

# New Musical Mappings for the MATRIX Interface

Dan Overholt

CREATE, Center for Research in Electronic Art Technology  
University of California at Santa Barbara  
*email:* dano@create.ucsb.edu

## Abstract

Traditional musical controllers, such as MIDI keyboards, provide an event-based model of music in which each note instantiates a copy of the sound. An alternative is the MATRIX (see Figure 1), which can modify the sound continuously, allowing for a wider range of sonic manipulations. I have developed a set of new musical mappings and synthesis models for this instrument in the SuperCollider language. The methods described generate audio output that is controlled by real-time updates from the MATRIX interface (Overholt 2000). Each mapping allows the performer to express complex musical intentions in a direct manner.

## 1 Introduction

There are two opposing models prevalent in musical mapping systems today. In the *discrete* model, sounds or notes are considered to be independent and unchanging events. In the *continuous* model, sounds or notes are modified by controlling volume, pitch, timbre, and other dynamic parameters. Discrete and continuous models can be seen in traditional music notation, computer music languages, MIDI, synthesis hardware, and synthesis software. Although the distinction between discrete and continuous models is fundamental, it is not often made, and a large portion of today's electronic music interfaces are primarily discrete.

While many controllers concentrate on discrete mappings, virtually every music synthesis system exhibits both models to some degree. In other words, synthesis systems have aspects of both the discrete *and* continuous models. However, most interfaces provide only limited access to real-time continuous control (e.g., mod wheels, foot pedals, after touch). Furthermore, these controllers can only modify a sound or note that has been instantiated by a discrete controller, and are often given a small set of pre-defined mappings that are confined to high-level parameters in the synthesis engine.

Armed with an array of 144 continuous controllers, it is possible to develop musical mappings that shed new light on existing synthesis models, exposing simple yet effective methods, revealing subtle sonic manipulations, and unmasking limitations of discrete controllers. I will begin by

explaining the motivation and background of this research, and then describe a set of new musical mappings that incorporate the continuous controllers of the MATRIX interface. Both continuous and combinatorial discrete-continuous mappings are implemented by means of a variety of synthesis techniques. Finally, I describe the application of frequency domain analysis to a new mapping system for music analysis and re-synthesis.



Figure 1: The **MATRIX** Multipurpose Array of Tactile Rods for Interactive eXpression

## 2 Motivation and Background

In electronic music composition, it is necessary to use a variety of synthesis techniques to generate audio in musically interesting ways. These synthesis techniques can be seen as modern corollaries to the

different orchestral instruments. However, they have evolved in recent years largely as software based systems, or virtual instruments. As a result, there are no physical interfaces tied to each new synthesis model, as was dictated by the physics of an orchestral instrument's method of sound production. What we are left with are many *headless* instruments that, for lack of a better interface, are routinely controlled by a MIDI keyboard or a standard computer keyboard (through the use of a score file).

As previously mentioned, the drawback with using a MIDI keyboard is the discrete nature of its mapping—it leaves musicians without useful handles for many expressive synthesis parameters. And while this is satisfactory for some situations (after all, an acoustic piano has the same limitations), there is no way to manipulate a sonic texture at even the level of a violin or wind instrument. Likewise, the problem with the computer keyboard approach is that it is too painstaking to generate score files that describe the nuances of micro-sound parameters, and it is not an interactive (real-time) process.

Instead of losing much of the expressiveness of our new *orchestra* of synthesis techniques, we should extend our sonic control to a more meticulous level than is possible even with any acoustic instrument. Many synthesis algorithms would be capable of this by employing a large number of continuous controllers to modify their parameters. The next generation of musical instruments should allow musicians to more directly manipulate the *physics*, or synthesis parameters of these virtual instruments in a real-time performance. This modus operandi is the inspiration behind the development of the MATRIX and its different mappings as a new musical instrument.

## 2.1 System Architecture

The MATRIX interface consists of a 12x12 array of spring-mounted rods that move vertically in their guides. The position of each rod is determined using opto-electronics and a FPGA (Field Programmable Gate Array), which sends the data to a host PC as a serial stream. The PC acts as a visual display (see Figures 2 & 4), and translates the serial data into the Open Sound Control (OSC) format (developed at U.C. Berkeley's Center for New Music and Audio Technologies). The OSC data is sent across an Ethernet network to a Macintosh running SuperCollider (an audio synthesis programming language developed by James McCartney), where the control data from the MATRIX is mapped to musical parameters in a real-time audio synthesis algorithm.

## 3 The Mapping Models

This section will outline three different synthesis algorithms that have been developed as musical mappings that utilize the array of continuous controllers offered by the MATRIX.

### 3.1 Additive Synthesis Mapping

The technique of combining sine waves to produce a complex signal is well known—the fact that any periodic waveform can be expressed as the sum of one or more sine waves was proven by Jean Baptiste Fourier. This phenomena is exploited in the additive synthesis mapping with one sine wave assigned to each of the MATRIX rods. In theory, this method can produce any sound wave imaginable that has 144 partials or less.

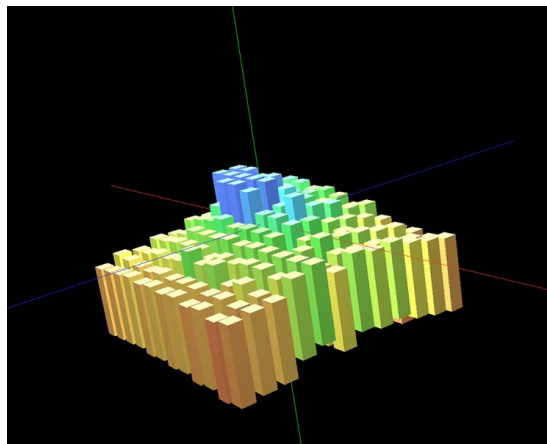


Figure 2: The position of each rod controls the amplitude or frequency of one sine wave

The algorithm is implemented as a bank of 144 sine oscillators, and has two different modes. The first allows the performer to control the frequency of each partial, and the second controls the volumes of a predetermined set of overtone frequencies. The audio output is the just the addition of all 144 oscillators.

### 3.2 Scanned Synthesis Mapping

The technique of scanning synthesis (Verplank, Mathews, and Shaw 2000) was developed as a way of generating a wave shape by scanning along a surface. In this implementation, it is adapted for use with the MATRIX by mapping the height of each rod to individual sample values.

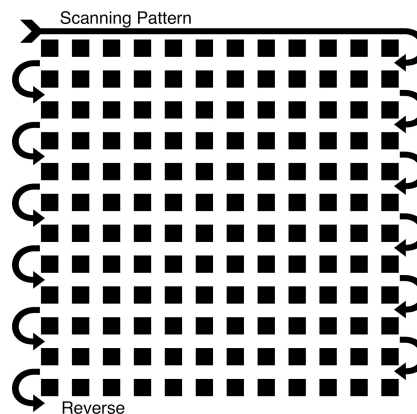


Figure 3: The Scanned Synthesis sampling pattern

To avoid discontinuities in the audio signal, the rod values are stepped through in a zig-zag pattern (see Figure 3), and then the entire array is read in reverse. This provides a closed path for the periodic sample values. This mapping can be looked upon as a dynamic wave table controlled by the performer—it provides a method for directly manipulating the waveform of a sound by human movements. It is used in conjunction with a MIDI keyboard to provide discrete pitch data that simply changes the rate at which the array is read.

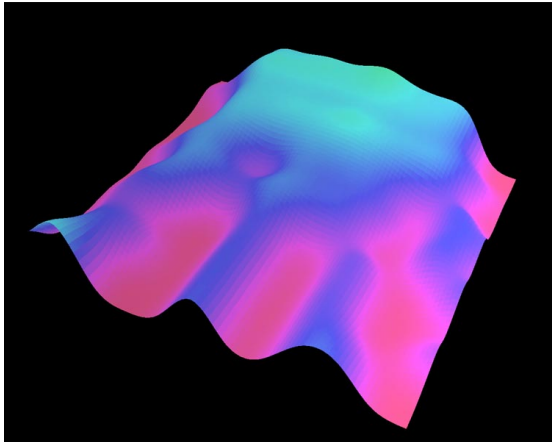


Figure 4: Scanned synthesis or wave-terrain synthesis uses the surface generated by the MATRIX rods

This mapping is actually a cross between scanned synthesis and wave-terrain synthesis (Bischoff, Gold, and Horton 1978), where the wave table is drawn from a path in a static instead of a dynamic surface.

### 3.3 Frequency Analysis Mapping

By using analysis software to generate a mapping between short segments of a sound and the MATRIX rods, this model uses the frequency components of a signal to organize the content for a granular synthesis algorithm. The analysis software is a Macintosh OS-X program that is based on FFTW, a library for computing the Discrete Fourier Transform, and the PVD phase vocoder (Ramakrishnan, 2001). The application analyzes sound files in the frequency domain with a user-specified window size (FFTW allows for any window size—not just powers of 2), and generates the results by classifying the FFT windows according to their frequency content.

The program's output is a text file that lists the windows categorized by their spectral characteristics. This text file is then read by SuperCollider, and used to determine which rod each analysis window should be assigned to on the interface. Indexing a sound in this way results in the frequency-dependent mapping of the sound grains controlled by the MATRIX, where each FFT window becomes a grain.

The re-construction of sonic material is implemented using 144 discrete audio buffers that loop through short segments of the audio file at the

start points specified by the analysis software. The volume level and repeat rate of each segment, or grain is controlled by the vertical position of the corresponding rod on the MATRIX. A performer interacting with this mapping can choose to pick out individual grains, small groups, or entire clusters of sound thereby creating calm or agitated moments during a musical piece.

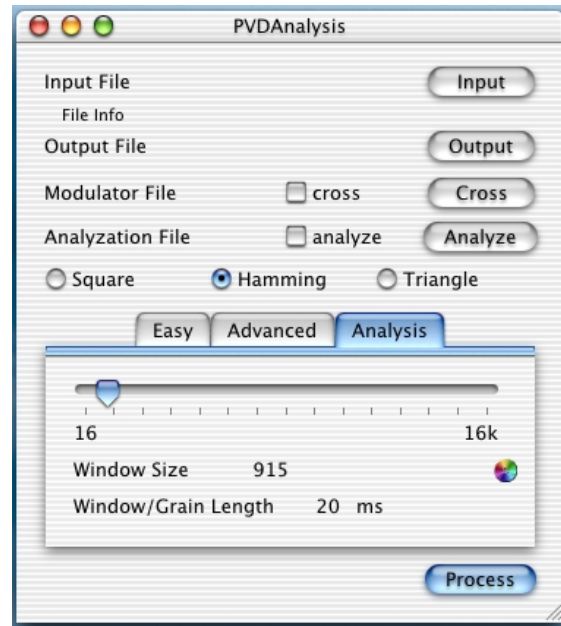


Figure 5: Frequency-domain analysis software used for granular mapping

## 4 Conclusion

The musical mappings described here are adaptations and extensions of existing synthesis techniques that take advantage of the substantial number of continuous controllers provided by the MATRIX interface. The author is continuing the exploration of signal processing techniques that yield the most extensive control of an audio signal, and is using the results of this research to compose and perform new music using the MATRIX.

## 5 Acknowledgments

I would like to thank Professor Curtis Roads, Stephen Travis Pope, and Ioannis Zannos, and many others for their support and help with this research.

## References

- Bischoff, J., Gold, R., and Horton, J.: "A microcomputer-based network for live performance." *Computer Music Journal* 2(3), pp. 24-29. 1978.
- Overholt, D. "Master's Thesis." MIT Media Lab, 2000. <http://www.media.mit.edu/~dano/matrix/>
- Ramakrishnan, C.: "PVD." Vocoder Framework. <http://www.mat.ucsb.edu/~c.ramakr/pvd.html>
- Verplank, B., Mathews, M., and Shaw, R.: "Scanned Synthesis." 2000.