Evolutionary music generation based on geometry*

1st Justus Huebotter
MSc Artificial Intelligence
Vrije Universiteit
Amsterdam, Netherlands
huebotter@outlook.com

2nd Thomas Maaiveld
MSc Artificial Intelligence
Vrije Universiteit
Amsterdam, Netherlands
tmaaiveld@gmail.com

3rd Stefan Wijtsma
MSc Artificial Intelligence
Vrije Universiteit
Amsterdam, Netherlands
stefanwijtsma@gmail.com

***Abstract*—The aim of this paper is to explore methods of generating changing and interesting music with using evolutionary algorithms. Specifically, we exploit the concept of Sacred Geometry and its relations to musical harmony. We propose a framework that uses visualizations of rotating geometric shapes as a trigger for live music generation. An adapted evolutionary programming strategy is used let the system evolve and create changing, interesting and aesthetically pleasing or appealing music in real time. We will describe our setup, and further elaborate on how we adapted the standard evolutionary strategies for the case of generating music from geometry. The main contribution of this paper is exploratory, it being a proof of concept for evolving music from geometry. We'll describe the experiments we did using a multi-population representation, various selection mechanisms, and multi-objective fitness evaluation that is based on a moving optimum.**

***Index Terms*—evolutionary computing, evolutionary art, evolutionary strategies, music generation**

I. Introduction

Evolutionary algorithms — computer systems that take their inspiration from Darwinian evolution — have been an inspiration for many artistic and musical projects [?], [?], [?]. The field of evolutionary art applies computational methods from Evolutionary Computing to create interesting and beautiful artistic expressions, that come in the form of images, video, audio, text, animations or complete performances. Many researchers as well as artists have recombined, modified, and extended these techniques, beginning the exploration of possible applications of evolution to aesthetic design [?]. Central in evolutionary approaches is the concept of fitness, as one needs some way of determining which individuals from the population survive. Therefore, the field of evolutionary art is invested in translating aesthetics into well-defined measurements that could be used for fitness evaluation. In other words, a key aspect of this field is finding ways of converting "what is beautiful" into a mathematical function that describes what is aesthetically pleasant to the human. Modelling this function may be achieved by having a "human in the loop", such as learning optimal aesthetic parameters by estimating fitness through feedback from a user or system designer. Evolving aesthetically pleasing art without a human in the loop presents unique challenges when combined with a sufficiently large search space to allow for creativity and when applied in a domain where aesthetic measures of quality are abstract and poorly defined. Moreover, in a setting of continuous music generation, the optimal fitness (what is considered pleasant) might even change over time.

One way of avoiding having a human in the loop in harmonic music generation is by exploiting the concept of "Sacred Geometry" [?], [?], describing the similarities of mathematical relations in geometry and harmonics in music. The key idea is to let the relations between different geometric shapes form a basis for musical tonality and rhythm. This paper proposes a framework for applying evolutionary programming techniques to combine music and geometry into an evolving art installation. The research is focused towards creating a system able to generate and evolve music using fixed aesthetic measures of fitness inspired by geometrical proportions. The goal is to explore methods of music generation and fitness estimation without human interference in order to take a step towards completely autonomous, continuous music generation. The experiments conducted are aimed at answering the following research questions:

1) What properties of geometry are translatable into the domain of music creation?
2) How can evolutionary methods help to generate harmonic and dynamic music?
3) Can an evolutionary algorithm generate musical novelty, interest and structure through a predefined set of geometrical rules?

This paper presents related work in the domain of evolutionary art, gives a complete description of our implementation and the adapted evolutionary programming strategy, and present experiments with different parameter settings. In conclusion, findings are discussed to formulate a conclusion and interesting directions for future research.

II. Related Work

Evolutionary programming has proven to be fruitful technique to simulate evolution computationally [?]. A summary review of Johnson *et al.* provides an overview of previous research and existing implementations concerning the formulation and measurement of aesthetic fitness measures in evolutionary art [?]. The review outlines a variety of theoretical approaches to measuring aesthetic fitness. Theory of form within evolutionary art, focused on "use of symmetry, and the balance between order and complexity", functioned as a theoretical resource for the implementation described in this paper, as encoding aspects of form algorithmically and using measures of form as fitness drivers for evolution were considered key to answering the research questions specified in the introduction. McCormack specified five open questions in evolutionary music and art, of which the second and third questions, designing a representable aesthetic fitness function producing recognizably artistic works by evolutionary means, are directly relevant to this research [?].

Johnson's taxonomy classifies the body of literature describing previous implementations by fitness scope and fitness measure [?]. Situating this implementation within the "Aesthetic Measure" category of this taxonomy allowed for the investigation of similar existing implementations. Some previous evolutionary music implementations applied abstracted statistical measures to quantify the aesthetic value of the produced music, which could be derived either by using human response [?] [?] [?] or by examining properties inherent to the music [?] [?]. Others were focused on directly evaluating musical attributes. A multitude of melodic and rhythmic features found in the literature are listed in [?]. Bilotta *et al.* implemented a distribution-based musical sequence generator with qualitative fitness measures by devising a scoring scheme to examine consonance between different notes of the Western chromatic scale [?]. Browne and Fox used a pre-trained, undirected neural network to measure tension and release in evolved compositions [?] [?]. In these implementations, fitness evaluation is often focused around the elimination of bad results, relying on rich representation and a broad search space to generate musical interest.

Previous work concerning implementation of rhythmic aesthetic is sparse. The aforementioned implementations omitted rhythmic information as part of an aesthetic measure and focused on consonance. Horowitz utilized an interactive genetic algorithm to model and parametrize user preference of rhythm [?], but lack of previous examples of formal fitness estimation necessitated a novel approach to estimating the fitness of rhythmic combinations. Nonetheless, geometric proportions and their relationship to musical rhythm have been explored in the domain of computer science [?] and provided a useful basis for constructing music through a representation that may be evolved. The other approaches provided general inspiration for some of the attributes of the evolutionary algorithm and fitness measure. Challenges specific to the representation, implementation and research objectives specified in this paper necessitated adaptation and exploration of novel approaches to evolution and fitness measurement. The style of implementation and using a graphical interface with rotating polygons and UI control surface was inspired by the work of musician and sound artist Rui Gato[1].

III. System Description

The system architecture consists of several elements, visualised in Figure 1. The system code was written in Python 3, while music generation was performed using Sonic Pi[2], an open source music tool allowing for "live coding". Music is generated by triggering of real-time music events through internal OSC commands from Python to Sonic Pi. Apart from some Python library dependencies, the system runs on a standard Python distribution on Windows and MacOS using the default Sonic Pi synthesizers. Sonic Pi also plays custom samples, some of which are included in the git repository[3].

This section gives an overview of the system implementation and describes the procedure for producing sound from evolved geometric shapes, followed by a specification of the genotypic and phenotypic representations of the individuals and a description of the population structure. The algorithmic procedure and equations utilized to perform variation and selection are described to give an overview of the evolutionary algorithm. Lastly, the implementation of presets containing initialization parameters and a sequential parameter control system ('autopilot') will be described, which enable the system to create continuous, dynamically evolving music.

Rotating polygons are visualized on a screen and shown to the user. They trigger a musical note whenever one of their vertices crosses a vertical line through the middle of the upper half of the display. To determine what polygons should be played, the system reads from an array containing the phenotypes of genes selected to be played by the evolutionary algorithm. The process is initialized using a set of initial parameters retrieved from the configuration file of the active preset, and . The preset can be set using a GUI and can be written to by the autopilot.

A. From Shape to Sound

The display system visualizes individuals as polygonal shapes with vertex counts (hereafter referred to as the 'order') ranging from 3 to 12. Each polygon may have several replicated polygons circumscribing it, meaning the angles of the inner polygon are tangent to the sides of the outer polygon. The properties of the shapes are derived from the phenotype representation of corresponding individuals currently being played. Apart from the order, some important geometric qualities encoded in the phenotype are the radius, rotation speed, (relative) offset and number of copies to be produced. The radius of the polygon is related to the frequency of the played tone. The radius of a shape is set to correspond with a note of the chromatic scale; a larger radius results in a higher tone played. Any shape may assume a radius corresponding to any note.

The phenotype of an individual may also encode a mapping for a series of 'copies', circumscribed shapes increasing in size. The amount of circumscribed shapes is encoded as the number of copies to be produced ('number'), . For instance, a phenotype could encode a series of hexagons, which will result in a tonal pattern of notes increasing by a fixed interval. The width of the interval depends on the order of the shape; the higher the order, the smaller the increment in the radius of the circumscribed copies. The ratio between the radius of

[1]http://www.ruigato.info/blog/. A workshop and live demonstration of his work can be found at https://www.youtube.com/watch?v=7UwhtvD4s9Q

[2]http://sonic-pi.net/

[3]https://github.com/st33f/evo-art-yoshi

a polygon and and the radius of its circumscribed polygon is equivalent to the ratio between the apothem and radius of the circumscribed polygon (since its apothem is equivalent to the radius of the inscribed polygon). This ratio depends on the order of the shape and is given by $\cos\left(\frac{180°}{n}\right)$, where n represents the number of vertices. This ratio corresponds with a tonal interval, meaning the interval differs for polygons of varying orders. Thus, a different order produces a different repeating harmonic pattern of intervals (the note corresponding to the innermost shape). Furthermore, the angular orientation of the shapes' vertices relative to the innermost shape is also rationally divided for circumscribed shapes (by ratios of 2, 3 or 4), resulting in a regular rhythm. Thus, the rhythm of the music is determined by the order, number, rotation rate and offset parameters of the playing polygons; higher values result in a more densely populated rhythmic axis. By combining multiple polygons with different amounts of vertices, and creating multiple copies of the same shape with an increasing size, a wide spectrum of tonality, harmony and rhythm can be represented.

Certain non-geometric properties evolved by the system and encoded in the phenotype are not encoded in the graphical representation and relate solely to the music, such as the instrument type and panning. For forming longer musical compositions, the structural aspect of what elements to repeat and what to take away or change is the main challenge. Crucial to producing aesthetically pleasing music is combining the right instruments, and evolving the set of playing instruments in a dynamic manner. This is handled by a combination of the representation, evolutionary process, the selected preset and the autopilot. These will be explained in the remainder of this section.

B. Representation

An individual may be represented by its genotype and corresponding phenotype mapping, which in turn provide the information necessary to represent it visually and musically. To create the desired mix of different polygons with pleasing musical properties, 20 different parameters were encoded in the genotype [ADD TABLE? OR PHENO TABLE]. The genotypes are a real-numbered vector of 20 values in [0, 1). These contain information on both the geometric shape as well as its musical properties. Since some instruments have very different ranges of pleasant musical properties and may require restriction to a certain range, genotypes are converted into phenotypes using a custom mapping function for each instrument. Phenotypes are represented as a vector of 22 values, extending the 20 values of the genotype with the type of instrument ('nature') defined by the population the individual is a member of, and pitch of the sound, dependent on the note properties of the genotype. Multiple populations are evolved separately, one for each instrument set; a separate population is defined for *kicks, snares, hats, percussion, bass, guitar* and *synths*. Genotypes are equivalent across all populations to ensure that they evolve in a similar fashion, while different phenotypes are produced by the mapping function variant associated with each population.

Genotype & Phenotype				**Phenotype**
amp	attack	bpm	cutoff	nature
initial_offset	instrument	mix_echo	mix_reverb	pitch
mod_phase	mod_range	number	order	
pan	release	rootnote	rootoctave	
total_offset	red	green	blue	

Populations	7 ($\mu_1, ..., \mu_7$)
Population Size	10 (n)
Representation	Real-valued Vectors
Initialisation	Random [0.0, 1.0)
Parent Selection	Probabilistic (p_m)
Recombination	None
Mutation	Polynomial Bounded (η)
Fitness	Multi-objective
Specialty	Moving Optimum
Survivor Selection	Select Best n ($\mu + \lambda$)
Playing Selection	Select Best k
Termination Condition	Manual (none)

C. Evolutionary Process

The populations are randomly initialized with a size μ. Individuals are selected with a probability p_m to be mutated by bounded polynomial mutation with a step size of η. The fitness values of the original population and mutated offspring are calculated, of which the best μ individuals are selected. These steps are equivalent to a standard implementation of a $(\mu + \lambda)$-ES algorithm.

Once all populations have evolved and a new generation is prepared, the next step is to select k population members to be played. k is defined as the sum of the amount of members drawn from each population, which is controlled by the preset and the autopilot. The selected population members' phenotypes are compiled and used to generate the visuals and audio. Since the algorithm has no predefined termination condition, evolution continues until the user terminates the process.

D. Fitness Evaluation

The aesthetic fitness of an individual i is measured by computing the weighted sum of metrics of fitness as shown in Formula 1. The formulations of these three metrics are described in this section. The fitness function objective is to minimize this weighted sum.

$$f_i = w_{dist} \cdot f_{dist,i} + w_{sym} \cdot f_{sym,i} + w_{age} \cdot f_{age,i} \quad (1)$$

The distance metric f_{dist} represents the distance between the order of a given shape and a defined optimal shape. Rather than being fixed, the optimum value is modulated over time to generate varying musical patterns. For any given individual i, its f_{dist} is defined as the Euclidian distance between the order o_i encoded in its phenotype representation and the optimal order $o(t)$ at a given time point. A higher distance from the optimum incurs a large penalty to the individual's fitness, reducing the likelihood of being selected for the next generation. Several optima may also be configured, in which case f_{dist} represents the distance to the nearest optimum. The generalised formula is given in Equation 2.

$$f_{dist,i} = \min_k (o_i - o_k(t))^2 \quad (2)$$

In order to formulate an aesthetic estimation of rhythmic fitness of shapes, a custom encoding scheme was devised to evaluate the rhythmic purity of two encoded ratios. The encoding penalizes rhythms that feature conflicting rhythmic intervals. As two shapes rotate, their order (a shape's number of angles) determines the temporal intervals at which samples are played. Certain polyrhythmic combinations may produce interesting combinations explored in virtually all musical cultures worldwide, such as 6 on 4 or 12 on 8, or might produce unfavourable intervals that are rarely used, highly irrational and would range from difficult to impossible for a human listener to interpret, such as 5 on 11, 11 on 12, or 3 on 7. The assigned value from the encoding scheme is retrieved to find the penalty for a specific pair of rhythmic subdivisions, and the sum is normalized to average over the encoding value for all shapes currently playing. The encoding scheme is given in Table III-D.

$$f_{sym,i} = \frac{1}{n} \sum_{j=1}^{n} enc(p_i, p_j) \quad (3)$$

Ratio	Whole	Half	Third/Quarter	Eighth/Tenth	Other
Encoding	0	1	2	5	10

Lastly, the age metric f_{age} is defined as the number of algorithm cycles an individual has been in the set of individuals being played. Consequently, f_{age} is initialized at 0 for all genes in a population, and only begins to increase incrementally for those genes selected to be currently playing. This metric ensures individuals that have been playing for a long time are eventually phased out, even if they are considered highly fit. Formula **??** gives the substituted form of the fitness formula of an individual.

E. Presets

The challenge now is to maintain diversity, while keeping some (musical) elements constant.

F. Autopilot

IV. Experimental setup

A. Fitness Evaluation

V. Experimental Results

A. Evaluation of system

VI. Discussion

[...]

A. Limitations

limitations of study + focus on no human in loop prohibited user-centred aesthetic evaluation of produced music (no questionnaires done, no measure of psychological aesthetic perception on listener side)

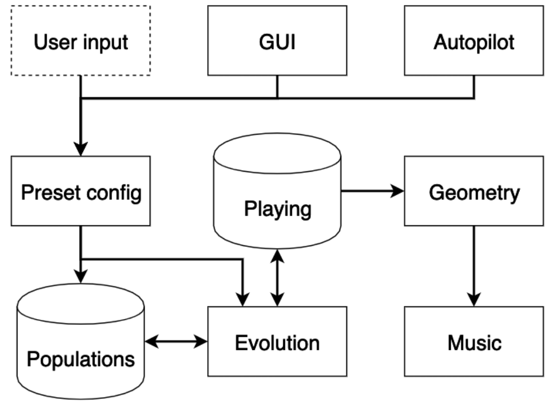

Fig. 1. Overview of system architecture

VII. Conclusion & Further Work

VIII. Acknowledgements