Virtual Performance Modelling

Brad Garton
Music Department
Columbia University
New York, NY 10027
brad@woof.music.columbia.edu

ABSTRACT

This paper presents a "layered" approach to the creation of artificial performers intended to mimic human performance characteristics. Building upon synthesis algorithms conceived as physical models of actual dynamical systems, this performance model employs rules ranging from constraints on possible actions (i.e. the amount of time it takes for a human player to shift hand positions, etc.) to rules governing harmonic and melodic unfolding. The layered model differs from hierarchical models in that control within the program is not governed from a fixed level (neither top-down nor bottom-up), but instead flows between layers as particular ad-hoc local decisions are made. Programs based upon this model were used to create the pieces Rough Raga Riffs *and* Almost Real *-- the realization of these pieces is discussed along with possible directions for future work.*

Introduction

This paper discusses two related computer programs (Piece-o-Matic and Riff-o-Matic) I have developed recently. I should probably begin with a few disclaimers: although the programs are intended to model human performance characteristics within certain musical styles, they are not intended to be a generalized set of performance rules like those developed for "traditional tonal music" by Friberg, et. al. (Friberg, et. al. 1991). Nor are these programs meant to function as algorithmic models of human composition or improvisation in the same sense as the connectionist approach taken by Peter Todd (Todd, 1989) or any of the cognitive models described by Otto Laske (Laske, 1988). Piece-o-Matic and Riff-o-Matic have no "deep knowledge" of how music should be constructed.

Instead, these programs grew as I was working on specific pieces of music. Any "knowledge" imbedded into the programs in the form of rules or procedures was coded to meet particular musical demands. Thus these programs were written on a very ad-hoc basis, with no guiding model existing prior to the programming. The performance model discussed in the next section came from an analysis of the working programs -- it did not dictate their development.

The Model

Piece-o-Matic and Riff-o-Matic are intended to simulate the improvisational behavior of musicians working within the "string folk band" idiom and the "solo rock guitar" idiom, respectively. Riff-o-Matic (the first one written) was originally intended to test various combinations of parameters for Charles Sullivan's "strum" sound synthesis algorithm (Sullivan, 1990). Soon I noticed that the simple test procedures were producing music which was unusual and quite interesting. I began to add more and more procedures for making note-by-note decisions, mostly based on idiomatic playing techniques I heard in solo rock guitar playing. Piece-o-Matic was an elaboration of Riff-o-Matic, with the focus being a simulation of an ensemble of folk musicians performing on various stringed instruments. It is worth noting that all of my work is predicated upon the existence of a sophisticated synthesis technique such as the "strum" algorithm. In order for my virtual performers to work, there must be some good virtual instruments for them to play.

The types of rules included in the programs can be divided conceptually into four separate "layers":

-- *the physical layer:* Rules at this layer consist of decision procedures related to the physical actions involved in producing sound on an actual instrument. A slight delay before each note in a strummed chord reflecting the travel time of the pick from one string to another is an example of this type of rule. Timbral differences due to different note articulations (such as an up-pick or a down-pick) are part of this level. The sounding of intermediate notes during a single-string glissando across a fretted guitar neck is another example of a rule at this level. There are also a number of rules for choosing parameter values probabilistically within certain ranges. This reflects the "imperfection" of human performance, plus these rules can be used to simulate statistical tendencies, such as "pushing" certain beat values, or flattening particular scale degrees.

-- *the inflection layer:* This layer built upon the previous layer to encompass particular stylistic articulations such as pitch-bends, vibrato, hammer-ons, etc. It is at this level that I feel much of the idiomatic-specific information was coded. Ways of articulating small groups of notes, such as the "Van Halen" hammer-on technique in rock guitar playing, or a double-picking effect in Irish folk music seem to be a large part of the stylistic cues we hear.

-- *the riff layer:* Stringing sets of inflections together into longer sets of notes happens at this layer. Inflection rules are used to guide the intersection of pitch and rhythm templates to produce short, motif-like musical units. The idea behind this level came from observations of how rock and folk guitarists learn musical gestures. Rhythm and pitch patterns (colloquially known as "riffs") are practiced repeatedly, and then used to build longer musical passages. The pitch and rhythm templates encoded at this layer are meant to function as these motivic building blocks.

-- *the shape layer:* Rules at this level are meant to establish a context for the sequencing of riffs. Most harmonic and melodic knowledge comes from this layer. In Riff-o-Matic, rules at this level established melodic trajectories and determined the level of rhythmic activity. Piece-o-Matic also included rules governing the behavior of a group of virtual performers.

Implementation

Although I have presented the model as a nicely-structured hierarchy, the actual parameter decisions are made in a very tangled manner. It is easy to think that the procedures are invoked logically through the layers I have described -- the harmonic and melodic context is set by the shape layer, which then constructs sequences of riffs, which call upon particular sets of inflections, etc. In actuality, however, the note-by-note decisions are made at an extremely local level.

Whether or not a particular rule or procedure from any layer is being used is usually determined probabilistically. The programs work by writing cmix scorefiles. To write a single note, Piece-o-Matic (or Riff-o-Matic) uses a list of instrument parameters to invoke procedures necessary for the assignment of numerical values. Which set of procedures is being used for each parameter might be determined by the riff currently being played, or by a particular inflection being chosen, or because of some physical-level constraints. All of these are controlled by probabilistic choice. In other words, it is extremely difficult to predict which rules will be used to determine the pitch, duration or timbre of a given note because coins are being tossed at some fundamental level in the program. I suspect that this is what makes the output interesting.

Some Observations

I was surprised at how few rules it took (and how "dumb" the rules were) to produce some stylistically-passable music. Having a powerful synthesis algorithm with the appropriate "handles" for hooking in a set of physical performance constraints was probably the reason for this. Even a simple "windowed" pitch trajectory up and down a scale (i.e. pitches are chosen randomly within a window which slides along a list of pitches) sounded musical when performed by a program with some rudimentary knowledge of what was physically possible. After the programs had reached a certain level of complexity, it seemed that I could almost make up any rules I wanted governing the unfolding of the riffs, and some sort of bizarre music would result -- but the music would be stylistically coherent.

I had great fun creating pieces by interacting with these programs. I doubt that they contain many ideas which could be applied to the synthesis of "music in general". A criticism from those wishing to discover fundamental principles underlying human musical behavior might be that these programs are so highly idiosyncratic as to be almost piece-specific. For me, however, the real "kick" of doing computer music is the ability to design and implement particular working methodologies for the creation of specific pieces. Designing Piece-o-Matic and Riff-o-Matic was a part of the compositional process; I don't consider writing them to be separate from the actual composing of the music I created with them.

I was also intrigued by the notion that with these programs I was in a sense simulating different cultures. As our world becomes more homogenized and more disconnected from diverse cultural traditions, virtual recreations of what we lose may come to represent and replace the original. Through music, people can vicariously participate in particular cultures and societies. It may be that simulations of folk idioms will meet human needs for communal continuity by providing a nostalgic representation of communitarian ideals. I'm not sure that this is a real good thing.

References

Friberg, A., Fryden, L., Bodin, L., and Sundberg, J. 1991. "Performance Rules for Computer-Controlled Contemporary Keyboard Music." *Computer Music Journal* 15(2): 49-55.

Laske, O. 1988. "Introduction to Cognitive Musicology." *Computer Music Journal* 12(1): 43-57.

Sullivan, C. 1990. "Extending the Karplus-Strong Algorithm to Synthesize Electric Guitar Timbres with Distortion and Feedback." *Computer Music Journal* 14(3): 26-37.

Todd, P. 1989. "A Connectionist Approach to Algorithmic Composition." *Computer Music Journal* 13(4): 27-43.