David Wessel and Matthew Wright
Center for New Music and Audio Technologies
Department of Music
University of California, Berkeley
Berkeley, California 94720 USA
www.cnmat.berkeley.edu
[wessel,matt]@cnmat.berkeley.edu

Problems and Prospects for Intimate Musical Control of Computers

When asked what musical instrument they play, few computer musicians respond spontaneously with ''I play the computer.'' Why not? In this report, we examine the problems associated with the notion of the computer as musical instrument and the prospects for their solution.

We begin with a discussion of our goals and requirements for computer-based musical instruments, raising the main issues that inform our work. Then we discuss a variety of metaphors for musical control that we have found to be powerful. Finally, we discuss some of the technology we have developed for implementing these instruments.

Goals and Requirements

Musical instruments and their gestural interfaces succeed for a variety of reasons, most of which are social in character. These sociological aspects, such as the development of a repertoire for the instrument, are beyond the scope of this article. Here we will concentrate on factors such as ease of use, potential for development of skill, reactive behavior, and coherence of the cognitive model for control.

Relationships of Gestures to Acoustic Results

The image immediately called to most people's minds by the statement ''I play the computer'' is of a performer physically interacting with the QWERTY keyboard and pointing device of a typical computer workstation live on stage. Many musicians do indeed perform live music with this interface, but we prefer to interact with more specialized gestural interfaces, because they offer lower latency, higher precision, higher data rates, and a broader range of physical gestures than the keyboard/mouse-type interface. There is also the issue of the visual appearance of a performance and

the association of standard computer interfaces with office work (Zicarelli 1991).

At the onset, it would be useful to consider some of the special features that computer technology brings to musical instrumentation. Most traditional acoustic instruments such as strings, woodwinds, brass, and percussion place the performer in direct contract with the physical sound production mechanism. Strings are plucked or bowed, tubes are blown, and surfaces are struck. The performer's gesture plays a direct role in exciting the acoustic mechanism. With the piano and organ, the connection between gesture and sound is mediated by a mechanical linkage (and in some modern organs by an electrical connection). But the relationship between gesture and acoustic event remains in what one might call a ''one-gesture-to-one-acoustic-event'' paradigm.

One obvious feature of computer technology is immense timbral freedom. Although skilled players of acoustic instruments can produce a wide variety of timbres, they are nevertheless constrained by the sound production mechanism. In contrast, computers can produce arbitrary sampled or synthesized sounds. A sample-playback synthesizer can reproduce any recorded sound in response to any gesture. The extreme case can be found in tape music: with the single gesture of pressing the ''play'' button, the acoustic result is an entire composition. A more interactive middle ground is the kind of piece where a performer steps through a sequence of prerecorded computer sound cues—for example, with a foot pedal. An even more interactive example would be the typical sampling keyboard, where each key depression triggers a potentially different sound. In each of these examples, the acoustic result of a gesture may be arbitrarily complex, perhaps consisting of multiple perceived sonic events, but the control model is still a one-to-one mapping from gestures to acoustic results.

When sensors are used to capture gestures and a computing element is used to generate the sound, an enormous range of possibilities becomes avail-

Figure 1. A conceptual framework for our controller research and development.
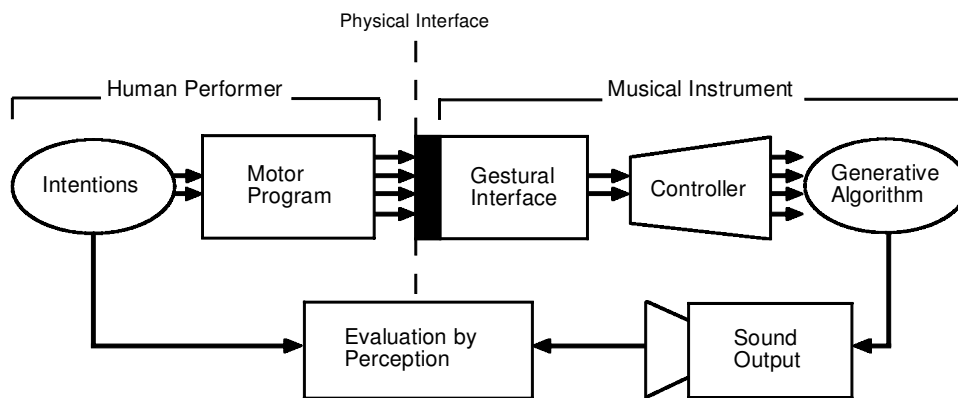
able. Sadly but understandably, the electronic music instrument industry, with its insistence on standard keyboard controllers, maintains the traditional paradigm.

Figure 1 provides a conceptual framework for our controller research and development. Our human performer intends to produce a certain musical result. These intentions are communicated to the body's sensorimotor system (''motor program''). Parameters are sensed from the body at the gestural interface. These parameters are then passed to controller software that conditions, tracks, and maps them to the algorithms that generate the musical material. Admittedly, this diagram is schematic and incomplete. One aspect that is not well captured by this diagram is the way in which performers' intentions are elaborated upon by discovery of new possibilities afforded by the instrument. Experimental and otherwise exploratory intentions are certainly dear to us. We find that this albeit schematic framework allows us to view the roles of human motor learning, controller mapping, and generative software as an overall adaptive system (Lee and Wessel 1992).

Unlike the ''one-gesture-to-one-acoustic-event'' paradigm, our framework allows generative algorithms that can produce complex musical structures consisting of many events. In this model, the performer's gestures can guide and control high-level parameters of these generative algorithms rather than directly triggering each event. One of our central metaphors for musical control is that of driving or flying about in a space of musical pro-

cesses. Gestures move through time, as do the musical processes.

Low ''Entry Fee'' with No Ceiling on Virtuosity

We believe that ''getting started'' with a computer-based instrument should be relatively easy, but this early stage ease-of-use should not stand in the way of the continued development of musical expressivity. Most traditional acoustical musical instruments are not easy to play at first but afford the development of a high degree of musicality. On the other hand, many simple-to-use computer interfaces proposed for musical control seem—after even a brief period of use—to have a toy-like character. By this we mean that one quickly ''outgrows'' the interface by discovering the limits of how it can be used. Many such simple-to-use interfaces do not invite continued musical evolution.

Is a low ''entry fee'' with no ceiling on virtuosity an impossible dream? We think not and argue that a high degree of control intimacy can be attained with compelling control metaphors, reactive low latency variance systems, and proper treatment of gestures that are continuous functions of time. With the potential for control intimacy assured by the instrument, musicians will be much more inclined to develop their performance skills and personal styles.

Latency Requirements for Control Intimacy

Latency is the elapsed time between a stimulus and the response. Some kinds of latency are rela-

tively easy to measure—for example, the time between a computer's receipt of a control message and the beginning of the audio event generated in response (Freed et al. 1997). The conceptual model in Figure 1 includes elements whose latency is very difficult—if not impossible—to measure, but presumably all latencies within the human performer have already been compensated for by the timing of the motor program.

Few practitioners of live computer music would deny that low latency is essential, but just how low is the subject of considerable debate. We place the acceptable upper bound on the computer's audible reaction to gesture at 10 msec, and the systems described in this article provide measured latencies closer to 7 msec.

Low variation of latency is also critical, and we argue that the range of variation should not exceed 1 msec. Grace note–generated events such as flams can be controlled by percussionists with temporal precision of less than 1 msec. This is accomplished by controlling the relative distance of the sticks from the head during the strokes. Timbral changes in the flams begin to become audible when the variations in the time between the grace note and the primary note exceed 1 msec. Psychoacoustic experiments on temporal auditory acuity provide striking evidence for this criterion (Ronken 1970; Henning and Gaskell 1981). As we will argue later, prospects for the solution to the latency variation problem can be found by using time tags or by treating gestures as continuous signals tightly synchronized with the audio input/output (I/O) stream.

Representing Control Gestures as Events and Signals

All music, in the end, exists as sound—that is, continuous variations in air pressure. However, the notion of discrete events is a very powerful and effective metaphor for musical control, providing much-simplified reasoning about rhythms, entrances and exits, notes, and many other aspects of music. The long-term success of extremely note-centric music languages such as the Music-N family (Roads 1996) attests to the usefulness of this model, and we believe that the representation of

some control gestures as discrete events is an important part of the toolbox for building computer music instruments. OpenSound Control (described later) and MIDI are network protocols that can represent musical events and communicate them to our software.

Nevertheless, many musical gestures are continuous functions of time and should be treated as such—for example, the position along the string of a finger on a violinist's left hand. We note that almost all traditional acoustic instruments afford (and require) continuous control, and, for the most part, it is this continuous control that makes them expressive. Many sensing technologies exist today that can transduce a variety of continuous physical gestures and digitize them at high data rates; we will describe some of our favorites later.

How do we get continuous gestural information into our software? Although it is common to sample continuous gestures asynchronously (at relatively low rates) and treat each change of the value as an event (for example, MIDI continuous controllers), we are not satisfied with this approach. In addition to the loss of the higher-frequency components of our control signals from the long times between ''control change'' events, latency variation destroys the fine shapes of control gestures and also introduces an unpredictable timing error between the control gesture input to our software and any audio signals that may also be input to our software.

Our solution is to represent continuous control gestures as audio signals and get them into our software with the same high-bandwidth isochronous data paths that we use for audio input. Because most control gestures do not have frequency components in the upper ranges of the audible spectrum, we can multiplex lower-rate control gestures into a single audio channel. The new generation of software synthesis programming environments such as Max/MSP (Zicarelli 1998, 2002), SuperCollider (McCartney 2000), Pd (Puckette 1996), and Open Sound World (Chaudhary, Freed, and Wright 1999; see cnmat.CNMAT .Berkeley.EDU/OSW) provide multi-rate signal processing. In these programming environments, it is quite natural to treat gestures with a sample-

synchronous signal processing approach. An additional advantage is the ability to apply existing audio DSP modules (for example, filters, Fast Fourier Transforms, and linear predictors) to process, condition, and otherwise manipulate our control-gesture signals.

CNMAT's connectivity processor (Avizienis et al. 2000) described below provides a mechanism for getting continuous gestures into the computer in a manner that is very tightly synchronized with the audio sample stream. With this system, we demonstrate a significant increase in control intimacy.

Musical Features Desired for All Instruments

The traditional ''one-gesture-to-one-acoustic-result'' paradigm automatically provides many important musical features that must sometimes be implemented explicitly. One of the most critical features of any musical control system is a ''silencer,'' a mechanism that allows the performer to gracefully shut down a musical process. More generally, the performer should always have control of the overall volume and density of the sound an instrument is producing, even when a semi-autonomous generative algorithm is handling the details.

We believe that there should be some sort of correspondence between the ''size'' of a control gesture and the acoustic result. Although any gesture can be mapped to any sound, instruments are most satisfying both to the performer and the audience when subtle control gestures result in subtle changes to the computer's sound and larger, more forceful gestures result in dramatic changes to the computer's sound.

Another feature we find very important is predictability. Although we want our generative algorithms to fill in musical details not directly specified by the control gestures, we always want to feel that we have complete control, at least at a high level of abstraction, over the sounds our instruments produce. We find the analogy of the conductor useful: individual instrumentalists in the orchestra determine fine details, while the conductor exercises overall control.

The possibility of a performer's controlling a generative algorithm raises issues about memory and state. If the output of the algorithm depends in a complicated way on the history of the control gestures that have guided the algorithm, then the ''predictability'' goal requires the performer to know the algorithm's state at any given time. It is too easy to build interesting interfaces that place a large burden on the performer's memory. One alternative is for the computer to provide some sort of display, typically visual, of its current state. Ideally, the performer does not need to look at a computer screen during the entire performance and will be able to look at fellow performers and at the audience. One of our design goals has been to create stateless or mostly stateless interfaces; the challenge in this case is to be able to produce complex results without accumulating information from multiple gestures.

Metaphors for Musical Control

As suggested in the introduction, metaphors for control are central to our research agenda. We have found the work of George Lakoff and his collaborators (Lakoff and Johnson 1999; Lakoff and Núñez 2000) on embodied cognition to be particularly applicable. They argue that abstract concepts like time and space and even the loftier concepts of mathematics are grounded in sensorimotor experience. We now present some of the metaphors that have inspired the development of our controller software.

Spaces of Musical Sounds and Processes

Spatial metaphors can play an important role in characterizing the materials of music. The basic concept is that sounds that are similar to each other are placed proximate to each other in the spatial representation.

A rich variety of spatial representations that go beyond the one-dimensional layout of a traditional keyboard have been proposed for pitch. Fred Ler-

dahl, in his book *Tonal Pitch Space* (2001), provides a review of the work in music theory and music perception and cognition on the spatial representation of pitch. W. A. Mathieu (1997) in his book entitled *Harmonic Experience*, uses a variety of pitch lattices to pedagogical advantage. Two-dimensional pitch space representations have been used as control structures in a variety of musical instruments, most notably the button keyboards of Bandoneons and accordions. Another example is Harry Partch's Tonality Diamond (Partch 1949) used to lay out the bars of his Diamond Marimba. Keislar (1987) provides a review of keyboard layouts for microtonal pitch systems and points out the importance of the way transpositional invariance of pitch material corresponds to translational invariance of the geometry of gesture. For example, on the two-dimensional keyboard of Bosanquet's enharmonic harmonium (Bosanquet 1875), chords and melodic patterns are fingered identically in all transpositions.

Timbre space (Wessel 1979) has been proposed as a multidimensional musical control structure. Following the basic metaphor that similar timbres are represented as proximate in the multidimensional control structure, timbre space concepts have found their way into a variety of control schemes. One novel implementation of timbre space as a musical control structure involves attaching a pointing device to the tip of a wind controller. The wind controller then becomes a pointing device itself and can be used in a very natural manner to navigate timbre space. We have implemented this idea by attaching a Buchla Lightning infrared-based pointing transmitter to the tip of a Yamaha WX-7 wind-controller.

The metaphor of geometric translational invariance can be generalized to control attributes other than pitch. A number of experiments (McAdams and Cunibile 1992; Wessel 1979) suggest that timbral sequence transpositions can be heard and predicted by parallel paths in timbre space. We suggest that translational invariance in the spatial representation of musical material is a compelling metaphor for the design of interfaces.

The spatial metaphor that maps perceptual simi-larity to spatial proximity in the control interface can be generalized to more complex musical processes and structures, such as those involved in the generation of melody, rhythm, and complex textures. One notable example is the musical texture space provided by David Huron (Huron 1989, 2001). Georg Hajdu (1994) has demonstrated the use of Huron's texture space as a control structure.

It should be noted that meaningful spatial control configurations can be produced without the multidimensional scaling of tediously collected similarity judgments for all pairs of sounds in the set under consideration. We have found that placing the sounds intuitively in a two-dimensional space works well, and we have developed a convenient interface to assist in this task. With this direct placement method, we have laid out useful timbre spaces consisting of more than 120 sounds in a few hours—a real gain in efficiency over making more than 7,000 pair-wise judgments of similarity. Experimental verification of the validity of the intuitive spatial layout approach to the collection of similarity judgments has been provided by Goldstone (1994).

The direct placement method for laying out a controller configuration can be carried out with a variety of criteria. In the loosest sense, the layout need not have a priori dimensions associated with it. The material can be laid out by similarity alone with the possibility that an interpretable dimensional structure will emerge in an ad hoc manner. At another extreme, one could arrange things with an a priori dimensional structure in mind. Yet another possibility would involve layout constraints based on the metaphor of translational invariance. Whatever the constraints, one should not loose sight of the goals of such interfaces: they should facilitate navigation in performance and suggest musically interesting trajectories for gesture.

"Drag and Drop"

The "drag and drop" metaphor is well known to users of the Apple Macintosh computer. An object is selected, picked up, and dropped upon a process.

This is a natural application of Lakoff's movement and container metaphors. Our use of "drag and drop" is based on gestural interfaces with two-dimensional surfaces and accurate position sensing. We represent both musical material and musical processes as objects occupying area of the surface. For example, the musical material might consist of a collection of samples, and the musical processes might be different ways of playing the samples with or without looping, pitch shifting, etc. The performer selects a musical object, drags it to the appropriate part of the surface, and drops it onto the desired musical process. This metaphor can be extended with musical objects representing other parameters of the musical processes, for example, filter shapes.

Scrubbing and Its Variants

Sinusoidal, granular, and other models allow arbitrary time-scale manipulation without changes in pitch or spectral shape. "Scrubbing" interfaces require controllers with very accurate detection of one-dimensional position; this position parameter maps to the time index of the sound model. Moving gradually from left to right at the appropriate rate results in a resynthesis with the original temporal behavior, but any other gesture results in an alteration of the original. This interface allows a performer to play more arbitrary musical material while preserving the fine continuous structure of the original input sounds. This kind of interface has been used in live performance contexts with classical Indo-Pakistani singing (Wessel, Wright, and Khan 1998), trombone samples with expressive glissandi, and saxophone quartet material. Other dimensions of sensed data can be mapped to synthesis parameters such as loudness and spectral shape.

"Dipping"

In the "dipping" metaphor, the computer constantly generates musical material via a musical process, but this material is silent by default. The performer controls the volume of each process—for example, by using a poly-point touch-sensitive interface with the pressure in each region mapped to the volume of a corresponding process. Other gestural parameters control other parameters of the musical processes.

An advantage of this metaphor is that each musical event can be precisely timed, regardless of the latency or jitter of the gestural interface. Once a given process is made audible, its rhythm is not dependent on the performer in an event-by-event way. This kind of interface is most satisfying when there are multiple simultaneous musical processes of different timbres, allowing the performer to orchestrate the result in real time by selecting which processes will be heard.

"Catch and Throw"

The "catch and throw" metaphor involves the notion of selectively trapping musical phrases from an ongoing performance, transforming them in some way, and then sending these transformed materials back into the performance. Transformation of musical material is critical for this kind of performance; literal echoing grows old very quickly.

An early application of this concept was used in a work for Roscoe Mitchell and David Wessel entitled *Contacts Turbulants* (Wessel et al. 1987), which was commissioned by IRCAM in 1986. Here, a standard MIDI keyboard controller was the interface to a MIDI recording and playback system. The lower octave of the keyboard became a 12-channel MIDI sequence recorder, and the remaining six octaves were used for playback. The standard pitch meanings of the keyboard were completely abandoned. If a phrase produced by the MIDI pitch tracker on Roscoe Mitchell's saxophone were recorded into the key normally associated with, for example, the note C, then the C's in all octaves of playback keys referred to that material. A different kind of transformation of the recorded data was associated with each playback octave.

The essential idea of this performance situation is that the performer at the keyboard captures material from the companion performer and then

transforms and re-injects this material into the performance. The recording and playback processes run in parallel. In fact, the keyboard is fully polyphonic, and if one had enough fingers, all 12 recording processes could run while all of the remaining keys were playing the transformed material. The keyboard performer has complete control over the initiation and termination of a recording process, but there will certainly be some delay between the beginning of the phrase to be captured and the decision that it should be captured. In these improvisatory contexts, the keyboard performer is unable to decide if a phrase is interesting to capture until after the phrase has begun. For this reason, some computer-implemented forms of an echoic memory mechanism (Snyder 2001) are required so that the performer can reach back in time to the beginning of the desired phrase. Automatic phrase segmentation can be used to start and stop the recording process at musically appropriate times.

One of the challenges of the "catch and throw" situation is the memory load on the performer, who is required to remember the caught phrases and where they are placed. A visual or tactile cueing system would help solve this problem. Another solution would be an adaptation of the headphone-on-one-ear technique used by disk jockeys who listen to material as they cue it for introduction into the ongoing mix.

"Catch and throw" is a rich metaphor and lies at the heart of musical dialog. In its most general sense, "catch and throw" is really "listen and respond."

Technology

In this section, we describe some of the underlying technology we have developed for building computer music instruments.

Open Sound Control

Open Sound Control (OSC; Wright and Freed 1997) is a discrete event protocol for communication among controllers, computers, sound synthesizers, and other multimedia devices that is optimized for modern networking technology. OSC is an open protocol, and OSC messages can be carried by any networking protocol, including UDP and TCP/IP. Entities within a system are named individually by an open-ended, URL-style, symbolic, hierarchical address space. OSC includes a pattern-matching language to specify multiple recipients of a single message. There is also a mechanism for dynamically querying an OSC system to find out its capabilities and features. OSC has been integrated into Max/MSP, OSW, SuperCollider, Pd, Jmax (Dechelle et al. 1999), Csound (www.csounds.com), Native Instruments' Reaktor (www.nativeinstruments.de), Siren (Pope 2002), RTCmix (Garton and Topper 1997), Mike Berry's Grainwave (www.7cities.net/users/mikeb/GRAINW.HTM), and SodaConstructor (www.soda.co.uk/explore/osc.htm). (See www.cnmat.berkeley.edu/OSC for more details and downloadable OSC software.)

OSC for Distributed Processing on Local Area Networks

OSC was originally developed to enable distributed processing on local area networks. In particular, we designated the role of a "synthesis server," a computer whose entire processing power would be devoted to sound synthesis. This frees the machine physically connected to the gestural interfaces to devote all of its resources to controller I/O, gestural data conditioning and mapping, and implementation of control structures based on the metaphors described previously. Many precedents exist for a "black box" synthesizer producing sound in response to control messages; it is conceptually easy to provide an interface and a layer of abstraction between deciding what should be sounded and the implementation of that sound.

To achieve our overall latency and jitter goals with this kind of system architecture, OSC provides high-resolution time tags and a mechanism for specifying groups of messages whose effects should occur simultaneously. Time tags allow the receiving device (in this case, the synthesis server)

to follow a scheduling discipline (Dannenberg 1989) that reduces jitter by trading it for latency.

Benefits of OSC Addressing

OSC's symbolic names and open-ended hierarchical name space grew out of our frustration with addressing limitations inherent in protocols such as MIDI and ZIPI (McMillen, Wessel, and Wright 1994) that rely on short, fixed-length bit fields. As authors of synthesis servers, we wanted to name our control parameters as we designed them rather than pick from control parameter names pre-selected by the authors of a networking protocol. Likewise, we wanted to organize the control interface according to project-specific structures without being bound by fixed architectures such as ''notes'' within ''channels.''

The features that make OSC attractive for representing synthesizer control messages also make OSC good for representing the sensed gestures from physical controllers (Wright et al. 2001). OSC's symbolic and numeric argument types are sufficient to represent the parameters of all the controllers we have used. Symbolic parameter names make programs easier to read and understand. The pervasive use of OSC provides a uniform syntax and conceptual framework at all stages of the gesture-to-sound mapping problem, from controller data input to sound control output. An OSC interface to a gestural-sensing or signal-processing subprogram is a powerful form of abstraction that can expose all of the important features while hiding the implementation.

A Programmable Connectivity Processor

The conventional approach for communicating gesture and sound to real-time performance systems is to combine a microcontroller or DSP chip with A/D and D/A converters and a network interface, such as a MIDI serial controller. We have developed an alternative, more flexible approach that supports scalable implementations from a few channels of audio and gestures to hundreds of channels.

Our new system to address computer music and audio connectivity problems is based on integrating all digital functions on a single field-programmable gate array (FPGA). All functions are determined by compiling high-level hardware descriptions (in VHDL) into FPGA configurations. This approach allows the considerable investment in developing the interface logic to each peripheral to be easily reused with a wide variety of FPGAs from different vendors and of different sizes. Since FPGAs are now available in sizes greater than one million gates, entire DSP chips and microcontrollers can also be integrated if required.

We have developed and tested VHDL descriptions for processing serial audio data for the S/PDIF, AES/EBU, AES-3, and ADAT lightpipe industry standards. Other VHDL modules support the Synchronous Serial Interface (SSI) protocol used for communicating with codecs; this allows us to provide multichannel analog I/O for both audio and continuous gestures. We also provide modules for multiple MIDI input and output streams. Although such descriptions have been developed for proprietary systems, this library of modules represents the first complete, independent suite available in VHDL.

This suite makes possible some unusual cross-codings such as embedding gestural data in audio streams, thereby increasing temporal precision by exploiting isochronous data paths in the control processor. We sample continuous gestural signals at a submultiple of the audio sampling rate, multiplex the channels, and represent them as audio input signals. This allows us to read gestural signals into our software with the same low latencies as audio input, and it guarantees that gestural and audio input signals will be tightly synchronized.

A novel module of particular importance in portable computer-music performance systems implements Fast Ethernet all the way up through IP to the UDP protocol of TCP/IP. Owing to the importance of Internet performance, Fast Ethernet implementations are extremely reliable and finely tuned on all modern operating systems.

A key feature of the connectivity processor is the analog subsystem for continuous gesture acquisition. We currently provide 32 channels of analog-

to-digital conversion. Voltage ranges of the converters are selectable, as are the sampling rates and the appropriate anti-aliasing filter cutoff frequencies.

Our system combines VHDL connectivity modules that multiplex 8 channels of bidirectional audio, MIDI, S/PDIF and transduced gestures into UDP packets which are exchanged with a portable computer using new, customized ASIO drivers in Max/MSP. (See www.cnmat.berkeley.edu /connectivity for a more detailed description of CNMAT's connectivity processor.)

Digitizing Tablets

Throughout history, people have adapted whatever objects were in their environment into musical instruments. The computer industry has invested significant resources in creating broadly available, low-cost gestural controllers without any particular musical application in mind; thoughtful adaptation of these controllers for music is a fruitful yet overlooked route.

We find the latest incarnations of the venerable digitizing tablet (also known as the "artist's tablet") very interesting for musical control. Tablets offer accurate and fast absolute position sensing of cordless devices in three dimensions. Additionally, pressure, orientation, tilt, and rotation estimates are available. The tablet we use allows simultaneous sensing of two devices, usually one in each hand. This rich, multidimensional control information can be mapped to musical parameters in a variety of interesting ways.

The most direct kind of mapping associates a single synthesis parameter with each control dimension—for example, vertical position controlling loudness, horizontal position controlling pitch, etc. The high resolution of the tablet's position sensing makes this kind of mapping capable of fine subtlety, but the results of instruments with this kind of mapping tend to be very homogenous; the challenge for the performer is to produce drastically different sonic results.

More interesting interfaces define regions of the tablet associated with particular behaviors. For example, one region might consist of a grid providing access to a large palette of musical material, while other regions represent musical processes that can operate on selected musical material. Repeating rhythmic cycles can be represented graphically on a region of the tablet, and sonic events can be placed at particular time points within the cycle (Wright and Wessel 1998).

One way to silence a process is for the performer to place the pen on the process, and, using a circular motion like the traditional copy editor's cursive "delete" sign, the performer can silence the process at a rhythmically appropriate point in time. Other interfaces use the "eraser" end of the pen to silence processes.

We have created software in the Max/MSP environment that we use to develop control structures for the two-handed digitizing tablet. Examples include navigation in timbre space, multidimensional synthesis control, note stream synthesis, and emulations of the gestures of strumming, plucking, and bowing strings.
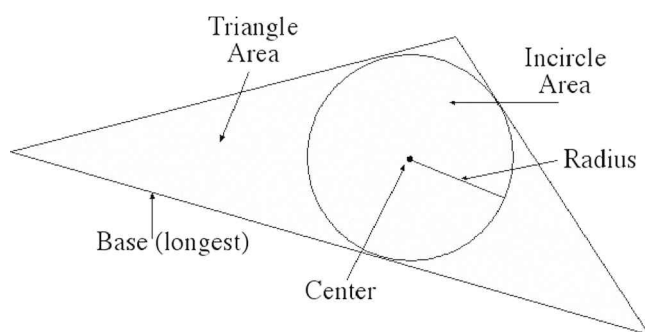
"Catch and throw" interfaces have also been implemented for digitizing tablets. Regions on the tablet represent buffers that store the "caught" musical material and can be thought of as virtual containers. Touching the pen to the tablet in one of these regions begins recording. We map the horizontal position of the contact point within the region to the amount of time to go back in the echoic memory: touching the pen to the rightmost part of the region begins recording from the current time, while positions further left in the region begin the recording from increasingly further back in time.

Joysticks

USB joysticks used for computer games also have good properties as musical controllers. One model senses two dimensions of tilt, rotation of the joystick, and a large array of buttons and switches. The buttons support "chording," meaning that multiple buttons can be pressed at once and detected individually.

Ali Momeni at CNMAT developed a modal interface in which each button corresponds to a particular musical behavior (Wright et al. 2001). With

Figure 2. *A visual interpretation of Tactex's MTCExpress controller, for three fingers.*

Table 1. OSC Messages Output from Tactex Triangle Detection

/area ⟨*area_of_inscribed_triangle*⟩
/averageX ⟨*avg_X_value*⟩
/averageY ⟨*avg_Y_value*⟩
/incircle/radius ⟨*inscribed circle radius*⟩
/incircle/area ⟨*inscribed circle area*⟩
/sideLengths ⟨*side1*⟩ ⟨*side2*⟩ ⟨*side3*⟩
/baseLength ⟨*length_of_longest_side*⟩
/orientation ⟨*slope_of_longest_side*⟩
/pressure/average ⟨*avg_pressure_value*⟩
/pressure/max ⟨*maximum_pressure_value*⟩
/pressure/min ⟨*minimum_pressure_value*⟩
/pressure/tilt ⟨*leftmost_Z-rightmost_Z*⟩

no buttons pressed, no sound results. When one or more buttons are pressed, the joystick's tilt and rotation continuously affect the behaviors associated with those buttons.

Some Custom Controllers

At CNMAT, we have developed applications for variety of custom controllers. Don Buchla's Thunder is a polyphonic, continuous, multidimensional interface that provides a two-dimensional surface containing 12 continuously pressure-and position-sensitive strips that are all sensed simultaneously. This polyphonic continuous control is excellent for interfaces based on the "dipping" metaphor.

Tactex's MTC Express controller (see www.tactex.com) senses pressure at multiple points on a surface. The primary challenge of using the device is to reduce the high dimensionality of the raw sensor output (over one hundred pressure values) to a small number of parameters that can be reliably controlled. We interpret the output of the tactile array as an "image" and use computer vision techniques to estimate overall pressure and 2D position for each finger of the hand. The anatomy of the human hand makes it impossible to control these three variables independently for each of five fingers. We have developed another level of analysis based on interpreting the parameters of three fingers as a triangle, as shown in Figure 2. This results in the parameters shown in Table 1. These parameters are particularly easy to control and have the advantage of working with

any orientation of the hand—an application of the metaphor of translational invariance.

Several of our current research projects exploit a key feature of the previously described connectivity processor, namely the synchronization of control signals with the audio I/O stream. These projects include an organ keyboard with continuous sensing of each key's position (Freed and Avizienis 2000), a variety of micro-accelerometer projects, and new force-sensing resistor-type (FSR) devices. In addition, we have developed new sensor systems for multidimensional string motion (Freed and Isvan 2000) and have made some advances in extracting control signals from vocal sounds.

Conclusions

We have described our efforts toward the development of live-performance computer-based musical instrumentation. Our design criteria include initial ease of use coupled with a long-term potential for virtuosity, minimal and low variance latency, support for both event-based and signal-based control, and clear and simple strategies for programming the relationship between gesture and musical result. We believe that compelling metaphors are the key to rich and expressive gesture-to-sound mappings and presented some of our favorites. We also presented some of the underlying technology we have developed in the course of this work, includ-

ing custom controllers and unique adaptations of standard gestural interfaces, a programmable connectivity processor, and the Open Sound Control communications protocol.

Acknowledgments

References

Avizienis, R., et al. 2000. ''Scalable Connectivity Processor for Computer Music Performance Systems.'' *Proceedings of the 2000 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 523–526.

Bosanquet, R. H. M. 1875. ''Temperament; or, the Division of the Octave.'' *Proceedings of the Musical Association* 1(4–17):112–158.

Chaudhary, A., A. Freed, and M. Wright. 1999. ''An Open Architecture for Real-Time Audio Processing Software.'' *1999 Audio Engineering Society 107th Convention*, preprint #5031. New York: Audio Engineering Society.

Dannenberg, R. 1989. ''Real-Time Scheduling and Computer Accompaniment.'' In M. V. Mathews and J. R. Pierce, eds. *Current Directions in Computer Music Research*. Cambridge, Massachusetts: MIT Press, pp. 225–261.

Dechelle, F., et al. 1999. ''jMax: An Environment for Real-Time Musical Applications.'' *Computer Music Journal* 23(3):50–58.

Freed, A., and R. Avizienis. 2000. ''A New Music Keyboard Featuring Continuous Key-Position Sensing and High-Speed Communication Options.'' *Proceedings of the 2000 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 515–516.

Freed, A., A. Chaudhary, and B. Davila. 1997. ''Operating Systems Latency Measurement and Analysis for Sound Synthesis and Processing Applications.'' *Proceedings of the 1997 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 479–481

Freed, A., and O. Isvan. 2000. ''Musical Applications of New, Multi-axis Guitar String Sensors.'' *Proceedings of the 2000 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 543–546.

Garton, B., and D. Topper. 1997. ''RTcmix: Using CMIX in Real Time.'' *Proceedings of the 1997 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 399–402.

Goldstone, R. L. 1994. ''An Efficient Method for Obtaining Similarity Data.'' *Behavior Research Methods, Instruments, & Computers* 26:381–386.

Hajdu, G. 1994. Personal communication.

Henning, G. B., and H. Gaskell. 1981. ''Monaural Phase Sensitivity Measured with Ronken's Paradigm.'' *Journal of the Acoustical Society of America* 70(6):1669–1673.

Huron, D. 1989. ''Characterizing Musical Textures.'' *Proceedings of the 1989 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 131–133.

Huron, D. 2001. ''Tone and Voice: A Derivation of the Rules of Voice Leading from Perceptual Principles.'' *Music Perception* 19(1):1–64.

Keislar, D. 1987. ''History and Principles of Microtonal Keyboards.'' *Computer Music Journal* 11(1):18–28.

Lakoff, G., and M. Johnson. 1999. *Philosophy in the Flesh: The Embodied Mind and Its Challenge to Western Thought*. New York: Basic Books.

Lakoff, G., and R. E. Núñez. 2000. *Where Mathematics Comes From: How the Embodied Mind Brings Mathematics into Being*. New York: Basic Books.

Lee, M. A., and D. Wessel. 1992. ''Connectionist Models for Real-Time Control of Synthesis and Compositional Algorithms.'' *Proceedings of the 1992 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 277–280.

Lerdahl, F. 2001. *Tonal Pitch Space*. New York: Oxford University Press.

Mathieu, W. A. 1997. *Harmonic Experience: Tonal Harmony from Its Natural Origins to Its Modern Expression*. Rochester, Vermont: Inner Traditions International.

McAdams, S., and J. C. Cunibile. 1992. ''Perception of Timbral Analogies.'' *Philosophical Transactions of the Royal Society* Series B 336:383–389.

McCartney, J. 2000. ''A New, Flexible Framework for Audio and Image Synthesis.'' *Proceedings of the 2000*

International Computer Music Conference. San Francisco: International Computer Music Association, pp. 258–261.

McMillen, K., D. Wessel, and M. Wright. 1994. ''The ZIPI Music Parameter Description Language.'' *Computer Music Journal* 18(4):52–73.

Partch, H. 1949. *Genesis of a Music*. New York: Da Capo Press.

Pope, S. T. 2002. ''Music and Sound Processing in Squeak Using Siren.'' In M. Guzdial and K. Rose, eds. *Squeak: Open Personal Computing and Multimedia*. Upper Saddle River, New Jersey: Prentice-Hall, pp. 377–421.

Puckette, M. S. 1996. ''Pure Data.'' *Proceedings of the 1996 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 269–272.

Roads, C. 1996. *The Computer Music Tutorial*. Cambridge, Massachusetts: MIT Press.

Ronken, D. 1970. ''Monaural Detection of a Phase Difference Between Clicks.'' *Journal of the Acoustical Society of America* 47(4):1091–1099.

Snyder, B. 2001. *Music and Memory*. Cambridge Massachusetts: MIT Press.

Wessel, D., M. Wright, and S. A. Khan. 1998. ''Preparation for Improvised Performance in Collaboration with a Khyal Singer.'' *Proceedings of the 1998 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 497–503.

Wessel, D. 1979. ''Timbre Space as a Musical Control Structure.'' *Computer Music Journal* 3(2):45–52.

Wessel, D., et al. 1987. ''MIDI-Lisp: A Lisp-Based Programming Environment for MIDI on the Macintosh.'' *Proceedings of the 1987 AES 5th International Conference: Music and Digital Technology*. New York: Audio Engineering Society, pp. 185–197.

Wright, M. and A. Freed. 1997. ''Open Sound Control: A New Protocol for Communicating with Sound Synthesizers.'' *Proceedings of the 1997 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 101–104.

Wright, M., et al. 2001. ''Managing Complexity with Explicit Mapping of Gestures to Sound Control with OSC.'' *Proceedings of the 2001 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 314–317.

Wright, M., and D. Wessel. 1998. ''An Improvisation Environment for Generating Rhythmic Structures Based on North Indian 'Tal' Patterns.'' *Proceedings of the 1998 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 125–128.

Zicarelli, D. 1991. ''Music for Mind and Body.'' *Electronic Musician* 7(2):154.

Zicarelli, D. 1998. ''An Extensible Real-Time Signal Processing Environment for Max.'' *Proceedings of the 1998 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 463–466.

Zicarelli, D. 2002. ''Max/MSP''. Available at www.cycling74.com/products/maxmsp.html.