# A System for Evolving Art Using Supervised Learning and Aesthetic Analogies


**Aidan Breen and Colm O'Riordan**

**Abstract** Aesthetic experience is an important aspect of creativity and our perception of the world around us. Analogy is a tool we use as part of the creative process to translate our perceptions into creative works of art. In this paper we present our research on the development of an artificially intelligent system for the creation of art in the form of real-time visual displays to accompany a given music piece. The presented system achieves this by using Grammatical Evolution, a form of Evolutionary Computation, to evolve Mapping Expressions. These expressions form part of a conceptual structure, described herein, which allows aesthetic data to be gathered and analogies to be made between music and visuals. The system then uses the evolved mapping expressions to generate visuals in real-time, given some musical input. The output is a novel visual display, similar to concert or stage lighting which is reactive to input from a performer.

**Keywords** Genetic algorithms · Evolutionary art and design · Genetic programming · Hybrid systems · Computational analogy · Aesthetics

## 1 Introduction

Analogy is the comparison of separate domains. The process of analogy has strong applications in communication, logical reasoning, and creativity. A human artist will often take some source material as inspiration and create an equivalent, or related art piece in their chosen artistic domain. This process of metaphor is the equivalent of making an artistic analogy and has been used successfully in a literal form by artists like Klee [1], Kandinsky [2] and more recently Snibbe [3]. Similar approaches

A. Breen (✉) · C. O'Riordan
National University of Ireland, Galway, Ireland
e-mail: a.breen2@nuigalway.ie
URL: http://www3.it.nuigalway.ie/cirg/

C. O'Riordan
e-mail: c.oriordan@nuigalway.ie

© Springer Nature Switzerland AG 2019

J. J. Merelo et al. (eds.), *Computational Intelligence*,
Studies in Computational Intelligence 792,
https://doi.org/10.1007/978-3-319-99283-9_3

are often taken in a less direct form by stage lighting designers or film soundtrack composers.

Our aim is to make computational analogies between the domains of music and visuals by making use of aesthetic models, computational analogy, and grammatical evolution.

This work has direct practical applications for live performance and stage lighting design. The work in this paper may also have less direct applications in user interface and user experience design with particular use in the automatic generation of user interfaces and subconscious feedback mechanisms. Beyond these application domains, our research motivation also includes gaining insight into aesthetics and analogical reasoning.

## *1.1 Creating Aesthetic Analogies*

One of the major challenges of computational art is to understand what makes an art piece *good*. Indeed the cultural and contextual influences of an art piece may define what makes it emotive, such as Duchamp's Fountain [4] or René Magritte's The Treachery of Images [5], but beyond that we rely on the aesthetics of an object to decide if it is pleasurable to perceive. Aesthetics provide an objective description of this perception. We use this objective description as a tool upon which to build our analogies.

Every domain has its own aesthetic measures—musical harmony, visual symmetry, rhythm and combinations thereof. In some cases, these measures can be used to describe objects in more than one domain. Symmetry, for example, can describe both a visual image, and a phrase of music. The example we demonstrate in this paper is harmony. Musical harmony can be measured by the consonance or dissonance of musical notes. Visual harmony can be measured directly as the harmony of colours.

The analogy we are hoping to create is described as follows: given some musical input with harmony value $x$, a *mapping expression* can be created to generate a visual output with a harmony value $y$ such that $x \simeq y$. Furthermore, we posit that when performed together, both input music and output visuals will create a pleasing experience. In other words, can we take music and create a visual with a similar harmony and will they go together?

For this simple example, it is clear that a suitable expression could be created by hand with some knowledge of music and colour theory. However, if we extend the system to include more aesthetic measures, such as symmetry, intensity, contrast or granularity, defining an analogy by use of a *mapping expression* becomes far more complex. While developing a system to capture more complex mappings is beyond the scope of this paper, we aim to build the system such that it may be extended to do so.

### *1.2 Grammatical Evolution and Mapping Expressions*

A genetic algorithm (GA) provides a useful method of traversing an artistic search space, as demonstrated by Boden and Edmonds in their 2009 review [6]. Grammatical evolution (GE) [7], in particular allows us to provide a simple grammar which defines the structure of *mapping expressions* which can be evolved using a GA. This allows us to flexibly incorporate aesthetic data, operators and constants while producing human readable output. Importantly, we make no assumptions about the relationships between input and output. This approach does not restrict the output to any rigid pattern; potentially allowing the creation of novel and interesting relationships between any two domains, music and visuals or otherwise.

No single set of *mapping expressions* would be capable of creating pleasing output in every circumstance. In this respect, we intend to find suitable expressions for a specific input, such as a verse, chorus or phrase. Expressions may then be used in real-time when required and would handle improvisation or unexpected performance variations. Expressions produced by Grammatical Evolution are naturally well suited to this task as they can be stored or loaded when necessary, and evaluated in real-time.

### *1.3 Contributions and Layout*

The main contribution of this work is an implementation of Grammatical Evolution using music and empirically developed aesthetic models to produce novel visual displays. Secondary contributions include a structural framework for aesthetic analogies used to guide the gathering of data and development of evolutionary art using *mapping expressions*, and preliminary results produced by our implementation of the system.

The layout of this paper is as follows. Section 2 outlines related work in the areas of computational analogy, computational aesthetics, and computational art. Section 3 introduces our proposed method including a general description of our analogy structure, aesthetic models and the structure of our evolutionary system. Section 4 presents the details of our implementation in two distinct phases, the evolutionary phase (Sect. 4.1) and the evaluation phase (Sect. 4.2). Our results are presented in Sect. 5 followed by our conclusion in Sect. 7 including a brief discussion of future work (Sect. 6.3).

## 2 Related Work

"Analogy underpins language, art, music, invention and science" [8]. In particular, Computational Analogy (CA) combines computer science and psychology. CA aims to gain some insight into analogy making through computational experimentation.

As a research domain, it has been active since the late 1960s, accelerated in the 1980s and continues today. Computational analogy systems historically fall into three main categories: symbolic systems, connectionist systems and hybrid systems. Symbolic systems make use of symbolic logic, means-ends analysis and search heuristics. Connectionist systems make use of networks with spreading activation and back-propagation techniques. Hybrid systems often use agent based systems taking aspects of both symbolic and connectionist systems. For further reading, see [9, 10].

Birkhoff is often cited as one of the first to consider aesthetics from a scientific point of view. His simplistic 'aesthetic measure' formula, $M = O/C$, was simply the ratio of order ($O$) to complexity ($C$) [11]. Of course, this is over simplified and abstract, but it did begin a long running discussion on aesthetics and how we can use aesthetics to learn about the higher functions of human cognition.

More recently, the discussion has been reignited by Ramachandran who has outlined a set of 8 'laws of artistic experience' [12]. In this paper a number of factors are outlined which may influence how the human brain perceives art. Some of these factors are measurable, such as contrast and symmetry, but others remain more abstract such as the grouping of figures. Nonetheless, it has inspired further discussion [13–16].

Within specific domains, heuristics can be formalized and used to generate derivative pieces in a particular style or to solve particular challenges in the creation of the art itself. GAs in particular have proven to be quite effective due to their ability to traverse a large search space. In music for example, GAs have been used to piece together particular musical phrases [17], generate complex rhythmic patterns [18] or even generate entire music pieces [19]. Similar systems have also been used to create visuals [20, 21], sculpture [22] and even poetry [23].

Indeed the use of aesthetic measures in combination with GAs has also been reviewed [24] and an approach has been outlined to demonstrate the potential application of Multi-Objective Optimization to combine these measures [25]. While the system does produce computational art that may be described as aesthetic, it is also limited strictly by the aesthetic measures used, without any artistic context.

It is clear that computational systems can work in tandem with aesthetics to generate art and explore the possible applications of computational intelligence. Up to this point however, popular approaches have been remarkably rigid. Our work aims to explore a more flexible approach and perhaps discover a more natural artistic framework through analogy.

## 3 Proposed Method

### *3.1 Analogy Structure*

We make use of a conceptual structure to provide a basis for our aesthetic data and analogies. The structure is shown in Fig. 1 with measurable aesthetic attributes in separate domains which are connected by a set of *mapping expressions* which may

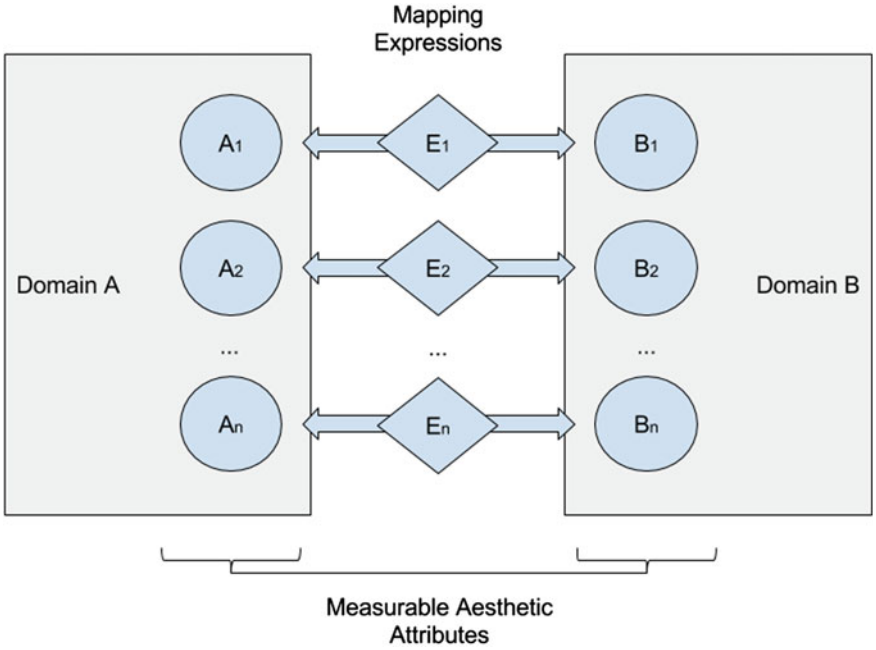


**Fig. 1** Analogy structure overview. The *Mapping Expressions*, $E_1$ to $E_n$, are encoded as chromosomes and evolved using the genetic algorithm. Extracted from [26]

be evolved using grammatical evolution. The implementation in this paper uses a single attribute in each domain, however, the structure is not restricted to a bijective mapping.

Some aesthetic attributes may be more suitable than others for use in a structure as described in Fig. 1. Harmony is selected for use in this paper as it has a strong impact on the overall aesthetic quality of an art piece, and can be measured quite easily in separate domains. In music, the harmony of notes being played is often referred to as the consonance—or conversely, dissonance—of those notes. While the timbre of notes has an impact on consonance, an estimate can be obtained from pitch alone. In the visual domain, colour harmony, or how pleasing a set of colours are in combination, can be measured as a function of the positions of those colours in some colour space, such as RGB (red, green and blue dimensions), CMYK (cyan, magenta, yellow and black dimensions), HSV (hue, saturation and value dimensions) or LAB (one lightness dimension and two opposing colour dimensions of red/green and yellow/blue). The conceptual similarities between musical consonance and visual harmony provide a convenient and understandable starting point.

Consonance values for any two notes have been measured [27–29] and numerous methods have been proposed that suggest a consonance value can be obtained for larger sets of notes [30–33]. The simplest general approach is to sum the consonances for all pairs of notes in a set. This provides a good estimation for chords with the

same number of notes and can be normalized to account for chords of different cardinalities.

For this preliminary implementation, we enforce a number of restrictions. Firstly, we restrict the number of inputs to two musical notes at any one time. This simplifies the grammar and allows us to more easily analyse the output *mapping expressions*. Secondly, musical harmony is calculated using just 12 note classes within a single octave. This helps to avoid consonance variations for lower frequencies.

The consonance values for each note classes were gathered in a study whereby test subjects were presented with two note pairs and asked to select the note pair that sounded more consonant. By using this *two alternative forced choice* approach, together with a novel ranking algorithm, a number of issues which affected the results of previous studies were avoided.

The subjectivity of responses was reduced by forcing two pairs to be directly compared. Contextual issues, where the order in which pairs were ranked might affect their actual ranking, were also avoided in this way. The ranking algorithm used a graph based approach which allowed a full ranking of all 12 note classes to be found with a minimal number of comparisons. This was particularly important as subject fatigue had a large impact on the quality of responses and can increase in a very short space of time. The performance of the ranking algorithm has been further analysed and compared to other ranking approaches demonstrating the effectiveness of the algorithm for this application [34].

The observed consonance values were then compared to previous studies and historical observations, showing a high correlation. Figure 2 shows the consonance values for note pairs used based on this study [29].

Similarly, colour harmony values can be measured and modelled [35–37]. While the harmony of more than 2 colours may be obtained with a similar approach to music

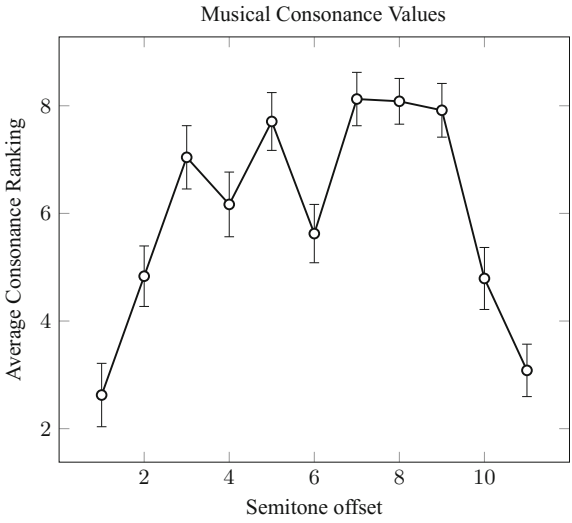


**Fig. 2** Consonance Values for musical intervals used to calculate musical Harmony Values. Extracted from [26]

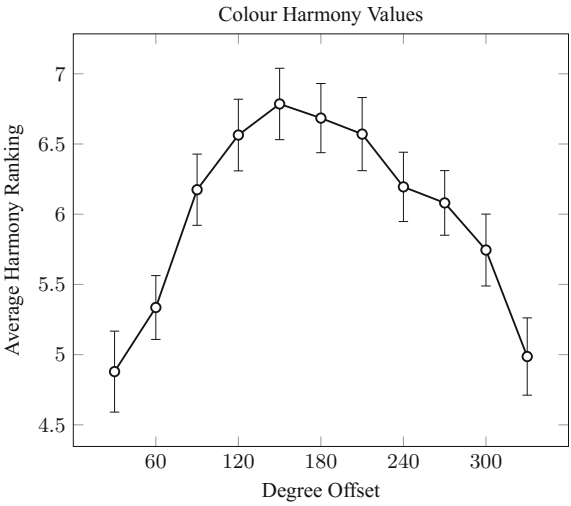


**Fig. 3** Average Colour Harmony values. Extracted from [26]

chords, the pattern in which colours are displayed adds an extra level of complexity. To combat this, we assume our visual display is not a strict two dimensional image, but rather a pair of lights emitting coloured light into some space. For example, a pair of LED stage lights for a small musical performance.

The harmony values for colour pairs were obtained using the same methodology used to obtain musical data. A *two alternative forced choice* style study using the same ranking algorithm was employed to prevent subjectivity and contextual pitfalls and to reduce subject fatigue. The results were compared, once again, to previous studies and historical observations. Figure 3 shows the harmony values obtained in this study and used in the work presented here.

## *3.2 Evolutionary System*

Our evolutionary approach is based upon Grammatical Evolution [7]. We use a Genetic Algorithm (GA) to evolve *mapping expressions* based upon a given grammar. The evolved expression allows us to create a real time system rather than an offline visual output. In this way, any particular performance is not limited to a strict musical input thereby allowing improvisation, timing and phrasing variation, and handling of human error.

Another advantage of this particular GA approach is the flexibility by which we can incorporate aesthetic data. By using a grammar, we are decoupling the structure of the output of the GA from the actual process of evolution. To extend the current implementation to include intensity as an aesthetic attribute, for example, we simply edit the grammar to accommodate this. We can also extend the grammar to include

other operators, such as predefined functions if we feel their inclusion may improve the performance of the system.

Further, the human readability of the output expression is extremely valuable, not only as a sanity check to debug unexpected behaviour, but also to analyse the output of a particular evolutionary run. The goal of this work is not simply to create analogies, but to understand the process by which they are created. The readability of output expressions at each stage of the evolutionary process is therefore a huge advantage over other *black box* AI techniques which produce results without any way of understanding how or why they were produced.

**Mapping Expressions**. *Mapping expressions* are created by using an individual chromosome to guide the construction of a symbolic expression by use of the given grammar. The following is an example of a symbolic expression representing a nested list of operators (addition and multiplication) and parameters (2, 8 and 5) using prefix notation. This is the same structure used by *mapping expressions*.

```
(+ 2 (* 8 5))
```

The symbolic expression is evaluated from the inside out, by evaluating nested expressions and replacing them with their result. In this example, the nested expression multiplying 8 by 5 will be evaluated producing the following intermediate expression.

```
(+ 2 40)
```

Once all nested expressions are evaluated, the final expression can return a result, in this case, 42.

**Grammar**. The grammar defines the structure of an expression using terminal and non-terminal lexical operators. Terminals are literal symbols that may appear within the expression, such as $+$, $*$ and the integers in the example expression above. Non-terminals are symbols that can be replaced. Non-terminals often represent a class of symbols such as operators of a specific cardinality, other non-terminals, or specific terminals. In the example expression above, the $+$ and $*$ symbols might be represented by a single *operator* non-terminal that accepts two parameters. Parameters may be an integer, or a sub expression. Finally, the integers in the expression above could be replaced by an integer non-terminal. This leaves us with the simple grammar shown in Table 1 which is capable of representing not only the example expression, but also any other expression, of any size, that adds and multiplies integers.

Given a defined grammar, such as the example grammar in Table 1, an expression can be built from a chromosome using the following approach. Beginning with a starting non-terminal, each value in the chromosome is used in series as the index of the next legal terminal or non-terminal. This mapping continues until either the expression requires no more arguments, or a size limit is reached. If the chromosome is not long enough to complete the expression, we simply begin reading from the start of the chromosome again.

**Table 1** Example grammar

| Non-terminals | Possible replacements |
|---|---|
| Parameter | Integer, (Operator Parameter Parameter) |
| Operator | +,* |
| Integer | Any integer |
| Terminals | |
| + | |
| * | |
| Any Integer | |

**Evolution**. The evolution process is straightforward thanks to the simple structure of chromosomes. Each *mapping expression* becomes an individual within out population possessing a single chromosome. Beginning with a population of individuals with randomly generated chromosomes, known as Generation 0, successive generations are produced using Selection, Crossover and Mutation genetic operators.

Tournament selection wth elitism is used in this implementation. At generation $G_n$, the fitness of each individual is calculated. A small number of the highest fitness individuals are moved to generation $G_{n+1}$. This elitism is used to prevent the maximum fitness of the population from declining. The selection phase now begins to population the rest of generation $G_{n+1}$. Two individuals are selected at random. The individual with more favourable fitness is placed into generation $G_{n+1}$. The process continues until the desired size of generation $G_{n+1}$ is reached.

When selection is complete, crossover begins using both single point and double point crossover. Two individuals $P1$ and $P2$ are selection at random from the newly selected generation $G_{n+1}$. If single point crossover is to be used, a random crossover point is selected and two children, $C1$ and $C2$ are produced. These children take the place of their parents in the new generation.

$C1$ will receive a chromosome made up of the $P1$ codons to the left of the crossover point, and the $P2$ codons to the right of the crossover point. $C2$ will receive the alternative, that is, a chromosome made up of the $P1$ codons to the right of the crossover point, and the $P2$ codons to the left of the crossover point. If double point crossover is to be used, the same steps are taken, but twice. This results in one child receiving outer codons from one parent and inner codons from the other.

The effect of double point crossover is quite strong when using the described implementation of Grammatical Evolution. Due to the tree-like, recursive way in which an expression is built, single point crossover invariably has large effects on the final expression. Large portions of the expression are likely to be completely different, even to the parent expression, as previous codons determine the meaning of following codons. Double point crossover alternatively, can operate similar to subtree substitution, affecting only a small portion of the final expression and retaining the overall structure of the expression tree.

Finally, once generation $G_{n+1}$ is complete, a mutation operator is applied. Mutation consists of setting a very small number of codons to new values. Both the mutated codon and the new value are randomly selected.

This process relies heavily on the fitness function. Calculating the fitness of any *mapping expression* without some guidelines would be extremely subjective. In our implementation we take a heuristic approach that rewards solutions that produce outputs with a similar normalised aesthetic value as inputs. An in-depth description of the implemented fitness function is presented in Sect. 4.1.

## 4 Implementation

We now discuss how the structure introduced above together with the data gathered has been implemented. We demonstrate how the following system has been used to evolve *mapping expressions* that generate a visual output when given a musical input. The system may be used to generate visuals in time with music by use of a time synchronized subsystem utilizing a music synthesizer and visualization server.

### *4.1 Evolution Phase*

Figure 4 shows the structure of the Evolution Phase. This phase is centred about the GE algorithm. In our implementation we use a population of 50 chromosomes. Chromosomes are stored as 8 bit integer arrays, with values between 0 and 255. A chromosome length of 60 integer values was used in the work presented in this paper.

Musical input is taken in the form of *Musical Instrument Digital Interface* (MIDI) data. The MIDI protocol represents digital music signals, originally designed as a transmission protocol to allow musical signals to be sent between instruments and synthesizers. Musical notes are sent as packet pairs (*note on* and *note off*) containing the note pitch and the 'velocity', or strength of the note which is often translated to volume. The MIDI protocol also allows data to be stored as a file with each packet

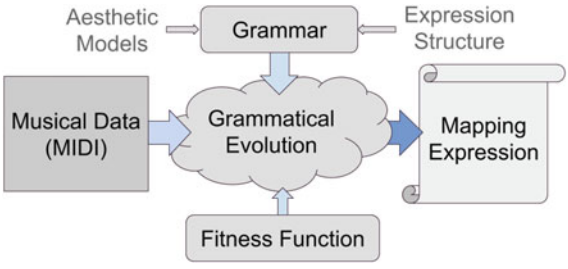


**Fig. 4** Evolution Phase overview. Extracted from [26]

**Table 2** Grammar terminal operators

| Expression | Arguments |
|---|---|
| Plus 90° | 1 |
| Plus 180° | 1 |
| Sin | 1 |
| Cos | 1 |
| Log | 1 |
| Addition | 2 |
| Subtraction | 2 |
| Multiplication | 2 |
| Division | 2 |
| Music harmony constant | 2 |
| Visual harmony constant | 2 |
| Ternary conditional operator | 3 |

**Table 3** Grammar terminal values

| Expression | Range |
|---|---|
| Constant integer value | 0–255 |
| Musical input 1 | 0–255 |
| Musical input 2 | 0–255 |

containing a timing value. We use a file to store a sample musical input using this format and determine which notes are being played using the timing value.

The implemented grammar contains a list of operators, and values (variables and constants) which are presented in Tables 2 and 3. Of note here are the aesthetic values for music and visuals which can be inserted directly into an expression as constants, or read at run-time as variables in the case of musical input. The aesthetic models use normalised values based on the values shown in Figs. 2 and 3. Aesthetic constant expressions such as *Musical Harmony Constant* and *Visual Harmony Constant*, accept two arguments representing two music notes or two colour hues. The expression returns the aesthetic value of those two notes or colours.

Our fitness function, as introduced above, aims to maximise the similarity between input and output harmony. The fitness for any $n$ pairs of input musical notes is calculated as follows, where $M$ is a function representing the musical harmony of a pair of notes, and $V$ is a function representing the visual harmony of a pair of colour hues.

$$fitness = \frac{1}{n} \sum_{i=1}^{n} 255 - |M(input) - V(output)| \tag{1}$$

Both $M$ and $V$ are normalised between 0 and 255, which produces a fitness range of 0–255.

Tournament selection is carried out to select individuals for evolution. A combination of single point and double point crossover is used to build a succeeding generation. Elitism is used to maintain the maximum fitness of the population by promoting the best performing individuals to the next generation without crossover or mutation.

Mutation is applied at the gene level. A gene is mutated by randomly resetting its value. The mutation rate is the probability with which a gene will be mutated. The mutation rate is varied based on the number of generations since a new peak fitness has been reached. This allows us to optimise locally for a period, and introduce hyper-mutation after an appropriate number of generations without any increase in peak fitness. We call this the Mutation Threshold. The standard mutation rate ($Mut_1$) is calculated as:

$$Mut_1 = \left(\frac{0.02}{70}\alpha\right) + 0.01 \tag{2}$$

where $\alpha$ represents the number of generations since a new peak fitness was reached.

After the Mutation Threshold is reached, indicating a local optima, hyper-mutation ($Mut_2$) is introduced to encourage further exploration of the fitness landscape.

$$Mut_2 = 1.0 \tag{3}$$

If a fitter solution is discovered, mutation is again reduced to $Mut_1$ to allow smaller variations to occur.

At each generation, the *mapping expressions* represented by chromosomes are stored. This allows us to monitor the structure and size of the expressions as they are created. We can also use the stored expressions to compare earlier generation to later generations.

Evolution is halted after a Halting Threshold has been reached. The Halting Threshold is measured as the number of generations without an increase of peak fitness. Details of the parameters used can be found in Table 4.

The output of this process is a set of *mapping expressions* representing the final generation. From this set, the fittest expression is selected for evaluation and comparison to the fittest expression from previous generations.

**Table 4** Genetic algorithm parameters

| Parameter | Value |
|---|---|
| Population size | 50 |
| Chromosome length | 60 |
| Crossover rate | 0.8 |
| Standard mutation Rate ($Mut_1$) | see Eq. (2) |
| Hyper-Mutation rate ($Mut_2$) | 1.0 |
| Mutation threshold | 100 |
| Halting threshold | 200 |

### *4.2 Evaluation Phase*

In order to evaluate the performance of an evolved *mapping expression*, we must play both music and visuals together. To this end, we have built the evaluation system as outlined in Fig. 5. While the evolved expression is capable of generating visual output in realtime, the evaluation for this work is conducted offline, using a pre-calculated file containing the original musical data, and the generated visual data.

In order to perform music in synchrony with generated visuals, an *extended MIDI player* subsystem is required. Musical data (MIDI) and visual data are combined in an *extended MIDI file* (MIDIX). The *extended MIDI player* then parses this file and uses an internal time synchronisation process to send MIDI signals to two separate systems: a music synthesizer and a visualization server. Both systems are capable of reading MIDI signals and producing output in real time.

The music synthesizer is a common tool in music creation and performance. Historically, synthesizers existed as hardware devices connected to some digital instrument producing MIDI signals. The synthesizer would listen for input from a physical MIDI cable, and produce a signal which could be sent to an amplifier and speakers to create audio. Modern synthesizers are typically digital systems that listen to digital MIDI ports for message packets and produce audio using modern audio interfaces. While hardware based synthesizers are highly sought after by electronic musicians, software based synthesizers are indistinguishable in most circumstances.

We make use of digital synthesizers in this work due to their flexibility and cost effectiveness. We use a standard MIDI port to send and receive musical data to an open source software synthesizer, part of the Reaper digital audio workstation [38]. The signals received by the synthesizer are used to produce realistic sounding music.

The visualization server is a software system created specifically for this implementation. The server works in a similar fashion to a music synthesizer; however, rather than using a MIDI port, the server uses HTTP and websockets. With this approach, a webserver accepts HTTP messages sent from the *extended MIDI player* and uses websockets to update a javascript application running in a web browser. This technology was chosen due to the cross platform support for javascript applications and also due to the ease of rendering a visual display using standard web technologies. When the javascript application receives an update through the websocket, the display is updated accordingly. This ensures the visuals remain synchronized with the audio being played.

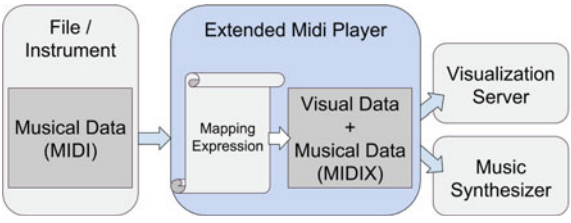


**Fig. 5** Evaluation Phase overview. Extracted from [26]

### 4.3 Supervised Fitness

Using the evaluation system outlined above, we can interactively evaluate the performance of a particular *mapping expression*. We can either use a static MIDI file to compare individual *expressions* or we can use a live MIDI instrument to send live MIDI signals to evaluate how it performs with improvised and varying input.

Expressions that are deemed fit by human supervision may then be reintroduced to the evolution phase to continue the process. This step is independent of the fitness function in order to capture aesthetic results beyond its capabilities.

## 5 Results

### 5.1 Evolutionary Phase

Using the approach outlined above we successfully evolved *mapping expressions* capable of mapping musical input to visual output.

Many of the random seed expressions such as the following example simply produced constant values:

```
['plus180',['plus90',['sin', 215]]]
```

In later generations however, we see more complex expressions producing better fitting results:

```
['add',56, ['musicalHarmony',94,['cos','mus2']]]
```

Here we see the expression makes use of the input variable `mus2` and the musical harmony constant `musicalHarmony` which produces a dynamic output. The example chosen here is one of the smallest expressions created.

Further visual analysis of the smallest evolved expressions shows regular use of dynamic input variables. A script was used to search through the entire set of final expressions from all evolutionary runs. The smallest expressions were selected for manual evaluation and visual inspection. The sampled expressions achieved high fitness scores, above 200 in all cases, while using less than 10 sub-expressions. Leaf nodes of these sampled expression trees consist of at least one input variable in all cases. All sampled expressions also made use of musical or visual harmony constants at least once.

Figure 6 shows the distribution of fitness values for randomly generated expressions versus evolved expressions. We see the distribution for random expressions is heavily skewed towards the minimum value of 100. This is due to the number of expressions which produce a constant output. Evolved expressions however show a much tighter distribution with significantly higher fitness values.

The distribution of intervals in the input $M$ will affect the fitness of the evolved expression. An evolved expression may be directly compared to the target visual

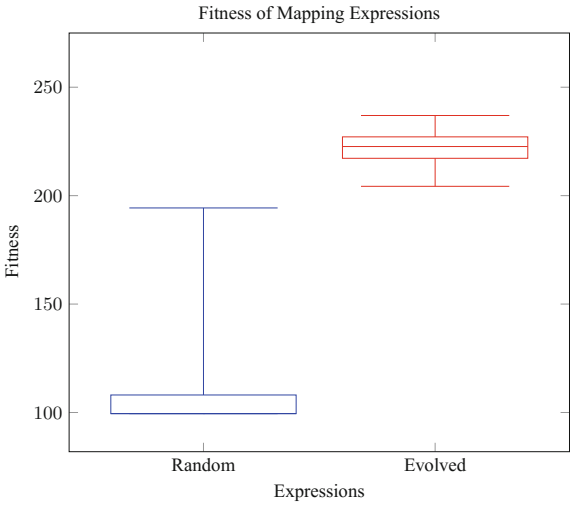


**Fig. 6** Fitness of 100 randomly generated *Mapping Expressions* versus Evolved expressions. Extracted from [26]

harmony by using an equally distributed input. Our input is a set of 11 note intervals, 1–11, excluding the unison and octave intervals 0 and 12 respectively. In Fig. 7 we see a demonstration of this comparison. Previous generations are shown as dotted lines with the final fittest individual in solid black. The target output, the output that would produce an optimum fitness value, is shown in red. We see as the generations pass, the output matches the target more closely. Of note here are the horizontal dotted lines indicating older generations producing constant outputs which have been superseded by generations producing closer matching dynamic outputs.

Figure 8 shows the fitness of a single population across a number of generations. In blue we see the maximum fitness of each generation. This value never decreases

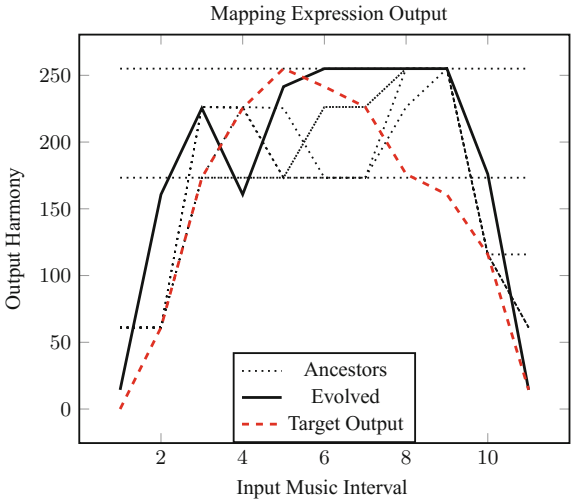


**Fig. 7** Output of one evolved expression and its ancestors compared to target visual harmony. Extracted from [26]

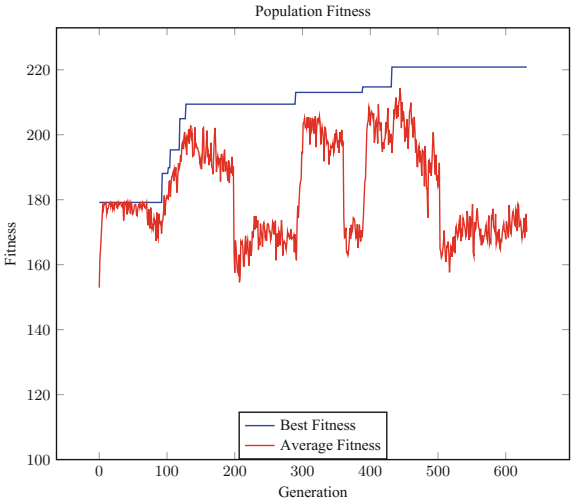


**Fig. 8** Population fitness for 631 generations of a typical run. Extracted from [26]

due to the use of elitism. Highly fit individuals are brought from one generation to the next to prevent the population from decaying and losing a possible fit solution.

In red we see the average fitness of the population. During early generations we see dramatic improvements in fitness followed by a series of incremental increases punctuated by dramatic decreases in average fitness. These incremental increases in fitness are an indication of local optima being discovered with low mutation. Hyper-mutation is then introduced causing the dramatic reduction in average fitness as more individuals mutate in more extreme ways. While this seems to have a negative effect on the population as a whole, it allows us to find fitter solutions and prevents premature population convergence at a local optima.

## *5.2 Evaluation Phase*

Preliminary results have been obtained based on the initial implementation described in Sect. 4.2. These results demonstrate that a visual display, however rudimentary, can be produced based on musical data in real time. The *extended MIDI player* was used to play a file containing a 10 s music piece with musical intervals of varying harmony. Visuals generated by a *mapping expression* were displayed on a computer screen. Visuals were observed to be in time with the synthesized music. An example of the visual display with screenshots taken at 2 s intervals are shown in Fig. 9. The colour pattern used in the visual display was similar to that used to collect colour harmony data. This ensures that there is no variation between observed colour harmony and generated colour harmony.

Our initial observations indicate the the evolved expressions produce an analogy that results in more enjoyable visuals than randomly generated colours. The colours

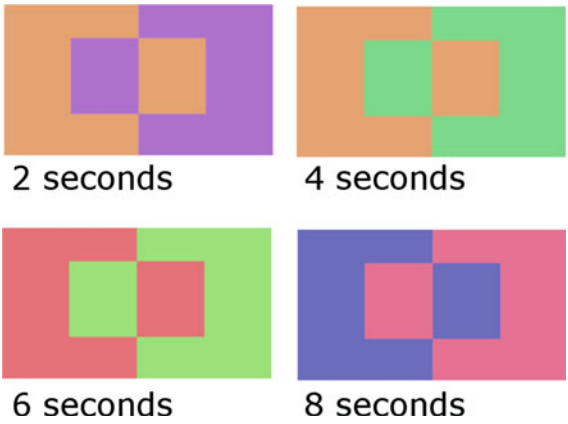


**Fig. 9** Generated visual display. Extracted from [26]

tend to follow a general pattern where similar note pairs, with similar intervals, produce similar colours.

## 6 Discussion

In this work we aim to lay the foundation for further developments in the use of aesthetic analogy and supervised learning in art. Our work takes a pragmatic approach, with the objective of using these tools to create an output which can be built upon and extended in further studies.

In this regard, the results presented above show a positive outcome. We were successful in defining an analogical structure which was codified such that Grammatical Evolution could be utilised to produce an output that represents an analogy. That analog—or its representation—can then be used to create a visual output where before there was only sound—or the digital representation of sound.

Realistically, of course, the visual output of the system is lacking in a number of ways. It might be a stretch to call the display shown in Fig. 9 a piece of art, but it *is* undeniably a visual display which has been created by a computer system. This system knows only the given parameters of aesthetics, a grammar and a way to compare results in the form of a fitness function. In this respect, the results might be compared to the exploratory play of a toddler as she learns. While the results might not be on par with more advanced or mature systems, what we are shown is really the beginning of what might be possible.

### *6.1 Analogical Structure*

One of the clear drawbacks of the work presented here is the restriction in the structure of the analogy used. Currently, the system is designed to create an analogy using only

one aesthetic attribute—harmony. This attribute was chosen specifically because it was clearly present and easily measured in both domains.

Harmony is however, only one of many potential attributes within the context of the aesthetics of music and visuals. The resulting analogy could therefore be described as a fractional aesthetic analogy, or an analogy of harmony alone which may contribute to the overall aesthetic quality. We hope to achieve a full analogy of aesthetics by making use of more than one attribute. To this end, we have highlighted a number of potential candidates for future attributes to be included.

Symmetry, intensity, contrast and complexity have specifically been highlighted as candidates for our future work. All four attributes can be observed in each domain and we believe they all may be measured, to some degree, without a great deal of subjectiveness or computational expense. Contrast is perhaps the easiest attribute to measure in both domains with contrast in visuals being measurable on a per-pixel basis and in music on a per-note basis. Symmetry is similarly directly measurable in visuals, and potentially measurable in music using heuristics such as matching beat patterns or relative pitch changes. Intensity in visuals may be measured as a function of colour contrast and fractal index while in music it may be measured by volume and tempo. Finally, complexity is commonly measured in visuals using a fractal index and may be measured in music simply by measuring the number of notes being played in a period of time.

These attributes may not represent the full aesthetic gamut but we believe that they will provide a strong toolset, beyond harmony alone.

## *6.2 Evaluation*

The evaluation phase, as described above, refers to the evaluation of the output visual display in tandem with the input music. We report a positive preliminary result however, fully evaluating the output of the system in any reasonably objective way would require a great deal of work beyond the scope of this paper.

A study of human subjects would be required to compare randomly generated visual displays, to displays created with *mapping expressions* of various fitness. Due to the subjective nature of human aesthetic response, a reasonably large study would be required. A similar problem was faced when gathering the data used to build the aesthetic models used in this work. Even when tasked with ranking simple musical note pairs—or colour pairs—the participants in the study suffered from fatigue and boredom extremely quickly which may have had strong effects on the data gathered.

In the case of musical note pairs, however, the study could be conducted at a reasonably quick speed as each note pair would play for only a few seconds. To evaluate the output of the system presented in this paper, a longer segment of music must be played, significantly increasing the time it takes to gather information from any individual subject. This has a knock-on effect of increasing fatigue and boredom, reducing the quality of the data gathered. It is possible that any data gathered in a

study like this would simply by too strongly affected by subject fatigue and boredom to be useful in any meaningful way.

In practice however, no visual display is created in this manner. A display for a live music performance would typically be created at the discretion of the lighting designer or artist, and a small number of advisors with very little specific feedback from the audience. This observation should serve to guide the development of the system as an artistic tool, rather than a replacement of the artist.

## *6.3 Future Work*

We have shown that *mapping expressions* can be evolved using a fitness function based on empirically developed aesthetic models. However, we have not evaluated the perceived aesthetic differences between expressions of varying fitness. Further research is required to fully evaluate the strength of this correlation.

At present we restrict the number of input musical notes to simplify the grammar and allow analysis of the evolved expressions. This clearly limits the application of this system greatly. Future iterations should accommodate varying musical input lengths.

The results presented were obtained using only one *mapping expression* between musical consonance and colour harmony. We have not explored the possibilities of using multiple mapping expressions incorporating many attributes. We believe this will improve the quality of generated visuals dramatically.

As shown in Sect. 5, the fitness of a population has certain limitations. We hope to improve the speed at which fitness increases and also increase the maximum fitness achievable by any individual by tuning the parameters of the genetic operators.

The *extended MIDI* format has a number of useful applications beyond its use in this implementation. The format may also be useful for predefined visual displays and synchronised performances. With this in mind, we would like to fully define our version of the protocol and make it available to the public.

In a similar vein, the visualization server, which uses the *extended MIDI* format may also be improved. Most immediately, it should be able to handle all of the attributes used by *mapping expressions* to generate varied and immersing visual displays. Also, the server is currently restricted to displaying visual displays on a computer screen, which is not suitable for a live performance. We hope to develop functionality to allow the visualization server to accept an *extended MIDI* signal and control stage lighting hardware using industry standard protocols.

The outlined system is certainly capable of producing some visual output. Whether that output is deemed aesthetically pleasing is still an open question. In order to determine the actual performance of the final output of the system, we hope to conduct a study with human subjects. Our hypothesis here is: *the system produces more pleasing visual displays than random colour changes*.

The proposed study would demonstrate if we are moving in the right direction, however, the overall goal of this research is to create a system that can create art, and perform it. To this end, the success of the system should be evaluated with a live performance.

## 7 Conclusion

The reported results show the effectiveness of the analogical structure as shown in Fig. 1, which can be used in combination with Grammatical Evolution to produce novel visual displays. This analogical structure successfully guides the collection of data in each aesthetic domain. This structure also allows the creation of a fitness function which serves to allow the generation of effective mapping expressions which maximise the aesthetic similarity between input music and output visuals over the course of an evolutionary run. The use of mapping expressions, evolved using this fitness function, allows the creation of real-time aesthetic analogies based solely on human aesthetic experience, without assuming any structure or weighting between aesthetic values across domains.

Results show mapping expressions that have achieved similarity in harmony between input music and output visuals using this approach. We also show the increase in fitness over time across an evolutionary run, demonstrating the effectiveness of Grammatical Evolution in this application.

**Acknowledgements** The work presented in this paper was kindly funded by the Hardiman Scholarship, National University of Ireland, Galway. The authors would also like to thank the staff and students of the Computational Intelligence Research Group (CIRG), Department of Information Technology, and the School of Mathematics, Statistics and Applied Mathematics, NUIG for their guidance and support throughout the development of this work.

## References

1. Klee, P.: Pedagogical Sketchbook. Praeger Publishers, Washington (1925)
2. Kandinsky, W., Rebay, H.: Point and Line to Plane. Courier Corporation, (1947)
3. Snibbe, S.S., Levin, G.: Interactive dynamic abstraction. In: Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering, ACM, 21–29 (2000)
4. Cameld, W.A.: Marcel Duchamp's fountain: Its history and aesthetics in the context of 1917. Artist of the century, Marcel Duchamp (1990)
5. Magritte, R.: The treachery of images. Oil Canvas **231**, 1928–1929 (1928)
6. Boden, M.A., Edmonds, E.A.: What is generative art? Digit. Creat. **20**, 21–46 (2009)
7. O'Neil, M., Ryan, C.: Grammatical evolution. In: Grammatical Evolution. Springer (2003) 33–47
8. Gentner, D., Forbus, K.D.: Computational models of analogy. Wiley Interdiscip. Rev.: Cogn. Sci. **2**(3), 266–276 (2011)
9. French, R.M.: The computational modeling of analogy-making. Trends Cogn. Sci. **6**, 200–205 (2002)
10. Hall, R.P.: Computational approaches to analogical reasoning: a comparative analysis. Artif. Intell. **39**(1), 39–120 (1989)

11. Birkhoff, G.: Aesthetic Measure. Cambridge University Press (1933)
12. Ramachandran, V.S., Hirstein, W.: The science of art: a neurological theory of aesthetic experience. J. Conscious. Stud. **6**, 15–35 (1999)
13. Goguen, J.A.: Art and the brain: editorial introduction. J. Conscious. Stud. **6**, 5–14 (1999)
14. Huang, M.: The neuroscience of art. Stanf. J. Neurosci. **2**, 24–26 (2009)
15. Hagendoorn, I.: The dancing brain. cerebrum: the dana forum on brain. Science **5**, 19–34 (2003)
16. Palmer, S.E., Schloss, K.B., Sammartino, J.: Visual aesthetics and human preference. Ann. Rev. Psychol. **64**, 77–107 (2013)
17. Todd, P.M., Werner, G.M.: Frankensteinian methods for evolutionary music. Musical networks: parallel distributed perception and performace, 313 (1999)
18. Eigenfeldt, A.: The evolution of evolutionary software: intelligent rhythm generation in Kinetic Engine. In: Applications of Evolutionary Computing, pp. 498–507. Springer (2009)
19. Fox, R., Crawford, R.: A hybrid approach to automated music composition. In: Artificial Intelligence Perspectives in Intelligent Systems, pp. 213–223. Springer (2016)
20. Heidarpour, M., Hoseini, S.M.: Generating art tile patterns using genetic algorithm. In: 2015 4th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), pp. 1–4. IEEE (2015)
21. Garcia-Sanchez, P., et al.: Testing the diferences of using RGB and HSV histograms during evolution in evolutionary Art. In: ECTA. (2013)
22. Bergen, S., Ross, B.J.: Aesthetic 3D model evolution. Genet. Program. Evol. Mach. **14**, 339–367 (2013)
23. Yang, W., Cheng, Y., He, J., Hu, W., Lin, X.: Research on community competition and adaptive genetic algorithm for automatic generation of tang poetry. Math. Probl. Eng. 2016 **2016**, (2016)
24. den Heijer, E., Eiben, A.E.: Comparing aesthetic measures for evolutionary art. In: Applications of Evolutionary Computation, pp. 311–320. Springer (2010)
25. den Heijer, E., Eiben, A.E.: Evolving art using multiple aesthetic measures. In: Applications of Evolutionary Computation, pp. 234–243. Springer (2011)
26. Breen, A., O'Riordan, C.: Evolving art using aesthetic analogies: evolutionary supervised learning to generate art with grammatical evolution. In: 8th International Conference on Evolutionary Computation, Theory and Applications. Porto, Portugal, pp. 59–68.(2016)
27. Malmberg, C.F.: The perception of consonance and dissonance. Psychol. Monogr. **25**, 93–133 (1918)
28. Kameoka, A., Kuriyagawa, M.: Consonance theory part I: consonance of dyads. J. Acoust. Soc. Am. **45**, 1451–1459 (1969)
29. Breen, A., O'Riordan, C.: Capturing and ranking perspectives on the consonance and dissonance of dyads. In: Sound and Music Computing Conference, pp. 125–132. Maynooth (2015)
30. Von Helmholtz, H.: On the Sensations of Tone as a Physiological Basis for the Theory of Music. Longmans, Green (1912)
31. Plomp, R., Levelt, W.J.M.: Tonal consonance and critical bandwidth. J. Acoust. Soc. Am. **38**, 548–560 (1965)
32. Hutchinson, W., Knopoff, L.: The acoustic component of Western consonance. J. New Music Res. **7**, 1–29 (1978)
33. Vassilakis, P.N.: Auditory roughness as means of musical expression. Sel. Rep. Ethnomusicol. **12**, 119–144 (2005)
34. Breen, A., O'Riordan, C.: Capturing data in the presence of noise for artificial intelligence systems. In: Irish Conference on Artificial Intelligence and Cognitive Science, pp. 204–216. Dublin (2016)
35. Chuang, M.C., Ou, L.C.: Influence of a holistic color interval on color harmony. COLOR Res. Appl. **26**, 29–39 (2001)
36. Szabó, F., Bodrogi, P., Schanda, J.: Experimental modeling of colour harmony. Color Res. Appl. **35**, 34–49 (2010)
37. Schloss, K.B., Palmer, S.E.: Aesthetic response to color combinations: preference, harmony, and similarity. Atted. Percept. Psychophys. **73**, 551–571 (2011)
38. Cockos: Reaper (2016)