An Electronic Musical Controller Based on the Trombone
C. Chris Erway (cce3@cornell.edu), Spring 2002
Music 302 Independent Study – Ann Warde, Advisor

Introduction

When designing a new electronic musical instrument, in the absence of traditional restrictions, it is helpful to use as an existing instrument as a model. The slide trombone, unique among wind instruments for being able to produce a continuous range of pitches (in contrast to instruments with keys or valves), offers interesting possibilities for electronic modeling. This project aimed to explore these possibilities and create a single-hand touch-sensitive "trombone keyboard."

Design goals

Since the controller design is based on the trombone's slide positions and harmonic series, it is useful here to describe this system. The trombone slide has seven positions, each marking a different semitone, from "first position," not extended at all, to "seventh position," completely extended to produce a sound a tritone lower. Additionally, as with all brass instruments, the player can isolate higher and lower partials on the harmonic series. These partials, combined with the seven slide positions, produce the trombone's full range. Note that the same pitch may be produced by several different combinations of slide position and partial.

Our controller design is a fingering method based on this system of slide positions and partials. In this design, the seven slide positions are represented horizontally (from left to right) along the x-axis, with different partials represented along the vertical y-axis (higher partials higher, lower partials lower). Possible implementations of this design could be discrete (using perhaps a row of seven keys for the slide positions) or continuous (using a ribbon controller or tablet device), but all have the benefit of being hand-operated and significantly smaller in size than the trombone itself. This allows for the performance of fast sequences of pitches common to keyed and valved instruments, but generally uncharacteristic of the trombone.

Tablet input device

The first implementation decision made was the choice of an input device. A tablet – Tactex Controls, Inc.'s "MTC Express" multi-touch controller – was chosen for its continuous control and ease of integration (making this a software project, rather than electrical engineering). The MTC Express consists of a flat rectangular 5.75"x3.75" tablet that is pressure-sensitive and accurate to .05 inches. It is not a traditional single-pointer tablet such as those used with a stylus by graphic artists, but instead is able to sense multiple points of pressure, each at different pressure intensities. It connects to a computer via a serial port connection, and sensor data can be sampled at a maximum rate of 200 Hz.

Software environment

Tactex supplies input drivers for only the Macintosh and Windows operating systems, so the Microsoft Visual C++ environment was chosen for development. For MIDI output, the Improv library (developed by Craig Sapp of Stanford CCRMA) was used. The software development was all done in C++.

Processing approach

Creating MIDI messages in real time from sensor data received from the MTC Express was the central focus of the project. Various approaches at processing the sensor data to produce musical notes were explored, and will be detailed here. In all of the approaches, however, an emphasis was placed on keeping the processing algorithms fast and clean, so as to maximize the sampling rate and improve responsiveness. We discuss now four approaches to implementing our design, their evolution, and the progression between them.

I. Simple mapping

The most straightforward of implementations of our "trombone keyboard" idea would simply be to take the tablet, divide it into seven columns of slide positions and the desired number of rows of partials, and map each rectangular region directly to its corresponding musical pitch.

		slide positions					
		1	2	3	4	5	6	7
partials	F	65	64	63	62	61	60	59
	D	62	61	60	59	58	57	56
	Bb	58	57	56	55	54	53	52
	F	53	52	51	50	49	48	47
	Bb	46	45	44	43	42	41	40

Figure 1. MIDI note values in a simple mapping (five partials shown).

In Figure 1 we see an example of this simple mapping, with the first five partials of the trombone (excluding the fundamental) shown. Already we get an idea for the impressive range (two octaves and a semitone) covered in such a small space.

Our very basic algorithm is as follows: constantly poll the tablet sensors, waiting for pressure above a certain threshold (for noise elimination). When pressure has been recorded, compute a pointer (x, y) value and its corresponding pressure intensity. Map the pointer to its corresponding note value, and send a MIDI "note on" message to begin playing the pitch. Once pressure has ended for that region, send the MIDI "note off" message.

Drawbacks to this algorithm were evident immediately. The rigid boundaries set by this simple mapping are rather inflexible to a player "missing the mark" or, worse, playing too near one of these boundaries. When pressure was applied to a boundary area, the pointer's coordinates would often be found to fluctuate rapidly between two notes, causing an unintended trill until the player's finger was finally moved to the correct region. Even with these boundaries labeled on the instrument (somewhat like fret markings), it was found impossible to avoid occasionally placing one's finger directly on a boundary, causing this rapid trill effect.

II. Adaptive methods

The problems above, coupled with the knowledge that a player, when placing his finger down on the tablet, would never intend to press on a boundary area, but rather on a specific note, gave way to exploration of adaptive methods for assigning pitches. These methods attempt to assign a new touch to a certain pitch – not a boundary – and then assign future touches notes based on that first pitch. The

central issue of these methods is deciding whether a new touch constitutes a note outside a boundary (i.e., in a different partial, slide position), or inside a boundary (i.e., in the same partial, but a different slide position).

The first adaptive method was a fix for the simple mapping: that of assigning a new touch a note based on the simple map, and then establishing a "safe zone" around that touch where touches pressed in that region would still produce that same note. This prevented the annoying boundary problems from earlier.

Still, however, the fixed mapping of Figure 1 does not seem very adaptive; we'd like something that would use a mapping only for an initial value, and for the rest of the touches, assign pitches based on their relative distance from the first. One concept that is useful is that of a "tracking window," which surrounds the initial note and defines a wide strip of area that comprises touches produces pitches in the same partial as the first.

The first adaptive method was a fix for the simple mapping: that of assigning a new touch a note based on the simple map, and then establishing a "safe zone" around that touch where touches pressed in that region would still produce that same note. This prevented the annoying boundary problems from the first case from cropping up again.

Still, however, the fixed mapping of Figure 1 does not seem very adaptive; we'd like something that would use a mapping only for an initial value, and for the rest of the touches, assign pitches based on their relative distance from the first. One concept that is useful is that of a tracking window, which surrounds the initial note and defines a short, wide strip of area that defines the region successive touches must fall into to be considered in the same partial.
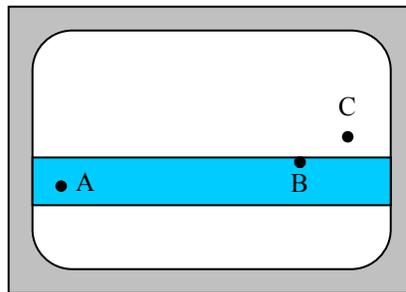
Figure 2. An adaptive tracking window demonstrating regions around touch A of the same partial.

In Figure 2 here, assume touch A is pressed first. A tracking window is established around that note. Next, touch B is pressed. It not that far above A on the y-axis; most likely it is meant to be in the same partial as A. Touch C, however, is too far above A to be considered as being in the same partial as it.

This tracking window allows the player to perform sequences of notes running up or down partials in a relatively smaller area than would be possible with a fixed mapping. The height of this window, width of a slide position, and the number of partials in the starting map are just a few of the many parameters can be tweaked to fine-tune this approach, depending on the player's ability to control small regions.

How, then, to decide when a sequence of touches has completed and a note should no longer be assigned based on the one before it? A phrase timeout was

implemented, waiting for long pauses between notes and concluding that after a certain amount of time (usually less than second), a musical phrase has ended and a new one begins.

The adaptive approach is interesting and works well, with one hitch. Consider the case of Figure 3 below. Here, let us say that touch A is struck first; a window is constructed around it as
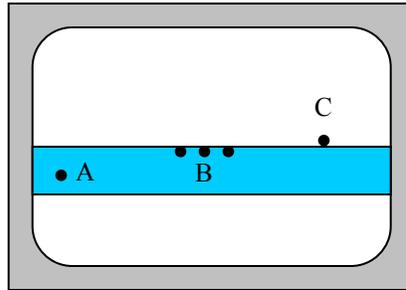
Figure 3. A problem with the adaptive window approach.

shown. The touches in group B follow, several this time; now, in comparison to these touches, it seems that perhaps touch A was a little low, and that the player really visualizes the partial around group B rather than A. Finally, touch C arrives, which the player may well have intended to sound in the same partial as touches A and B, but because of the initial low A, C is deemed "outside" the window and a new partial is established around C.

III. Statistical method

The statistical method described here rose out of this problem with the adaptive method. It is in fact the same as the adaptive method, with the exception of using a different rubric for defining the current partial. The mistake made in Figure 3 is averted by comparing new touches to a window centered on the average y-value across all touches in the partial, rather than the initial touch alone. For example, in Figure 3, by the time touch C arrived, touches A and B would have been averaged together, recentering the window somewhere between A and B and thus averting the mistake of keeping C out, but rather taking it in and assigning it a pitch in the same partial as A and B.

IV. Continuous controller

The final and most exciting input method implemented was that which most closely paralleled the trombone's continuous nature. Since the MTC Express offers very fine-grained continuous control, it seemed a waste to always be returning only discrete pitches. Thus this method uses along the x-axis not discrete slide positions, but a continuum of pitch. The y-axis, the harmonic series, is locked in this case to the five partials in Figure 1.

The bands of pitch are accomplished through use of MIDI pitch wheel continuous controller messages, which are sent for each x-axis change on the tablet. The result is a much more realistic variety of pitch bending effects which now become available; characteristic trombone slide effects are possible alongside the fast runs made possible by the controller's size.

Does this continuum of pitch coupled with the controller's small size make the new instrument hard to keep in tune? From my own experience, I do not believe so; some practice and a bit of using one's ear is necessary, just as with any non-electronic instrument, but it can be played in tune. It is interesting to point out this fact, as tuning "by ear" is not something one usually does with a MIDI controller or, say, an electronic keyboard.

Conclusion

This new instrument offers much opportunity for further research. Simply more practice will be necessary to both learn to play and improve the instrument. Ideas for further research include breaking free of MIDI and developing with digital audio; allowing for multiple touches and thus "chords" to be played rather than single notes; creating more realistic brass effects through gestural controls on the tablet; exploring better adaptive techniques for "learning" one's playing style; and using the actual values of partials on the harmonic series rather than their closest approximants.

The instrument has many features which seem opposing: electronic, yet based on a traditional instrument; continuous as well as discrete; tuned by ear, yet electronic; small, yet with a large range. These peculiarities make it an interesting controller.

References

Bromwich, M. A. 1997. "The Metabone: An Interactive Sensory Control Mechanism for Virtuoso Trombone." In *Proceedings of the 1997 International Computer Music Conferenc*e. San Francisco, International Computer Music Association, pp. 473-475.
Instut de recherche et coordination acoustique/musique (IRCAM), 2000. "Trends in Gestural Control of Music." (e-book)
Sapp, Craig. 2001. "Improv documentation" at http://improv.sapp.org.
Tactex Controls, Inc. 2000. "MTC Express Owner's and Developer's Guide."