POTENTIAL APPLICATIONS OF MIDI SOFTWARE IN IMPROVISED MUSICAL PERFORMANCE

Emile Toubenfeld
Dr. T’s Music Software
35 Beaconwood Road
Newton
MA 02161
USA

ABSTRACT: Software programs that can be used to control MIDI devices in improvised musical performance are divided into categories. Possibilities for the development of programs in each category are discussed. Potential applications of both new and existing programs to performance are suggested. New possibilities for the organization of improvising ensembles using these programs are briefly discussed.

Historically, most computer music has been composed music. This was a matter of necessity when the only available computers were mainframes which were unable to generate music in real time. We currently have available a wide variety of powerful, fully interactive computer systems, and a command protocol (MIDI), that allows the computational labour to be distributed among a variety of devices (MIDI controllers, synthesizers, effects processors), freeing up the computer itself for high-level command and control tasks. Such systems lend themselves very well to applications for improvisation, but it is our impression that such applications are receiving much less attention than applications for composition. Certainly, it is our experience in the commercial software world that most titles are intended for the creation of fully composed pieces, and that titles aimed at improvisers do less well, on the average, than those aimed at composers.

Let us divide music improvisation programs into categories, bearing in mind that the borders between categories are quite fuzzy, and that a single program may have aspects that fall into several categories. The categories that we shall consider are Sequence Playback Programs, Algorithmic Music Generators, Gestural Instruments, Input Processors, Simulated Players and Dedicated Programs. We limit our discussion to programs which control external devices through MIDI.

Sequence Playback Programs

The most straightforward and most commonly used program in musical improvisation is the Sequence Playback Program. This program plays back pre-sequenced music, which the live performer(s) use as an accompaniment to their improvisation. It can be the playback portion of a sequencing program designed for composing, or a program designed specifically to playback sequences with interaction from the performer.

If the sequences are played back without modification, this is no more interesting than live improvisation with pre-recorded tapes. It has the same basic limitation in that the sequences cannot respond to the live performer(s), and the live performers are thus constrained to follow the sequences. Such performances often feel more like an overdubbing session in a recording studio, than a concert performance.

Sequence Playback Programs are by no means limited to such simple applications. Commercially available programs give the performer the ability to modify the playback process by doing such things as muting and unmuting parts, choosing the sections to be played and the order in which to play them, transposing parts in pitch or velocity, and shifting parts against each other in time. Either the software or external control can be used to choose new instrument timbres, modify volume levels, apply pitch bends or other controllers to prerecorded sequences, and add effects processing. The performer who is thus controlling the playback process can be responding to the live portion of the performance, providing the interactive feedback loop that is an essential characteristic of ensemble improvisation.

The Sequence Playback Program could also be used to control effects devices, providing a time-varying environment for an improvising performer on an acoustic instrument.

Some programs allow composers to specify a degree of randomness in the playback process. This...
does not provide an interactive feedback loop, but it does create a unique situation for each performance.

There are many additional possibilities for such programs. If the live performers are using MIDI controllers, a computer could allow changes to the sequenced performance in response to the performer’s actions. Parts might be muted, unmuted, or modified depending on the register, dynamics or density of the live performance, attacks might be made to follow the tempo of the performance, pitches might be made to fit the instrument that the software perceives the performer to be using, etc. All of these processes could be specified in advance by the composer, or the performer might be able to modify the processes as part of the performance. Acoustic players can be included in the feedback loop, either by having them play instruments that are simultaneously physical sound generators and MIDI controllers, or by using existing or future devices to extract information about the acoustic performance and convert it to the MIDI domain.

Once we have allowed the presequenced parts to react to the live players, it is a simple step to allow them to react to each other as well. The Sequence Playback Program can thus become a complex dynamic system, as parts react to other parts, which are in turn reacting to other players or to the live players. If these reactions are under the control of a performer, the Sequence Playback Program alone could provide a fertile environment for musical improvisation, and start to take on the role of an Algorithmic Music Generator.

Algorithmic Music Generators

Let us define an Algorithmic Music Generator as a program which generates music data in a continuous process using the program’s algorithms and data that has been entered by the composer prior to the performance. For such programs to be improvisationally interesting, they must also allow the performer to modify the output by interacting with them using the computers input devices (keyboard, mouse, etc.), and perhaps MIDI. “M” and “Jam Factory” are the best known commercially available examples of such programs.

We make a distinction between these programs, which produce music by a process with which the performer interacts, and Gustural Instruments, which produce music directly in response to the performer’s gestures. The performer using an Algorithmic Music Generator acts more in the role of a conductor than a player, turning parts on and off, modifying the flow of performance, changing rhythms and timbres, etc.

There are many possible approaches to designing an Algorithmic Music Generator, too many to discuss here. Computers with fast, flexible screen displays can provide the performer with complete visual displays indicating the state of the system and allow him to manipulate the algorithms with the computer’s keyboard, a mouse, or a MIDI controller, and/or an array of MIDI switches, foot pedals and sliders. The performer can change the display during the piece, allowing him to manipulate different aspects of the music.

Gustural Instruments

We call programs which create music in direct response to the performer’s actions Gustural Instruments. Such instruments translate the performer’s actions using the computer’s input devices (perhaps augmented with MIDI controllers) into MIDI data for controlling instruments. They have much more in common with traditional instruments than the other programs we have discussed, as musical sounds are produced in direct response to the performer’s actions.

The potential power of such programs can be appreciated if we consider that the computer can provide the performer with an arbitrary, complex and, if desired, time varying screen display, complete with graphical or textual cues and can track complex gestures made with the mouse, keyboard or other input devices, and convert these gestures into MIDI data using almost any imaginable algorithm.

For example, a stringed instrument simulation could present a large array of lines to the performer, notes (or other sections) could be initiated by dragging the mouse across one of the lines, and sustained notes could be modified by further mouse movements. The computer can detect the line that was chosen, the position at which the line was crossed, and the velocity and angle of crossing, and use these to control the attack parameters of the note. The computer can track the further motions of the mouse and use this data to control the sustain portion of the note. More control can be achieved if the performer uses his other hand to control the computer keyboard and/or MIDI wheels or sliders, and uses his feet on MIDI foot pedaled.

The number of degrees of freedom available to the performer here seems to match the number available to players of even the most expressive acoustic instruments. The problems of designing software to effectively map such gestures onto MIDI data are challenging, and the computer and performer will have the perhaps equally daunting task of programming instruments to effectively respond to the data sent by the computer.

A very desirable input device for such programs would be a transparent, touch-sensitive surface that fits over the computer display. This would have some physical advantages of the mouse, as the player could position different fingers over different parts of the display.

ICMC GLASGOW 1990 PROCEEDINGS
402
I have been working for over two years with an unpublished program called the MIDI-Ax, which has characteristics of all three categories discussed so far. The MIDI-Ax generates note messages using a variety of algorithms in response to rouse clicking and dragging. Previously recorded sequences can be played as is, or have their pitch, velocity, loop points, and tempo altered by rouse or keyboard action. The program has a powerful recursive feature, which allows it to record, playback and modify the results of its own gestures. A MIDI keyboard, or an array of sliders and foot pedals, can be used to modify any aspect of the gestures performed.

The program is in an unfinished state, and has a very limited graphical interface. Even so, I have found it an enormously powerful and flexible tool for solo and ensemble improvisation.

Input Processors

We write programs which create music in response to MIDI input data Input Processors. Such programs have a great deal in common with Gestural Instruments, in that they produce action in direct response to gesture. They also have a great deal in common with the more advanced features of Sequence Playback Programs, as any process that can modify the playback of a sequence can also be used to modify the data produced in response to the performer’s actions.

The computer can produce an arbitrary complex mapping of input gesture onto MIDI output. Possibilities range from simple echoes and repetitions, through triggering pre-recorded sequences following the pitch and velocity of the notes played, to complex processes responding to aftertouch, wheel and pedal data, as well as note data.

The computer could also respond to acoustic input data that has been converted into the MIDI domain. Commercially available hardware can extract the pitch (with varying reliability), volume, and attack time of an acoustic sound and convert it to MIDI data for computer processing. It is easy to imagine hardware that could provide the computer information about the amount of energy in different parts of the audio spectrum; either through MIDI or some higher bandwidth protocol. Such data could be especially useful in controlling a Sequence Playback Program, or in directly controlling effects devices. The possibilities of feedback loops between the acoustic performer and MIDI equipment are tantalizing.

Simulated Players and Dedicated Programs

A simulated player is a program that is designed to improvise as an independent agent, responding to its environment, which presumably includes MIDI and/or audio information from the other players. The characteristics of such a program will depend on the style that the programmer wanted to simulate. A human performer could interact directly with this program by changing its parameters as part of the performance.

A Sequence Playback Program with some of the advanced features discussed above would act much like a Simulated Player. The distinction is that a Sequence Playback Program would be responding to ways that were directly programmed by the composer, while the Simulated Player would be acting more as an independent agent.

A Dedicated Program is a program designed to assist in the performance of a specific composition. For example, I have composed a piece for sequence called Increment. The score for Increments requires that each phrase or pattern in the piece be derived from the previous phrase by either adding or subtracting an element. It does not specify anything about the content of the elements or phrases. One could create a piece of software designed specifically to perform Increments. Such a program would allow the performer a variety of choices, while constraining the result to stay within the restrictions of the score.

New Ensemble Possibilities

Software instruments of this type allow performing ensembles to be organized in ways that are not possible with traditional instruments. In a traditional ensemble, each player is solely responsible for the sounds produced by his own instruments, and the number of sounds he can produce at one time is quite limited.

A single performer (solo or part of an ensemble) using software can control a large number of complex processes at once, and thus achieve a level of sonic complexity that was only available to a full ensemble. He may choose to control a number of ongoing processes with high-level commands, directly create musical gestures with physical actions on the computer, or do both at once.

An ensemble could have one or more such players, possibly with players of acoustic instruments. It could also consist of a number of players, each of whom had a defined role, but none of whom are solely responsible for any one sound.

For example, an ensemble might consist of a performer on a MIDI controller, another performer on an input Processor acting on the data stream from the first, or a Sequence Playback Program responding to the data, and a third performer using an Algorithmic Music Generator to generate additional sounds. Such ensembles could take on many different configurations, and could include Simulated Players in addition to (or instead of) live performers.

ICMC GLASGOW 1999 PROCEEDINGS

403