The "Blues-o-Matic" Real-time Interactive Performance Model

Brad Garton and Damon Horowitz
Columbia University Music Department
703 Dodge Hall
New York, NY 10027 USA
brad@woof.music.columbia.edu; damon@woof.music.columbia.edu

This paper describes the features of the "Blues-o-Matic" performance system, which consists of a PowerGlove controlling the evolution of a simulated blues-guitar performance. The performance model is implemented on a NeXT machine, using the MusicKit to generate sound on the internal Motorola DSP chip. Gestural data coming from the PowerGlove is interpreted by the performance model, and used to influence the unfolding performance in a relatively "high-level" manner. A discussion of the choice of hardware used in this system along with an overview of the software performance model is included.

Overview

Newer and more sophisticated synthesis algorithms require relatively intelligent control of synthesis parameters in order to be used effectively. For the real-time use of these synthesis techniques, this difficulty can be addressed by building more complex control devices, thus enabling the human performer direct access to the generation of sound. An alternative approach is to imbed more knowledge about how to use the synthesis algorithms in a program designed to interpret relatively high-level gestures and information in the context of an on-going performance. The "Blues-o-Matic" system described in this paper takes the latter approach. The program controls a real-time performance model based on idiomatic blues-guitar playing techniques by interpreting data from a gestural interface (the interface device we currently use is a Nintendo PowerGlove). The interface design was also motivated by an interest in exploring methods for codifying some portion of the intuitive relationship between the gestural movement of a hand and musical expression. This allows a very high-level control of the music, as opposed to the literal control (such as the production of individual notes) provided by most traditional instruments. The PowerGlove user functions in a similar fashion to a conductor in shaping a performance, except with the further ability of being able to determine the actual musical material present by steering the direction of the evolving improvisation (somewhat in the fashion of the leader of a free improvisation jam session).

Sound Synthesis and The Performance Model

The synthesis algorithm we used in our system is a real-time implementation of Charles Sullivan's "strum" algorithm [Sullivan, 1990] developed by Rick VanderKam at Stanford University. VanderKam's version was written using the NeXT MusicKit; the synthesis is accomplished on the Motorola 56001 DSP chip included in the older NeXT hardware. The synthesis program gives fairly high-level control over a distorted electric guitar sound, including the ability to specify pitches for two independent guitar strings and modification of several timbral parameters (distortion coefficients and feedback gain).

To achieve our goal of a "guided improvisation", we elected not to have the PowerGlove control the synthesis algorithm directly. This proved to be a rather serendipitous decision, as the latency of the PowerGlove in parsing gestures (described later in this paper) would have made direct control of all relevant parameters in a musically satisfying way very difficult. Instead we chose to use a separate program, or virtual performer, to act as an interpretive agent. This virtual performer interprets data coming from the PowerGlove as high-level musical directives. In certain cases, our system does allow for the direct specification of particular synthesis parameters, but these act in a global manner and are not confined to a single musical events. This mode of operation is generally used to set or modify baseline default values used by the synthesis algorithm.

The virtual blues performer program was built using ideas we had explored in earlier non-real-time performance models (see [Garton, 1992] for a more complete description of this approach to modelling musical performance). The model uses several interconnected layers: the *physical* layer constrains the synthesis according to "real world" principles -- it takes a finite amount of time for a human performer to move from one note to another on a guitar fretboard, certain combinations of notes are impossible on a real guitar, etc. Idiomatic playing techniques are coded in the *inflection* layer of the model -- pitch-bending methods, the "blues" scale itself, particular picking techniques, etc. The *gestural* layer contains information about specific musical patterns which occur within the performance idiom being modelled. These gestures are the building blocks used by the program to construct longer musical passages.

Our original intention was to have the final layer of the performance model (the *shape* layer -- controlling the context for the choice of gestures) dictated entirely by data from the PowerGlove. In actual practice, however, this approach proved to be far too difficult. Attempting to control the harmonic context, the pacing of the gestures, the "energy level" of the music as well as other unfolding musical factors simultaneously overloaded the capabilities of the PowerGlove (and the PowerGlove user as well!). We decided to give the virtual performer many of the decision-making capabilities included in our earlier performance models and opted to let the PowerGlove influence the direction of the musical decisions rather than make the decisions directly. We did leave open the option to specify exactly a particular gesture or musical choice, however, thinking that it would be wise to have this capability in the hands of the human user.

Use and Design of the PowerGlove Interface

A description of the PowerGlove apparatus is appropriate here, for our approach to the project was significantly affected by its structure and limitations. The glove on the user's hand emits a signal to a receiver which is mounted on the computer monitor. The data output from this hardware is captured by a driver using the NeXT's DSP. The information provided by the glove is as follows: the glove's position in three-dimensional space with respect to the receiver, its degree of clockwise rotation along the z-axis (between the performer and the monitor), and the amount of flex (three degrees of freedom) of each of the fingers. This presents a drastic restriction on the type of physical movement which is meaningfully detectable with this apparatus; there is no facility for determining the rotation of the glove with respect to the other two axes, rotation which is present in most "natural" hand gestures. Furthermore, there is a large error factor in the data which is transmitted; the position in space on any given axis is unreliable (will fluctuate) to a factor of one-tenth of the total range (i.e. for a range of 250 "units", an error of up to 25 in any reading is expected), and the degree of flex in the fingers can only reliably distinguish a fully flexed position from a fully open hand position. In addition, there is a latency of approximately .5 seconds from the movement of the glove to the receipt of the data by our program.

Initially, we wanted to use the glove as a real-time continuous controller producing immediate musical response to a movement. Such an interface provides the intuitively pleasing feel of "sculpting" the music because the sound responds as though it is literally being touched by the glove. The latency of the glove eliminated the possibility of this approach. Instead, we implemented the project in two separate modes: an interactive improvisation steering mode, along the lines of our original conception of the project except modified to accommodate the limitations of the glove; and a gesture parsing mode to take advantage of the possibilities of figuratively painting gestures with the glove. This latter approach is not preempted by the limitations discussed above. These two modes are explained in detail below.

Parsing Gestures

Mapping physical gestures into musical gestures is a difficult task, for neither physical gestures nor musical gestures are well-defined. On the musical side, our approach to the problem was to collect a range of typical blues "riffs" (these are coded in the *gestural* layer of the performance model), and use these as a set of hard-coded (with the exception of some randomness between different versions of a given riff) target gestures. To give us some higher-level control over these riffs from the PowerGlove, they were classified according to a few descriptive parameters -- pitch level, energy, sharpness, direction, etc. These parameters were then addressed from the PowerGlove by a type of motion which bears some literal relation to the parameter. For example, pitch level is determined by position on the y-axis, and a change can be directly signaled by a notch-like movement at a given level; energy is similarly reported by a "zigzag" motion. Alternately, the values for each of the parameters are used as atoms in a simple gestural language, which can parse any given gesture based upon its score for each of the parameter tests.

In performance, this mode begins by marching through a harmonic template of a standard blues form with a default pattern. Once global values have been directly set for the parameters described above, gestures can be entered into the template by a checkmark motion with the glove. This tells the virtual performer to insert a gesture with the current parameters at the indicated point in the template. In addition, any glove gesture can be parsed based upon our descriptive parameters and sent to the virtual performer as a specific musical gesture to be entered into the template. Once the template is replete with gestures to be played at specific points, the glove can be used in a "transformation" mode. The space the glove moves through is here sectioned into a grid, with pitch level values along the y-axis and energy values along the x-axis. When moving the glove through this space, its current position is sent to the performer, which transforms whatever gesture exists in the current position of the template by changing the gesture's pitch level and energy values to correspond to the glove's position (this mode can be toggled to affect all gestures or only those which were not specified directly when entered into the template). This technique of parsing gestures and then steering the transformations of them successfully provides a method for establishing a form (by entering gestures into the template) and then controlling its evolution (transformation) while still maintaining the original structure. However, there is currently no provision for creating new musical material (outside of the set of gestures available) or altering the harmonic structure of the entire template. These additions would help to create a more meaningful and flexible musical tool.

Interactive Improvisation

The goal in interactively steering the computer's improvisation was to provide an interface which would allow for real-time entry of parameters to control the evolution of the music. This objective is related to that of the "transformation" mode above, which directs the course of repeated passes through the harmonic template of gestures, except that here the harmonic template is abandoned in favor of a free improvisation in which the musical material itself is created by the performer. The low-level musical material available in this type of interaction is provided by the virtual performance model, which defines the manner in which sequences of notes can be produced in a blues style given a general direction and certain restrictions. In the interactive improvisation mode, the performer's gestures indicate ranges in which the improvisation can unfold. The parameters include those used in gesture-parsing mode, but are extended to offer more control in steering the actual musical material. The size of the pitch window restricting the music, the pitch level of this window, the direction of gestures operating within this window, and the energy, speed, and rhythmic constituents of the improvisation are all controllable by the performer through simple gestures with the PowerGlove. Each of these parameters can be sent individually, or any given physical gesture can be parsed and given values for each parameter (which are then all sent together). This latter form of control is essentially a mapping between a gesture and a specific type of improvisation -- a type defined by the ranges of the parameters. Through these forms of control, this mode effectively provides the performer with a means of steering the evolving improvisation based upon high-level changes in general parameters. The virtual performer handles the low-level specifics (of note choices and inflections) to realize intentions signaled by the performer. However, the latency present in our glove apparatus reduces the intuitive satisfaction by reducing its potential as a continuous controller. It also requires that the control of the performer be at more of a "meta" level than would perhaps be desirable. Ideally the present level of control should be maintained while more direct musical control is made available. This would allow the performer to achieve specific rhythmic or melodic lines within the context of the evolving improvisation.

Comments

One of the interesting aspects of this project was the fact that the technology used was relatively low-scale, but the system produced some complex and intriguing musical output. We believe that this resulted from the inclusion of an interpretive agent with some imbedded "knowledge" about music in the interface. The richness of the interaction which developed (despite the limitations of the PowerGlove) was constrained mainly by the sophistication of the performance model and not by the precision or complexity of the controlling hardware. Too often designers of musical interfaces (and possibly VR interface technologies in general) place a premium on the design of the interface hardware while neglecting consideration of the system or processes which the hardware is intended to be interfaced *to*. Taking "us humans" as a model again -- our hands have only five fingers, but the world presents an infinite number of ways to use them.

References

Garton, B. 1992. "Virtual Performance Modelling." *Proceedings of the 1992 International Computer Music Conference.* International Computer Music Association, San Francisco.

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