know the performance of the system such that further optimizations can be made. Two main experiments test the performance:

- 1. *Time usage***

Consumption of time is the main measurable unit of search performance. Time is measured from initialization of search, and until the desired fitness value is reached.

- 2. *Following the evolution***

Visually following the evolution to find out when the EA converge and how the number of generations affect the results.

The time usage experiment is a good indicator for evaluating how the system performs and whether or not the time usage is reasonable. "Following the evolution" experiments gives information about the generations that are the most critical for the system, in order to target optimizations at the right points.

Time usage

Time usage experiments were performed using the fitness function "optimizing towards a user preferred mapping table" (5.3.2.1). Time is measured per generation, with a mean duration of 23.44 milliseconds per generation. Figure 45 shows mean duration to reach a set of desired fitness values. Approximately 35 seconds is required to reach the lowest set fitness value of 2.5. This is reasonable amount of time consumed to find genotypes that are close to user preference.

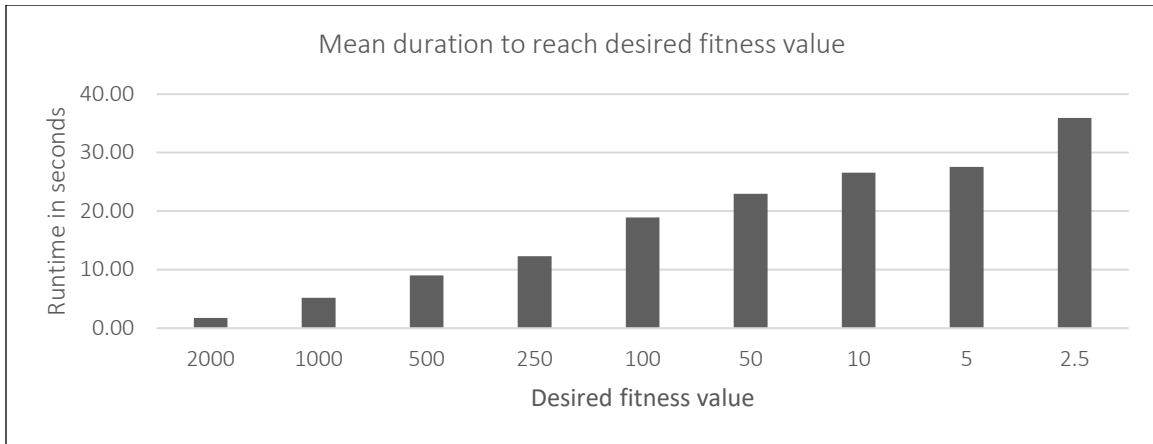


Figure 45. Mean duration to reach desired fitness value

Following the evolution

This experiment is performed to investigate how the evolutionary algorithms operate and how they evolve solutions to this specific project. The default configuration and fitness function (see Section 6.3) is used in this evolution. A set of generations on which the EA should pause its evolution and generate current best results obtained, was selected before the evolution. Considering previous experiments it is clear that notable fitness improvements occur in the early generations, thus it can be interesting to look at a few more early generations than late. Table 9 contains results after some selected numbers of generations. Looking at the initial generation (generation 1), it has hardly, if any, relation to the selected songs. The colors and figures are random and follow no patterns. At generation 250 colors are getting closer to user specification and the images are more aesthetically pleasing compared to the first generation. However, the figures and colors still do not match desired features. Song C – I Promise, is a calm musical piece and does not fit with energetic colors as in generation 250. In generation 500 the angles used while drawing figures stabilize, but the colors are still off. Generation 1000 is very close to expected results, and the color and figure properties converge towards expected properties. However, some elements are still off. The easiest missing property to spot is the color shading. Comparing generation 1000 to 2000, the last generation has a richer set of colors in the palette, giving the end result more depth. Generation 1500 is an interesting generation. Here the figure properties are very close to the expected end result (as in generation 2000), but the colors are different from the best individual in generation 1000. There are arguments that the color palettes in generation 1500 fit to the music, however,

they do not fit user expectations. Such anomalies in the generation may be interesting to explore for novelty in art generation. In generation 2000 the evolution is terminated. The mapping table values are very close to user expectation and the final images are as expected, where both color palettes and figure properties are as preferred. After generation 1000 the evolution can be terminated as the results are close to user preference. However, generation 1500 shows that some genotypes may be numerically close to user preference, but visually far off even after 1500 generations.

Generation	A	B	C	D	E	F
1						
250						
500						
1000						
1500						
2000						

Legend: A – Billie Jean, B – Chained to The Rhythm, C – I Promise, D – Kaleidoscope, E – No time for Caution, F – The Imperial March.

Table 9. Results after a number of generations during the evolution. Optimizing towards user specified mapping table.

6.3.7 E.7 – Experiments with user involvement

As the aesthetic judgement is subjective it is necessary to involve humans in the learning and evaluation process. In this project, humans will be involved through a small survey (see Appendix D) and interviews with one or two people. The interview will be used to generate mapping tables the system can optimize towards. The questions are straight forward about relations between tool parameters and music variables, as well as numeric intervals. The survey is created to obtain a more general overview of people's expectations on how the images should look, and to get a small feedback on already generated images. Interesting feedback would be how well the system has generated a set of images that matches the input music set, and the overall aesthetics of the resulting images.

For the initial mapping table used in experiments E.1 – E.6 one user has been questioned about how various musical variables should affect the image parameters. A fitness function is created (see Section 5.3.2.1) that guides the evolution towards the mapping preference of this user. Some fitness functions that intend to let the system explore freely, i.e. introduce novelty, are also based on this initial mapping table. To conclude, this user's expectations are parameterized into individual results, and some of these data is used in the experiments.

The user feedback presented below is a combination of individual responses and a summary of all users' responses. User expectations of how an image should look after listening to a specific song are collected into a table. Four categories are used to describe each image for each user:

1. *Base color*

Base color used for palette generation. Illustrated by taking the user's response color and making it darker or lighter if requested.

2. *Aggressiveness*

A measure from 1-5 where 1 is relaxing and 5 is aggressive. This scale is used to get an indication of how figures and brush strokes should be placed in the image.

3. *Brush strokes*

The amount of brush strokes to use on the image on a scale from 1-10, where 1 is very few and 10 is lots of brush strokes.











4. *Palette*

Expected coloring where the used palette should have:

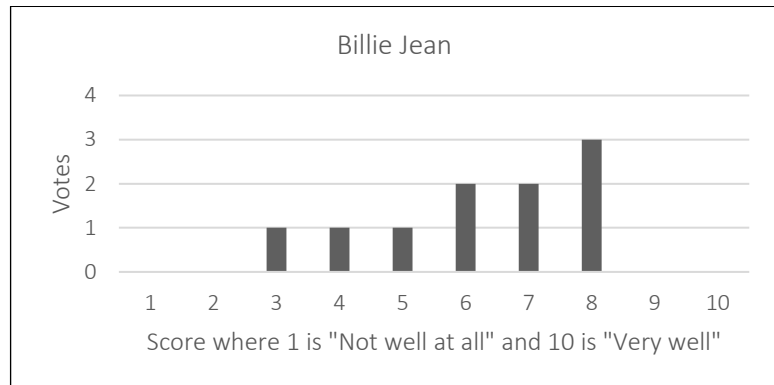
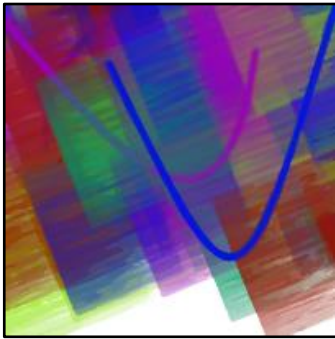
- O1 – One color with small changes in shades
- F1 – Few adjacent colors with small changes in shades
- M1 – Many adjacent colors with small changes in shades
- O2 – One color with high variance in shades
- F2 – Few adjacent colors with high variance in shades
- M2 – Many adjacent colors with high variance in shades

Billie Jean – Michael Jackson

Individual user expectations:

User	1	2	3	4	5	6	7	8	9	10
Base color										
Aggressiveness	3	2	4	2	4	3	4	3	4	4
Brush strokes	7	3	8	5	9	5	8	7	7	8
Palette	M1	F1	M2	F1	F2	F1	F2	M1	F2	M1

User feedback summary:



Positive











- color palette
- repetition of colors
- relationship between colors
- follows the rhythm
- amount of brush strokes

Negative

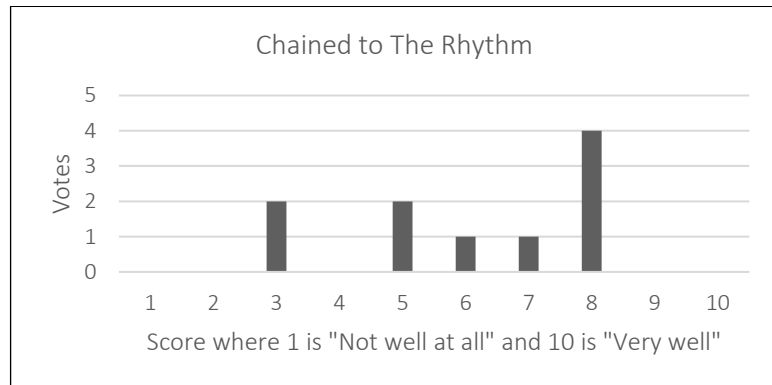
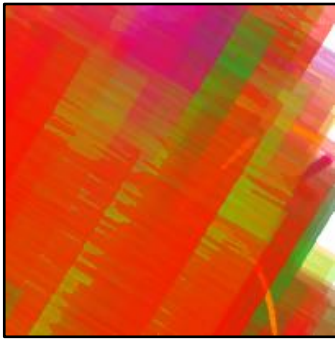
- too uniform
- geometrical
- too many colors
- exposed canvas
- too dark

Chained to The Rhythm – Katy Perry

Individual user expectations:

User	1	2	3	4	5	6	7	8	9	10
Base color										
Aggressiveness	2	3	3	2	3	4	4	2	4	4
Brush strokes	8	8	7	7	6	7	7	8	8	6
Palette	F1	F2	F2	M2	F1	F1	F1	F2	M2	F1

User feedback summary:



Positive











- light colors
- warm colors
- energetic
- lively

Negative

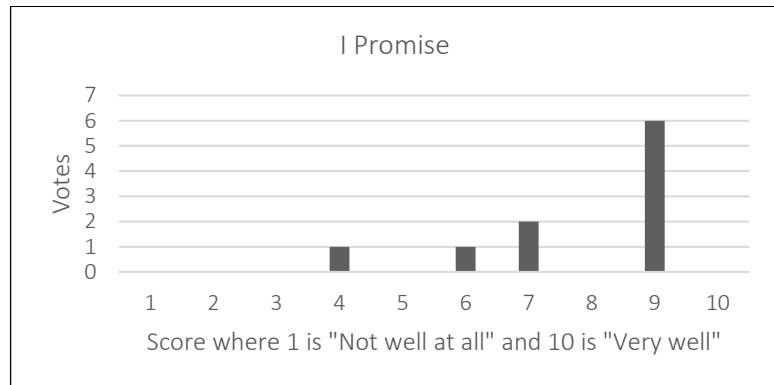
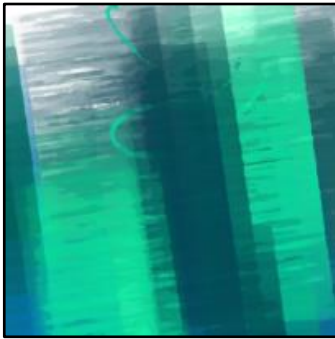
- too much red
- one user said "nothing particular"
- several pointed out the dominance of red colors

I Promise – Alex Kozobolis

Individual user expectations:

User	1	2	3	4	5	6	7	8	9	10
Base color										
Aggressiveness	1	1	1	1	1	2	2	1	1	1
Brush strokes	3	2	4	3	3	3	4	2	4	3
Palette	O1	O1	O2	O1	F1	F1	F1	F1	F1	O2

User feedback summary:



Positive











- correct color temperature
- relaxing
- harmonic colors

Negative

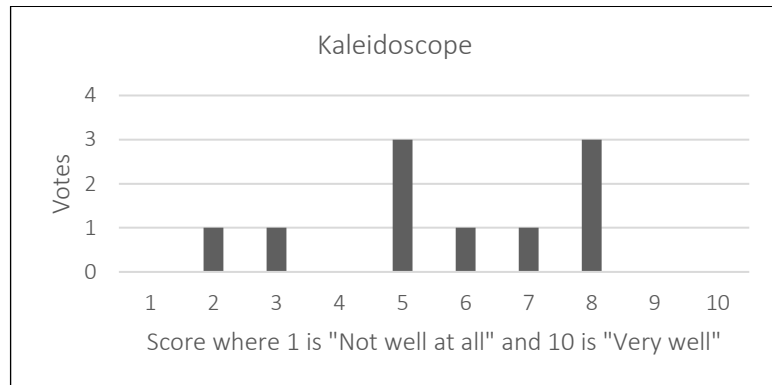
- several said there was not much to dislike
- some pointed out the geometrics do not fit the song

Kaleidoscope – Kasbo

Individual user expectations:

User	1	2	3	4	5	6	7	8	9	10
Base color										
Aggressiveness	2	4	2	1	2	3	3	2	4	2
Brush strokes	5	7	5	5	9	6	6	6	7	5
Palette	M1	F2	F1	F1	M2	M2	F2	M1	M2	O2

User feedback summary:



Positive











- color
- shade variations
- gives the right “feel”

Negative

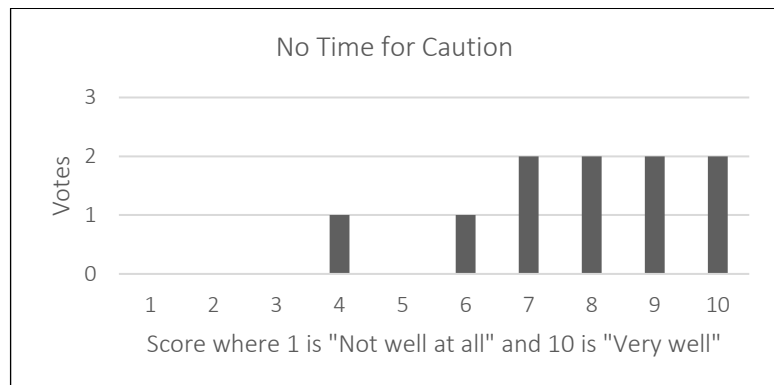
- too many brush strokes
- needs more colors and shades
- does not fully represent the music

No Time for Caution – Hans Zimmer

Individual user expectations:

User	1	2	3	4	5	6	7	8	9	10
Base color										
Aggressiveness	2	3	5	3	2	3	5	1	4	5
Brush strokes	1	6	9	7	2	5	9	2	8	6
Palette	O1	F1	F2	F2	F1	O1	F2	F1	F2	F2

User feedback summary:



Positive











- colors and color temperature
- reflects the mood of the music
- aesthetically pleasing image

Negative

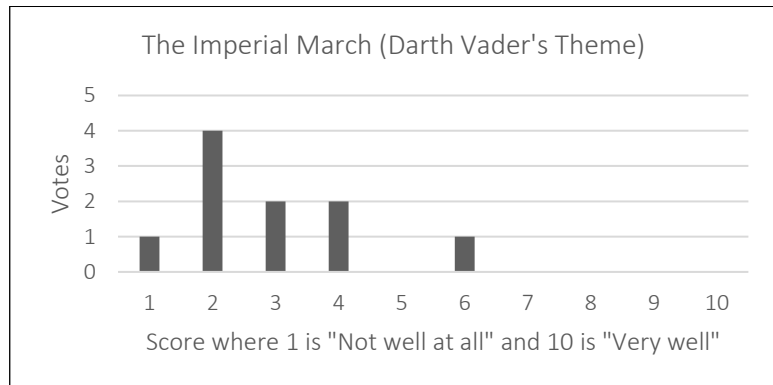
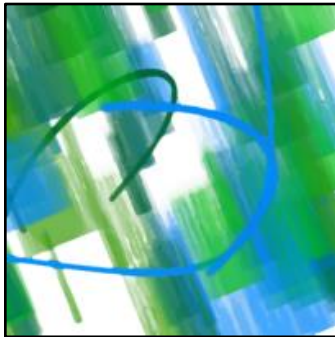
- few negative responses
- a bit too dark
- needs more aggressive colors

The Imperial March (Darth Vader's Theme) – John Williams

Individual user expectations:

User	1	2	3	4	5	6	7	8	9	10
Base color										
Aggressiveness	5	4	5	4	4	4	5	4	5	5
Brush strokes	10	8	10	4	6	6	9	8	10	8
Palette	F2	M1	F1	F1	F1	M1	M2	F1	M2	F2

User feedback summary:



Positive

- dynamic
- creative
- lots of responses on small brush strokes

Negative

- wrong colors
- not aggressive enough
- wrong mood

7 DISCUSSION

This chapter reflects on the results and other achievements of this thesis. The discussion deals with the following topics: (a) overall view on the work, (b) how the project goals are achieved, (c) how the research questions are answered, (d) learnings and experiences from the research and development work, (e) future prospects.

7.1 The Project from the High-Level Perspective

This project was a good theoretical, practical (software design and implementation), and research experiment-driven study. The study was supported by user interactions, and several approaches for collecting direct and indirect feedback have been applied. This research study gave insight into several important research and engineering aspects, including:

- use of AI approaches in art,
- formalization of aesthetics and novelty for machine-driven exploitation (for the benefit of the end user),
- evolutionary algorithms (theory and practice),
- concluding with software engineering principles (software framework approach), which facilitate reuse of the developed software functionality and interaction with existing knowledge and software systems.

This work was a journey in the research and development activities, where the existing knowledge body gave valuable insight, knowledge and interesting ideas, where some of them are tested in this work by the experiments. Experience of many research groups gave interesting suggestions for design of the evolutionary algorithms, how to influence them and finetune them by various techniques, e.g. by various implementations of the fitness function, seeding techniques, selection of the genotype, etc. Many of these statements and claims have been tested by the experiments presented in Chapter 6.

The author hopes that also this work slightly contributes to the overall "AI in Art" community. Some of the design achievements that should be emphasized are: (a) the metadata-driven design of the genotype, (b) the metadata-driven "coupling" between the evolutionary algorithm and the

painting tools, and (c) the generic design of the fitness function (that gives a possibility to experiment with various fitness evaluation approaches). The software framework for music-to-image creation has the properties of a generic and flexible software system, which can be easily upgraded and "coupled" to additional music and image interfaces. It also represents a good basis for future improvements and functional enrichments.

7.2 Project Goals and Research Questions

How did this study answer the major project goals? Several important project activities have contributed to fulfilling the project goals, starting with the intensive theoretical analysis, overview of the state-of-the-art in the field, design and implementation experiences, results of the experiments and trials (listed in Table 5), and ending with the user feedback and evaluations.

The major goals of this project are:

1. Design the computer artist system

The computer artist as a system has been designed, implemented and subjected to intensive testing and experimental trials. They helped improve the design and implementation of the system, but also to finetune of the evolutionary algorithms. The implemented software framework for music-to-image creation has the properties of a generic and flexible software system, which can be easily upgraded and "coupled" to additional music and image interfaces. It is also a good basis for future improvements and functional enrichments.

2. Implement evolutionary algorithms in the system to create images based on musical input

Evolutionary algorithms have been implemented in a relatively generic way. They can be used not just in the area of music-to-image transformations, but also other application areas. EA should not be considered just as a software engineering system, but also as solvers in the search for solution and optimization in the complex domain space. The series of experiments helped understanding the importance and sensitivity of various techniques that influence the domain space search, e.g. various fitness function designs, seeding techniques, mutation pressure, cross-over approaches, etc.

Project activities mentioned above also helped answering the research questions of this project (supported by the experiments and implementation tests, listed in Table 5):

1. How can the computer artist system be analyzed and evaluated?

Tests and experiments were considered the best approach for analyzing the system. Qualitative analysis and software tests were performed to analyze the system from a software perspective, while research experiments were performed to analyze the system as an AI system. The performance and integration of EA towards the music-to-image problem was the main part of the system to analyze. EA can be analyzed through experiments with different configurations and parameters (evaluating how they affect the performance and the search results). Experiments E.1-E.7 contain this analysis in Section 6.3. Increasing the number of evolutionary art modules, such as different fitness functions, selection strategies, mutation mechanisms, etc., can further improve the analysis and demonstrate which parts of the system are sensitive to change and to what degree.

Evaluation of the system can be performed through user feedback. As aesthetic judgement is subjective, it is difficult to create an automated evaluation process. The system cannot, at this point, evaluate the performance sufficiently only through a fitness function. Some additional techniques have to be used together with a fitness function. Experiments E.1 and E.7 (experiments with fitness functions and user involvement), showed that the system was able to find a mapping table that is very close to the user preference, but the resulting images sometimes were not optimal considering the user's expectations.

2. Can the system learn about user preferences and user notion of aesthetics, and how?

The system can partially learn about the user preferences through earlier experiences. Collecting genotypes and user expectation data gives the necessary information about user preferences. The best genotypes after each evolution can be stored and reused through the seeding. Data analysis techniques can then be utilized to collect the key elements of one or more user preferences.

The system can accurately reuse (single) user's preference data through the seeding (experiment E.2, p.61). Experiment E.1 shows that the results may come from the evolution optimizing directly towards the user preference, but also from the fitness functions optimizing for novelty. The novel results can be approved by the user and included in the experience data as a successful run.

Diverse user preferences (experiment E.7) make it difficult to generate a mapping table that fits every user's preferences. It means that at this stage of development, the system can at best learn the individual single user's preferences.

Currently, not enough data analysis modules are implemented to take full advantage of earlier experiences. Intelligently merging previously generated genotypes (based on how different parameters (music and image) affect the end result and their sensitivity) could produce more accurate solutions.

Implementing additional image analysis modules to analyze also the resulting images can guide the evolution towards more fit solutions. Image analysis can also be used to investigate which parameters are essential for the user's notion of aesthetics. However, image analysis within the evolution will result in a more time-consuming evolution, as previously discussed.

3. How can user feedback be collected and used in the system?

User feedback is necessary to modify and train the system to produce accurate results considering user preferences and a subjective notion of aesthetics. Feedback can be collected both directly (through surveys and interviews) and indirectly (through input parameters and preferences). Surveys and interviews give ***direct answers*** to questions targeting specific parts of the system, while the input parameters and preferences give the system a requested ***direction*** of user preferences. Multiple datasets of input parameters and preferences give the system a ***pointer*** to expected results. Comparison between different inputs can give the system information of how the user adapts the input to improve perceived quality of results. Analysis of the system input gives therefore a rough feedback or expectations of the end results. Experiment E.2 illustrates how the input data and the resulting genotypes (after an evolution) can be used to increase the performance of the system. Survey and interview analysis give the direct feedback of the system and its performance. This feedback can either be used to directly modify the mapping tables, or to evolve towards user preference (by modifying fitness functions, seeding techniques and the painting tools). Experiment E.1 (Optimizing towards the user preference without knowing the mapping table) yields promising results for the evolution with direct use of feedback data from experiment E.7.

4. Can the system create aesthetically pleasing and meaningful images without user involvement during evolution?

Experiment E.1 shows multiple fitness functions that are able to find solutions that fit the user preferences as specified, or result in a solution that is close to the user preference. Assuming that the provided user preference through the metadata is accurate enough, the system can create meaningful images. Experiment E.7 confirms that some images are both aesthetical and meaningful. In some cases, there was some negative feedback on the image aesthetics and the music match. This is mainly due to subjective preferences of image quality. Even though users are not directly involved in the evolution, some user interaction can be introduced through the seeding. The seeding shows promising results as described in experiment E.2. The seeding could be introduced mid-evolution to push the evolution in a specific direction (by choosing the genotypes with specific/wanted properties).

Several projects (see Section 3.1.1 and 3.2) use interactive evolution to guide the evolution towards user preferences. This can be a more effective process of searching for subjective solutions, but also introduces challenges as mentioned by McCormack (2005). An alternative to interactive evolution may be a hybrid system that only involves humans in critical parts of the evolution and not through the whole evolution process. Such a system may modify the population through user selection or seeding.

Due to subjective judgement of aesthetics and cultural differences worldwide, the system cannot create aesthetically pleasing and meaningful images without any user involvement. This is also pointed out by Galanter (2010). This system must therefore involve end users in the initial stage of the evolutionary algorithms such that the system has some guidelines towards user preferences. The system is able to generate aesthetically pleasing and meaningful images for end users that share at least some notions of aesthetics, such that the differences between the user's preferences can be utilized positively. The system can generate images based on one user's preferences and thus be considered as novelty by another user. In this scenario, the second user has no involvement in the system. However, there is no implemented fitness function that is able to cover every user's preferences.

7.3 Creativity

This project belongs to the field of computational creativity. An essential question to ask is: is the software producing the images creative? Boden stated that: “A creative idea is one which is novel, surprising, and valuable (interesting, useful, beautiful...)” (Boden, 1998, p.347). Boden emphasized that computational creativity should focus on P-creativity, which means novelty for the creative individual. Following this definition there are some arguments towards the software being creative. Novel and surprising elements can be achieved in multiple ways. Every generated image will most likely be unique due to stochastic parameters in the painting algorithm. This means that novelty is present in the generated images, considering the figure structures and placements. Another direction for novel images is through the mapping tables. Experiments in Section 6.3.1 showed that evolution towards novelty can generate novel mapping tables that are surprising for the end user. The surprising elements occur in the feature mapping, e.g. how colors relate to specific music features. There are less surprising elements in the painting style as it is repetitive due to limited amount of painting tools and tool parameters. During the experiment “Optimizing towards user preference without knowing the mapping table” (p.58) the system generated several mapping tables and images that are considered novel and surprising. Most of the novel elements in this experiment are due to randomness. It is hard to argue that randomness is creativity, even though it sometimes is essential to discovery of novel solutions.

Further following Boden’s definition of creativity, the resulting images should be valuable. There are multiple arguments that the images generated by this software are valuable. Figure 1 shows multiple use cases for this system. The images are pleasing and interesting enough to use as decoration. The images can also be used as therapeutic art in combination with the music. The images generated with music as inspiration could also be useful for people with hearing difficulties, as they can get an understanding of how the music is.

Another way to tell if the software is creative could be to follow Colton and Wiggins’ definition of computational creativity: “The philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviors that unbiased observers would deem to be creative” (Colton & Wiggins, 2012, p.1). No unbiased group of people was gathered to answer whether the behaviors of the software could be deemed creative. Some of the end users

deemed the software to be creative in both the interviews and through the user survey, while other users indicated that the software lacks creativity.

Whether the software is creative or not is objectively inconclusive as several arguments for creativity are subjective arguments.

7.4 Learnings and Experiences

Positive experiences

Experience and the knowledge obtained from the design and implementation “from scratch” is invaluable for both SW development of the new AI systems and the use of the existing systems, tools and libraries. Before reusing the existing functionality, it is always good knowing the essentials and the basic principles.

Intensive state-of-the-art study helped grasping the challenging field, and good suggestions from researchers gave ideas for various experiments and finetuning approaches. Experiments and tests gave very useful information about the behavior of EA and the sensitivity of search finetuning techniques. It will be very useful for future improvements of the EA functionality (fitness techniques, genotype and phenotype manipulation approaches, seeding and cross-over approaches, to mention just a few), but also for the cases where one has to use existing AI libraries.

Challenges

The existing knowledge body and the papers from the research community helped grasping how difficult and enormous the problem of the fitness in art might be. Being highly end-user specific and subjective, but demanding at the same time the artistic aesthetic value, elements of surprise and novelty. The fuzziness of the problem and the topic required creative software engineering approaches.

7.5 Future Prospects

This is work a good basis for further work considering all perspectives (artificial intelligence in art, R&D enabled SW framework, evolutionary algorithms in concrete applications etc.). It is a foundation for interesting improvements, and should be a start of intensive testing, evaluation and

finetuning activities, in order to give good support for various Use Cases (Figure 1). Some good candidates for future work activities, might be:

- Improvements of the evolutionary algorithms,
 - Various new fitness functions,
 - Changing the fitness function during the evolution/search,
 - Implementing the interactive evolutionary algorithms,
 - Experimenting with different mutation strategies,
 - Etc.
- Implementation of alternative AI approaches for solving the music-to-image transformation problem
 - Incorporating constraint satisfaction problems with the EA (Colton, 2012),
 - Including the artificial neural networks during the fitness evaluation.
- This system currently uses linear scaling to calculate how a music parameter value is converted to a tool's parameter value. In the future, this work can take inspiration from artificial neural network activation functions. There are several activation functions in neural networks that can be used: linear functions, binary step, Sigmoid, Gaussian, etc. In combination with the implemented linear scaling, these functions can be used to calculate the output parameter values for different tools. For instance, having a binary step function can act as a conditional function that only draws figures if a given music parameter is above some threshold.
- Interfacing other music and image systems and platforms,
- Interacting with the end-user in new ways – creating web platforms that can learn by interaction with user groups (inspired by Trujillo et al. (2013) and García-Valdez et al. (2013)).

8 CONCLUSION

This thesis explores computer-generated art based on musical input. It combines theoretical and practical approaches. It is a design, implementation and experiments-driven exercise and focuses on the use of evolutionary algorithms in image generation. The end-user involvement, survey results and user interaction tests, contributed to the system design of the evolutionary algorithm functionality, e.g. seeding techniques, feedback functions, and mappings between genotype and phenotype information.

The evolutionary algorithms for music-to-image transformation operated on music and image meta-data, rather than raw data (music and image media). This required good metadata structure and organization, as well as good solutions for metadata and parametric representation of music and images.

The computer system design in this work is organized as a software framework, i.e. it is modular and flexible, with well-defined workflows, interfaces and protocols. This enables that different AI methods and generation tools can be easily applied and facilitates future improvements. On the other side, it is suitable for the interaction with legacy system and services, making it industry-relevant.

Experiment and test-driven development gave not just the insight in the behavior of the evolutionary algorithms, but also sensitivity to various functionality, processes and input parameters.

The most important design tests and experiments focused on: (a) using mapping tables to match music and image features and (b) experiments with various fitness functions that combined user preferences with art novelty aspects.

The obtained experimental results and the end user interaction tests and evaluations are promising and motivate for further development of the tool set, as well as finetuning of the evolutionary algorithms.

The experience and the knowledge obtained from the design and implementation “from scratch” is invaluable for both SW development of new AI systems and the use of existing systems, tools

and libraries. Before reusing existing functionality, it is always good knowing the essentials and basic principles.

The framework enables improvements in several directions. For instance, evolutionary algorithm improvements (different genotype, phenotype and fitness function solutions), alternative AI approaches for solving music-to-image transformation problem, and interfacing other music and image systems and platforms and using additional information and knowledge they can offer. Future efforts might also include:

- Improvements of the evolutionary algorithms, e.g. (a) new fitness functions, (b) changing the fitness function during the evolution/search, (c) implementing interactive evolutionary algorithms, (d) experimenting with different mutation strategies, to mention just a few.
- Implementation of alternative AI approaches for solving music-to-image transformation problems
 - Incorporating constraint satisfaction problems with the evolutionary algorithms (Colton, 2012)
 - Including artificial neural networks during the fitness evaluation
- Interfacing other music and image systems and platforms
- Interacting with the end-user in new ways – creating web platforms that can learn by interaction with user groups (inspired by Trujillo et al. (2013) and García-Valdez et al. (2013)).

BIBLIOGRAPHY

- Adami, C., Ofria, C. & Collier, T.C., (2000). Evolution of biological complexity. *Proceedings of the National Academy of Sciences*, 97, pp.4463-68.
- Andrienko, Y.A., Brilliantov, N.V. & Kurths, J., (2000). Complexity of two-dimensional patterns. *The European Physical Journal B-Condensed Matter and Complex Systems*, 15, pp.539-46.
- Arnheim, R., (1969). *Visual thinking*. Univ of California Press.
- Ashmore, L. & Miller, J., (2004). Evolutionary Art with Cartesian Genetic Programming. *Technical Online Report*.
- Baluja, S., Pomerleau, D. & Jochem, T., (1994). Towards automated artificial evolution for computer-generated images. *Connection Science*, 6, pp.325-54.
- Barkhall, H. et al., (2016). *Growdio - Project Report*. Trondheim: NTNU.
- Bates, J.E. & Shepard, H.K., (1993). Measuring complexity using information fluctuation. *Physics Letters A*, 172, pp.416-25.
- Bense, M., (1969). Kleine abstrakte ästhetik [small abstract aesthetics], Rot edn. *E. Walther, March*, 38.
- Berlyne, D.E., (1971). *Aesthetics and Psychobiology*. New York: AppletonCenturyCrofts.
- Birkhoff, G.D., (1933). *Aesthetic measure*. Cambridge: Harvard University Press.
- Boden, M.A., (1998). Creativity and artificial intelligence. *Artificial Intelligence*, 103(1), pp.347-56.
- Boden, M.A., (2009). Computer models of creativity. *AI Magazine*, 30(3), p.23.
- Bredesen, T. et al., (2016). *Sågawave, Creating visualizations based on attributes in music*. Trondheim: NTNU.
- Britannica, A., (2010). *art, philosophy of*. [Online] Available at: <http://academic.eb.com/levels/collegiate/article/art-philosophy-of/108546> [Accessed 18 April 2017].
- Channon, A.D. & Damper, R.I., (2000). Towards the evolutionary emergence of increasingly complex advantageous behaviours. *International Journal of Systems Science*, 31, pp.843-60.
- Ciesielski, V., Berry, M., Trist, K. & D'Souza, D., 2007. Evolution of animated photomosaics. In *Workshops on Applications of Evolutionary Computation.*, 2007.
- Clune, J. & Lipson, H., 2011. Evolving three-dimensional objects with a generative encoding inspired by developmental biology. In *ECAL.*, 2011.
- Cohen, H., 1988. How to Draw Three People in a Botanical Garden. In *AAAI.*, 1988.

- Colton, S., (2008). Automatic invention of fitness functions, with application to scene generation. In *Workshops on Applications of Evolutionary Computation*. Springer. pp.381-91.
- Colton, S., (2012). The Painting Fool: Stories from Building an Automated Painter. In J. McCormack & M. d'Inverno, eds. *Computers and Creativity*. New York: Springer. pp.3-36.
- Colton, S. & Wiggins, G.A., (2012). Computational creativity: The final frontier? In *ECAI Vol. 12*. pp.21-26.
- Dawkins, R., (1996). *Climbing mount improbable*. New York: W.W. Norton & Company.
- den Heijer, E., (2013). Autonomous Evolutionary Art, Ph.D. Thesis.
- den Heijer, E. & Eiben, A.E., (2010). Comparing Aesthetic Measures for Evolutionary Art. In C. Di Chio et al., eds. *Applications of Evolutionary Computation: EvoApplications 2010: EvoCOMNET, EvoENVIRONMENT, EvoFIN, EvoMUSART, and EvoTRANSLOG, Istanbul, Turkey, April 7-9, 2010, Proceedings, Part II*. Berlin, Heidelberg: Springer Berlin Heidelberg. pp.311-20.
- Dumoulin, V., Shlens, J. & Kudlur, M., (2016). A Learned Representation For Artistic Style. *arXiv*, Available at: <https://arxiv.org/abs/1610.07629>.
- Eiben, A.E. & Smith, J.E., (2015). Interactive evolutionary algorithms. In *Introduction to Evolutionary Computing*. Berlin Heidelberg: Springer. pp.215-22.
- Floreano, D. & Mattiussi, C., (2008a). Artificial Evolution. In *Bio-Inspired Artificial Intelligence*. Cambridge, Massachusetts; London, England: The MIT Press. pp.5-38.
- Fornari, J., Maia Jr, A. & Manzolli, J., 2007. Creating soundscapes using evolutionary spatial control. In *Workshops on Applications of Evolutionary Computation.*, 2007.
- Frade, M., de Vega and Fernandez, F. & Cotta, C., 2010. Evolution of artificial terrains for video games based on accessibility. In *European Conference on the Applications of Evolutionary Computation.*, 2010.
- Galanter, P., (2010). The Problem with Evolutionary Art Is. In C. Di Chio et al., eds. *Applications of Evolutionary Computation: EvoApplications 2010: EvoCOMNET, EvoENVIRONMENT, EvoFIN, EvoMUSART, and EvoTRANSLOG, Istanbul, Turkey, April 7-9, 2010, Proceedings, Part II*. Berlin, Heidelberg: Springer Berlin Heidelberg. pp.321-30.
- Galanter, P., (2012). Computational aesthetic evaluation: Past and future. In *Computers and Creativity*. Springer. pp.255-93.
- Galván-López, E., McDermott, J., O'Neill, M. & Brabazon, A., (2011). Defining locality as a problem difficulty measure in genetic programming. *Genetic Programming and Evolvable Machines*, 12, pp.365-401.

- Galvan, E., Trujillo, L., McDermott, J. & Kattan, A., (2013). Locality in continuous fitness-valued cases and genetic programming difficulty. In *EVOLVE-A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation II*. Springer. pp.41-56.
- García-Valdez, M. et al., 2013. Evospace-interactive: A framework to develop distributed collaborative-interactive evolutionary algorithms for artistic design. In *International Conference on Evolutionary and Biologically Inspired Music and Art.*, 2013.
- Gedeon, T.T.D., 2007. Neural network for modeling esthetic selection. In *International Conference on Neural Information Processing.*, 2007.
- Gould, S.J., (1998). Full house: the spread of excellence from Plato to Darwin. *Senior Managing Editor*, 5(2), p.68.
- Greenfield, G., (2006). Robot paintings evolved using simulated robots. In F. Rothlauf et al., eds. *Applications of Evolutionary Computing: EvoWorkshops 2006: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoINTERACTION, EvoMUSART, and EvoSTOC, Budapest, Hungary, April 10-12, 2006, Proceedings*. Heidelberg: Springer. pp.611-21.
- Heggedal, A. et al., (2016). *WordPainter*. Trondheim: NTNU.
- Hoover, A.K., Szerlip, P.A. & Stanley, K.O., 2011. Generating musical accompaniment through functional scaffolding. In *Proceedings of the Eighth Sound and Music Computing Conference (SMC 2011).*, 2011.
- Jaśkowski, W., Krawiec, K. & Wieloch, B., 2007. Learning and recognition of hand-drawn shapes using generative genetic programming. In *Workshops on Applications of Evolutionary Computation.*, 2007.
- Javid, M.A.J., al Rifaie, M.M. & Zimmer, R., 2015. An informational model for cellular automata aesthetic measure. In *The AISB15's 2nd International Symposium on Computational Creativity (CC2015).*, 2015.
- Johnson, R.E., (1997). Components, frameworks, patterns. *ACM SIGSOFT Software Engineering Notes*, 22(3), pp.10-17.
- Johnson, C.G., (2012). Connotation in Computational Creativity. *Cognitive Computation*, 4, pp.280-91.
- Johnson, C.G., (2016). Fitness in evolutionary art and music: a taxonomy and future prospects. *International Journal of Arts and Technology*, 9, pp.4-25.
- Kampis, G. & Gulyás, L., (2008). Full body: The importance of the phenotype in evolution. *Artificial Life*, 14, pp.375-86.
- Kaya, N. & Epps, H., (2004). *Relationship between color and emotion: A study of college students*. Georgia: The University of Georgia.
- Klinger, A. & Salingaros, N.A., (2000). A pattern measure. *Environment and Planning B: Planning and Design*, 27, pp.537-47.
- Kötter, T. & Berthold, M.R., (2012). (Missing) concept discovery in heterogeneous information networks. In *Bisociative Knowledge Discovery*. Springer. pp.230-45.

- Kowaliw, T., Dorin, A. & McCormack, J., 2009. An empirical exploration of a definition of creative novelty for generative art. In *Australian Conference on Artificial Life.*, 2009.
- Lehman, J. & Stanley, K.O., 2008. Exploiting Open-Endedness to Solve Problems Through the Search for Novelty. In *ALIFE.*, 2008.
- Lehman, J. & Stanley, K.O., 2010. Revising the evolutionary computation abstraction: minimal criteria novelty search. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation.*, 2010.
- Lewis, M., (2008). Evolutionary visual art and design. In *The art of artificial evolution.* Springer. pp.3-37.
- Liapis, A., Martínez, H.P., Togelius, J. & Yannakakis, G.N., 2013. Adaptive game level creation through rank-based interactive evolution. In *Computational Intelligence in Games (CIG), 2013 IEEE Conference on.*, 2013.
- Liapis, A., Yannakakis, G.N. & Togelius, J., 2013. Sentient world: Human-based procedural cartography. In *International Conference on Evolutionary and Biologically Inspired Music and Art.*, 2013.
- Li, Y. & Hu, C.-J., 2010. Aesthetic learning in an interactive evolutionary art system. In *European Conference on the Applications of Evolutionary Computation.*, 2010.
- Lynch, M., (2007). The frailty of adaptive hypotheses for the origins of organismal complexity. *Proceedings of the National Academy of Sciences*, 104, pp.8597-604.
- Machado, P. & Cardoso, A., 1998. Computing aesthetics. In *Brazilian Symposium on Artificial Intelligence.*, 1998.
- Machado, P. & Cardoso, A., (2002). *All the truth about NEvAr.* Portugal: Applied Intelligence 16.
- Machado, P., Martins, T., Amaro, H. & Abreu, P.H., (2016). Beyond Interactive Evolution: Expressing Intentions Through Fitness Functions. *Leonardo*, 49, pp.251-56. Available at: http://dx.doi.org/10.1162/LEON_a_01103.
- Machado, P., Romero, J. & Manaris, B., (2008). Experiments in Computational Aesthetics. In J. Romero & P. Machado, eds. *The art of artificial evolution: a handbook on evolutionary art and music.* Berlin: Springer.
- Machado, P. et al., (2007). On the development of evolutionary artificial artists. *Computers & Graphics*, 31, pp.818-26.
- Majidal-Rifaie, M., 2016. Correlation Between Human Aesthetic Judgement and Spatial Complexity Measure. In *Evolutionary and Biologically Inspired Music, Sound, Art and Design: 5th International Conference, EvoMUSART 2016, Porto, Portugal, March 30--April 1, 2016, Proceedings.*, 2016.
- McCormack, J., 2005. Open problems in evolutionary music and art. In *Workshops on Applications of Evolutionary Computation.*, 2005.

- McDermott, J., Griffith, N.J.L. & O'Neill, M., (2005). Toward user-directed evolution of sound synthesis parameters. *LNCS*, 3449, pp.517–26.
- Miconi, T., (2008). Evolution and complexity: The double-edged sword. *Artificial life*, 14, pp.325-44.
- Mills, A., 2016. Animating Typescript Using Aesthetically Evolved Images. In *International Conference on Evolutionary and Biologically Inspired Music and Art.*, 2016.
- Moles, A.A., (1968). *Information Theory and Esthetic Perception: Transl. by Joel E. Cohen.* University of Illinois Press.
- Monmarché, N., Mahnich, I. & Slimane, M., (2008). Painting Ants. In J. Romero & P. Machado, eds. *The Art of Artificial Evolution, A Handbook on Evolutionary Art and Music.* Berlin: Springer. pp.235-47.
- Mordvintsev, A., Olah, C. & Tyka, M., (2015). *Inceptionism: Going Deeper into Neural Networks.* [Online] Available at: <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html> [Accessed 1 December 2016].
- Nagel, U. et al., 2011. Bisociative discovery of interesting relations between domains. In *International Symposium on Intelligent Data Analysis.*, 2011.
- Nixon, M.S. & Aguado, A.S., (2008). *Feature Extraction and Image Processing.* Academic Press.
- Østdahl, A. et al., (2016). *Pixel-ate.it.* Trondheim: NTNU.
- Re, A., Castelli, M. & Vanneschi, L., 2016. A Comparison Between Representations for Evolving Images. In *International Conference on Evolutionary and Biologically Inspired Music and Art.*, 2016.
- Riehle, D., (2000). *Framework Design: A Role Modeling Approach.* Ph.D. Thesis, No. 13509. Zürich, Switzerland: ETH Zürich.
- Rigau Vilalta, J., Feixas Feixas, M. & Sbert, M., (2008). Informational Aesthetics Measures. *IEEE Computer Graphics and Applications*, 28(2), pp.24-34.
- Rigau, J., Feixas, M. & Sbert, M., 2007. Conceptualizing Birkhoff's Aesthetic Measure Using Shannon Entropy and Kolmogorov Complexity. In *Computational Aesthetics.*, 2007.
- Ross, B.J., Ralph, W. & Zong, H., 2006. Evolutionary image synthesis using a model of aesthetics. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on.*, 2006.
- Rumbaugh, J., Jacobson, I. & Booch, G., (2004). *The Unified Modelling Language, SBN 0-321-24562-8.* Second Edition ed. Addison-Wesley.
- Sandve, J., Mathisen, T., Hjermand, T. & Olsen, T., (2016). *Computer generated music with basis in visual input.* Trondheim: NTNU.
- Saunders, R. & Gero, J.S., 2001. The digital clockwork muse: A computational model of aesthetic evolution. In *Proceedings of the AISB.*, 2001.

- Saunders, R. & Gero, J.S., (2004). Curious agents and situated design evaluations. *AI EDAM: Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 18, pp.153-61.
- Schmidt, D.C., (1997). Applying patterns and frameworks to develop object-oriented communication software. *Handbook of programming languages*, 1.
- Secretan, J. et al., (2011). Picbreeder: A case study in collaborative evolutionary exploration of design space. *Evolutionary Computation*, 19, pp.373-403.
- Semet, Y., (2002). Interactive Evolutionary Computation: a survey of existing theory. *University of Illinois*.
- Silberschatz, A. & Tuzhilin, A., (1996). What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and data engineering*, 8, pp.970-74.
- Sims, K., (1991). *Artificial evolution for computer graphics*. ACM.
- Spehar, B., Clifford, C.W.G., Newell, B.R. & Taylor, R.P., (2003). Universal aesthetic of fractals. *Computers & Graphics*, 27, pp.813-20.
- Stadler, P.F., (2002). Fitness landscapes. In *Biological evolution and statistical physics*. Springer. pp.183-204.
- Standish, R.K., (2003). Open-ended artificial evolution. *International Journal of Computational Intelligence and Applications*, 3(02), pp.167-75.
- Takagi, H., (2001). Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. *Proceedings of the IEEE*, 89, pp.1275-96.
- Todd, S. & Latham, W., (1992). *Evolutionary art and computers*. London: Academic Press, Inc.
- Todd, P.M. & Werner, G.M., (1999). Frankensteinian methods for evolutionary music. *Musical networks: parallel distributed perception and performance*, pp.313-40.
- Trujillo, L., García-Valdez, M., Fernández-de-Vega, F. & Merelo, J.-J., 2013. Fireworks: Evolutionary art project based on evospace-interactive. In *Evolutionary Computation (CEC), 2013 IEEE Congress on.*, 2013.
- Vinhas, A. et al., 2016. Fitness and novelty in evolutionary art. In *International Conference on Evolutionary and Biologically Inspired Music and Art.*, 2016.
- Vitányi, M.L.a.P., (2008). *An Introduction to Kolmogorov Complexity and Its Applications*. 3rd ed. Springer.
- Wackerbauer, R. et al., (1994). A comparative classification of complexity measures. *Chaos, Solitons & Fractals*, 4, pp.133-73.
- Whitelaw, M., (2004). *Metacreation: art and artificial life*. Mit Press.
- Wilson, D.J., (1939). An experimental investigation of Birkhoff's aesthetic measure. *The Journal of Abnormal and Social Psychology*, 34, p.390.

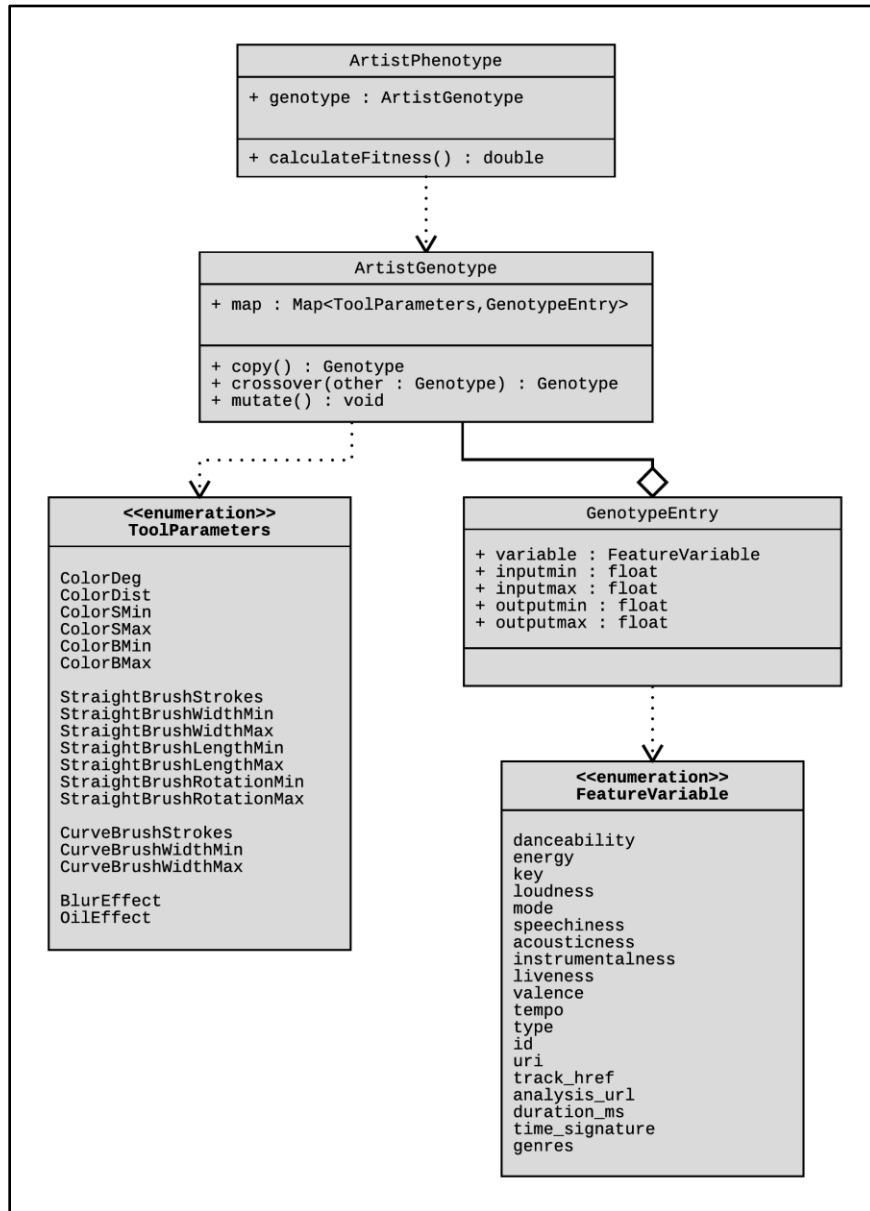
- Woolley, B.G. & Stanley, K.O., 2011. On the deleterious effects of a priori objectives on evolution and representation. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation.*, 2011.
- Woolley, B.G. & Stanley, K.O., (2012). Exploring promising stepping stones by combining novelty search with interactive evolution. *arXiv preprint arXiv:1207.6682*.
- Xu, Q., D'Souza, D. & Ciesielski, V., (2007). *Evolving images for Entertainment*. Australia: RMIT University.
- Yuan, J. & Gong, D., 2008. Large population size IGAs with individuals' fitness not assigned by user. In *International Conference on Intelligent Computing.*, 2008.
- Zhang, J., Taarnby, R., Liapis, A. & Risi, S., 2015. DrawCompileEvolve: Sparking interactive evolutionary art with human creations. In *International Conference on Evolutionary and Biologically Inspired Music and Art.*, 2015.
- Zurek, W.H., (1989). Algorithmic randomness and physical entropy. *Physical Review A*, 40, p.4731.

APPENDIX A: SPOTIFY AUDIO FEATURES TABLE

KEY	VALUE DESCRIPTION
acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.
duration_ms	The duration of the track in milliseconds.
energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
instrumentalness	Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
key	The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C#/Db, 2 = D, and so on.
liveness	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
loudness	The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). Values typical range between -60 and 0 db.
mode	Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
speechiness	Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.
tempo	The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
time_signature	An estimated overall time signature of a track. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure).
valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

APPENDIX B: PHENOTYPE & GENOTYPE

UML Diagram of the phenotype and genotype classes



APPENDIX C: SPOTIFY ANALYSIS RESULTS

Relevant data from Spotify audio features objects for used songs. Data description in Appendix A.

<p>Billie Jean</p> <p>danceability = 0.92 energy = 0.654 key = 11 loudness = -3.051 mode = 0 speechiness = 0.0401 acousticness = 0.0236 instrumentalness = 0.0158 liveness = 0.0359 valence = 0.856 tempo = 117.046 id = 5ChkMS80tdzJeqyybCc9R5 duration_ms = 293827.0 time_signature = 4</p>	<p>Chained To The Rhythm</p> <p>danceability = 0.451 energy = 0.809 key = 0 loudness = -5.327 mode = 1 speechiness = 0.141 acousticness = 0.073 instrumentalness = 0.0 liveness = 0.201 valence = 0.468 tempo = 189.888 id = 5pD9x23lsoQr464mlmaKNJ duration_ms = 237915.0 time_signature = 4</p>
<p>I Promise</p> <p>danceability = 0.304 energy = 0.248 key = 11 loudness = -29.531 mode = 1 speechiness = 0.0461 acousticness = 0.972 instrumentalness = 0.65 liveness = 0.129 valence = 0.14 tempo = 60.419 id = 2L8ZiTLLejiY907qhhoday duration_ms = 265142.0 time_signature = 4</p>	<p>Kaleidoscope</p> <p>danceability = 0.538 energy = 0.452 key = 4 loudness = -6.67 mode = 0 speechiness = 0.0343 acousticness = 0.0754 instrumentalness = 0.0279 liveness = 0.107 valence = 0.0655 tempo = 160.119 id = 5f2CgxUSiRtuEMcY6mmAW4 duration_ms = 301340.0 time_signature = 4</p>
<p>No Time for Caution</p> <p>danceability = 0.207 energy = 0.101 key = 0 loudness = -28.244 mode = 1 speechiness = 0.0538 acousticness = 0.964 instrumentalness = 0.92 liveness = 0.197 valence = 0.023 tempo = 66.785 id = 2iNRom9yU8lY7SEw6JCiBA duration_ms = 146182.0 time_signature = 4</p>	<p>The Imperial March (Darth Vader's Theme)</p> <p>danceability = 0.585 energy = 0.354 key = 7 loudness = -14.02 mode = 0 speechiness = 0.0304 acousticness = 0.948 instrumentalness = 0.941 liveness = 0.0998 valence = 0.309 tempo = 103.256 id = 6qTJtlUrLTrZLUuS81SPI7 duration_ms = 183733.0 time_signature = 4</p>

APPENDIX D: USER SURVEY

5/7/2017

Music-to-Image Survey

Music-to-Image Survey

* Required

Billie Jean - Michael Jackson -

<https://open.spotify.com/track/5ChkMS8OtdzJeqyybCc9R5>

1. Choose one color that fits the music *



0 1 2 3 4 5 6 7 8 9 10

Mark only one oval.

0 1 2 3 4 5 6 7 8 9 10

2. The above selected color should be *

Mark only one oval.

- Darker
- As it is
- Lighter

3. How do you expect the image to be? *

Mark only one oval.

1 2 3 4 5

Relaxing Aggressive

4. How many brush strokes? One brush stroke is one continuous line on the canvas. *

Mark only one oval.

1 2 3 4 5 6 7 8 9 10

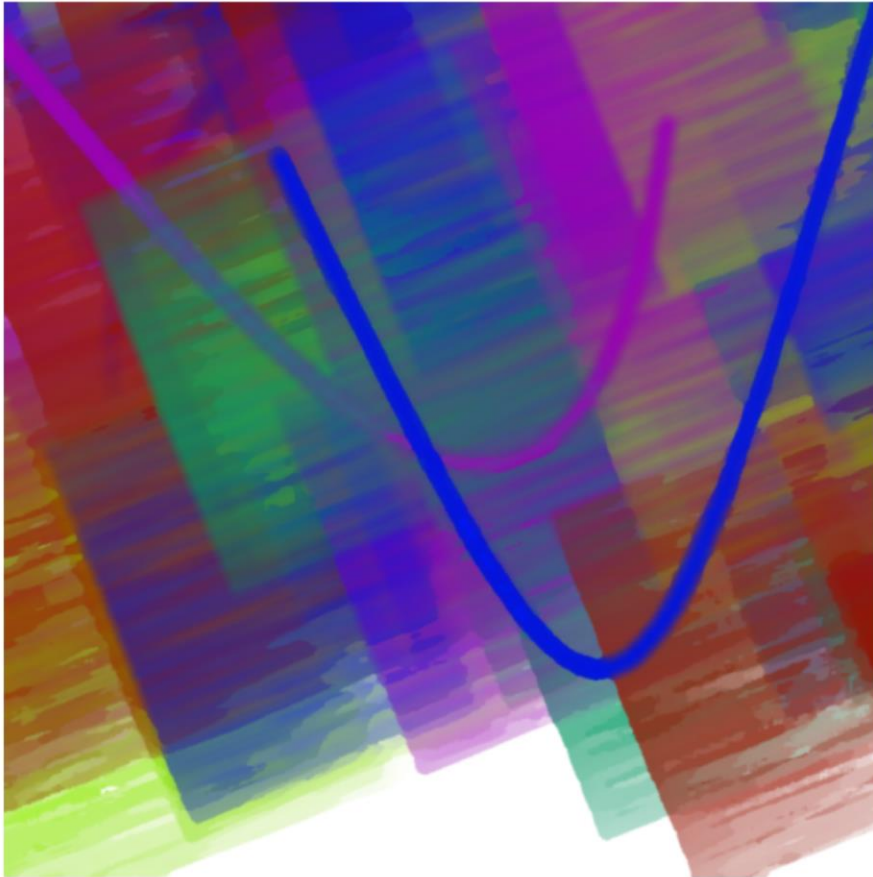
Few brush strokes leaving much empty space on the canvas Lots of brush strokes filling the whole canvas

5. How should the coloring be? *

Mark only one oval.

- One color with small changes in shades
- Few adjacent colors with small changes in shades
- Many different colors with small changes in shades
- One color with high variance in shades
- Few adjacent colors with high variance in shades
- Many different colors with high variance in shades

6. How well does this image fit this song? *



Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Not well at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very well

7. What do you like about this image, with respect to the music? *

8. What do you dislike about this image, with respect to the music? *

9. What would you change in order for the image to better fit the music? *

Chained To The Rhythm – Katy Perry -

<https://open.spotify.com/track/5pD9x23IsoQr464mlmaKNJ>

10. Choose one color that fits the music *



0 1 2 3 4 5 6 7 8 9 10

Mark only one oval.

0 1 2 3 4 5 6 7 8 9 10

11. The above selected color should be *

Mark only one oval.

- Darker
- As it is
- Lighter

12. How do you expect the image to be? *

Mark only one oval.

1 2 3 4 5

Relaxing Aggressive

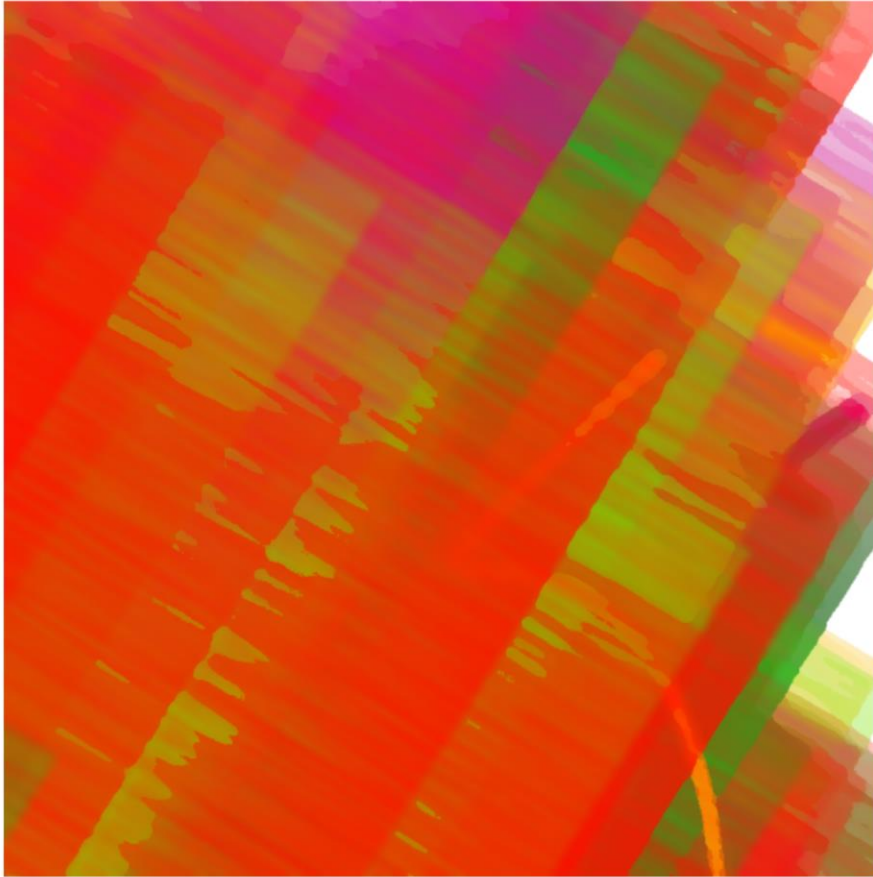
13. How many brush strokes? One brush stroke is one continuous line on the canvas. **Mark only one oval.*

	1	2	3	4	5	6	7	8	9	10	
Few brush strokes leaving much empty space on the canvas	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Lots of brush strokes filling the whole canvas

14. How should the coloring be? **Mark only one oval.*

- One color with small changes in shades
- Few adjacent colors with small changes in shades
- Many different colors with small changes in shades
- One color with high variance in shades
- Few adjacent colors with high variance in shades
- Many different colors with high variance in shades

15. How well does this image fit this song? *



Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Not well at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very well

16. What do you like about this image, with respect to the music? *

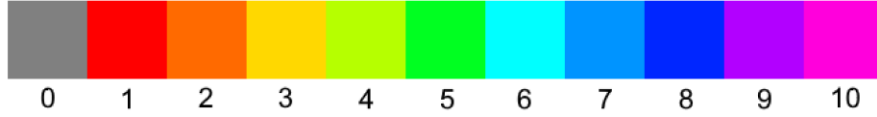
17. What do you dislike about this image, with respect to the music? *

18. What would you change in order for the image to better fit the music? *

I Promise – Alex Kozobolis -

<https://open.spotify.com/track/2L8ZiTLLejiY9O7qhhoday>

19. Choose one color that fits the music *



Mark only one oval.

0 1 2 3 4 5 6 7 8 9 10

20. The above selected color should be *

Mark only one oval.

- Darker
- As it is
- Lighter

21. How do you expect the image to be? *

Mark only one oval.

1 2 3 4 5

Relaxing Agressive

22. How many brush strokes? One brush stroke is one continuous line on the canvas. *

Mark only one oval.

1 2 3 4 5 6 7 8 9 10

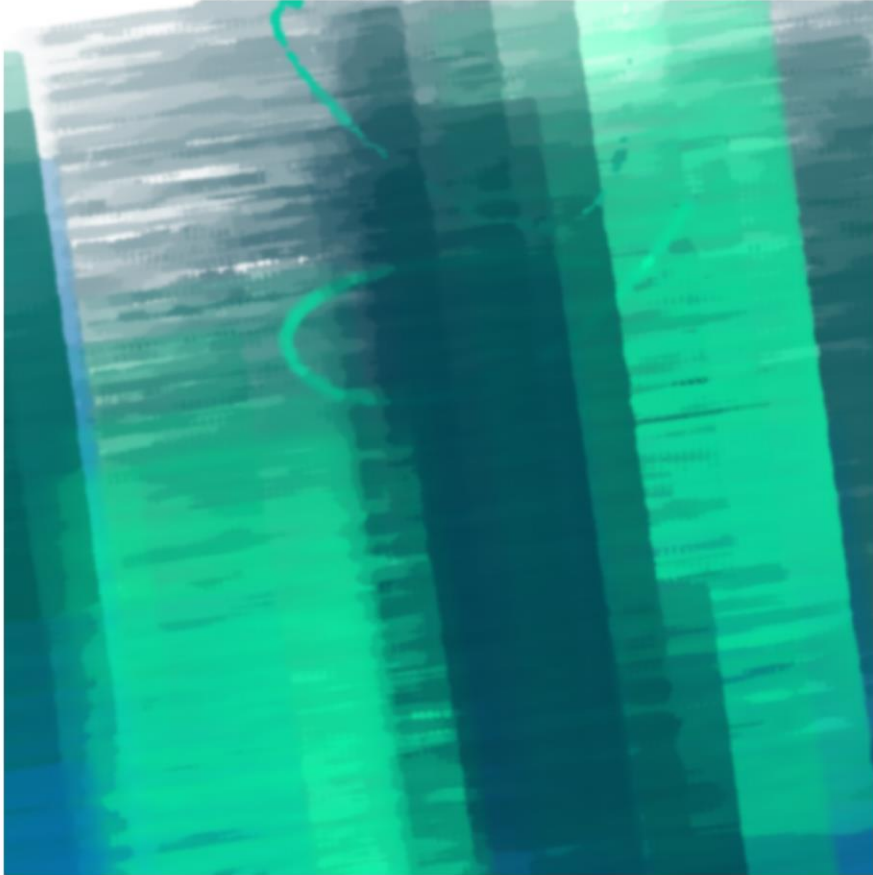
Few brush strokes leaving much empty space on the canvas Lots of brush strokes filling the whole canvas

23. How should the coloring be? *

Mark only one oval.

- One color with small changes in shades
- Few adjacent colors with small changes in shades
- Many different colors with small changes in shades
- One color with high variance in shades
- Few adjacent colors with high variance in shades
- Many different colors with high variance in shades

24. How well does this image fit this song? *



Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Not well at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very well

25. What do you like about this image, with respect to the music? *

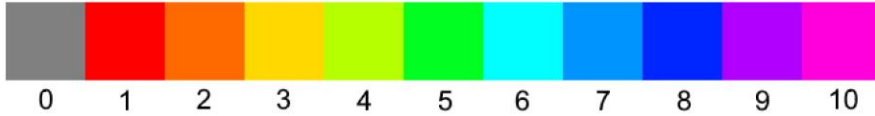
26. What do you dislike about this image, with respect to the music? *

27. What would you change in order for the image to better fit the music? *

Kaleidoscope – Kasbo -

<https://open.spotify.com/track/5f2CgxUSiRtuEMcY6mmAW4>

28. Choose one color that fits the music *



Mark only one oval.

0 1 2 3 4 5 6 7 8 9 10

29. The above selected color should be *

Mark only one oval.

- Darker
- As it is
- Lighter

30. How do you expect the image to be? *

Mark only one oval.

1 2 3 4 5

Relaxing Aggressive

31. How many brush strokes? One brush stroke is one continuous line on the canvas. **Mark only one oval.*

	1	2	3	4	5	6	7	8	9	10	
Few brush strokes leaving much empty space on the canvas	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Lots of brush strokes filling the whole canvas

32. How should the coloring be? **Mark only one oval.*

- One color with small changes in shades
- Few adjacent colors with small changes in shades
- Many different colors with small changes in shades
- One color with high variance in shades
- Few adjacent colors with high variance in shades
- Many different colors with high variance in shades

33. How well does this image fit this song? *



Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Not well at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very well

34. What do you like about this image, with respect to the music? *

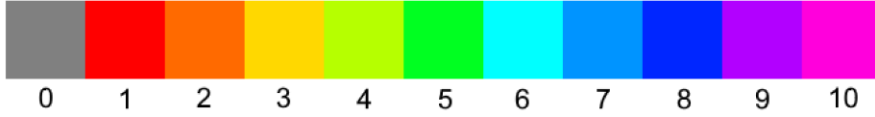
35. What do you dislike about this image, with respect to the music? *

36. What would you change in order for the image to better fit the music? *

No Time for Caution – Hans Zimmer -

<https://open.spotify.com/track/2iNRom9yU8lY7SEw6JCiBA>

37. Choose one color that fits the music *



Mark only one oval.

0 1 2 3 4 5 6 7 8 9 10

38. The above selected color should be *

Mark only one oval.

- Darker
- As it is
- Lighter

39. How do you expect the image to be? *

Mark only one oval.

1 2 3 4 5

Relaxing Aggressive

40. How many brush strokes? One brush stroke is one continuous line on the canvas. *

Mark only one oval.

1 2 3 4 5 6 7 8 9 10

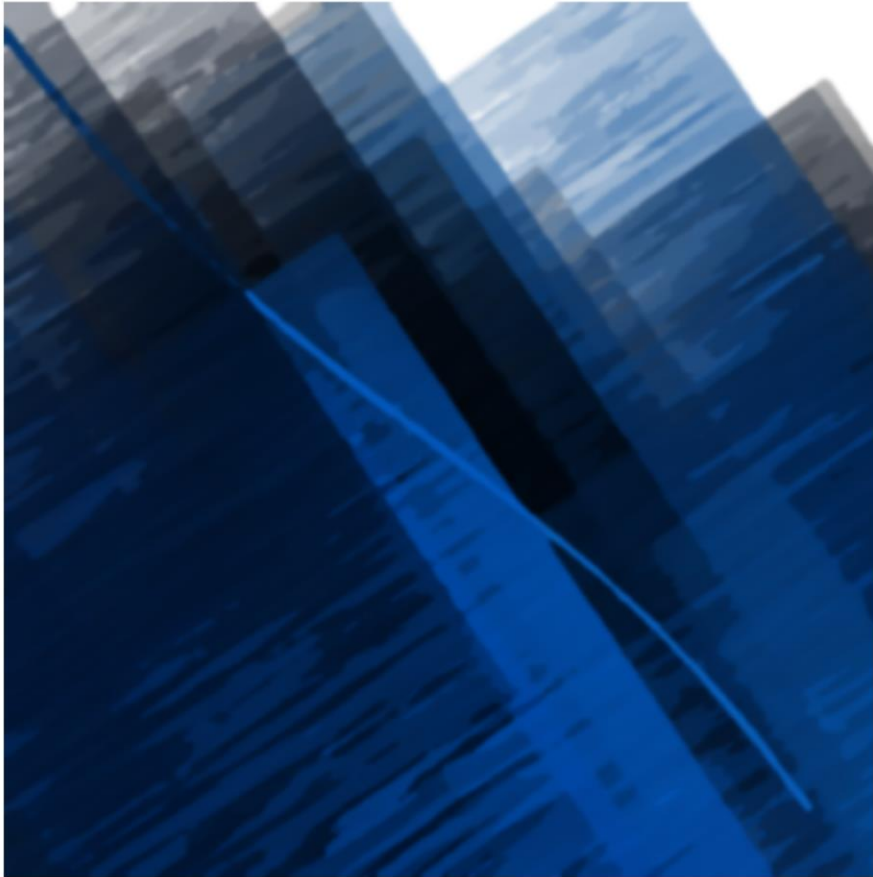
Few brush strokes leaving much empty space on the canvas Lots of brush strokes filling the whole canvas

41. How should the coloring be? *

Mark only one oval.

- One color with small changes in shades
- Few adjacent colors with small changes in shades
- Many different colors with small changes in shades
- One color with high variance in shades
- Few adjacent colors with high variance in shades
- Many different colors with high variance in shades

42. How well does this image fit this song? *



Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Not well at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very well

43. What do you like about this image, with respect to the music? *

44. What do you dislike about this image, with respect to the music? *

45. What would you change in order for the image to better fit the music? *

The Imperial March (Darth Vader's Theme) – John Williams - <https://open.spotify.com/track/6qTJtIUrLTrZLUuS81SPI7>

46. Choose one color that fits the music *



Mark only one oval.

0 1 2 3 4 5 6 7 8 9 10

47. The above selected color should be *

Mark only one oval.

- Darker
- As it is
- Lighter

48. How do you expect the image to be? *

Mark only one oval.

1 2 3 4 5

Relaxing Aggressive

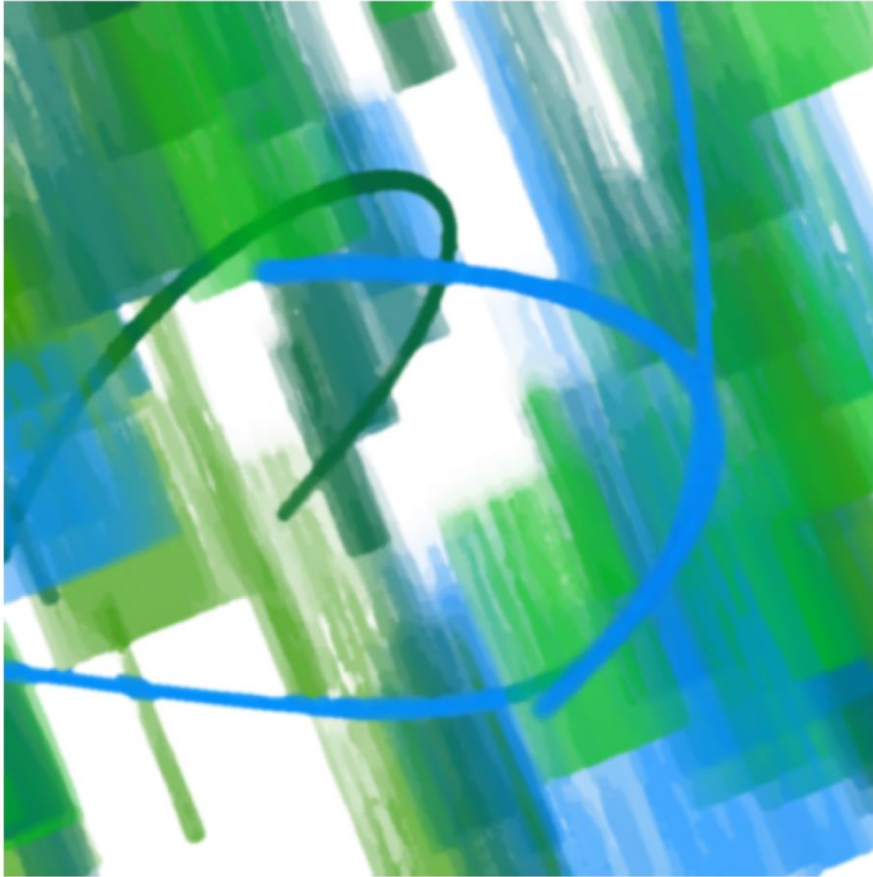
49. How many brush strokes? One brush stroke is one continuous line on the canvas. **Mark only one oval.*

	1	2	3	4	5	6	7	8	9	10	
Few brush strokes leaving much empty space on the canvas	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Lots of brush strokes filling the whole canvas

50. How should the coloring be? **Mark only one oval.*

- One color with small changes in shades
- Few adjacent colors with small changes in shades
- Many different colors with small changes in shades
- One color with high variance in shades
- Few adjacent colors with high variance in shades
- Many different colors with high variance in shades

51. How well does this image fit this song? *



Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Not well at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very well

52. What do you like about this image, with respect to the music? *

53. What do you dislike about this image, with respect to the music? *

54. What would you change in order for the image to better fit the music? *

Thank you for your time

55. Is there anything you would change about this survey? If yes, please give a short explanation

Powered by
 Google Forms