

Prof. V.J. Manzo

Alternative Instruments for Music Education

Kean University, Montclair State University, Temple University
TMEA Conference 2009 – Wednesday, February 11, 2009
5:00 PM – 6:00 PM

e-mail: vj@vjmanzo.com

websites: www.vincemanzo.com | www.eamir.org

Introduction

Imagine a room where music is produced by touching the wall or the floor. Imagine a room where physical gestures that you've used since birth are mapped to notes and harmony. Imagine a room with new musical instruments designed to let individuals without any formal music training create and perform meaningful music.

EAMIR (Electro-Acoustic Musically Interactive Room) is an interactive music system that allows students and individuals with mild and profound disabilities to create a unique, tonal musical expression without the physical and technical limitations found in the performance of traditional acoustic instruments. EAMIR instruments are unmodified alternate controllers and sensors that connect with original software designed to allow these individuals to create music by using physical gestures that individuals are familiar with. The EAMIR instruments are accessible interfaces such that can be played by stepping on a foam floor tile, touching an LCD screen, waving a hand in the air. Each of these gestures produces a meaningful series of musical events or sounds that can be mapped to a single 7-note mode.

The software is limitlessly customizable and can be used to produce pitches melodically or harmonically. Teachers specify the tonic and mode and can easily perform with students that would otherwise have great difficulty performing and composing.

In 2007, the first Electro-acoustic Musically Interactive Room was proposed and created by Vincent Joseph Manzo.

In order to make EAMIR more accessible, all of the commercial controllers/interfaces are unmodified, making EAMIR primarily a software project. This allows the end-user to obtain any controller/interface associated with an EAMIR software title and simply plug it in and begin making music on any Windows or Macintosh computer. This software is freely available from <http://www.eamir.org>, EAMIR's website which also provides information regarding setup, installation, and use as well as where to purchase controllers and sensors. The source code is also available upon request allowing users to easily adapt the software for their personal needs.

Software Design

EAMIR's software is written primarily in Max/MSP/Jitter, or LISP. Data is received from each controller/sensor connected to a computer. Depending on the EAMIR software you are using, the typical EAMIR program will interpret the range of numbers received from a controller and normalize its output

to a desired range: the pitches of a scale across any number of octaves, the complete MIDI pitch range (0-127), velocity (0-127), etc. This is expressed in LISP as:

```
(defun normalize-list (data-list lowest-data-point the-scaler midi-lowest)
  (if (null data-list) ()
      (cons (abs (floor (+ (* (- (first data-list) lowest-data-point) the-scaler) midi-lowest)))
            (normalize-list (rest data-list) lowest-data-point the-scaler midi-lowest))))
```

And expressed in Max/MSP/Jitter as:

```
[scale lowest_#_received highest_#_received lowest_#_scaled highest_#_scaled]
```

With each stream of controller data being normalized, the typical EAMIR program will interpret some streams to control pitch and others to control velocity and other parameters. In order to give each program the ability to interpret pitches outside of an atonal context, the Modal Object Library, a collection of algorithms to control and define modality, is employed. This library, which V.J. Manzo created in 2006, contains numerous objects that allow tertian harmony to be expressed easily within EAMIR programs. The center of the Modal Object Library is the Modal_Change object which interprets all of the scale data for the major scale, harmonic minor scale, harmonic major scale, and melodic minor scale and each of the 7 modes associated with these scales (28 modes all together). From within each program, a user selects a tonic and mode using the traditional names (e.g. C Major, D Lydian, F# Mixolydian, etc.). The mode name equates to a pattern of whole steps and half steps representing the distance between each scale degree. When a tonic is selected, it is reduced to a pitch class (as a result of modulo 12) and the scale degree distances are applied to the tonic building a complete scale and establishing a tonal center. The table below represents the scale degree distances in which each integer is equivalent to a semitone.

	Major	Harmonic Major	Harmonic Minor	Melodic Minor
Mode 1	2 2 1 2 2 2 1	2 2 1 2 1 3 1	2 1 2 2 1 3 1	2 1 2 2 2 2 1
Mode 2	2 1 2 2 2 1 2	2 1 2 1 3 1 2	1 2 2 1 3 1 2	1 2 2 2 2 1 2
Mode 3	1 2 2 2 1 2 2	1 2 1 3 1 2 2	2 2 1 3 1 2 1	2 2 2 2 1 2 1
Mode 4	2 2 2 1 2 2 1	2 1 3 1 2 2 1	2 1 3 1 2 1 2	2 2 2 1 2 1 2
Mode 5	2 2 1 2 2 1 2	1 3 1 2 2 1 2	1 3 1 2 1 2 2	2 2 1 2 1 2 2
Mode 6	2 1 2 2 1 2 2	3 1 2 2 1 2 1	3 1 2 1 2 2 1	2 1 2 1 2 2 2
Mode 7	1 2 2 1 2 2 2	1 2 2 1 2 1 3	1 2 1 2 2 1 3	1 2 1 2 2 2 2

Once a tonal center has been established, incoming controller/sensor data that is received is normalized and checked against the notes of the selected mode. For example, if the normalized data yields the C# above middle C (MIDI Note 61), but the key of C Major has been selected by the user, the

note is reduced to a pitch class (modulo 12). Then the pitch class is checked against a list of data where the pitch class received acts as an address to yield the number to its right. This is expressed as:

0, 0; 1, 0; 2, 2; 3, 4; 4, 4; 5, 5; 6, 5; 7, 7; 8, 7; 9, 9; 10, 11; 11, 11;

The second block of data shows [1,0;]. The C# (1) will yield the pitch to its right (0), and thus any non-diatonic notes received will be filtered to diatonic notes. Once the notes has been filtered, it is returned to its proper octave designation.

With a tonic and mode established, the program calls on functions within the Modal Object Library that use tertian harmony to build chords and interpret functional harmony. Several objects in the library establish relationships between the modes allowing tonicizations and modulations, as well as perform certain types of real-time analysis. The control interface's design uses these functions in different ways, but the key mapping and pitch filtering is typically consistent. By default, all EAMIR programs are in the key of C Major, but contain menus for easily selecting new modes. Each program also contains the ability to store and recall presets.

Output

Most EAMIR programs output standard MIDI messages and use the host computer's internal MIDI core to synthesize notes. This allows the user to change MIDI parameters such as program and channel according to the General MIDI specification. The MIDI output of each program can also be routed to any software synthesizer (softsynth) or external synthesizer allowing for unlimited timbres to be used with the software.

All EAMIR programs have the option to record a performance as a standard MIDI file and, if applicable, a standard *wav* or *aiff* audio file. MIDI files can then be brought into any program that works with MIDI such as Pro Tools, Logic, and Garage Band, as well as notation programs like Finale and Sibelius. Audio files can be edited in similar programs that deal with audio such as Pro Tools, Logic, and Garage Band.

EAMIR Software

Lazy Guy

Lazy Guy is an interactive music system for composition and performance. Using a webcam, the software displays two windows on the screen. The first window shows the webcam view and allows a user to click on any point in the window to get the color of that point. The color of that point is displayed in a small window marked "Tacking this color". The software then maps out a perimeter for all instances of that color and builds a composite in the second window. The composite will resemble the shape and size of all instances of that color.

The orientation of the composite is then used to yield pitch and velocity. Typically, composites showing near the left of the window will trigger low pitches while higher ones (0-127) will sound as the

composite moves to the right. In the same manner, the vertical orientation of the composite will determine velocity; higher orientation on the screen will yield a higher velocity value (0-127).

Like all EAMIR software, these parameters can be changed and the software can be used to output the pitches of a given tonic and mode within any number of octaves. If the software is mapped to only one or two octaves, less precision is needed to play a scale since more points in the composite window will be associated with a given pitch.

The software outputs standard MIDI which it internally synthesizes. This allows the user to change programs according to the General MIDI specifications. The MIDI output can be also routed to any external synthesizer or softsynth. An additional feature allows the MIDI message to be converted to a sine wave produces a smooth Theremin-like sound. This audio is sent out through the host computer's default audio playback device which can be modified by click on the dac~ icon.

Lazy Guy is best performed with a laser pointer since its color is bright and can easily be turned off. Pointing the laser directly at the camera will cause refraction which will produce multiple pitches.

Monochrome

Monochrome is an interactive music system for composition and performance. Monochrome takes user input from a graphics tablet and generates a piece in reaction to the artist's gestures on the tablet. The software contains an embedded drawing program which responds to graphical table data. The amount of stylus pressure placed on the tablet will produce different types of shading (lighter if drawing with a black background and darker if drawing with a white background).

The software calculates user input such as pen orientation and pressure to control pitch and velocity. Pitch is determined on the horizontal axis identically to the way *Lazy Guy* functions and is can be mapped to any user-defined tonic and mode across any number of octaves. The amount of pressure placed upon the tablet will control velocity (0-127).

An additional feature allows users to enable common-tone modulation while performing. If selected, as the artists begins to fill the canvas, the software will look at which tonic and mode the user has selected and, using the Modal_Shift object from the Modal Object Library, will modulate to a tonic and mode sharing 6 of the 7 notes of the original tonic. It will also take the MIDI data transmitting on MIDI channel 1, by default, and route the data to an additional MIDI channel (up to 16).

The software outputs standard MIDI which it internally synthesizes. This allows the user to change programs according to the General MIDI specifications. The MIDI output can be also routed to any external synthesizer or softsynth.

An additional feature allows the MIDI message to be converted to a sine wave produces a smooth Theremin-like sound. This audio is sent out through the host computer's default audio playback device which can be modified by click on the dac~ icon.

Guitar EAMIR-o

Using the controller of the popular game Guitar Hero®, Guitar EAMIR-o allows the buttons of this controller to play the notes/chords of any diatonic scale.

The standard USB® Guitar Hero® controller has numerous buttons. In the Guitar EAMIR-o software, the first 4 buttons are mapped to the 8 notes of a user -selected diatonic scale with respect to

any tonic. Pressing the first button and holding the strum bar down will yield scale degree one; pressing the first button down and holding the strum bar up will produce scale degree two. The fifth button enables chord mode which causes each note to act as the root for the typical guitar bar chord voicing. The *back* button on the controller allows them to boost the sounding pitches up one octave. The *start* button doubles the scale degree/chord root at one octave below producing a bass voice. The whammy bar will send a pitchbend message causing any currently held notes to bend down a major second.

The software allows up to two controllers to be connected simultaneously. The MIDI data for controller one will transmit on MIDI channel one with the optional bass voice transmitting on channel two. Controller two will transmit data similarly on channels three and four.

The software outputs standard MIDI which it internally synthesizes. This allows the user to change programs according to the General MIDI specifications. The MIDI output can be also routed to any external synthesizer or softsynth.

DDR EAMIR

Using the controller of the popular game Dance Dance Revolution®, DDR EAMIR allows the buttons of this controller to play pre-recorded audio files and loop them. From within the software, a user simply loads up their original audio files (or use the default loops) into each of the playback slots. Each slot corresponds to one of the pads on the controller. A user then steps on each pad to begin playback of each file. By pressing the *select* pad, all of the loops will begin playing from their starting point. Pressing the *start* pad will stop playback of all loops. Additionally, the EQ and volume for each slot can be modified. Each file can also be played back in reverse.

All audio is sent out through the host computer's default audio playback device which can be modified by click on the dac~ icon.

EAMIR padKontrol

The EAMIR padKontrol system uses the Korg® padKontrol MIDI controller and PC or Mac compatible software to compose music. When the software is launched, the user can select between two modes of operation: Absolute Mode and Relative Mode. Both modes utilize the padKontrol's 16 touch-sensitive pads to create chords in some manner.

Absolute

In Absolute Mode, the first three rows of 4 pads will map to the tones of a chromatic scale beginning on C. The first pad will trigger the tone C, the second one C#, the third D, and so on. The fourth row of 4 pads will operate similarly, but instead of triggering tones, they will signify chord qualities. The first pad of the fourth row will enable a major chord voiced across several octaves to be built whenever a user presses one of the pads in the first three rows. If a user wants to play a d minor chord, he will press the second button in the fourth row to tell the software that the chord being built will be a minor chord. Then, when the user presses the third button in the first row, the d will be sent as the chord root and the chord will sound. The touch sensitive pads take the amount of pressure placed on each pad and yield a velocity value (0- 127) which will determine the volume of the synthesized sound.

Sustain

By default, the system is in Sustain On mode which means that chords will sustain indefinitely until a new chord is played. When a new chord is played, the old chords decays and only the new chord is heard. With Sustain Off, the chords will only last as long as long as the user is physically touching the pad. The controller is equipped with lights for each pad that illuminate when touched, so the user will know if he is holding down the pad. If two or more of the first 12 pads are held simultaneously, the latter of the pads will be the one that produces the chord. Pressing the first and second pads of the fourth row simultaneously will toggle between Sustain modes.

XY

Both Absolute and Relative mode make use of the X/Y pad at the left side of the controller. A tonic and mode must be selected before playing; this is accomplished in the same manner when in Relative Mode. By default, the key of C Major is selected. The pad then is used to play a monophonic melody in which the X value controls pitch classes of the specified mode, C Major, across a 2 octave range. The Y value controls velocity; a position held closer to the top will yield a higher velocity value. When there is no position held on the XY pad, the velocity will equal zero. This, therefore, yields a note-off state when the XY pad is not in use.

The octave displacement of the 2 octave range can be shifted manually from within the software. A sustain mode can be enabled by pressing the HOLD button on the controller. This will cause notes played on the XY pad to sustain until another XY coordinate replaces it. The velocity (Y) will only change when a new note is triggered.

A tapping mode can be accessed by selecting the tapping toggle within the software. In this mode, the tonic will sustain except when notes on the XY pad are played which replace the tonic. When the user stops playing the XY pad, the tonic will sustain once again. This allows the user to tap different points on the XY pad to achieve a unique sound.

Program Changes

The software can route the MIDI data to an external synthesizer, but it also implements General MIDI to achieve internal synthesis. The two knobs on the controller set the MIDI program values for both the XY pad and the 16 pads. These program changes will ultimately change the synthesis timbres.

Inversions

By clicking on the inversion buttons on the right, the user can have the pad generated chords played in different inversions.

Relative Mode

Relative Mode operates similarly to Absolute Mode, but In Relative Mode, the user selects a tonic and a mode before playing. The user may choose from a list of 12 pitch classes and the 28 diatonic modes of the major scale, the harmonic minor scale, the melodic minor scale, and the harmonic major scale (major scale b6). By default, the key of C Major is selected.

Each of the 4 pads in the first two rows is voiced according to the chords that naturally occur in our tertian system of harmony. When pressed, the first pad of the first row will yield a C Major chord,

the 1 chord in the key of C Major; the second pad of the first row will yield a d minor chord, the 2 chord in the key of C major. If a different tonality is selected, different tertian chords will be built.

When a pad is pressed and a chord is played, the software determines the chord function and quality as it relates to the selected mode and tonic, and looks for the chords that normally follow the selected function. When these are found, the lights beneath these pads will begin to blink. If a user pressed the 2 chord in C Major, pad 5 begins to blink and he can easily see that the 2 chord normally resolves to the 5 chords. If the 5 chord is pressed, the chords that normally follow the 5 chord will begin to blink. This blinking does not cause any change in sound, but provides the user with suggestions for their performance with regard to the function of chords in functional Western harmony.

Each of the 4 pads in the last two rows creates a secondary dominant (V/V, V/ii, etc.) for each of the chords created by pressing pads 1 – 8 (first two rows). The user can change this secondary dominant chord to a secondary leading tone chord, a minor subdominant or other borrowed chords that tonicize the 7 diatonic chords.

P5 Glove

The P5 glove controller sends 5 streams of continuous data for each of the five fingers on the glove. Data is also sent with respect to the glove's X Y and Z orientation. The EAMIR P5 Glove software uses Nicolas Fournel's P5midi application as a front-end to translate the finger bend data to chords from a user-defined tonic and mode. Each finger on the glove is divided into eight bend increments, each of which corresponds to a scale degree. A user can conceivably play a desired scale in ascending order across five octaves by bending each of the five fingers in order beginning with the thumb.

The glove can also be set to allow all fingers to function as one to control a single user-specified octave. In this mode, the glove yields chords derived from the chord function associated with each scale degree. Each chord is voiced across several octaves.

Buttons on the glove can be used to trigger modulation (common-tone, chromatic mediant, etc.) as well as make timbre changes. The glove also transmits X, Y, and Z data which can be used to control dynamics, arpeggiation tempo, and EQ respectively.

The software outputs standard MIDI which it internally synthesizes. This allows the user to change programs according to the General MIDI specifications. The MIDI output can be also routed to any external synthesizer or softsynth.

Slider

Slider is a touchscreen musical interface that converts graphical table data into a linear array of musical notes. The slider software is run on a touchscreen computer which allows table data to be entered by physically touching the graphical interface. Each point in the table is analyzed serially and corresponds to a note from a user-specified tonic and mode spanned across all octaves (MIDI values 0-127). Each point is read and the corresponding note is played back at user-defined rhythmic value which can be changed by touching a rhythmic value icon at the left. The tempo can be changed as well.

When a user touches the screen, the points are read and produce notes. The higher the line appears when it is read, the higher the note will sound. A keyboard appears beneath the table to reflect the pitch of the point currently being analyzed. By default, notes are read "up and down" that is,

from left to right to left. This option can be changed to have notes read up (from left to right only) and down (from right to left only).

The software outputs standard MIDI which it internally synthesizes. This allows the user to change programs according to the General MIDI specifications. The MIDI output can be also routed to any external synthesizer or softsynth.

Tiles

The EAMIR tiles take the amount force exerted into them and convert that energy it into music. Embedded in each foam floor tile, beneath the colored rectangle, is a sensor that measures force. Typically, the program is used to output the notes of the ser-defined tonic and mode from low to high depending on the amount of force exerted. The more pressure that is put on the pad, the higher the pitch will be.

Other uses allow the tiles to be used for pitch matching games and activities for memory reinforcement as well as extended options for musical performance. The tiles are moveable and may also be mounted vertically (attached to the walls) if desired.

The software outputs standard MIDI which it internally synthesizes. This allows the user to change programs according to the General MIDI specifications. The MIDI output can be also routed to any external synthesizer or softsynth.

Bibliography:

Brosius, Eric (Arlington, M. E., & Eran (Cambridge, M. (2008). *Patent No. 7,320,643*. United States of America.

Fournel, N. (2006, July). *P5midi*. Retrieved January 10, 2007, from <http://www.nicolasfournel.com/>:
<http://www.nicolasfournel.com/P5midi.htm>

Inc., K. (2005). Japan.

Manzo, V. J. (2007). *EAMIR: the Electro-acoustic Musically Interactive Room*. Retrieved January 1, 2009, from EAMIR.org: <http://www.eamir.org>

Manzo, V. J. (2006). *Implementing Modality in Algorithmic Composition*. in press.

Toyama, Motoki (Kobe, J. M., Shigehito (Kobe, J. O., Toru (Kobe, J. Y., & Tomoya (Kobe, J. (2001). *Patent No. 6,225,547*. United States of America.

Resources:

EAMIR – the electro-acoustic musically interactive room | www.eamir.org
or permalink: <http://www.vjmanzo.com/clients/eamir/>

The Modal Object Library – a collection of algorithms to control and define modality
http://www.vincemanzo.com/modal_change
or permalink: http://www.vjmanzo.com/clients/vincemanzo/modal_change/

Prof. V.J. Manzo – academic website | www.vincemanzo.com
or permalink: <http://www.vjmanzo.com/clients/vincemanzo/>