A Walk to Meryton : A Co-creative Generative work by Musebots and Musicians

Arne Eigenfeldt
School for the Contemporary Arts, Simon Fraser University, Vancouver, Canada
www.sfu.ca/~eigenfel
e-mail: arne_e@sfu.ca

Abstract

A Walk to Meryton is a co-creation between a generative system and its creator, as well as three improvising musicians and a sound poet. Building upon previous generative systems [1, 2], the system is routed in composition rather than improvisation, in that plans (frameworks) are created, then filled in by musical agents (musebots) by creating a score; musebots can edit their individual parts, making decisions based upon global structures and local events by other musebots. The final score is performed by audio musebots, and a version of it is presented to musicians as a lead sheet consisting of harmonic progressions, melodies, and overall structure. Finally, an additional set of videobots generate video using still images of various nature walks made by the author, overlaid with text from Jane Austen's *Pride and Prejudice* (the inspiration behind the entire work).

1. Background

Much of my creative research in the past decade has revolved around making my generative music systems more *compositional* rather than *improvisational*. My own attraction to generative systems is the opportunity for continual reinterpretation of processes to essentially create infinite versions of a single work; however, there is a delicate balance between limiting the immediate output of a closed system to produce similar results – what I might consider an improvisational approach – and one in which the limits are the result of what the system itself might produce – what I would consider a compositional approach.

My background is that of a composer rather than an improvising performer; composers tend to be concerned with the control of time through structure, while improvisers tend to be concerned with an immediacy of production. I noted in my own non-generative output that, somewhat ironically, my compositional approach was often to begin a work through improvisation, then taking a slow and methodical approach to sculpting that material into time-based structures.

My earliest generative systems also tended to be improvisational: I would manually set up constraints on harmony, melody, rhythm, and other musical elements, and allow my rudimentary musical agents to explore the sonic potential of the space, stepping in when necessary to make subtle (or not so subtle) changes when I, as a listener, deemed the mercurial results to be getting boring.

The difficulty in automating such high-level decisions is due to the nature of the decisions themselves: they are aesthetic judgements made entirely based upon contextual relationships. A certain harmony may get boring quickly if not enough melodic and rhythmic variation is occurring, yet the same harmony may be acceptable if the particular concurrent melodic/rhythmic generation is deemed "interesting"; the issue, of course, is how to determine what is “interesting” in a given situation – computational aesthetics remains, for the time being, an open problem [3].

A realisation occurred to me several years ago when a research group I was involved with began to consider generative methods of creating musical structure [4]; more specifically, it was a consideration of an alternative method of musical form, namely moment-form [5], which led to a proof-of-concept system described in 2017 [2]. The suggestion was to generate non-teleological structures by ignoring the Germanic tradition of formal development and embracing more static models found in various musical traditions, including ambient electronica, non-Western music, and experimental music as initially proposed by Stockhausen [6].

Most of my generative works since 2016 have embraced this method, generating entire musical formal structures prior to a performance, then allowing musical agents – musebots [7] – to fill in that form with unique details based upon their individual knowledge and abilities. Apart from being able to control high-level notions of formal repetition and variation, such a method allows individual musebots to benefit from having a musical precognition of the structure within which they are operating; knowing a section is two minutes in duration allows them to plan their activity within that time, for example.

A Walk To Meryton is my latest work that explores this compositional approach to generative music, with a new modification. One irony of my own exploration of generative works is the acceptance that, despite having the potential to explore multiple versions of a work, there is a tendency to find one output that is tremendously satisfying and keep that single version as a type of exemplar. I was looking for a method that could retain key aspects of a work – its structure – while allowing new details.

With *A Walk to Meryton*, the system separates the generation of its structure – what I consider to be a *framework* – from the final result completed by the audio musebots – what I consider to be the *score*: frameworks can be saved, with new and alternative details filled in by the musebots. This is similar, if not identical, to how *leadsheets* have been historically treated by jazz musicians: the leadsheet specifies the overall form (the number of measures, which measures are to be repeated, etc.), the harmonic structure, and the melodies. With each performance, a jazz group will retain the overall structure of a song by adhering to a repeating form that utilises a specific harmonic progression and recognisable melody but will vary the improvisational contributions of the individual musicians. The song remains recognisable due to what is retained, but different each performance due to the varied details provided by the improvising musicians. Similarly, a work generated by *A Walk to Meryton*'s system will generate what should be a recognisable form complete with a unique melody and harmonic progression, capable of having multiple versions in "performance" provided by varied musebot output.

1.1 User Control

Before explaining the various parts of the system, a brief mention is necessary to underline the limited user control over the generation; there are only two overall parameters that can be set by the user: *valence* and *arousal*. These two parameters influence decisions made by all aspects of the system, as valence can translate into complexity, and arousal can translate into activity level [8].

2. Generating Structure

An overall structure is the combination of several different phrases grouped into sections; each phrase is composed of an individual rhythmic structure.

2.1 Tala and Phrase length

This rhythmic structure can be considered as a repeating cycle, or *tala*; rather than utilising the Western notion of a divisive pattern – a length of time divided into equal parts – this system uses the South Asian method of additive cyclical patterns.

The first decision made is determining the length of the tala cycle (i.e., how many beats in a measure), with high valence (i.e., lower complexity) favouring 16 beats, and low valence (i.e., higher complexity) favouring an odd number of beats (i.e., 5, 11, 13, 7, 15). The limit of the tala is set at 16.

Next, a phrase length is determined: the number of individual cycles of each tala in a phrase. High valence favours 4 and 8 measures per phrase, while lower valence favours longer phrases (from 9 to 16).

In both cases, the actual valence generates a probability based upon the low and high valence vectors, and a roulette-wheel selection is used to select the individual values.

2.2 Generating Harmonic Progressions

For many years now, my generative systems have used a database of harmonic progressions using a corpus of jazz guitarist Pat Metheny's music [9]; Metheny's music is fundamentally tonal yet avoids many of the obvious progressions found in other jazz music. In its simplest form, a decision is made on the number of chords required in a progression (influenced by the number of measures in a phrase), adjusted by the overall arousal: a higher arousal would likely result in more chords in a phrase; a lower arousal would likely result in fewer chords.

Individual chords in the database are analysed for complexity, essentially the number of semitones and/or added extensions to the basic triad; a starting chord is selected from the possible range based upon the overall valence. The database, organised for Markov generation, then provides all possible probabilistic continuations for the initial chord, with adjustments made due to the overall valence. Thus, while chord Y may be the most likely chord to follow X, the current valence may require a more complex chord, and thus the probability for chord Z would increase.

As with all Markov generative methods, a sequence of selections is produced that makes sense from individual-to-individual element (or taking more than just the previous chord into consideration if using higher order Markov selection), but with little or no direction. Harmony is, however, very much based upon directed motion [10], so generating a progression using Markov strategies, without a goal, is problematic.

A Walk to Meryton attempts one solution by recognising that many progressions, particularly in popular music and jazz, are circular: a section will often consist of a single phrase containing a harmonic progression that is then repeated, with the final chord of the phrase leading back to the beginning (or the next phrase). Thus, four chords, for example, will logically follow one another – A to B to C to D – and the fourth chord, D, will need to logically transition back to the first.

To accomplish a system of circular harmonic progressions, the chord generating system was run for hours, creating hundreds of thousands of chord progressions in lengths of four to eight chords; each ending chord was then tested for the probabilistic movement to the first chord. Those progressions that passed a threshold were retained in a database, and each progression was then rated for overall valence.

During generation, the system decides the length of progression (with a maximum number of eight chords possible within a phrase) based upon the overall arousal, and selects probabilistically from the database based upon the overall valence.

2.3 Sections

The above determinations – tala, phrase length, harmonic progression – are made three times and stored, as the system generates an alteration of three possible sections: A, B, and C. In a related research project [11] it was determined that the vast majority of electronic dance music incorporates three basic sections: a main section (A) that operates similar to a verse in song form; a chorus-type section of high activity (C), and a low activity section that operates as a breakdown (B). Initial testing of structure generation limited to these three sections was found to produce enough variation coupled with balanced audible formal repetition.

A Markov probability table was created, by hand, to provide probabilities for each section. There are two additional sections – (I) for Introduction, and (O) for Outro; these are always (A) sections but with low arousal values.

An example probability vector for the first phrase (I) is as follows:

- 0.4 probability that it remain (I);
- 0.6 probability that it moves to (A);
- 0.3 probability that it moves to (C);
- 0.0 probability that it moves to (O);
- 0.0 probability that the composition ends.

A check is made to ensure that there are between three and sixteen phrases. An example structure generation could be as follows, given an arousal and valence of 0.4:

- I C C A A A B B C C B A B O

Repetitions of phrases are grouped into sections, thus the above phrase pattern results in the following overall structure of nine sections:

- Section 1 : I (Intro)
- Section 2 : C C
- Section 3 : A A A
- Section 4 : B B
- Section 5 : C C
- Section 6 : B
- Section 7 : A
- Section 8 : B
- Section 9 : O (Outro)

In this example generation, some sections consist of a single phrase (6, 7, 8), while the others consist of repetitions of phrases (i.e., Section 2, with two repetitions of phrase C).

2.3.1 Structural States

Once the overall formal structure has been generated, the states – on or off – for each musebot part is determined. Although the potential for self-organisation suggests that individual agents could determine on their own whether to perform within any section or phrase, an overriding "compositional approach" was favoured to ensure structural logic. Furthermore, the overall states and their unfolding over time contributes greatly to the perception of a single unified composition: the states are stored as part of the structure, so subsequent re-generations will contain the same states; for example, if the single Introduction phrase in the example above only contains a drone, a single Shapebot (described later), and a secondary percussion (also described later), each new generation will also only contain these parts.

There are eleven musebot parts: four Shapebots, Pad, Drone, Bass, Melody, Melody2, Percussion, and Percussion2. Each part has a probability for individual sections, set by hand, and adjusted by arousal. For example, the probability for ShapeBot1 in the Introduction is shown below:

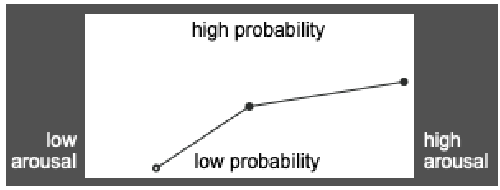

Figure 1. The probability for ShapeBot1 in the Introduction is determined by Arousal

The system will generate individual states based upon the current section, making sure that a minimum number of parts are active for that section. Once complete, a series of checks are made: for example, making sure that if specific parts becomes active in a given section, they remain active for the remainder of the phrases in that section; that within sections, phrases accumulate in activity level (i.e. the number of active states); and the elimination of duplicate phrases.

An example generation of track states is as follows:

		s1	s2	s3	s4	pad	drone	bass	mel	mel2	perc	perc2
0	I	0	1	0	0	0	1	0	0	0	1	0
1	C	0	0	1	0	0	1	1	0	0	0	0
2	C	0	0	1	0	0	0	1	0	0	1	0
3	A	1	0	0	0	0	0	1	1	0	0	0
4	A	1	0	0	0	0	1	1	1	0	0	0
5	A	1	0	0	1	0	0	1	1	0	0	0
6	B	0	1	0	0	1	1	0	0	0	0	0
7	B	0	1	0	0	1	1	0	0	0	0	1
8	C	0	0	1	0	0	0	1	1	0	1	0
9	C	0	0	1	0	0	1	1	0	0	1	0
10	B	0	1	0	0	0	1	0	0	0	1	1
11	A	1	0	1	1	1	0	1	1	0	0	0
12	B	0	1	0	0	1	1	0	0	0	1	1
13	O	0	0	0	0	0	1	0	0	1	0	0
14	end	0	0	0	0	0	0	0	0	0	0	0

Figure 2. An example of individual track states for a structure

For a complete framework, the talas, phrase lengths, and harmonic progressions for each section can be saved, along with the above track states.

3. Generating Parts: Individual Musebot preferences

As mentioned, there are eleven different parts, and eleven different musebots that generate material based upon the generated framework. Each musebot has a particular function; given the valence and arousal for each phrase, musebots will generate their material and write them to a score, which consists of event specifications (i.e. note onsets and durations).

Prior to generating individual parts, a general melodic shape is generated for a work, which melodic musebots (ShapeBots) follow.

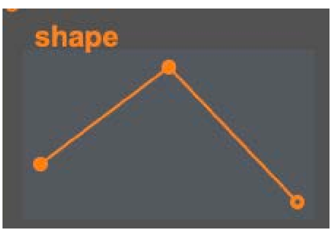

Figure 3. An Example Shape

The four ShapeBots interpret this shape when generating their melodic material. An example for three measures of ShapeBot1 using the above shape – displayed as piano roll notation – is given below:

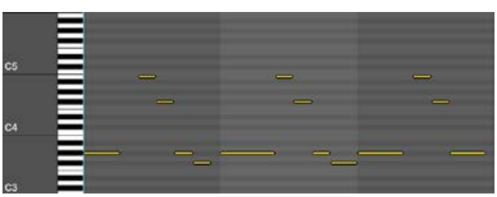

Figure 4. Piano roll notation for three measures of ShapeBot1

Compare this with the output of ShapeBot3 for the same three measures:

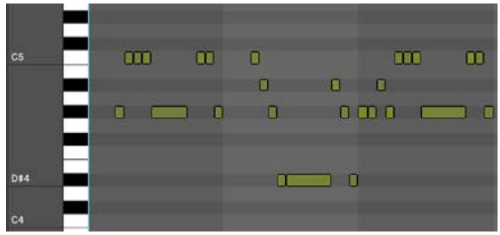

Figure 5. Piano roll notation for three measures of ShapeBot3

Hopefully it is apparent that the same melodic shape is generated by the two different musebots, however interpreted differently in terms of onset placement.

All musebots generate parts for the specific phrases set in the track states: each musebot writes its part as a clip in Ableton Live:

Figure 6. Generated clips for each part in Ableton Live

As mentioned, each musebot fulfils a separate function: the four ShapeBots provide melodic figurations based upon the composition's generated shape; the PadBot will generate held chordal tones; the DroneBot will generate long held pitches, attempting to minimise pitch changes by finding the common chord tones between harmonies; the BassBot will generate bass parts; the two Melodic bots will generate melodies (using a corpus of Pat Metheny melodies and 2nd order Markov generation); the PerBots will generate rhythmic material.

3.2 Sound selection

Once every part has been generated, each musebot selects a synthesiser voice to perform its part. Several synthesisers are available – Absynth, FM8, Kontakt, Massive, Omnisphere, and a variety of Ableton and bespoke synthesisers – and each synthesiser's preset sounds have been pre-analysed for spectral content, suitability, and range, creating a large database of available timbres. Each musebot has a distinct preference for specific patches within their available synthesisers.

Given a generated part with a limited pitch range, the musebot selects a synthesiser, then a timbre suitable to its generated part. Although there is a finite number of possible patch combinations, that number is very large; there are, for all practical purposes, a near-infinite number of possible sound worlds that the musebots can explore.

4. The Score

Figures 4 and 5 demonstrate how musebot write their parts into Ableton Live clips, which allows for this standard commercially available Digital Audio Workstation (DAW) to perform the collective music. At the same time, each part is stored in a single collective score with each part's onsets within the phrase, its MIDI pitch, velocity, and duration. This is stored along with the composition's framework as one example output.

With a single collective score, each musebot's data is available to every other musebot, allowing for musebots to make decisions based upon existing data. For example, the DroneBot examines the pitch ranges of other musebots active within a section and attempts to place its pitches in contrasting ranges.

5. Human Interaction: Improvising to Score

Although *A Walk to Meryton* generates complete compositions, the intention was always to allow for human interaction by improvising musicians. As mentioned, the system generates a framework for each composition, and this can be translated into standard musical notation. Figure 7 demonstrates the first two sections of "Room for a Moment", displaying the tala, the different harmonic progressions between the two sections, and the melody for the B section.

Figure 7. Leadsheet for "Room for a Moment"

These scores were given to three musicians: trumpeter John Korsrud; saxophonist Jon Bentley; and violinist Meredith Bates, each of which are expert improvisors. I determined which instruments would play during which sections and provided the recordings to the musicians before individual recording sessions. In sections with notated melodies, they were given the option of re-interpreting the melody, suggesting it, or ignoring it.

During the recordings with the musicians, I made very few suggestions or comments, treating the human musicians as I did the musebots: giving each an abundance of creative space.

Poet Barbara Adler was also asked to contribute and collaborate; Barbara and I had long conversations about walking, Jane Austin, musebots, and internal dialogs. Barbara then added her own take on these ideas and provided readings.

6. Video Generation

The generated frameworks for each composition are used in the generation of video for each work in *A Walk to Meryton*. Videos are generated by selecting from a database of photographs taken by myself on various nature walks (see Section 7.1). The database is sorted into individual walks, with subdirectories based upon specifics of the walk; for example, "Daisies"; "Fallen Tree"; "Ferns". Each subdirectory requires, at minimum, five photographs.

Videos are generated in realtime, after the generation of all audio data. Prior to performance, the video system selects a directory, and selects one photograph for each section: I A B C O. When that section is played, the corresponding selected photograph is selected.

Motion within the video is created through panning and subtle changes in video processing. At the start of each section, an initial start location and final end location is generated, the distance between them determined by the section's arousal value.

The amount of processing is similarly determined by the overall valence of a composition; the processing itself – erode and dilate processes – selected to suggest a painterly result. A posterize process is then added.

Text from Austen's *Pride and Prejudice* is superimposed on the image in one of two ways: individual lines randomly placed on screen, or several lines written onto a virtual sheet which billows using physical models. In some cases, text written and spoken by poet Barbara Adler is used instead of Austen's; the actual text written by Adler is loaded, and the position within the overall composition determines which lines are selected. As the correlation is not exact, the effect is the text sometimes preceding the spoken word, and other times following it.

7. A Complete Generative Composition

The described system generates complete compositions, including selecting timbres for playback. As mentioned, the user is only required to adjust overall valence and arousal parameter values, click "Generate", and then wait for the result.

My own role has been limited to curating the final output. The system produces music results that I find deeply moving and beautiful; in generating the ten works for *A Walk to Meryton*, I found that I was able to generate one new work a day, rejecting perhaps three generations and accepting on average the fourth. I had to make a conscious decision to stop generating new works, as I continued to discover very successful musical results. In the end, *A Walk to Meryton* consists of ten individual works, ranging in duration between four and ten minutes; the ten tracks eventually will be released on vinyl (a determining factor on the ten-track limit).

The collaboration that I have enjoyed with musebots, such as those within *A Walk to Meryton*, is discussed in detail elsewhere [12]; suffice to say, I consider musebots to be more than tools used in the creation of new music, but collaborative partners that have allowed me to produce music that I've always wanted to hear.

7.1 About the title

I will readily acknowledge that I have a very hard time coming up with titles for my compositions, generative or otherwise. As a result, I have used an algorithm to generate titles for almost a decade. Using a variation of the Markov algorithm described in Section 2.2, the complete text of Jane Austen's *Pride and Prejudice* – raw text that has been on my computer for many years – has been analysed for continuations. To generate a title, the algorithm selects a random word from the database, then produces a fixed number of continuants. When the first test score for this new system required a title, one of the generations was the phrase "A Walk to Meryton"; I decided that this was an ideal evocation of the emotions possible by the system: solo walks through nature, an individual lost in contemplation. Each movement had its own title generated in the same way.

8. References

1. Eigenfeldt, A. "Collaborative Composition with Creative Systems", International Symposium on Electronic Art (ISEA), Durban, 2018.

2. Eigenfeldt, A. “Designing Music with Musebots”, 5th Conference on Computation, Communication, Aesthetics, and X (xCoAx), Lisbon, 2017.

3. Galanter, Philip. "Computational Aesthetic Evaluation: Past and Future." *Computers and creativity* (2012): 255-293.

4. Eigenfeldt, Arne, Oliver Bown, Andrew R. Brown, and Toby Gifford. "Flexible Generation of Musical Form: Beyond Mere Generation." In *Proceedings of the seventh international conference on computational creativity*, pp. 264-271. 2016.

5. Eigenfeldt, Arne. "Exploring Moment-Form In Generative Music." In *Proceedings of the Sound and Music Computing Conference*. 2016.

6. Stockhausen, Karlheinz. "Momentform: Neue Beziehungen zwischen Aufführungsdauer, Werkdauer und Moment." *Texte zur Musik* 1 (1963): 189-210.

7. Bown, Oliver, Benjamin Carey, and Arne Eigenfeldt. "Manifesto for a Musebot Ensemble: A Platform for Live Interactive Performance Between Multiple Autonomous Musical Agents." Proceedings of the International Symposium of Electronic Art. 2015.

8. Eigenfeldt, Arne, Jim Bizzocchi, Miles Thorogood, and Justine Bizzocchi. "Applying Valence and Arousal Values to a Unified Video, Music, and Sound Generative Multimedia Work." In *Generative Art Conference (GA)*. 2015.

9. Eigenfeldt, Arne, and Philippe Pasquier. "Realtime Generation of Harmonic Progressions using Controlled Markov Selection." In *Proceedings of ICCC-X-Computational Creativity Conference*, pp. 16-25. 2010.

10. Salzman, Eric. *Twentieth-century Music: An Introduction*. Pearson College Division, 2002.

11. Eigenfeldt, Arne, and Philippe Pasquier. "Towards a Generative Electronica: Human-Informed Machine Transcription and Analysis in MaxMSP." In *Proceedings of Sound and Music Computing Conference, Padua*. 2011.

12. Eigenfeldt, Arne. "Musebots and I: Collaborating with Creative Systems." In *The Language of Creative AI*, pp. 207-216. Springer, Cham, 2022.