The aXiØ MIDI Controller

Brad Cariou
University of Calgary
Calgary, Alberta, Canada
carioub@evds.ucalgary.ca

Abstract

The aXiØ (alternative, eXpressive input Øbject) is a user-programmable MIDI controller offering several kinds of control surfaces where many dimensions of control can be accessed simultaneously. The controller and its programming interface are described here in some detail. Applications of the controller are very briefly discussed.

1 INTRODUCTION

The reasoning behind the design and human factors of the aXiØ (alternative, eXpressive input Øbject) MIDI controller have been discussed in a previous paper [Cariou, 1992].

The aXiØ has three distinct and different control surfaces, roughly assigned to the levels of control described by Schloss [1990] as the timbral level (continuous control of synthesis parameters in a number of dimensions), the note level (selecting or triggering specific pitches) and the musical process level (an abstract paradigm that allows the mapping of gestures to operations not available in acoustic instruments).

The aXiØ (Figure 1) is designed to rest lightly against the shoulder of the performer to promote a closer relationship between the performer and instrument than a free standing controller. The left hand operates an enhanced joystick while the right hand operates a 5-key chord keyboard. An array of switches on the "neck" of the controller is easily reached with either hand. A factor influencing the selection of the chord keyboard and joystick is that their basic operation are easy to understand, but enough depth exists to make mastering the controller a challenge.

2 SWITCH ARRAY

The musical process level of control is dealt with using an array of switches. Each switch is 33mm x 25mm. The switch array is a set of 12 switches divided into a group of 4 and a group of 8. This division allows the user to conceive of the two groups as having different functions, or as 4 banks of 8 for 32 possible selections. The switches can be programmed to send MIDI messages up to three bytes long (continuous controllers, program change, etc.).

Each switch is individually configurable and is capable of sending two different messages. The switches can also act as momentary or push-on/push-off switches. In the momentary mode, pressing a switch sends a MIDI message and releasing the switch

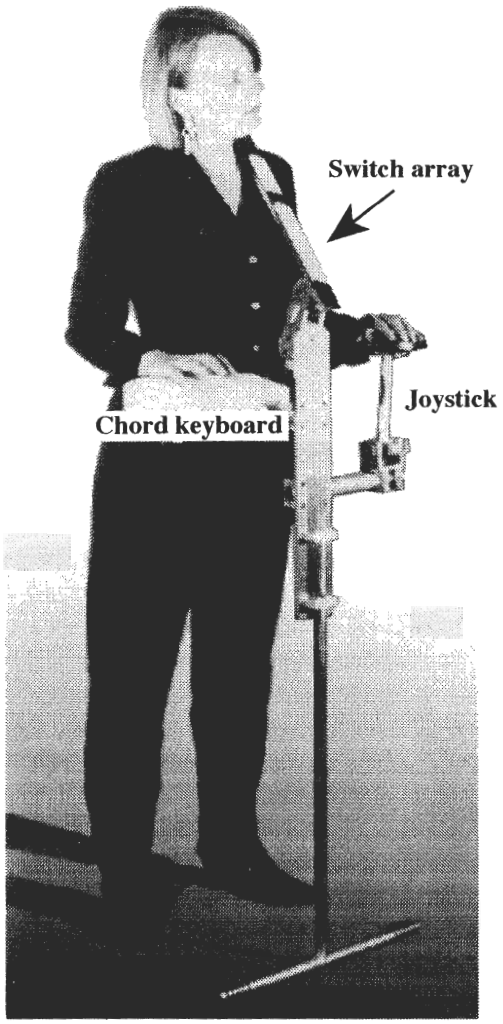

Figure 1: The aXiØ controller

sends the second programmed message. In the push-on/push-off mode, a switch sends a message only when pressed. The first press sends one message and a second press sends the second programmed message. This is useful for latching functions (such as sustain) where it is desirable to have a specific function associated with a single switch.

3 CHORD KEYBOARD

Pitch selection tasks are handled by a velocity sensitive five-key chord keyboard (Figure 2) for the right hand. The controller is designed to be monophonic so the musician focuses his or her attention on the manipulation of individual notes or sonic events. Notes are formed by using combinations of the keys. The 5 keys give 31 possible combinations (all keys up not included). This range is expanded through the use of octave switches located under the thumb of the left hand; 2 thumb switches transpose the range up by one or two octaves, and 2 more switches transpose down by the same amounts. This gives a possible pitch range of 78 notes using a total of 9 keys. The thumb switches also allow octave trills to be performed easily.

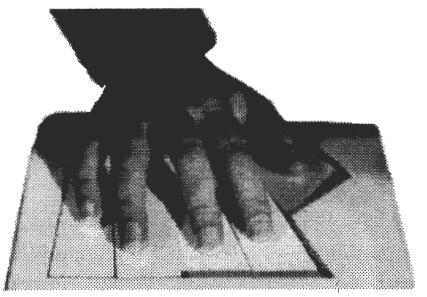

Figure 2: Chord Keyboard

The default fingering pattern follows a binary pattern with the thumb as the right-most digit of a 5 digit binary number (1), and the little finger as the left-most digit (16). The thumb key increases the pitch of already depressed keys by a semi-tone, the index finger by a whole tone, the thumb and index finger together by a minor third, the middle finger by a major third, and so on. The user is free to invent their own fingering patterns for particular scales or whatever purpose they have in mind.

The keyboard also generates MIDI aftertouch messages, and slight side-to-side movements generate MIDI data that might typically be used for pitchbend. The performer can also establish their own velocity curve for the keyboard.

Consider pressing two or more keys simultaneously to form a single note. Due to the vagaries of human physiology and mechanical variances in the key switches, the key switches will rarely close absolutely simultaneously. The problem arises of distinguishing between keys pressed "simultaneously" and a series of rapid individual key presses. A solution is to open an event window as soon as a key switch is closed and consider additional key switch closures taking place while the window is open to have occurred simultaneously. If keys are down and a new key press occurs after the event window closes, a new window is initiated and the process repeated. The same thing occurs when keys are released. If several keys are down and one (or more) are released, an event window is opened to capture key releases.

The length of the event window is adjustable to facilitate learning the fingering patterns. Unacceptable delays are introduced with longer window times as data is not sent until the window closes. As one becomes more skillful, the window time can be gradually shortened. A reasonable window seems to be about 30 milliseconds. This requires a precise and clean playing technique, but is not overly difficult to achieve. Even shorter window times can be used with sufficient practice.

3.1 Velocity Sensing

Velocity values are generated by measuring the time it takes to close a key switch. When keys are pressed simultaneously to form a single note, velocity is determined by the first key switch closed. If one or more keys are already down, depressing any additional keys produces a new velocity value. If several keys are down and then are released to form new notes, the last velocity generated by depressing a key is taken. If one depresses all 5 keys to form a note, lifting the fingers one by one will generate new notes that all have the velocity generated by the initial key depression. A future version of the keyboard electronics may use the speed at which keys are released to generate velocity values for the note formed by keys held down.

4 JOYSTICK

The joystick (Figure 3) has 4 dimensions of continuous control and 4 finger switches. The joystick employs a palmrest so a gripping action is not required, leaving the fingers free for other activities. The thumb is activates the keyboard octave switches (discussed above). The other switches are laid out in an arc to fall conveniently under the tips of the fingers. These switches operate in the same way as those of the switch array described above, and are typically used for sustain, portamento, or other functions that need to be accessed frequently.

The palmrest of the joystick rotates about 5 degrees about a point under the line formed by the knuckles. This provides a third dimension of continuous control and is somewhat analogous to keyboard aftertouch. Small raised bumpers in front of the switches prevent accidental activation of a switch while pressing on the front of the palmrest. A position-

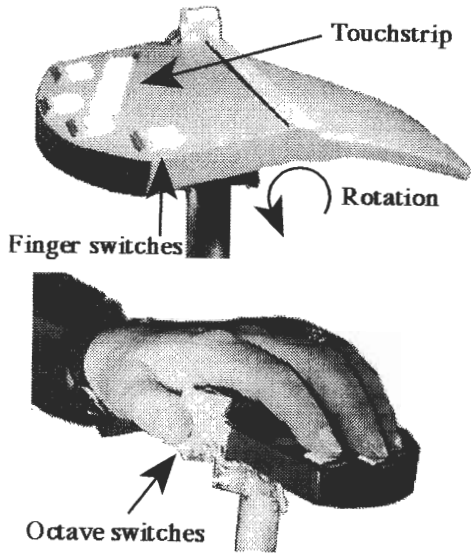

Figure 3: Joystick

sensitive touchstrip located behind the index and middle fingers is the fourth dimension of control. The strip requires only a very light touch and is therefore slightly recessed to prevent accidental activation. Data is generated by sliding one's finger along the strip, or by touching it in different places. This technique allows different values to be generated without passing through all the values in between, as is necessary for the other dimensions of control associated with the joystick. We are currently working on providing two additional dimensions of continuous control proportional to the speed at which the joystick is moving in the plane parallel to the floor.

5 PROGRAMMING INTERFACE

The software used to program the aXiØ was written using the MAX programming language and runs on a Macintosh computer. Figure 4 shows two of the screens. The interface is designed so that beginning users do not need any knowledge of MAX, but more advanced users have access to the MAX programming environment.

The controller electronics are programmable via SysEx, and each dimension of control has great flexibility. 8 setups each contain a complete configuration for the controller. The user can name each setup and set an associated MIDI program change message. The programming of the switches has already been described. For the 7 continuous control dimensions (the 4 of the joystick and keyboard ftertouch and side-to-side movement), the user can elect the MIDI controller (including aftertouch or pitchbend), minimum and maximum values and the MIDI channel on which data is transmitted. The sampling rate for each can be varied from 4ms to greater than 1 second. As well, two 127 byte tables are

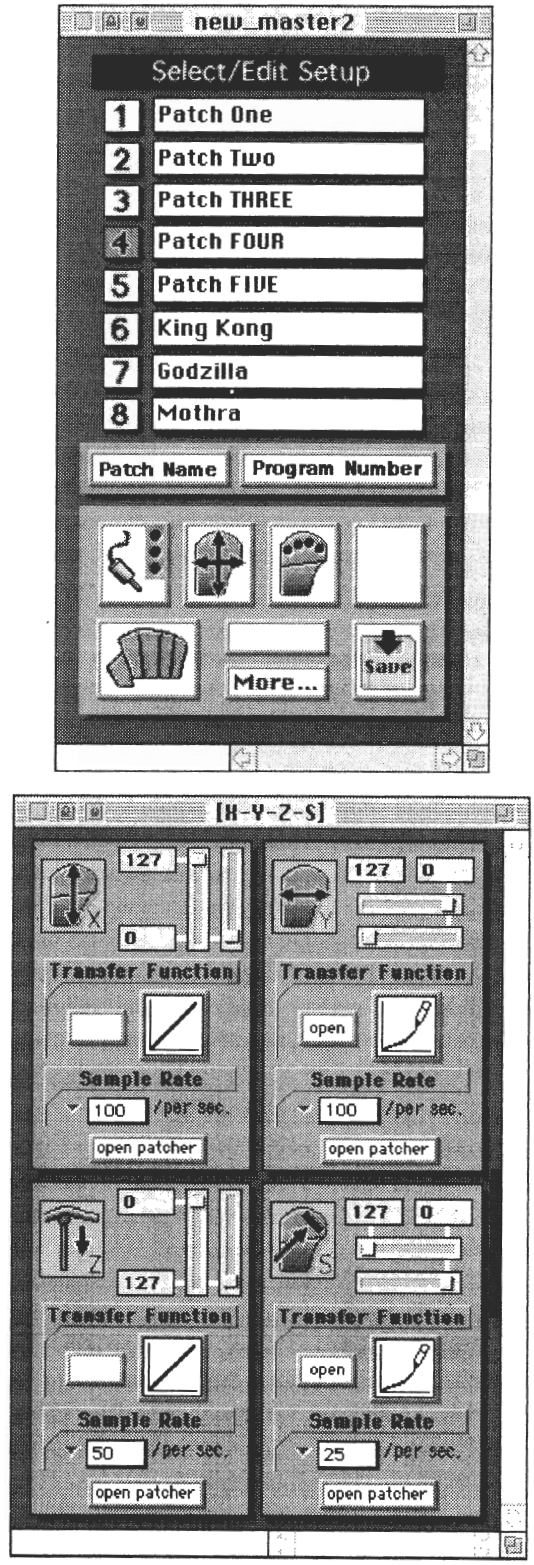

Figure 4: Screen shots

associated with each dimension of control. One of 8 pre-programmed tables (linear, log, anti-log, etc.) is

selected to set the overall response or "feel" of that dimension of control, and a second user-programmable table sets the actual data values generated.

The intent was to make the interface look and feel like a typical Macintosh application that someone could use without any specific knowledge of MAX programming. This goal was achieved, but not without some difficulties.

Several standard Macintosh interface elements are missing from MAX. A true Macintosh radio button for one-of-many selection tasks would be very useful. A scrolling text field could having many uses, such as extended instructions in help screens. As well, access to the different Macintosh window types (such as the non-resizable window) would be valuable in some situations. An argument for increased consistency in interaction can be made. Some user-editable objects respond to double-clicking with the mouse, while others respond to both double-clicking and an 'open' message. While there seems to be some logic behind this (only graphical representations such as the table object respond to the 'open' message), it is sometimes desirable to hide the underlying MAX patch from a user and rely on buttons for interaction purposes.

Some of the difficulties can be considered annoyances more than problems. For example, the ubutton object only highlights properly when placed over white space, making coloured buttons impractical (although they still work). The led object is a convenient indicator, but mouse-clicks on the led cannot be locked out, contradicting the behaviour of a real led. A useful interface detail has vanished with the introduction of coloured objects. Previously, the h- and uslider and dial objects could be "grayed-out" with a 'size 1' message, indicating to the user that the control is disabled.

Thanks to the adaptable nature of MAX, work-arounds have been found for the problems encountered. The interface does not have quite the look and feel originally envisioned, but the results are certainly acceptable.

6 ELECTRONICS

The controller uses two microprocessors; one is dedicated to the chord keyboard and the other to the switches and dimensions of continuous control. Initially two 8031s were used, but a 68HC11 micro is now employed for the switch and continuous control tasks. As illustrated in Figure 5, the controller is intended to be used in conjunction with a computer, although this not a requirement for use. The controller electronics are programmable via SysEx messages while being played, and if necessary, the machine code for the 68HC11 can be changed via SysEx as well.

The controller has most of the characteristics of a "Smart-Controller" as described by Settel et al., [1993] and we hope to explore applications in that area. The variety of control surfaces on the aXiØ suggest themselves to a wider range of control tasks than is possible with a bank of faders.

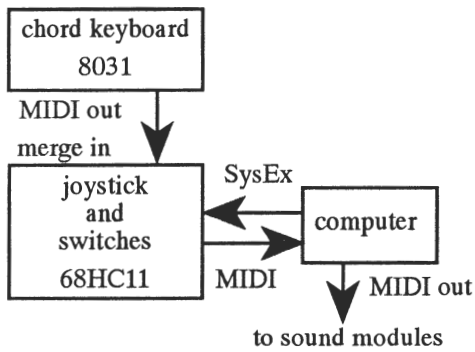

Figure 5: Controller electronics

7 CONCLUSION

The aXiØ controller has been used in composition and performance by Dr. David Eagle at The University of Calgary, Alberta, Canada. It has been found to provide ready access to many control parameters simultaneously, allowing the performer to do things not possible with a conventional keyboard. Preliminary steps in multimedia applications have been taken, and the aXiØ seems particularly suited to such uses. Improvements continue to be made to the aXiØ, and we plan to incorporate loudspeakers in the body of the controller in the near future.

ACKNOWLEDGMENTS

This project has received the support of a University of Calgary Research Services Grant and a grant from the Alberta Foundation for the Arts. I would also like to sincerely thank Dr. David Eagle for his continuing support and enthusiasm for the aXiØ.

REFERENCES

[Cariou, 1992] Brad Cariou. Design of an Alternative Controller from an Industrial Design Perspective. *Proceedings of the International Computer Music Conference* at San Jose, California, 1992.

[Schloss, 1990] W. Andrew Schloss. Recent Advances in the Coupling of the Language MAX with the Mathews/Boie Radio Drum. *Proceedings of the International Computer Music Conference* at Glasgow, Scotland, 1990.

[Settel et al., 1993] Zack Settel, Terry Holt and David Zicarelli. Remote Control Applications using 'Smart-Controllers' in Versatile Hardware Configurations. *Proceedings of the International Computer Music Conference* at Tokyo, Japan, 1993.