# A Force Sensitive Multi-touch Array Supporting Multiple 2-D Musical Control Structures

David Wessel, Rimas Avizienis, Adrian Freed
Center for New Music and Audio Technologies (CNMAT), UC Berkeley
Berkeley, CA 94720
(1) 510 643-9990
wessel@cnmat.berkeley.edu

Matthew Wright
CNMAT/UCB and CCRMA/Stanford
Stanford University
Stanford, CA 94305-8180
matt@(cnmat.berkeley, ccrma.Stanford).edu

## ABSTRACT

We describe the design, implementation, and evaluation with musical applications of force sensitive multi-touch arrays of touchpads. Each of the touchpads supports a three dimensional representation of musical material: two spatial dimensions plus a force measurement we typically use to control dynamics. We have developed two pad systems, one with 24 pads and a second with 2 arrays of 16 pads each. We emphasize the treatment of gestures as sub-sampled audio signals. This tight coupling of gesture with audio provides for a high degree of control intimacy. Our experiments with the pad arrays demonstrate that we can efficiently deal with large numbers of audio encoded gesture channels – 72 for the 24 pad array and 96 for the two 16 pad arrays.

**Keywords**

Pressure and force sensing, High-resolution gestural signals, Touchpad, VersaPad.

## 1. INTRODUCTION

### 1.1 Motivation

The first author of this paper has spent a considerable part of his life (1990-present) as a performing musician with a tactile music controller that senses both the location and force of his fingers: the *Thunder,* designed and constructed by Don Buchla, shown in Figure 1.

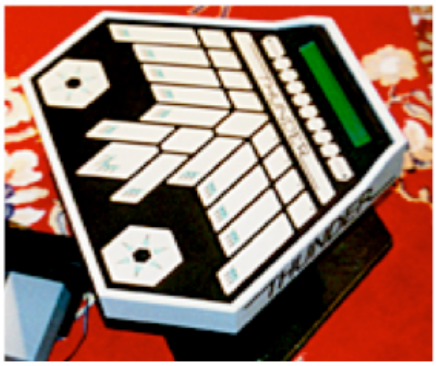

**Figure 1: Thunder by Don Buchla. The ridge-separated strips are sensitive to both location and finger pressure. It sends MIDI controller data and is both continuous and polyphonic in character.**

Building on what we learned using Buchla's *Thunder* we have created a touch location and force sensing system with the same continuous and polyphonic character but with two-dimensional sub-surfaces instead of the one-dimensional strips. We also wanted our new controller to perform with high reactivity and temporal precision. Two-dimensional sub-surfaces allow us to map finger locations to timbre spaces, rhythm spaces, melody spaces and a wide variety of other 2-D representations for the low-dimensional control of musical material [3]

We chose touchpad technology capable of sensing X, Y and Z (where Z is force) and designed a "brick-wall" array that was tightly packed and allowed each of the ten fingers to touch separate pads. Our 24 pad layout is shown in Figure 2.

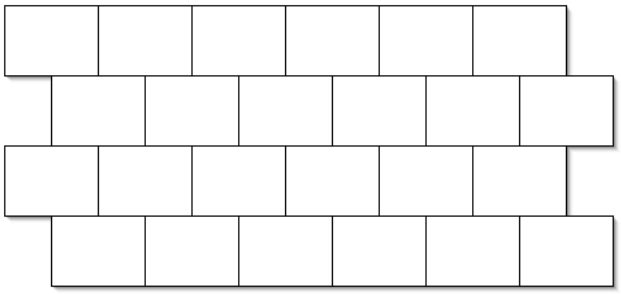

**Figure 2: Layout for a new control surface: 24 X, Y, and force sensitive pads separated by ridges for sightless control.**

Each of the two 16 pad arrays used a similar layout but oriented so the long sides of the pads functioned as the Y dimension.

### 1.2 Overview

We describe in considerable detail key issues in coupling gesture data with audio data by locking it to the audio sample clock. We also describe the Xylinx FPGA-based data acquisition system used in the project as well as additional design considerations. We characterize via physical measurement the characteristics of the touchpads and demonstrate that sampling gestures at high rates provides musically useful data particularly for the shaping of amplitude envelopes and related gain structures. We seek an instrumental like response with fine control of dynamics. We stress high reliability and travel worthiness and describe many musical applications.

## 2. ENGINEERING

### 2.1 Interlink VersaPad Touchpad

The *VersaPad* semiconductive touchpad[1] (by Interlink Electronics) estimates the X and Y position and applied force (Z) of an object touching it. It is most commonly used as a pointing device in laptop computers and in hand-written signature recognition applications. The VersaPad consists of a stack of four layers: a base, two sensor layers of resistive film, and a touchpad surface on top. Each sensor layer is made up of two conductive traces (at opposite ends of the device) which are connected to each other through a resistive material. The two sensor layers are rotated 90 degrees with respect to one another, so there is one conductive trace along each edge of the device. These 4 conductive traces are brought to the exterior of the pad by a short length of flexible flat cable. When an object touches the touchpad the two sensor layers come into contact at the point of applied pressure. This results in a pressure-dependant resistance between the two layers at the point of contact, and a position-dependant resistance between the point of contact and the conductive traces on each sensor layer. By measuring the resistances it is possible to calculate the location of the point of contact and the amount of force being applied.

Interlink utilizes measures these resistances using an inexpensive PIC based microcontroller circuit (with only a few passive components) to sample the X, Y and Z values 40 times a second. Their patented approach is economical for a single trackpad but would not satisfy our goals for a higher sampling rate of 6kHz. More importantly we want to sample the measurands concurrently. Interlink's approach is to steer currents through the array and infer resistance values from the charge rates of capacitors. This introduces temporal uncertainty and delay which we avoid in our novel circuit by supplying a small constant current through the series-connected X, Y, Z resistors and concurrently converting the induced voltages across these resistors into digital values using a a multi-channel ADC. Our solution is comparable in parts count to Interlinks and just as cost effective as we only have one microcontroller for all 24 trackpads.

### 2.2 Pressure = Force / Area

The only way to get the pad to output its maximum "pressure" value is to contact a relatively wide area of the pad. Therefore it is not strictly measuring pressure, or the same weight on a smaller area would be *higher* pressure. It is better thought of as a force sensor. Whatever the exact physical interpretation of the Z axis, the output increases monotonically with the performer's effort.

### 2.3 Data Acquisition Hardware

Our hardware system is made up of five discrete PC boards. Four of the boards are identical, with 6 touchpads mounted on top and analog conditioning circuitry and multichannel analog-to-digital converters on the bottom. These 4 sensor boards are connected to the controller board, which is based around a Xilinx Virtex4 FX12 FPGA. The FPGA has an embedded PowerPC core that runs at 300 MHz. In addition to the FPGA, the controller board has 64M of DRAM, 8M of flash memory, a gigabit Ethernet interface, clock oscillators and a power supply.

[1] http://www.interlinksensors.com/products/integratedmouse/versapadoem.html

We scale and normalize all measurements in hardware, so that X, Y, and Z are output in the range [0,l].[2] Also, since the touchpads' X and Y values are meaningless when the pressure goes to zero, the hardware keeps track of the most recent good values of X and Y, (i.e., where the finger was just before it came off the pad) holding them constant for as long as Z=0.

Our design outputs sensed data over the Ethernet interface as described in the next section. Our hardware also includes a word clock output to enable sample-rate clock synchronization (without resampling) between our device and a standard audio interface.

### 2.4 Sensor Measurements via OSC

As there are X, Y, and Z (force) measurements from each pad there are a total of 24 x 3 = 72 values to be acquired and transmitted. Presently, our design can operate in one of two modes. In the first mode, the touchpads are sampled at a low rate (0-200 Hz) and Open Sound Control (OSC) [8] packets containing the measurement data are transmitted as UDP packets over the Ethernet interface. Each OSC packet gives the current X, Y, and Z values only for the pads that are currently being touched.

### 2.5 Sensor Measurements via Audio Signals

The second mode provides for audio sample synchronous output from the pad array [2]. The touchpads are scanned at a one eighth the audio sampling rate (up to 6000 Hz), the data is up-sampled to audio rates (44.1 or 48 kHz), converted to 32-bit floating point, and then encapsulated in a stream of UDP packets which are sent over the Ethernet interface. A custom driver on the host computer receives these packets and presents them to the operating system as a collection of audio input channels, thereby enabling us to use these high-rate control signals in audio processing applications. Our chosen development environment, Max/MSP, provides for up to 512 audio input channels, more than enough for the 72 sensor inputs. In this mode, the current values of all sensor signals can also be polled asynchronously via CNMAT's /dev/osc mechanism [1].

Clearly this synchronous approach provides for more control intimacy as the gestures are encoded as jitter-free signals locked to the audio sample clock. Sampling the sensors at 1/8 the audio sampling rate results in high temporal resolution (or, equivalently, a wide bandwidth in the frequency domain) on the gestural signals produced by the human performer. This naturally brings up the question of whether there is any useful information in all the extra data output by the device, which we will address below. Surprisingly, the large number of channels does not appear to cause an unmanageable processor overload, though we need to do more evaluation with a larger variety of applications that themselves demand considerable processor resources. The conversion to floating point in the hardware avoids a huge amount of processing on the host system. We were so encouraged by the very low processor load imposed by the upwards to 100 channels of audio encoded gestural data that we concluded that the host

[2] The input range for the pressure scaling was calibrated empirically by touching a representative pad with a finger as lightly and heavily as possible. Likewise, for X and Y, we empirically found the edges of the area where the trackpads can reliably sense position; there is a very small "dead" area around the outside of each touchpad where position cannot be sensed.

application, in our case Max/MSP, was the place to do the polling that would selectively convert signals to asynchronous events. We have successfully implemented a variety of polling mechanisms in Max/MSP and have gained considerable experience with mappings that involve mixture of signal-synchronous-based control and asynchronous-event-based control.

*2.5.1 Examples of Short Taps*

Figure 4 shows various "envelopes" of sensed pressure/force for a few short example taps on one pad. Figure 5 is a detail of the same, focusing on the first 3 milliseconds after each attack. Note that different methods of tapping the device result in substantially different shapes of these curves even within the first millisecond.

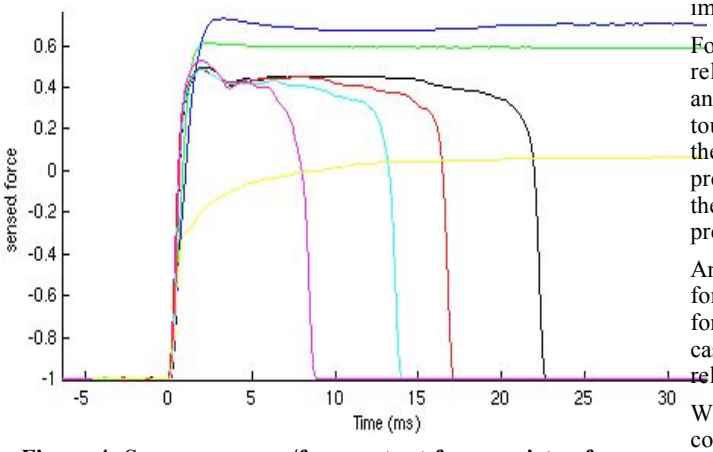


**Figure 4: Sensor pressure/force output for a variety of various tapping gestures. Note that a variety of short durations can be differentiated. Our experience demonstrations that with practice such fine differences in duration can be intentionally controlled.**

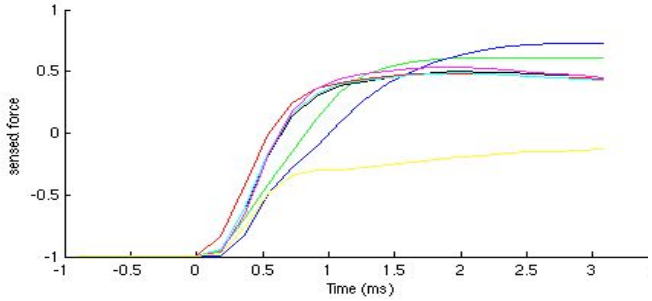


**Figure 5: Here we show the first 3 milliseconds of the above plot demonstrating sensed differences in the attack transient.**

*2.5.2 Signal-Domain Gestural Processing*

Many of the control inputs to MSP objects, for example, gain control, an oscillator's frequency, filter coefficients, etc., can be signals, thus avoiding the timing uncertainties of the Max event domain. Furthermore, there are some non-obvious techniques for handling signals in MSP such as gate~ that can be exploited with the pads. Finally, it's always possible for the MSP programmer to downsample any of these signals, e.g., by polling it with the snapshot~ object.

Keeping gesture data in the signal domain provides new opportunities for the study of the role of gesture in music making and its perception. These include joint time-frequency representations of gesture data.

### 2.6 Haptic Regularization

Two desirable features that we have identified for musical control of computers are predictability and a correspondence between the "size" of a gesture and the resulting acoustic result [5]. Towards these ends, we must consider the relationship between the performer's own perception of the input gestures and the outputs of the sensors.

For X and Y the relationship between actual position and output value is nearly linear, and a performer's perception of position closely matches the actual position. Thus the performer's internal notion of position has a linear relationship with the actual physical position as well as the touchpads' position outputs, and so the position variables are immediately usable for musical interface purposes.

For pressure applied vertically to the pad, however, the relationships among perceived effort, actual physical force, and sensed force output are more problematic. A very light touch moves sporadically and rather uncontrollably through the first half of the output range, then moderate amounts of pressure result in smoothly controllable output values, and then the difference between subjectively heavy and very heavy pressure has very little effect on the output.

An ideal force sensor has a linear relationship between input force and the output quantity (e.g., resistance). For real-world force sensors, this relationship may be nonlinear, in which case engineers will often apply the inverse function to this relationship so as to "linearize" the sensor.

When we consider the performer, things become still more complicated, as shown in figure 7. Physical linearization of the sensor may be an important part of the overall solution, but what we really require is a good mapping between intended effort and the output of our sensor, a process that we might term *Haptic Regularization*. In this case we are looking for the inverse function of the composition of the motor control function with the sensor function. The best way to choose the appropriate function here is by systematic trial and error in the musical context in which the specific controller is to be used.

To this end, we have implemented an environment that allows us to choose among a variety of non-linear function to achieve overall haptic regularization.

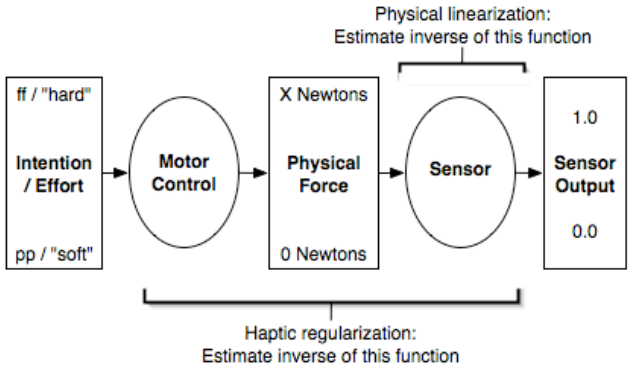


**Figure 7: The relationship between intention and effort and the sensor output appropriate to a given sound generating algorithm is complex, involving factors such as the resilience of sensing system and the behavior of the generative algorithm itself.**

*2.6.1 Force Hammer Measurements*

We now examine the problem of characterizing the relationship between the physical force applied to a pad and the output of the sensor with its analog data conditioning circuit. In other words, we want to characterize the transfer function labeled "Sensor" in Figure 7; this is a well-defined and non-subjective subtask that's an important part of the overall goal of haptic regularization.

We employed an impact force hammer to make the measurements to obtain a function relating the physical force applied to the pad to its output. To make these measurements appropriate to the manner in which the pads will be used in musical practice we placed the first author's index finger on the pad and tapped on his fingernail with the hammer. This technique yielded the scatter plot shown in Figure 6.

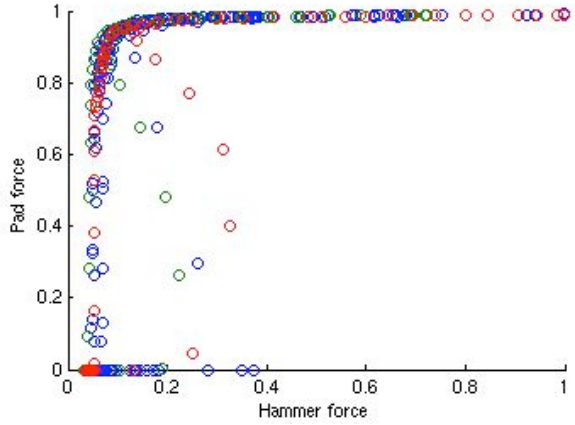


**Figure 6: Scatter plot of pad's "force" output as a function of the force output by a force hammer applied to a finger touching the pad. Note the nearly inverse exponential relationship.**

To make the relationship between hammer force applied to the finger linear we need to apply the inverse of function shown in Figure 6 to the sensor output.

## 3. ERGONOMICS

We wanted the pad array to fit nicely under the hands, provide some form of non-visual reference, provide for some form of displacement when applying finger pressure, and be mounted solidly so it can be played percussively.

### 3.1 "Brick Wall" Pad layout

At first glance, especially with the sensor layout of Buchla's *Thunder* (Figure 1) in mind, the "brick wall" layout we chose for the pads may not seem ergonomic. But the interleaved rectangular layout shown in Figures 2 and 8 provides the tightest packing possible and affords the possibility of touching ten separate pads one with each finger. A similar tight layout is used in each of the two 16 pad arrays.

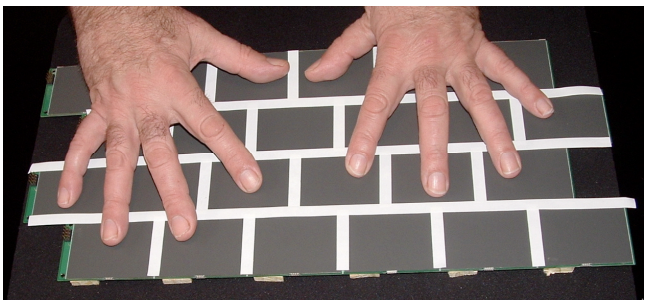

**Figure 8: The pads under the hands of the first author. The white grid overlay not only provides a tactile reference but also occludes the dead regions around the edges of each pad.**

### 3.2 Tactile Reference Strategies

In order to be able to play the interface without looking at it, we designed a tactile reference. The touch sensitive surface on the Buchla *Thunder* provides embossed ridges around each sensor area. We chose to place a laser-cut plastic overlay approximately 1.5 millimeters thick over the touchpad array. This overlay also covers the dead areas around the edges of the pads where contact cannot be sensed, making it physically impossible to touch them there.

The grid overlay also provides a way of holding down another material such as fabric, thin rubber, or other materials such as various grades of sandpaper used to change the texture and the force transfer properties.

### 3.3 Additional Inputs and Outputs

To complement the 24 touchpads already described, our complete device also contains 16 pushbuttons, each with 2 LEDs (green and red), one long position-sensing strip, and four inputs for 0-5V control voltages intended for footpedals and similar controllers. On the 16 pad controllers we added 4 LED's per pad as indicators.

## 4. MAPPING SCHEMES / MUSICAL METAPHORS

Our existing mapping schemes for the device treat each touchpad independently. Additional experiments involve having the behavior of the pads interact with each other.

### 4.1 Subjective 2D Spaces

One concept that works very well for each pad is to map the X and Y values to a subjective space of some kind [3]. This could be a timbre space [6], a melodic process space, or two dimensional space representing the similarities among various rhythms.

### 4.2 Dipping

Dipping is a control metaphor [5] for dynamics. The basic idea is that there is a parameterized musical process running silently until a force is applied to the appropriate pad. Finger force increases the dynamic level which typically includes an increase in spectral bandwidth associated with increased effort. MIDI or OSC-based slow control rates are sluggish. High rate sampling of finger force on the pad affords nimbleness. We provide a sound example of touchpad dynamics dipping with Shifty Looping [7].

### 4.3 Migrators

*Migrators* are Max/MSP instruments that use a large number of oscillators, usually a hundred or more. The oscillators behave like particles that move from one frequency location to another in a migratory manner. The frequency locations are described by a large table that is typically indexed in one-cent

(1/100 semitone) increments. These tables function as targets for the migratory behavior of the oscillators. They specify the overall spectral shape as well as the precise frequency locations. Performing with a *migrator* on a pad involves navigating in a subjective 2-D space of target tables. With the array several migrations can operate simultaneously. Again force is used to control dynamics. If no force to applied to the pad the instrument remains silent.

### 4.4 Granular

The array can be used in a variety of ways with granular synthesis. The key parameters to control are grain size, grain shape, grain rate, location from where the grain is taken in a buffer, etc. We have found it effective to use multiple pads to control a given granular process. We provide a few examples, one were the mean location of a grain in a buffer is controlled with the X axis and the variance of the location is controlled by the Y axis. Dynamics are again controlled by finger force. In another example we use the Pitch Synchronous Overlap Add formulation of granular synthesis and scrub along the X axis while using the Y axis to control the degree of heterophonic connection with a fixed voice.

### 4.5 2D 4-point crossfading

Using a 2D mapping one can easily implement classical vector synthesis by cross fading among four frequency synchronized sound sources.

### 4.6 Exciting Resonant Filters Directly with Pad Pressure

We demonstrated earlier that with the pads we can obtain a variety of envelope shapes and durations. If we use these gesture generated functions to amplitude modulate noise used to excite models made with sharply tuned resonators we can obtain a rich variety of percussion sounds. If the location on the pad is used to specify the frequency locations of the resonators an even larger variety of sounds can be obtained.

## 5. CONCLUSION

We have designed, implemented, and evaluated an array of touchpad sensors that affords continuous, polyphonic, reactive, and temporally precise synchronization with audio. There are two areas were our sensing system could be considerably improved. As shown in Figure 6 the relationship between force and sensor output is very steep at the onset arriving at a high output value quickly. Our haptic regularization efforts have produced useable results but we would prefer a more linear sensor for finger force. Also the VersaPad pads are hard and do not displace with force. A system that provides a better feeling of resilience would be desirable.

We also conclude that a large number of audio channels can be used to encode gestures as sample-synchronous signals with little in way of processor overhead. As a result we have decided to keep our gestures as signals until they are needed as asynchronous events by the host application.

## 6. ACKNOWLEDGEMENTS

Richard Dudas, Michael Gurevich, Julius Smith, Carr Wilkerson, Andy Schmeder, and Don Buchla. Partial support for this project has been provided by a COR research grant from the UC Berkeley Academic Senate.

## 7. REFERENCES

1. Freed, A., Avizienis, R. and Wright, M., Beyond 0-5V: Expanding Sensor Integration Architectures. in International Conference on New Interfaces for Musical Expression, (Paris, France, 2006), 97-100.
2. Freed, A. and Wessel, D., Communication of Musical Gesture using the AES/EBU Digital Audio Standard. in International Computer Music Conference, (Ann Arbor, Michigan, 1998), International Computer Music Association, 220-223.
3. Momeni, A. and Wessel, D., Characterizing and Controlling Musical Material Intuitively with Geometric Models. in New Interfaces for Musical Expression, (Montreal, Canada, 2003).
4. Smith, J.O., III Introduction to Digital Filters. W3K Publishing, 2003.
5. Wessel, D. and Wright, M. Problems and Prospects for Intimate Musical Control of Computers. Computer Music Journal, 26 (3). 11-22.
6. Wessel, D.L. Timbre space as a musical control structure. Computer Music Journal, 3 (2). 45-52.
7. Wright, M., Shifty Looping: meter-aware, non-repeating rhythmic loops. in International Computer Music Conference, (New Orleans, LA, 2006), International Computer Music Association, 44.
8. Wright, M. and Freed, A., Open Sound Control: A New Protocol for Communicating with Sound Synthesizers. in International Computer Music Conference, (Thessaloniki, Hellas, 1997), International Computer Music Association, 101-104.