A System for Computer Assisted Gestural Improvisation

Emile Tobenfeld
Dr. T's Music Software, Inc.
35 Beaconwood Road, Newton MA 02161

ABSTRACT

A system used by the author for computer-assisted gestural improvisation is described. The system uses a Buchla Thunder Controller and two foot pedals as its primary control devices, along with an array of wheel and faders, and the computer's keyboard and mouse. A C-language program running on an Atari ST maps gestures made on the input devices into MIDI data that is sent to a sampler and several synthesizers and effects devices. The computer screen provides the performer visual feedback about the state of the system.

The system allows the performer to dynamically control the pitches and timbres of individual notes with multiple virtual instruments available at one time, to create effects such as glissandos and repeating phrases, and to interactively control the playback of presequenced material in a wide variety of ways. The system also allows the performer to record gestures during a performance, and to control the playback of these gestures later in the piece.

BACKGROUND

I began composing and improvising electronic music on analog equipment in 1976, I have no conventional musical training, and my primary interest has always been the improvisational control of musical process and timbre. In 1984, I began writing Dr. T’s Keyboard Controlled Sequencer (Tobenfeld, 1984). It is, like all sequencers, primarily a compositional tool, but it includes a number of features that allowed the user to interact with the sequenced material using the computer keyboard. I have also worked with other music improvisation programs such as "M" and Fingers.

All of these programs allow a performer to start, stop and interact with musical processes, but none provide the one-to-one mapping between physical and musical gestures that is characteristic of conventional instruments. In 1988 I began work on the Midi-Ax, primarily out of the desire for a software instrument that combines the physicality of one-to-one mapping with the flexibility of a computer controlled system. The original Midi-Ax (Martynece, 1992) allows the performer to control synthesizers with gestures on the computer mouse and keyboard, and MIDI faders, foot-pedals and wheels. In 1990 I acquired a Buchla Thunder Controller, and have modified Midi-Ax to use Thunder as an input device. My process of development is as improvisational as my music, and both the original Midi-Ax and the current version have many features that have never been explored by the handful of users.

DESIRABLE FEATURES IN A SOFTWARE-BASED INSTRUMENT

Below is a list of desirable features in a software-based instrument, based on the needs of a free improviser who often works unaccompanied and is not much concerned with traditional structures.

1. Precise control of the timbre of every note played
2. Visual feedback from the computer screen
3. Gestural control of previously composed musical processes.
4. Simultaneous control of multiple instruments or processes.
5. The ability to start a process, and relinquish control of it, allowing the process to continue while other processes are started.
6. The ability to regain control of an ongoing process.
7. The ability to record and playback the output of the program itself.

THE HARDWARE CONTROL SYSTEM

The hardware that I use to control the system consists of a Buchla Thunder Controller, a Yamaha MGe2 (a control device containing two faders, two wheels, two foot-pedals, and several buttons), additional MIDI faders, and the computer keyboard and mouse. The computer monitor provides visual feedback on the state of the system.

The Thunder Controller contains 13 longitudinal pads that are independently sensitive to touch position and touch pressure, and 12 smaller pads that are sensitive only to pressure. The 13 long pads are used for performance gestures and the others for program commands.

ARCHITECTURE OF THE PROGRAM

The two most important data structures in the software are called presets and assignments. A preset maps a gesture on one of the 13 longitudinal pads onto a MIDI gesture sent to the synthesizers. The output gesture can consist of a note, chord, glissando or sequence, and can be modified by any of the attached MIDI devices using the modulation system described below. An assignment maps a preset, a MIDI channel, and a preset modulation amount to each of the 13 pads. The arrow keys on the computer keyboard are used to reconfigure the system by changing assignments, and the computer screen shows the current preset, MIDI channel and other data for each of the 13 pads.

Another key element of the system is the MPE, Dr.T's multi-programming environment. This permits sequencing software (KCS Omega) to be co-resident with MIDI-Ax, and allows data sharing between them.

The 12 smaller pads on Thunder are used to send commands to the software. These commands include octave shifts, tempo changes, modifications of sequence playback and recording commands. Program change buttons on the MGe2 and some keys on the computer keyboard are also used for commands.

ARCHITECTURE OF THE PRESETS

The presets that determine the result of a gesture are designed around a matrix modulation system that is an extension of those used in instruments such as the Oberheim Xpander/Matrix series. Sources include pressure and finger position on the thunder pad, the preset modulation amount from the current assignment, and any of the external MIDI continuous controllers attached to the system. Modulation destinations include any MIDI continuous controller, and virtually any aspect of the preset.

Any modulation path can use the instantaneous value of the source, the value at the start of the gesture or the difference between these values. A typical preset that I use to play a single note derives the pitch and velocity of the note from the location and pressure when the pad is first
touched. Continuous pressure is converted into breath controller (MIDI CC #2) data, and the two foot pedal values are converted into mod wheel and foot controller data. Finger motion up and down the pad after the note is started creates a pitch bend. I typically use this preset with a synthesizer or sampler on which the foot pedal, breath controller and mod wheel are opening filters, bringing in or cross-fading waveforms, or moving the sample loop point. This technique provides a great deal of control of the details of the timbre of each note played.

If this preset is assigned to all 13 pads, and a different channel is set for each, the performer can play notes over several octaves on 13 independent instruments, and can control the timbre of each note as it is played. The controller mappings are made only during a gesture, so a controller motion only causes data to be sent on the channel(s) or which notes(s) are sustaining. Additional possibilities supported by note presets include chords, glassndos (in which finger motion during a gesture triggers new notes), and doubled notes on two different channels, with other MIDI controllers mapped to the volume and timbre of the second note. A preset can also determine pitch by using the pitch modulation value as an index to a lookup table, instead of using it directly. This allows a sense of tonality to be imposed on the pitches produced. The sequence used by the lookup table can be dynamically selected using a modulation source.

PLAYING SEQUENCES

Presets can also play sequences created in KCS Omega. All of the timbral controls described in the previous section can also be used when playing sequences. A number of parameters that affect sequence playback can be set in the preset and modulated by any modulation source. These include the choice of sequence to play, pitch transposition, velocity shift, adjustment of durations and tempo relative to the recorded value, and the amount of the sequence that will play before looping.

A typical preset that I use to play a sequence derives MIDI controllers (except pitch bend) from foot pedals and finger pressure as described above. The sequence played is determined by the preset modulation amount, or by the position of one of the MIDI faders. Pitch transposition and velocity shift are derived continuously from finger position and pressure. Pitch-bend is sent if the pitch wheel on the MCS2 is moved.

There are a number of commands that can modify a sequence being played by a gesture. These include commands to loop the sequence at its current location, to double or halve the tempo, to 'palindrome' the sequence (play it alternately forward and backward) and to continue play of the sequence after the pad is released.

ONGOING PROCESSES AND RECORDING

MIDI-Ax provides three ways to create ongoing processes; sequences that the performer can interact with or leave to play indefinitely. This allows a solo performer to create a multi-layered improvisation without any completely predetermined parts. A sequence started by a gesture can continue after the gesture ends. Completely recorded sequences can be started with the computer keyboard. The program can record and playback its own output.

MIDI-Ax has several recording commands, including a command to start a recording of indefinite length, and a command to start a recording of the same length as the previous recording. A recording will only include the output of gestures made after the recording begins. It is thus possible to build up dense layers of independent material during an improvisation.
Each such sequence is shown on the computer screen, and can be manipulated using the computer keyboard and mouse or MIDI controllers. These sequences can be muted, unmuted, or canceled, have their tempo changed or be moved in pitch and velocity space.

CONCLUSIONS

I have found the system described in this paper to be a powerful tool for both structured and unstructured improvisations using a wide variety of control paradigms. It provides a degree of control of the nuances of musical gesture that is comparable to that provided by acoustic instruments, and provides control of complex and overlapping gestures that is not found on acoustic instruments. It is impossible to give a full description of the rich set of possibilities that the system provides in the limited space available.

A composer who wishes to use this kind of system to full advantage faces a number of tasks. To effectively control the timbres of individual notes, each sound must be carefully programmed to respond suitably to MIDI controllers. The software data structures must be set up to provide the configurations required for the piece, and organized so that the performer can make transitions between configurations. The performer must become accustomed to a virtual system, in which the response to a physical gesture may be completely context specific. In an improvisational situation, the performer may need to divide his attention between making musical gestures and changing the configuration to meet the needs of the changing context.

I believe that these problems are as inherent in gestural improvisation with virtual (i.e., software based) instruments as the problems of physical control in performance with acoustic instruments. I also believe that there is a great deal of valuable music to be made improvising with such instruments.

REFERENCES