A Real-Time Genetic Algorithm in Human-Robot Musical Improvisation

Gil Weinberg, Mark Godfrey, Alex Rae, and John Rhoads

Georgia Institute of Technology,
Music Technology Group
840 McMillan St, Atlanta GA 30332, USA
{gilw,mark.godfrey,arae3}@gatech.edu,
jfrhoads@gmail.com
http://music.gatech.edu/mtg/

Abstract. The paper describes an interactive musical system that utilizes a genetic algorithm in an effort to create inspiring collaborations between human musicians and an improvisatory robotic xylophone player. The robot is designed to respond to human input in an acoustic and visual manner, evolving a human-generated phrase population based on a similarity driven fitness function in real time. The robot listens to MIDI and audio input from human players and generates melodic responses that are informed by the analyzed input as well as by internalized knowledge of contextually relevant material. The paper describes the motivation for the project, the hardware and software design, two performances that were conducted with the system, and a number of directions for future work.

Keywords: genetic algorithm, human-robot interaction, robotic musicianship, real-time interactive music systems.

1 Introduction and Related Work

Real-time collaboration between human and robotic musicians can capitalize on the combination of their unique strengths to produce new and compelling music. In order to create intuitive and inspiring human-robot collaborations, we have developed a robot that can analyze music based on computational models of human percepts and use genetic algorithms to create musical responses that are not likely to be generated by humans. The two-armed xylophone playing robot is designed to listen like a human and improvise like a machine, bringing together machine musicianship with the capacity to produce musical responses on a traditional acoustic instrument.

Current research directions in musical robotics focus on sound production and rarely address perceptual aspects of musicianship, such as listening, analysis, improvisation, or group interaction. Such automated musical devices include both Robotic Musical Instruments — mechanical constructions that can be played by live musicians or triggered by pre-recorded sequences — and Anthropomorphic Musical Robots — humanoid robots that attempt to imitate the action of human

R. Kronland-Martinet, S. Ystad, and K. Jensen (Eds.): CMMR 2007, LNCS 4969, pp. 351–359, 2008.

musicians (see a historical review of the field in [4]). Only a few attempts have been made to develop perceptual robots that are controlled by neural networks or other autonomous methods . Some successful examples for such interactive musical systems are Cypher [9], Voyager [6], and the Continuator [8]. These systems analyze musical input and provide algorithmic responses by generating and controlling a variety of parameters such as melody, harmony, rhythm, timbre, and orchestration. These interactive systems, however, remain in the software domain and are not designed to generate acoustic sound.

As part of our effort to develop a musically discerning robot, we have explored models of melodic similarity using dynamic time warping. Notable related work in this field is the work by Smith at al. [11], which utilized a dynamic-programming approach to retrieve similar tunes from a folk song database. The design of the software controlling our robot includes a novel approach to the use of improvisatory genetic algorithms. Related work in this area includes GenJam [2], an interactive computer system that improvises over a set of jazz tunes using genetic algorithms. GenJam's initial phrase population is generated stochastically, with some musical constraints. Its fitness function is based on human aesthetics, where for each generation the user determines which phrases remain in the population. Other musical systems that utilize human-based fitness functions have been developed by Moroni [7], who uses a real-time fitness criterion, and Tokui [12], who uses human feedback to train a neural network-based fitness function. The Talking Drum project [3], on the other hand, uses a computational fitness function based on the difference between a given member of the population and a target pattern. In an effort to create more musically relevant responses, our system is based on a human-generated initial population of phrases and a similarity-based fitness function, as described in detail below.

2 The Robotic Percussionist

In previous work, we developed an interactive robotic percussionist named Haile [13]. The robot was designed to respond to human drummers by recognizing low-level musical features such as note onset, pitch, and amplitude as well as higher-level percepts such as rhythmic stability and similarity. Mechanically, Haile controls two robotic arms; the right arm is designed to play fast notes, while the left arm is designed to produce larger and more visible motions, which can create louder sounds in comparison to the right arm. Unlike robotic drumming systems that allow hits at only a few discrete locations, Haile's arms can move continuously across the striking surface, which can allow for pitch generation using a mallet instrument instead of a drum. For the current project, Haile was adapted to play a one-octave xylophone. The different mechanisms in each arm, driven either by a solenoid or a linear-motor, led to a unique timbral outcome. Since the range of the arms covers only one octave, Haile's responses are filtered by pitch class.

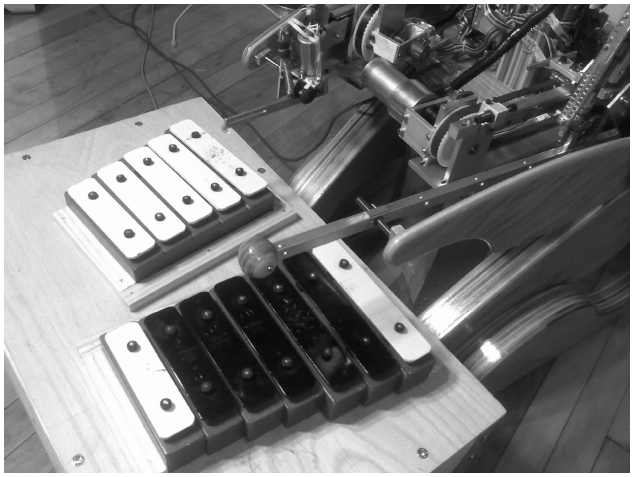


Fig. 1. Haile's two robotic arms cover a range of one octave (middle G to treble G.) The left arm is capable of playing five notes, the right arm seven.

3 Genetic Algorithm

Our goal in designing the interactive genetic algorithm (GA) was to allow the robot to respond to human input in a manner that is both relevant and novel. The algorithmic response is based on the observed input as well as on internalized knowledge of contextually relevant material. The algorithm fragments MIDI and audio input into short phrases. It then attempts to find a "fit" response by evolving a pre-stored, human-generated population of phrases using a variety of mutation and crossover functions over a variable number of generations. At each generation, the evolved phrases are evaluated by a fitness function that measures similarity to the input phrase, and the least fit phrases in the database are replaced by members of the next generation. A unique aspect in this design is the use of a pre-recorded population of phrases that evolves over a limited number of generations. This allows musical elements from the original phrases to mix with elements of the real-time input to create unique, hybrid, and at times unpredictable, responses for each given input melody. By running the algorithm in real-time, the responses are generated in a musically appropriate time-frame.

3.1 Base Population

Approximately forty melodic excerpts of variable lengths and styles were used as an initial population for the genetic algorithm. They were recorded by a jazz pianist improvising in a similar musical context to that in which the robot was intended to perform. Having a distinctly "human" flavor, these phrases provided the GA with a rich pool of rhythmic and melodic "genes" from which to build its own melodies. This is notably different from most standard approaches, in which the starting population is generated stochastically.

3.2 Fitness Function

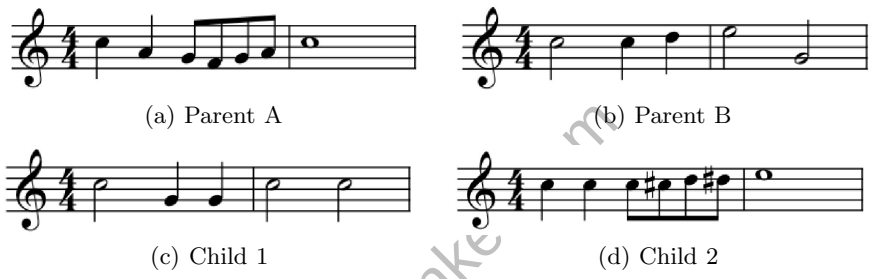
A similarity measure between the observed input and the melodic content of each generation of the GA was used as a fitness function. The goal was not to converge to an "ideal" response by maximizing the fitness metric (which could have led to an exact imitation of the input melody), but rather to use it as a guide for the algorithmic creation of melodies. By varying the number of generations and the type and frequency of mutations, certain characteristics of both the observed melody and some subset of the base population could be preserved in the output.

Dynamic Time Warping (DTW) was used to calculate the similarity measure between the observed and generated melodies. A well-known technique originally used in speech recognition applications, DTW provides a method for analyzing similarity, either through time shifting or stretching, of two given segments whose internal timing may vary. While its use in pattern recognition and classification has largely been supplanted by newer techniques such as Hidden Markov Models, DTW was particularly well suited to the needs of this project, specifically the task of comparing two given melodies of potentially unequal lengths without referencing an underlying model. We used a method similar to the one proposed by Smith [11], deviating from the time-frame-based model to represent melodies as a sequence of feature vectors corresponding to the notes. Our dissimilarity measure, much like Smith's "edit distance", assigns a cost to deletion and insertion of notes, as well as to the local distance between the features of corresponding pairs. The smallest distance over all possible temporal alignments is then chosen, and the inverse (the "similarity" of the melodies) is used as the fitness value. The local distances are computed using a weighted sum of four differences: absolute pitch, pitch class, log-duration, and melodic attraction. The individual weights are configurable, each with a distinctive effect upon the musical quality of the output. For example, higher weights on the log-duration difference lead to more precise rhythmic matching, while weighting the pitch-based differences lead to outputs that more closely mirror the melodic contour of the input. Melodic attraction between pitches is calculated based on the Generative Theory of Tonal Music model [5]. The relative balance between the local distances and the temporal deviation cost has a pronounced effect — a lower cost for note insertion/deletion leads to a highly variant output. A handful of effective configurations were derived through manual optimization.

The computational demands of a real-time context required significant optimization of the DTW, despite the relatively small length of the melodies (typically between two and thirty notes). We implemented a standard path constraint on the search through possible time alignments in which consecutive insertions or deletions are not allowed. This cut computation time by approximately one half but prohibited comparison of melodies whose lengths differ by more than a factor of two. These situations were treated as special cases and were assigned an appropriately low fitness value. Additionally, since the computation time is proportional to the length of the melody squared, a decision was made to break longer input melodies into smaller segments to increase the efficiency and remove the possibility of an audible time lag.

3.3 Mutation and Crossover

With each generation, a configurable percentage of the phrase population is chosen for mating. This "parent" selection is made stochastically according to a probability distribution calculated from each phrase's fitness value, so that more fit phrases are more likely to breed. The mating functions range from simple mathematical operations to more sophisticated musical functions. For instance, a single crossover function is implemented by randomly defining a common dividing point on two parent phrases and concatenating the first section from one parent with the second section from the other to create the child phrase. This mating function, while common in genetic algorithms, does not use structural information of the data and often leads to non-musical intermediate populations of phrases. We also implemented musical mating functions that were designed to lead to musically relevant outcomes without requiring that the population converge to a maximized fitness value. An example of such a function is the pitch-rhythm crossover, in which the pitches of one parent are imposed on the rhythm of the other parent. Because the parent phrases are often of different lengths, the new melody follows the pitch contour of the first parent, and its pitches are linearly interpolated to fit the rhythm of the second parent.



(a) Parent A (b) Parent B

(c) Child 1 (d) Child 2

Fig. 2. Mating of two prototypical phrases using the pitch-rhythm crossover function. Child 1 has the pitch contour of Parent A and rhythm pattern of Parent B while Child 2 has the rhythm of Parent A and the pitch contour of Parent B.

Additionally, an adjustable percentage of each generation is mutated according to a set of functions that range in musical complexity. For instance, a simple random mutation function adds or subtracts random numbers of semitones to the pitches within a phrase and random lengths of time to the durations of the notes. While this mutation seems to add a necessary amount of randomness that allows a population to converge toward the reference melody over many generations, it degrades the musicality of the intermediate populations. Other functions were implemented that would stochastically mutate a melodic phrase in a musical fashion, so that the outcome is recognizably derivative of the original. The density mutation function, for example, alters the density of a phrase by adding or removing notes, so that the resulting phrase follows the original pitch contour with a different number of notes. Other simple musical mutations include inversion, retrograde, and transposition operations. In total,

seven mutation functions and two crossover functions were available for use with the algorithm, any combination of which could be manually or algorithmically applied in real-time.

4 Interaction Design

In order for Haile to improvise in a live setting, we developed a number of human-machine interaction schemes. Much like a human musician, Haile must decide when and for how long to play, to which other player(s) to listen, and what notes and phrases to play in a given musical context. This creates the need for a set of routines to handle the capture, analysis, transformation, and generation of musical material in response to the actions of one or more musical partners. While much of the interaction we implemented centers on a call-and-response format, we have attempted to dramatically expand this paradigm by allowing the robot to interrupt, ignore, or introduce new material. It is our hope that this creates an improvisatory musical dynamic which can be surprising and exciting.

4.1 Input

The system receives and analyzes both MIDI and audio information. Input from a digital piano is collected using MIDI while the Max/MSP object pitch~ (`http://web.media.mit.edu/~tristan/maxmsp.html`) is used for pitch detection of melodic audio from acoustic instruments. The incoming audio is filtered and compressed slightly in order to improve results.

4.2 Simple Interactions

In an effort to establish Haile's listening abilities in live performance settings, simple interaction schemes were developed that do not use the genetic algorithm. One such scheme is direct repetition of human input, in which Haile duplicates any note that is received from MIDI input, creating a kind of roll which follows the human player. In another interaction scheme, the robot records and plays back complete phrases of musical material. A predefined chord sequence causes Haile to start listening to the human performer, and a similar cue causes it to play back the recorded melody. A simple but rather effective extension of this approach utilizes a mechanism that stochastically adds notes to the melody while preserving the melodic contour, similarly to the density mutation function described in Sect. 3.3.

4.3 Genetic Algorithm Driven Improvisation

The interaction scheme used in conjunction with the genetic algorithm requires more flexibility than those described above, in order to allow for free-form improvisation. The primary tool used to achieve this goal is an adaptive call-and-response mechanism which tracks the mean and variance of inter-onset times in the input. It uses these to distinguish between pauses that should be considered

part of a phrase and those that denote its end. The system quickly learns the typical inter-onset times expected at any given moment. Then the likelihood that a given pause is part of a phrase can be estimated; if the pause continues long enough, the system interprets that silence as the termination of the phrase.

If the player to whom Haile is listening pauses sufficiently long, the phrase detection algorithm triggers the genetic algorithm. With the optimizations described in Sect. 3.2, the genetic algorithm's output can be generated in a fraction of a second (typically about 0.1 sec.) and thus be played back almost immediately, creating a lively and responsive dynamic. We have attempted to break the regularity of this pattern of interaction by introducing some unpredictability. Specifically, we allow for the robot to occasionally interrupt or ignore the other musicians, reintroduce material from a database of genetically modified phrases generated earlier in the same performance, and imitate a melody verbatim to create a canon of sorts.

In the initial phase of the project, a human operator was responsible for controlling a number of higher-level decisions and parameters during performance. For example, switching between various interaction modes, the choice of whether to listen to the audio or MIDI input, and the selection of mutation functions were all accomplished manually from within a Max/MSP patch. In order to facilitate autonomous interaction, we developed an algorithm that would make these decisions based on the evolving context of the music, thus allowing Haile to react to musicians in a performance setting without the need for any explicit human control. Haile's autonomous module thus involves switching between four different playback modes. "Call-and-response" is described above and is the core. "Independent playback" mode is briefly mentioned above; in it, Haile introduces a previously generated melody, possibly interrupting the other players. In "Canon" mode, instead of playing its own material, the robot echoes back the other player's phrase at some delay. Finally, "Solo" mode is triggered by a lack of input from the other musicians, and causes Haile to continue playing back previously generated phrases from its database until both other players resume playing and interrupt the robotic solo.

Independently of these playback modes, the robot periodically changes the source to which it listens, and changes the various parameters of the genetic algorithm (mutation and crossover types, number of generations, amount of mutation, etc.) over time. In the end, the human performers do not know a priori which of them is driving Haile's improvisation or exactly how Haile will respond. We feel this represents a workable model of the structure and dynamic of interactions that can be seen in human-to-human musical improvisation.

5 Performances

Two compositions were written for the system and performed in three concerts. In the first piece, titled "Svobod," a piano and a saxophone player freely improvised with the robot. The first version of "Svobod" used a semi-autonomous system and a human operator (see video excerpts — `http://www.coa.gatech.`

edu/~gil/Svobod.mov). In its second version, performed at ICMC 2007, the full complement of autonomous behaviors described in Sect. 4.3 was implemented. The other piece, titled "iltur for Haile," also utilized the fully autonomous system, and involved a more defined and tonal musical structure utilizing genetically driven as well as non-genetically driven interaction schemes, as the robot performed with a full jazz quartet (see video excerpts http://www.coa.gatech.edu/~gil/iltur4Haile.mov).

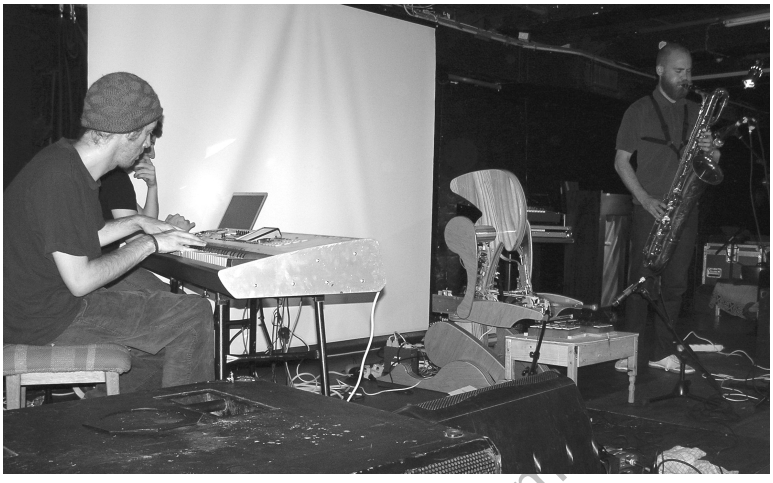


Fig. 3. Human players interact with Haile as it improvises based on input from saxophone and piano in "Svobod" (performed August 31, 2007, at ICMC in Copenhagen, Denmark)

6 Summary and Future Work

We have developed an interactive musical system that utilizes a genetic algorithm in an effort to create unique musical collaborations between humans and machines. Novel elements in the implementation of the project include using a human-generated phrase population, running the genetic algorithm in real-time, and utilizing a limited number of evolutionary generations in an effort to create hybrid musical results, all realized by a musical robot that responds in an acoustic and visual manner. Informed by these performances, we are currently exploring a number of future development directions such as extending the musical register and acoustic richness of the robot, experimenting with different genetic algorithm designs to improve the quality of musical responses, and conducting user studies to evaluate humans' response to the algorithmic output and the interaction schemes.

References

1. Baginsky, N.A.: The Three Sirens: A Self-Learning Robotic Rock Band (Accessed May 2007), http://www.the-three-sirens.info
2. Biles, J.A.: GenJam: a genetic algorithm for generation of jazz solos. In: Proceedings of the International Computer Music Conference, Aarhus, Denmark (1994)
3. Brown, C.: Talking Drum: A Local Area Network Music Installation. Leonardo Music Journal 9, 23–28 (1999)
4. Kapur, A.: A History of Robotic Musical Instruments. In: Proceedings of the International Computer Music Conference, Barcelona, Spain, pp. 21–28 (2005)
5. Lerdahl, F., Jackendoff, R.: A Generative Theory of Tonal Music. MIT Press, Cambridge (1983)
6. Lewis, G.: Too Many Notes: Computers, Complexity and Culture in Voyager. Leonardo Music Journal 10, 33–39 (2000)
7. Moroni, A., Manzolli, J., Zuben, F., Gudwin, R.: An Interactive Evolutionary System for Algorithmic Music Composition. Leonardo Music Journal 10, 49–55 (2000)
8. Pachet, F.: The Continuator: Musical Interaction With Style. Journal of New Music Research 32(3), 333–341 (2003)
9. Rowe, R.: Interactive Music Systems. MIT Press, Cambridge (1992)
10. Rowe, R.: Machine Musicianship. MIT Press, Cambridge (2004)
11. Smith, L., McNab, R., Witten, I.: Sequence-based melodic comparison: A dynamic-programming approach. Melodic Comparison: Concepts, Procedures, and Applications. Computing in Musicology 11, 101–128 (1998)
12. Tokui, N., Iba, H.: Music Composition with Interactive Evolutionary Computation. In: Proceedings of the 3rd International Conference on Generative Art, Milan, Italy (2000)
13. Weinberg, G., Driscoll, D.: Toward Robotic Musicianship. Computer Music Journal 30(4), 28–45 (2007)