The effects of Microsoft Kinect-based Interface on Live Guitar Performance

BY: GEORGE RYAN PEREZ

PROJECT IN LIEU OF THESIS

CHAIR: DR. BENJAMIN LOK

ADVISOR: DR. JAMES P. SAIN

INTRODUCTION

The effects units in guitarists' live performance setup is a popular way of altering musicians' audio source sounds. Currently, the form factor for adjusting the parameters for most of these units consists of switches (e.g., stompboxes), potentiometers (e.g., knobs or faders), treadles (e.g., volume pedal), or a combination thereof. These form factors can be categorized into two types: binary control and continuous control. When guitarists play their instrument, the level of control available to them gets reduced to binary on/off operations on foot switches as the finer control required by knobs and faders over a continuous range become inaccessible during live performance.

This project aimed to bring that level of control back to the guitarist by leveraging motion sensing capabilities of Microsoft Kinect. Specifically, the final application, dubbed the "Kinect/EQ Interface", will attempt to enable a musician to change equalizer levels in real-time with acceptable accuracy while playing their acoustic guitar.

MOTIVATION

The genesis for this project can be traced back to the author's venture as a solo acoustic guitarist and singer. During live performances, the author would use a DigiTech JamMan Looper/Phrase Sampler pedal on stage to record, overdub phrases, and layer multiple loops to create the illusion of having multiple guitarists playing simultaneously.

However the pedal presented challenges whenever the author wanted to access the continuous controls at the top of the pedal while playing the guitar. For example, when the author needed to select a live-recorded phrase several memory blocks away, the operation required turning the Select knob which was not reachable while playing[1]. Other needs such as adjusting the loop and instrument levels were equally inaccessible[2].

Figure 1. DigiTech JamMan Looper/Phrase Sampler pedal

[1] Instead, a separate footswitch pedal was needed for this operation and it required tapping of the footswitch pedal several times, leading to what's colloquially known as "pedal tap dancing".

[2] For these, no footswitch pedals were available.

Hence, in this and other effects units, the current form factors for continuous controls on pedals, e.g., knobs, faders, etc., presented limitations while playing the guitar. This raises the question: outside of the treadle form factor, are there any other operations that are available to guitarists for triggering effects while their hands are pre-occupied besides stepping on a pedal?

BACKGROUND

This led to research into new musical interfaces. A promising discovery of this research is the usage of motion sensing devices for musical purposes by leveraging spatial gestures provided by the WiiMote, Kinect, webcam, Leap Motion, and other devices (Heap, 2013; Kaltenbrunner, 2011; Leese, 2014; Nusz & Sanderson, 2012). These examples can be categorized into either:

1. making the entire body a musical instrument or
2. using gestures to create or manipulate virtual music instruments (VMI)

The Microsoft Kinect was by far most popular device among this initial research and it seemed suitable for onstage performance due to its wide range as well as its availability, popularity, and commercial support. Hence further research into research literature was undertaken towards this direction where a similar dichotomy was found in recent examples (Berg, Chattopadhyay, Schedel, & Vallier, 2012; Fan & Essl, 2013; Gillian & Paradiso, 2012; Sentürk, Lee, Sastry, Daruwalla, & Weinberg, 2012; Yoo, Beak, & Lee, 2011). However, examples of enabling musicians for spatial gestures while playing their traditional instruments was not found.

Knowledge gained from this research include insight into designing gestural controllers, which was currently described as a non-trivial task due to the infinite number of combinations to choose from and how such combinations could be mapped. Related to this, mirroring gestures to existing hardware was deemed inappropriate (Gillian & Paradiso, 2012). Common amongst the research is the need for visual feedback. These lessons, as well as popular technologies used in projects (i.e., Ableton Live, Max/MSP, OSC, etc.) were taken into consideration during the design of the current project.

For this project, a novel characteristic is its focus on allowing musicians to continue playing their instrument in the traditional sense while enabling them to perform operations they could not have done before the advent of spatial gestures. By making continuous controllers more accessible using gestures, it would open up new possibilities to their live performance on stage – either technically or aesthetically – with useful and meaningful movements that fall within the norms of their performance. To prove this concept, this project will attempt to enable a musician to change the equalizer (EQ) bands in real-time with acceptable accuracy while playing his/her acoustic guitar by leveraging the motion sensing capabilities of the Kinect.

PROTOTYPE

Interviews and sketches

During the early phases of this project, 12 solo acoustic guitarists were interviewed to gather requirements and find out the current limitations in their live stage performances. (It was hoped that their perceived limitations would have commonalities amongst each other and would align with the author's needs.) These clients ranged from music faculty members to local and touring artists. The answers revealed that most of the performers wished they could add another dimension to their solo performance by playing multiple instruments. Related to this was another common desire: the ability to control volume and effects settings in real time while playing their instrument.

Changing effect settings – specifically, the equalizer – was chosen for this project for several reasons:

1. one can reasonably assume that most musicians have knowledge of EQs,
2. the form factor for changing EQ settings through knobs or faders matches the author's motivation,
3. no such treadle form factor currently exists out-of-the-box for changing EQ settings, and
4. EQs represent a general effect that can be translated to other effect units such as distortion or phasers.

These reasons align with the goal that this project would inspire the performer to apply the resulting interface in other aspects of their live performance. If users perceive the project to be useful as demonstrated with changing EQ settings, the long-term goal of performing with a Kinect on stage for continuous control with their instrument could be realized.

A technique that was used to gauge user opinions of product design early on was rapid prototyping. Sketches were drawn on index card sketches to show users different interface options, sets of possible gestures while playing a guitar, the Kinect locations that were being considered (i.e., in front of the performer, on top of the ceiling, behind the performer, etc.), the three types of control aspects of the application (enabling, selection, and control of knobs) and the options for each. (See Appendix for the scan of all sketches drawn during this project.)

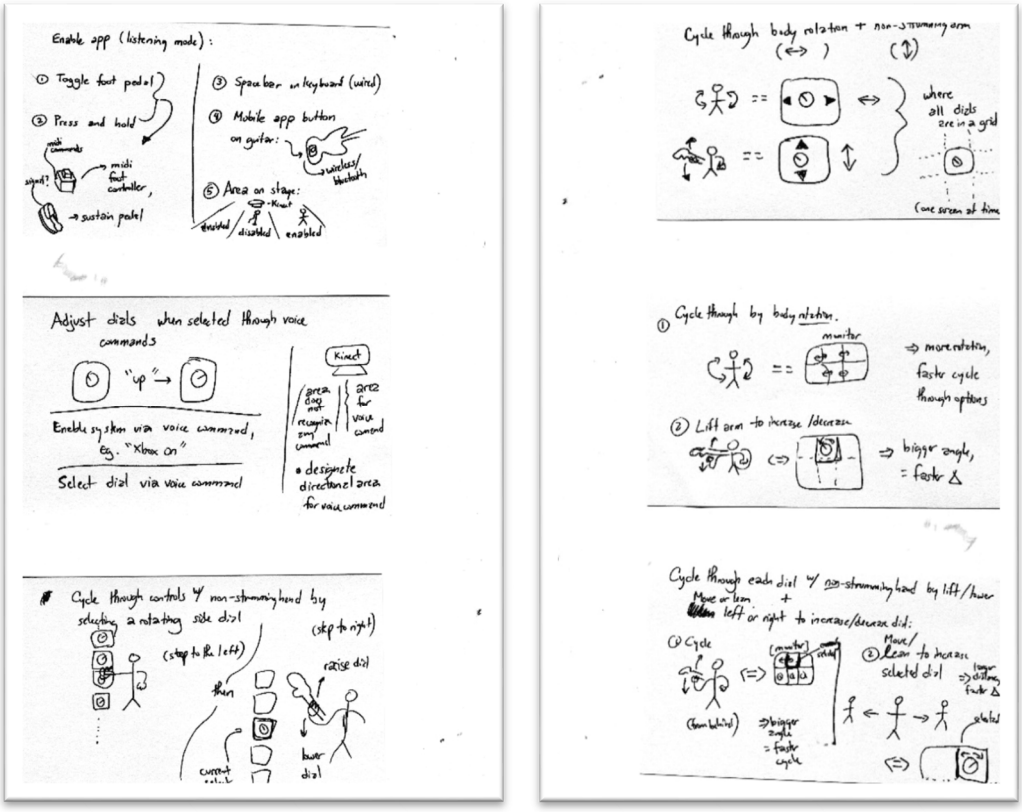


Figure 2. Scans of sketches showing control and selection schemes of EQ/volume Kinect functionality

All options were shown to a few students and colleagues in academia and in the entrepreneurship community, and each idea were individually scored.

A design decision made during this phase was the inclusion of a foot pedal to engage or enable the application: users expressed that enabling the application in such a way matches the current form factor for enabling effects units, and it was the manipulation of continuous effect parameters where the Kinect should be leveraged.

Once the scores were tallied, the initial set of candidate gestures were established from the highest scores. The final gestures for this project would be chosen from these, which were as follows:

1. Head roll
2. Body side lean
3. Guitar arm lift
4. Body rotation
5. Squats

User opinions expressed during the rapid prototype stage has shown that knobs were a familiar form factor for adjusting effect parameters, hence they were chosen for the interface. However, users were split between the one- or two-dimensional layouts. This was a concern for a layout of 4 or more knobs, and iterative feedback were continued throughout the development through the use of functional prototypes.

First functional prototype

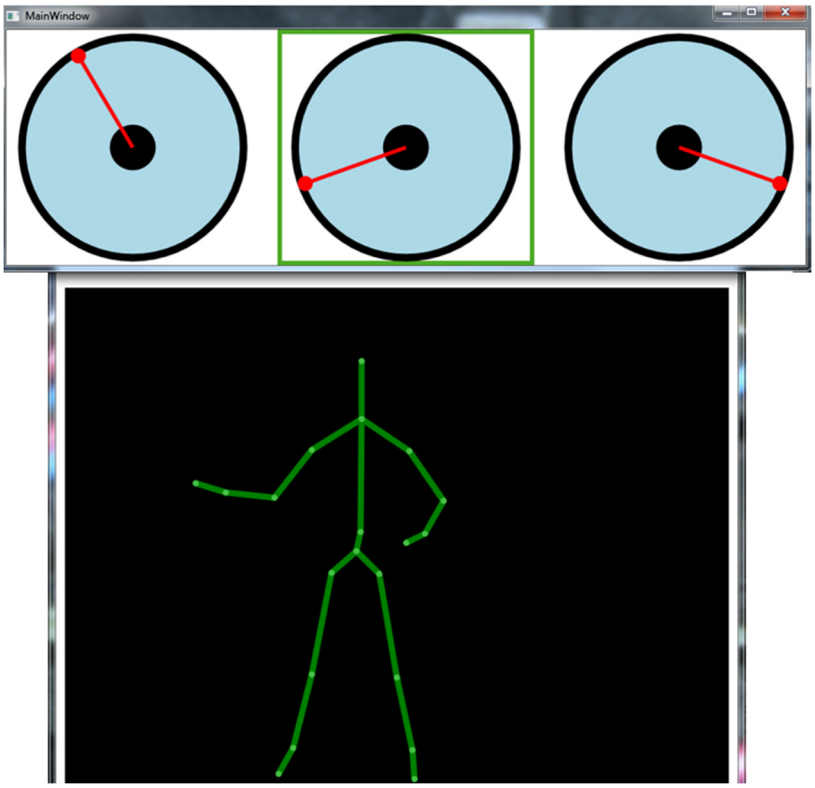


Figure 3. First functional prototype

A first functional prototype consisting of three knobs was written in WPF (C#) to whittle down the set of candidate gestures. The three knobs, known as the "three band EQ" which represent Bass, Mid and Treble, was the inspiration for this prototype since it is commonly seen on all sorts of musical gear and therefore having universal appeal.

The Kinect was not fully integrated for this version; i.e., it tracks their motion and displays a skeleton, but its functionality ended there. Instead, the Wizard-of-Oz prototyping technique (Green & Wei-Haas, 1985) was used whereby the functionality of a computer program was achieved by simulating it through the assistance of a human being, and in this case, the author. Specifically, the mechanisms for selection (i.e., moving to the next knob) and control (i.e., manipulating a selected knob) were simulated by using a wireless keyboard, which was operated by the author who would map user movements to the appropriate key commands for selection and control. This gestural mapping was determined when the interface was shown the users: users were asked to hold a make shift prop (e.g., a stick, a protracted umbrella, etc.) to simulate a guitar, and then they were asked what gestures were acceptable to them for selection and control. Based on user feedback from this round, the set of candidate gestures were whittled down to the following:

1. **Body rotation for knob selection**. Whenever the user rotates the makeshift prop along with their body towards the right or left, the knob selection would move from left to right or right to left, respectively.
2. **Guitar arm lift for knob control**. Whenever the user raises or lowers the left hand holding the guitar neck, the knob value would increase or decrease, respectively.

Second functional prototype

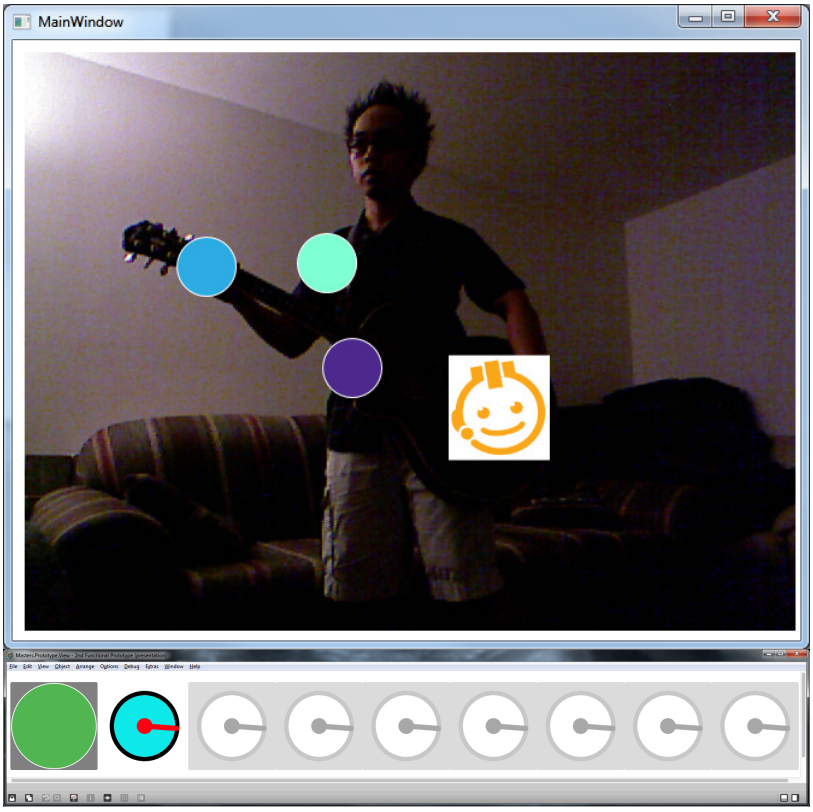


Figure 4. Second functional prototype

A second functional prototype consisting of 8 knobs was written in Max 6.1 to gather further user feedback such as the one- vs two-dimensional layout preferences. This was also the first version which enabled the user to control the interface using the Kinect directly. For this version, the joints used for gestures were as follows:

1. **Body rotation via left hand, left shoulder, center hip, and right hand tracking for knob selection**. Whenever the left hand holding the guitar neck was positioned past the left shoulder by rotating the body towards the right, the knob selection would move from left to right. Whenever the right hand was positioned past the center hip by rotating the body towards the left, the knob selection would move from right to left.
2. **Guitar arm lift via left hand tracking for knob control**. After the piano pedal was toggled to enable the current knob, whenever the left hand holding the guitar neck was raised or lowered, the knob's value would increase or decrease, respectively according to the current y coordinate of the left hand.

Pilot Test

A pilot test was conducted on the second functional prototype with 6 users prior to the user study.

A discovery made during pilot tests was the strong aversion towards the 8 knob layout due to its perceived complexity. After trying the prototype, most users felt overwhelmed by the number of knobs and asked for a simpler interface since there were "too many parameters to control", regardless of the layout choice. The one- vs two-dimension decision was thus upended by this request for a simpler layout, hence, the decision was made to revert back to the original 3-knob layout.

This version also revealed a significant weakness regarding any usage of joints near the guitar arm: the Kinect could not distinguish between a user's arm and the guitar arm. More critically, it was revealed that all Kinect joints obscured by the guitar were jittery and unreliable even after attempts to smoothen the joint positions. After research, it was theorized that the view obstruction (as well as the additional sheen) introduced by the guitar body and the guitar arm interfered with the IR depth sensor of the Kinect and its subsequent body position inference, thereby causing unreliable joint data in the area below the shoulders and above the knees. This made this version unusable by users, hence, this led to the design decision to capture data on three joints that were not obscured by the guitar: left shoulder, right shoulder, and head. Also, the set of candidate gestures was replaced with what would end up as the final gestures, which were as follows:

1. **Body rotation via left and right shoulder tracking for knob selection**. Whenever the left shoulder and right shoulder joints "touched" each other by rotating the body towards the right (or left), the knob selection would move from left to right (or right to left, respectively).
2. **Head roll (aka., head tilt) and/or body lean via head tracking for knobs control**. When the piano pedal was pressed and held, whenever the head rolled to the left or right, the knob's value would change according to the current x coordinate of the head. Additional body leaning would aid the user to reach the knob’s maximum and minimum values.

Lastly, a piano pedal was first used in this version as a toggle to enable/disable the ability to control the knob’s values. During the second functional prototype, users have suggested that the knob be adjustable only when they are currently pressing on the pedal. At the same time, it was suggested that

changing between knob selections be disabled while they are stepping on the pedal to prevent accidental knob switches. These suggestions were incorporated into the final design of the application.

FINAL DESIGN AND IMPLEMENTATION

To recap, a limited set of features were defined based on user interviews and prototypes. These features would then enable the user to assess the usefulness of the Kinect interface via changing EQ bands while playing his/her acoustic guitar. These features were as follows:

1. Selection of knobs through the Kinect
2. Control the value of knobs through the Kinect
3. Control the audio sound source based on the value of the knobs

Hardware implementation

Figure 5. Kinect/EQ Interface hardware setup

The hardware implementation of the Kinect/EQ Interface consisted of the following parts and signal flow:

- Sony Vaio VPCF115FM laptop using 2 USB 2.0 ports
- On-board internal soundcard w/ audio output headphone jack, and audio input microphone jack
- Kinect for Xbox 360 w/ USB cable and power supply
- M-Audio Oxygen8 v2 w/ USB cable
- Piano pedal
- Acoustic guitar with quarter inch output jack
- Guitar cable

- Quarter inch jack to 3.5mm adapter
- External speakers

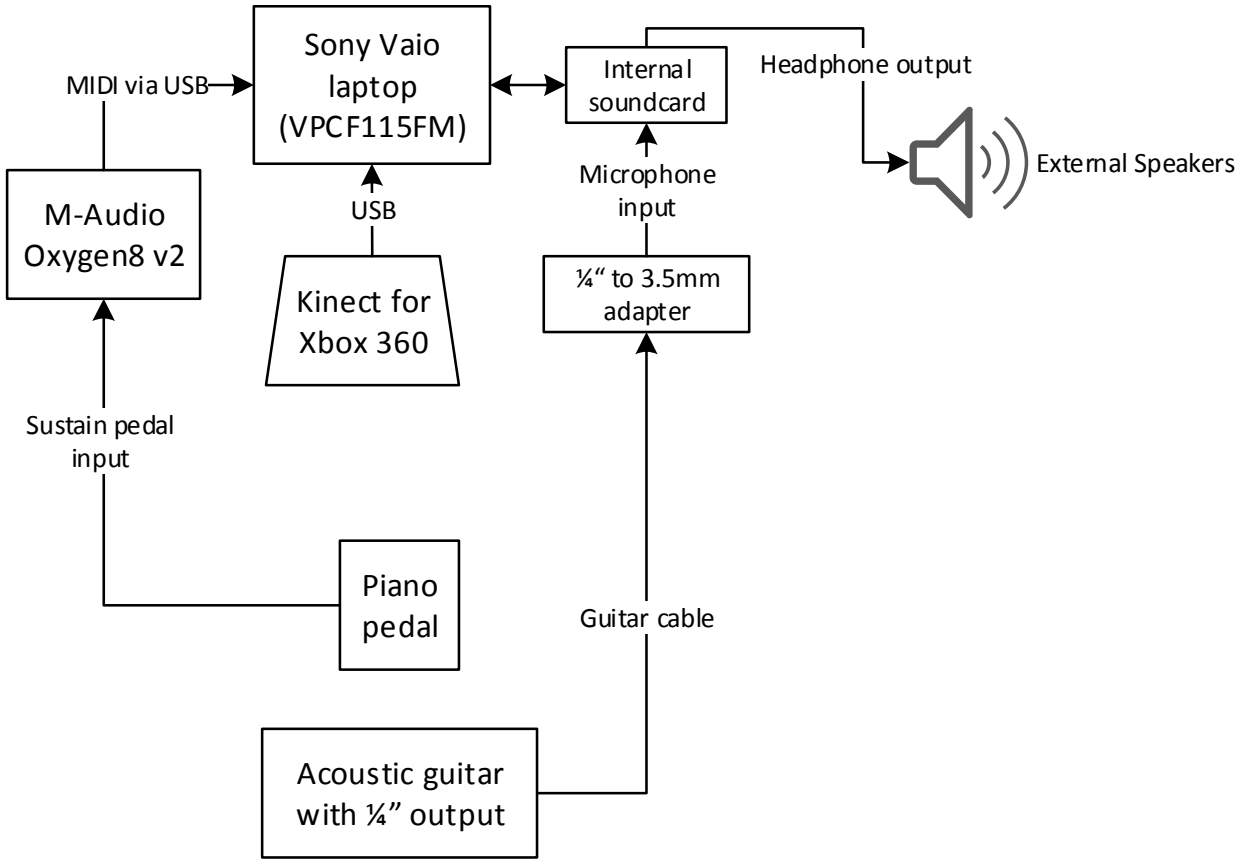


Figure 6. Kinect/EQ Interface signal flow

The M-Audio Oxygen8 v2 was used for two reasons: it was already owned by the author, and it converted the sustain pedal into MIDI commands which the laptop accepts via a USB connection. This component can be substituted by any hardware which accepts a piano pedal, converts it to MIDI, and sends it to the laptop for processing.

The internal sound card was used to simplify the hardware setup but with the drawback of lower fidelity input sound. This component can be replaced by additional hardware that maintains the fidelity of the input audio signal and can send it to the audio application in the laptop with the proper drivers.

Software implementation

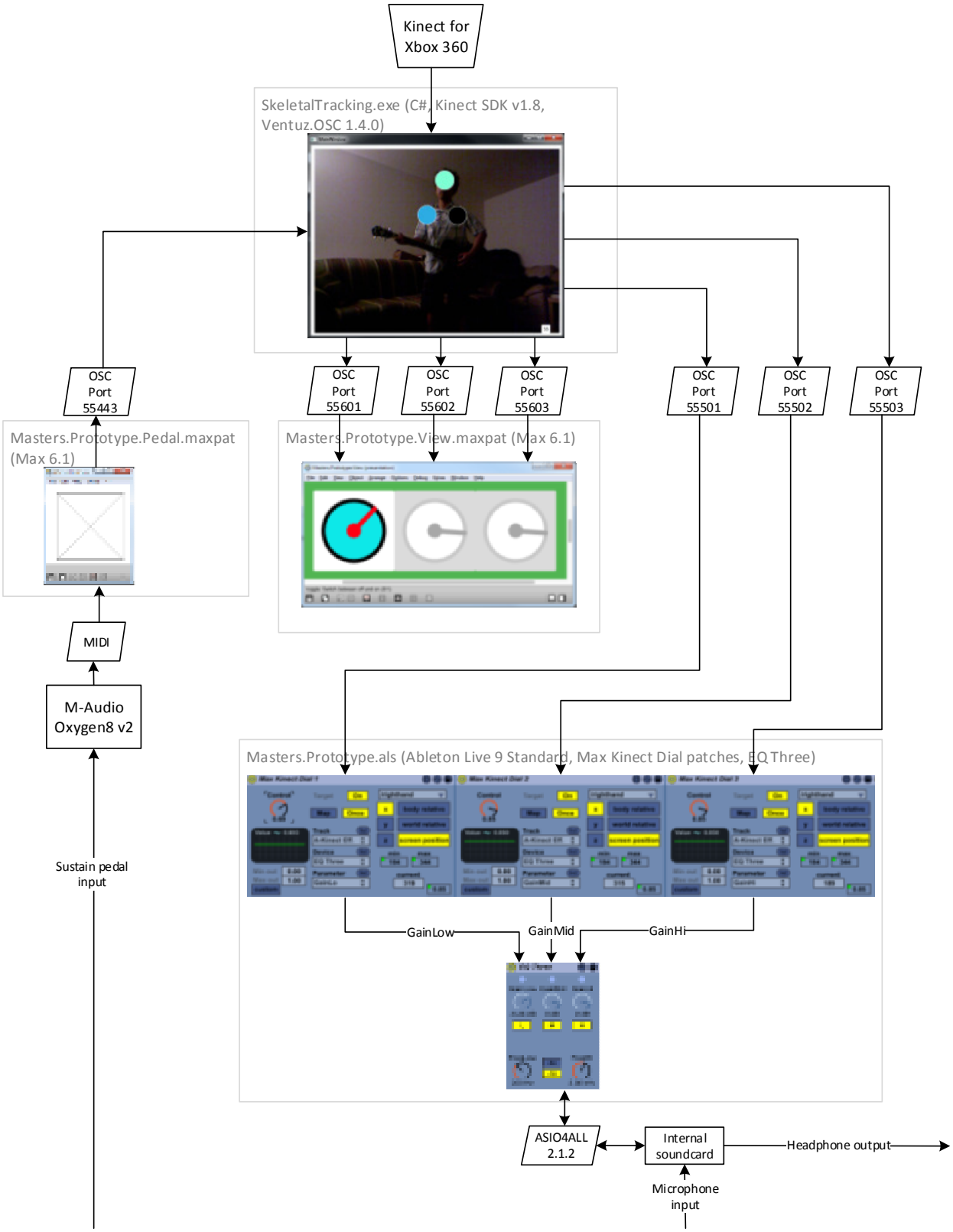


Figure 7. Kinect EQ/Interface architecture

The software implementation of the Kinect/EQ Interface consisted of the following parts and architecture:

- SkeletalTracking.exe (C# using Kinect SQK v1.8 and Ventuz.OSC 1.4.0)
- Masters.Prototype.Pedal.maxpat (Max 6.1)
- Masters.Prototype.View.maxpat (Max 6.1)
- Masters.Prototype.als (Ableton Live)
    - EQ Three (Ableton Live Audio Effect)
    - Min Max Init Rack (Ableton Live Audio Effect)
    - Max Kinect Dial 1, 2, and 3 (custom Max for Live patches)
- Open Sound Control
- ASIO4ALL 2.1.2

SkeletalTracking.exe (C#, Kinect SDK v1.8, Ventuz.OSC 1.4.0)

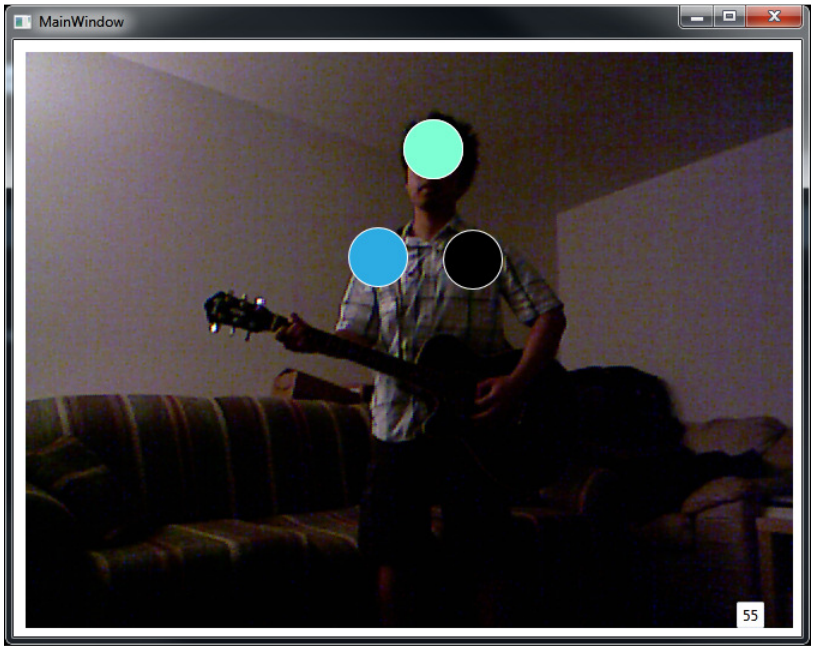


Figure 8. SkeletalTracking.exe

SkeletalTracking.exe was considered the main program as it contained the joint tracking via the Kinect SDK and corresponding business logic which affected all other software components of the Kinect/EQ Interface.

SkeletalTracking.exe used skeleton, depth, and color streams from the Kinect SDK v1.8 to track 3 joints: left shoulder, right shoulder, and head, which were superimposed in the bitmap by different-colored ellipses. For every frame cycle, these 3 joints' x and z depth point coordinate values were then used to implement the following workflow:

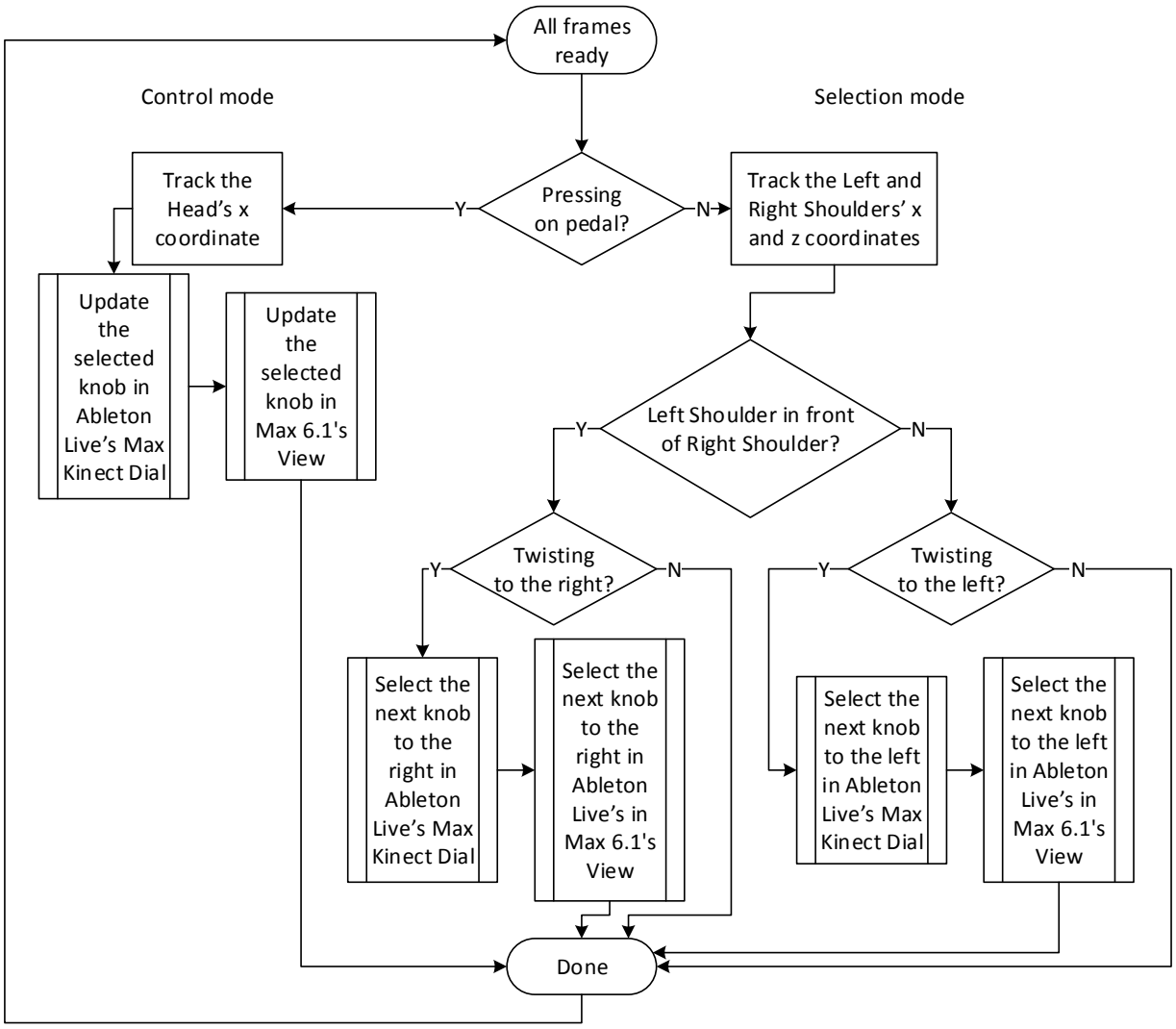


Figure 9. SkeletalTracking.exe workflow when all frames are ready

There were two main forks in SkeletalTracking.exe and this was determined by whether the user was pressing on the piano pedal.

When the user was not pressing and holding the piano pedal, the SkeletalTracking.exe entered what is known as Selection mode whereby the knobs on both Masters.Prototype.als (Ableton Live) and Masters.Prototype.View.maxpat (Max 6.1) were selected. As discussed, the gesture for selection was body rotation via left and right shoulder tracking. Hence, whenever the left shoulder and right shoulder joints "touched" each other by twisting the body towards the right, the knob selection would move to the immediate right. Likewise, whenever the left shoulder and right shoulder joints "touched" each other by twisting the body towards the left, the knob selection would move to the immediate left. The x coordinate values of the left and right shoulder joints were tracked to determine whether the user's shoulder were touching, whereas their z coordinate values were tracked to determine which shoulder was in front of the other, thereby distinguishing left vs right rotation.

This was implemented by keeping in memory an ordered list of OSC ports which correspond to the 3 knobs in both Masters.Prototype.als (ports 55501, 55502, and 55503) and

Masters.Prototype.View.maxpat (ports 55601, 55602, and 55603). These ports were also placed in a circular-array-like arrangement so that the rightmost knob would switch to the leftmost knob whenever the user twists to the right and goes out of bounds (and vice versa). Additionally, OSC commands were sent on port 55701 to change the corresponding color of the selected knob in Masters.Prototype.View.maxpat. This was done to make it clear to the user which knob was currently selected.

When the user was pressing and holding the piano pedal, SkeletalTracking.exe entered what is known as Control mode whereby the currently selected knob's value on both Masters.Prototype.als and Masters.Prototype.View.maxpat were updated. As discussed, the gesture for selection was head roll (aka., head tilt) and/or body lean via head tracking. Hence, while the piano pedal was pressed, whenever the user's head rolled to the left or right, the selected knob's value would change according to the current x coordinate of the head relative to the center of the camera. The x coordinate value of the head joint is tracked and sent to both Masters.Prototype.als and Masters.Prototype.View.maxpat which were interpreted as x screen positions. When the user was completely centered on the screen, the knobs were set to flat EQ settings, i.e., no dBs were cut or boosted from the signal. When the user leaned to the left, the knob's needle turned clockwise, which in turn cuts dBs from the selected knob's corresponding EQ band. Likewise, when the user leaned to the right, the knob's needle turned counterclockwise, which boosts dBs from the selected knob's corresponding EQ band. (Note: The maximum and minimum values of the EQ Three patcher in Ableton Live were unmodified and kept their defaults.)

When the user was pressing and holding the piano pedal in Control mode, the currently selected knob stayed in place, i.e., the knob to the immediate right or left cannot be selected, until the user stopped pressing on the piano pedal.

Masters.Prototype.Pedal.maxpat (Max 6.1)

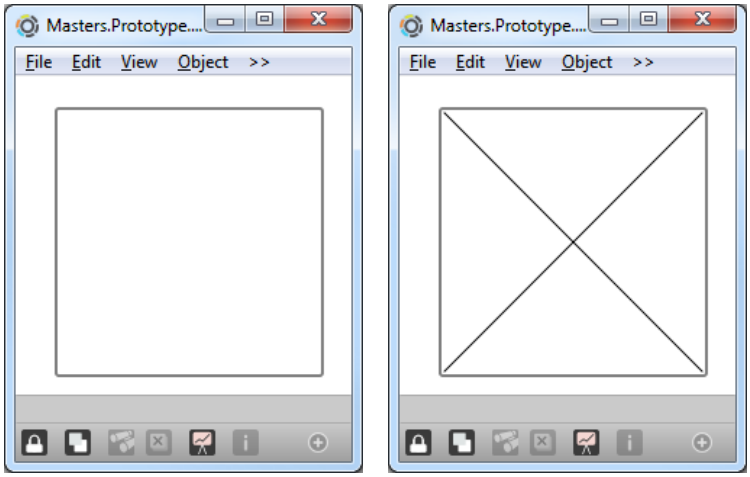


Figure 10. Masters.Prototype.Pedal.maxpat when user is not pressing or pressing on the pedal, respectively

The Masters.Prototype.Pedal.maxpat patcher is a helper application that detected whether the pedal was pressed by parsing a MIDI signal that originated from the piano pedal. It then toggled a flag whose value was then sent via OSC (port 55443) to SkeletalTracking.exe. SkeletalTracking.exe then used this flag's value to enter either Selection or Control mode. (See Appendix for the patch mode of this patch.)

Masters.Prototype.View.maxpat (Max 6.1)

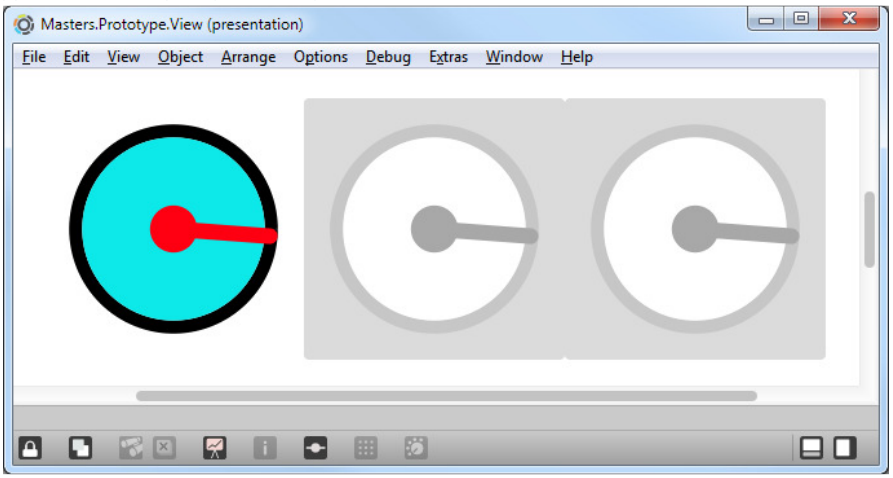


Figure 11. Masters.Prototype.View.maxpat when user is not pressing on the pedal

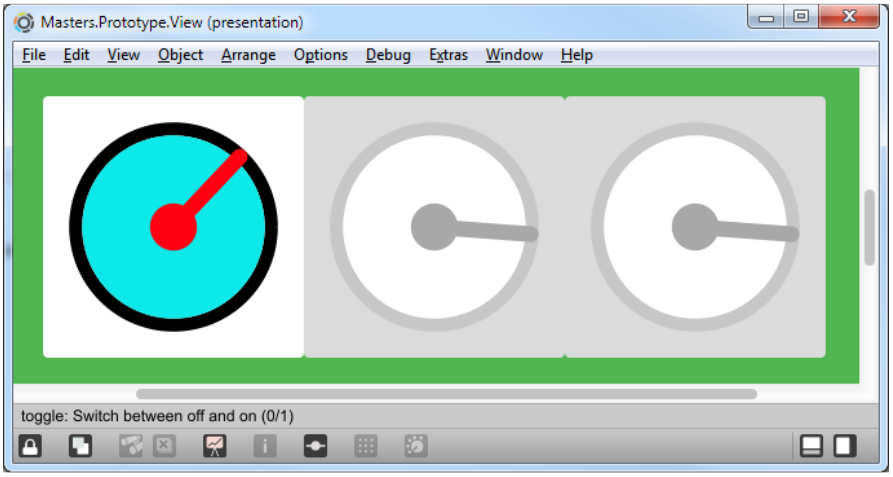


Figure 12. Masters.Prototype.View.maxpat when user is pressing on the pedal

The Masters.Prototype.View.maxpat patcher is a larger view of the corresponding knobs in the Masters.Prototype.als Max for Live patchers. This patcher was created for two reasons: (1) the knobs in the Max for Live patchers were too small from a distance and (2) to make it clear which knob was currently selected. (See Appendix for the patch mode of this patch.)

The logic behind the Masters.Prototype.View.maxpat patcher was the same logic that was used in the Max for Live patchers, except that the knobs were enlarged so that they could be viewed from an acceptable distance as determined by the Kinect sensor. The patcher listened via OSC for commands from SkeletalTracking.exe to determine which knob was selected in the ordered list as well update the value of the currently-selected knob. Additionally, it listened on port 55701 to set the currently selected knob's color accordingly (and resetting the colors of the other knobs).

Masters.Prototype.als (Ableton Live)

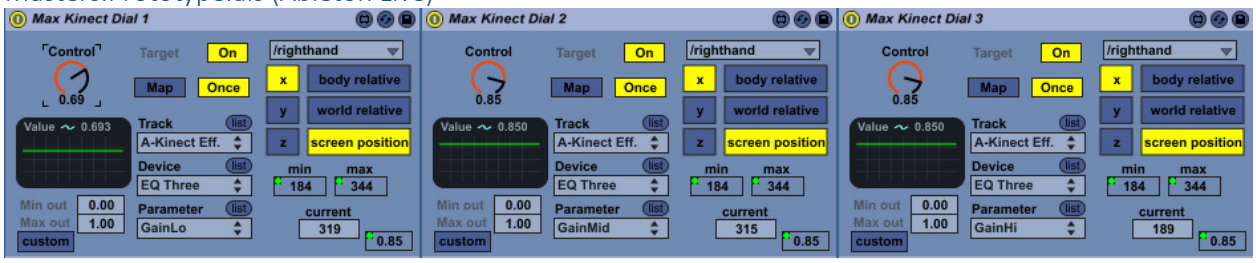


Figure 13. Max Kinect Dial 1, 2 and 3 custom Max For Live patchers in Masters.Prototype.als

The Masters.Prototype.als Ableton Live Set consisted of three patchers: (1) EQ Three, (2) Audio Effect Rack, and (3) custom Max For Live patchers. The EQ Three and Audio Effect Rack were out-of-the-box Ableton Live patchers and were left unmodified. Max Kinect Dial 1, 2 and 3, however, were Max for Live patchers that were originally part of the Ableton Live Example Set from Synapse (Challinor, 2011). The Max Kinect Dial patchers were copied and modified so that Max Kinect Dial 1, 2 and 3 were each listening on a different port: 55501, 55502, and 55503, respectively. The patchers were also defaulted to interpret x coordinate screen positions. Lastly, an input box was added to accommodate default dial values. All other settings were left as they were from Synapse. (See Appendix for the patch mode of this patch.)

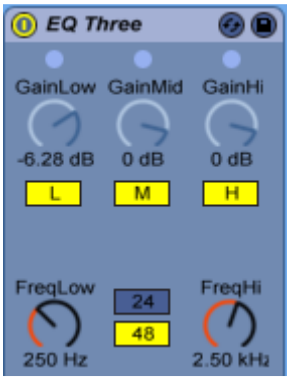


Figure 14. EQ Three in Masters.Prorotype.als

The Max Kinect Dial 1, 2 and 3 knobs were mapped to the GainLow, GainMid, and GainHi knobs in EQ Three, respectively. When the Max Kinect Dial 1, 2 and 3 knobs' values were updated via the OSC command from SkeletalTracking.exe's head tracking, the corresponding knobs in EQ Three get updated accordingly, thereby affecting in real time the audio sound source based on the value of the knobs.

The Min Max Init Rack was an Audio Effect Rack patcher that was mapped and used solely to set the min, max, and initial dial of all 3 Max Kinect Dial patchers. Hence, as a convenience patcher, this patcher could be removed and the audio signal path would not be affected.

Open Sound Control

Open Sound Control, or OSC, is a protocol for networking music and other multimedia devices similar to MIDI but with greater accuracy, speed, and flexible addressing over a network connection. This was used as the communication pipeline between all major components of the Kinect/EQ Interface. OSC messages were all addressed to localhost but on different ports so as to distinguish between components.

SkeletalTracking.exe read and sent OSC commands via the Ventuz.OSC 1.4.0 assembly. Max 6.1 and Max for Live patches read and sent OSC commands via the udpsend and udpreceive objects, respectively.

ASIO4ALL driver

The ASIO4ALL 2.1.2 driver ("ASIO4ALL - Universal ASIO Driver For WDM Audio," 2014) is a Universal ASIO driver for WDM Audio and was used as the Audio Device in Ableton Live. This driver bears a brief mention in this project because it produced a significantly lower latency than the audio driver originally installed in the laptop. During a pilot test, the original audio driver was deemed unacceptable by users due to the noticeable delay between the guitar and the resulting audio output. Without the ASIO4ALL driver, users would have been too distracted by the delay in sound and therefore unable to properly assess the Kinect/EQ Interface.

STUDY

A long-term goal of this project was to assess whether the Kinect/EQ Interface would be used in a real live performance on stage, and one predictor of actual usage was users' perceived usefulness of a new application. In order to measure the perceived usefulness of the Kinect/EQ Interface, an IRB-approved study was conducted at the University of Florida School of Music Building, Room 232. The study spanned for 3 weeks and had a total of 29 participants.

Population

Participants were 29 adult grad and undergrad students who were recruited from the University of Florida and adult residents from Gainesville, Florida. Candidates were screened and culled to ensure that all participants knew how to play basic chord progressions on the guitar. Each participant was compensated $10 for their participation in the study.

Procedure

All users followed the following IRB-approved procedure.

After a 5-minute tutorial, the user was asked to play a guitar with the Kinect/EQ Interface to complete 3 tasks totaling 20 minutes. For the first 10 minutes, the user was asked to adjust the sound quality to their liking using the Kinect and guitar. For the next 5 minutes, the user was asked to rehearse a song or chord progression of their choice in private. And for the last 5 minutes, the user was asked to perform the song or chord progression in front of the investigator and a video camera. At the end of the study, the user was asked to fill out a questionnaire.

Metrics

The metrics for this study was collected via a questionnaire containing 10 questions which were designed to measure the perceived usefulness of the Kinect/EQ Interface. All 10 questions used a Likert scale where 1=Strongly Disagree, 4=Don't Agree Or Disagree, and 7=Strongly Agree. The questions were derived from the Technology Acceptance Model (TAM) paper (Davis, 1993) which contained a questionnaire that measured perceived usefulness. (See Appendix for the questionnaire used in this project.) The TAM also concluded that perceived usefulness has a more positive influence than ease of use towards actual usage.

Results

Participants = 29, Mean = 51, Std Dev = 5.76, Range = 22, Minimum = 41, Maximum = 63, Confidence Interval (95.0%) = [48.81, 53.19]

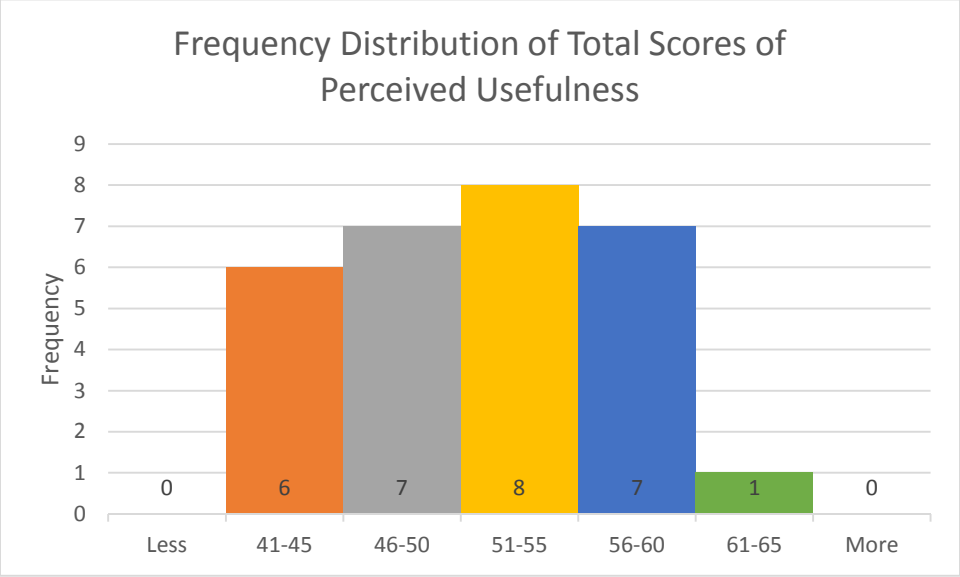


Figure 15. Histogram of Perceived Usefulness Total Scores

Discussion

User comments at the end of the questionnaire revealed insights regarding the Kinect/EQ Interface's perceived usefulness distribution.

Most users have reported to have difficulty when switching knobs by twisting their body in Selection mode. Users have said that the "moving of shoulders…[was] really difficult to pull off in a live setting", were concerned that "shoulders have to be pronounced", "felt awkward when actually performing", and that it was "somewhat cumbersome", etc. Conversely, users also reported that changing the knob values using head tracking in Control mode was satisfactory, with most users saying that it "worked well" and relatively better than twisting their body.

This suggested that the Selection mode would be a candidate area of improvement for future versions. Instead of making the users twist their body, perhaps the shoulder tracking could be used similarly to the head tracking. One could conceive and test other gestures which leverage the head and shoulder tracking used in the Kinect/EQ Interface, since these three joints were not obstructed by the guitar.

Some have expressed the desire to spend more time and get more acclimated with the system so that they can use it more effectively, with a few mentioning that the time allotted did not allow them to use the interface to its full potential. This either suggests that they actually wanted to use it at a later time to achieve its full potential (as some have indicated) or that the interface wasn't intuitive enough. At any rate, this also suggests that the usefulness rating of the interface was uncertain for these users.

There was some concern over the multitasking involved with the interface. With more movements, the multitasking aspect may "detract the ability of the performer to express his/herself emotionally" as well as "take focus away from playing". Some have requested that instead of two moves, one movement could be used in both Selection and Control modes since the pedal already distinguished between the

two modes. On the other hand, a few have expressed appreciation that the "knobs were more accessible" while performing and that it was a "fantastic tool for solo performers who already have their hands full." Hence, the multitasking aspect of the interface seemed mixed amongst the participants.

One user has mentioned that leaning the head while singing would be "difficult to use with a standard boom mic stand as it would take you away from the microphone." This would suggest replacing head tracking with another gesture for users who sing as well as play the guitar simultaneously. Related to this, some have expressed concern that the pedal constrained movement to one spot on the stage. Others also said that the requirement of facing the Kinect constrained the orientation to one position.

Users in general have positive reaction to the idea, saying that it's a "clever concept", "a great idea, and "has a lot of potential". Some has said they "loved experimenting quickly" with the interface and it "inspires creativity". Many have wanted other effects other than EQ such as phasers, distortion, chorus, etc. A few have appreciated its "clean interface" and "user friendliness".

FUTURE WORK

There is ample opportunity for improvements in the Kinect/EQ interface.

Since this project started before Kinect for Xbox One was released, the source code could be ported to the latest Kinect v2 SDK and tried on the newer hardware. Similarly, this project was developed using the Kinect for Xbox 360 hardware instead of the commercial Kinect for Windows hardware whose features could have been tested in this project (e.g., "near mode").

The Kinect EQ/Interface currently uses Max For Live patches in Ableton Live that were mapped to the EQ Three Audio Effect and can be expanded to use other effects or VST plugins. Similarly, Ableton Live could be removed from the architecture altogether and work solely within the Max 6.1 project space, which offers its own expansions for live performance and composition.

The Kinect EQ/Interface has not been taken to a live performance setting such as a venue or concert hall where lighting may affect the Kinect performance. User opinions in this outside-the-lab context would be a worthwhile to collect and analyze.

CONCLUSION

This project started with the observation of continuous controls' inaccessibility during performance due to its existing form factors, and the Kinect/EQ Interface was developed and studied to determine whether spatial gestures were a suitable replacement by measuring its perceived usefulness. Through iterative feedback and careful design decisions, the Kinect/EQ Interface enabled users to adjust and/or trigger effect parameters in real time while performing their instrument as opposed to interrupting their performance. It is hoped that the work and findings of this project would serve to impart lessons and inspire others to experiment with other means of enabling the performer to do more while their hands are already occupied.

REFERENCES

ASIO4ALL - Universal ASIO Driver For WDM Audio. (2014). from http://www.asio4all.com

Berg, T., Chattopadhyay, D., Schedel, M., & Vallier, T. (2012). *Interactive music: Human motion initiated music generation using skeletal tracking by Kinect*.

Challinor, R. (2011). SYNAPSE for Kinect. from http://synapsekinect.tumblr.com/post/6307790318/synapse-for-kinect

Davis, F. D. (1993). User acceptance of information technology: system characteristics, user perceptions and behavioral impacts. *International Journal of Man-Machine Studies, 38*(3), 475-487.

Fan, X., & Essl, G. (2013). Air Violin: A Body-centric Style Musical Instrument. *Ann Arbor, 1001*, 48109-42121.

Gillian, N., & Paradiso, J. A. (2012). *Digito: A fine-grain gesturally controlled virtual musical instrument.* Paper presented at the Proc. NIME.

Green, P., & Wei-Haas, L. (1985). *The rapid development of user interfaces: Experience with the wizard of oz method.* Paper presented at the Proceedings of the Human Factors and Ergonomics Society Annual Meeting.

Heap, I. (2013). The Gloves Project. from http://theglovesproject.com/about-the-project

Kaltenbrunner, M. (2011). Therenect - Kinect Theremin. from http://vimeo.com/17330186

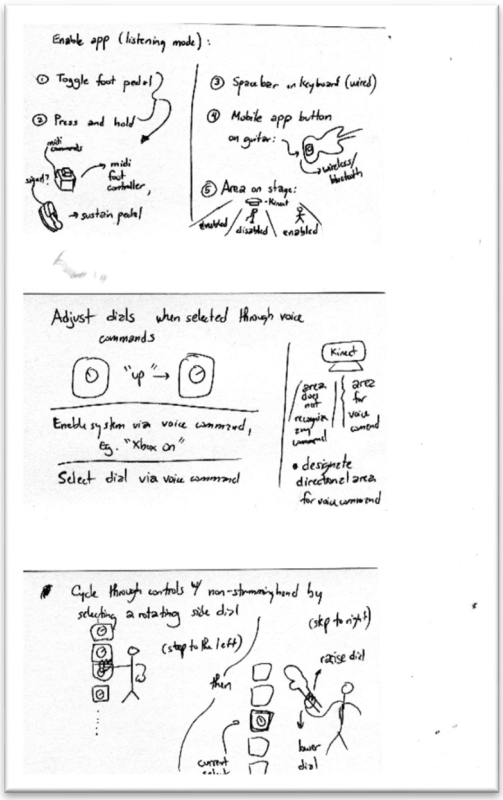
Leese, A. (2014). Leap Turntable - Prototype. from https://www.youtube.com/watch?v=rbyXG_Q29Tg

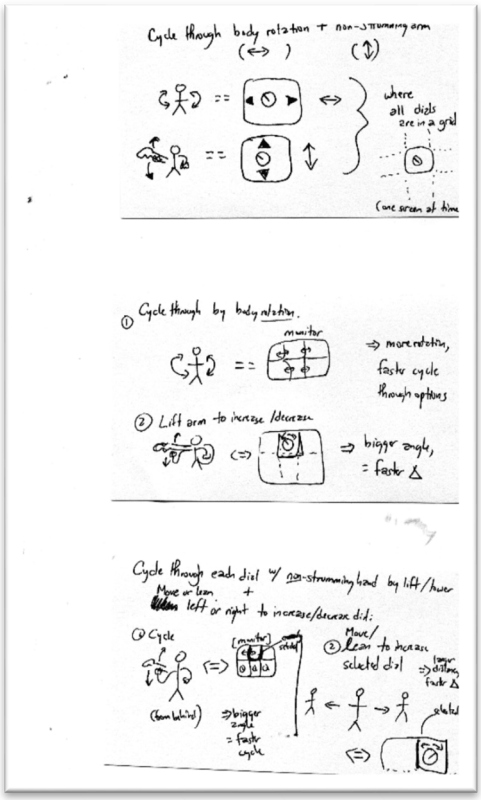
Nusz, J., & Sanderson, P. (2012). The V Motion Project. from http://vimeo.com/45417241

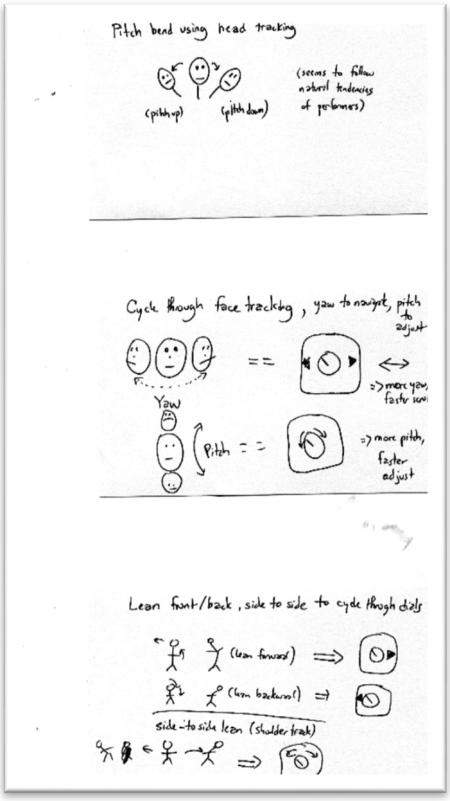
Sentürk, S., Lee, S. W., Sastry, A., Daruwalla, A., & Weinberg, G. (2012). Crossole: A gestural interface for composition, improvisation and performance using kinect. *Proc. NIME'12*.

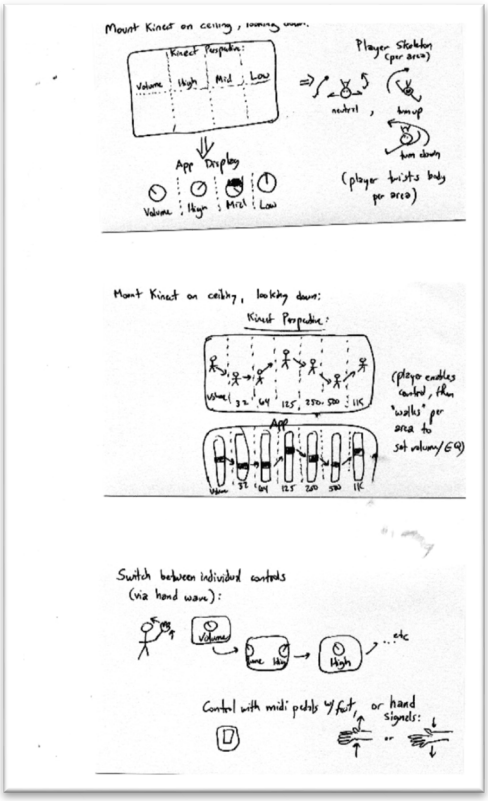
Yoo, M.-J., Beak, J.-W., & Lee, I.-K. (2011). *Creating musical expression using kinect.* Paper presented at the Proc. NIME.

APPENDIX


Enable app (listening mode):
① Toggle foot pedal
② Press and hold
③ Spacebar on keyboard (wired)
④ Mobile app button on guitar:
⑤ Area on stage:
Adjust dials when selected through voice commands
"up"
Enable system via voice command, Eg. "Xbox on"
Select dial via voice command
Kinect
area for voice control
designate directional area for voice command
Cycle through controls w/ non-strumming hand by selecting a rotating side dial
(step to the left)
(step to right)
then
raise dial
lower dial


Cycle through body rotation + non-strumming arm
where all dials are in a grid
(one screen at time)
① Cycle through by body rotation.
monitor
⇒ more rotation, faster cycle through options
② Lift arm to increase/decrease
⇒ bigger angle, = faster Δ
Cycle through each dial w/ non-strumming hand by lift/lower
+
Move or lean left or right to increase/decrease dial:
① Cycle
(turn behind)
⇒ bigger angle = faster cycle
② Move/lean to increase selected dial
selected


Pitch bend using head tracking
(pitch up)
(pitch down)
(seems to follow natural tendencies of performers)
Cycle through face tracking, yaw to navigate, pitch to adjust
Yaw
Pitch
=> more yaw, faster scroll
=> more pitch, faster adjust
Lean front/back, side to side to cycle through dials
(lean forward)
(lean backward)
side-to-side lean (shoulder track)


Mount Kinect on ceiling, looking down:
Kinect Perspective
Volume
High
Mid
Low
Player Stance (per area)
neutral
turn up
turn down
(player twists body per area)
App Display
Mount Kinect on ceiling, looking down:
Kinect Perspective:
(player cradles control, then "walks" per area to set volume/EQ)
App
Switch between individual controls (via hand wave):
etc
Control with midi pedals w/ foot, or hand signals:

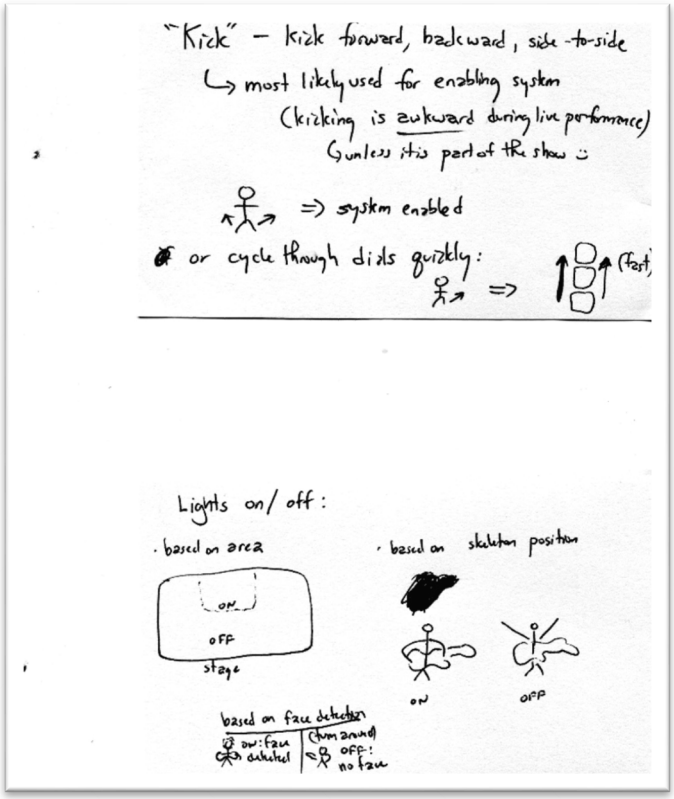


Figure 16. Scans of all sketches

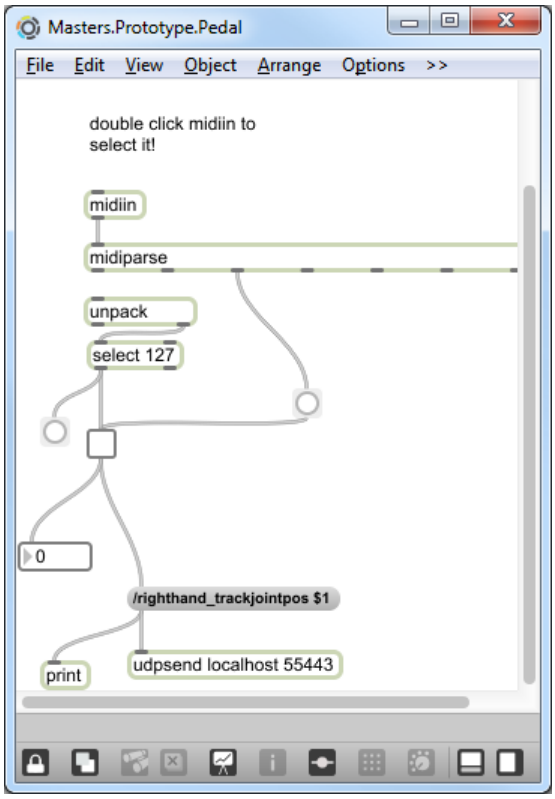


Figure 17. Masters.Prototype.Pedal.maxpat patch

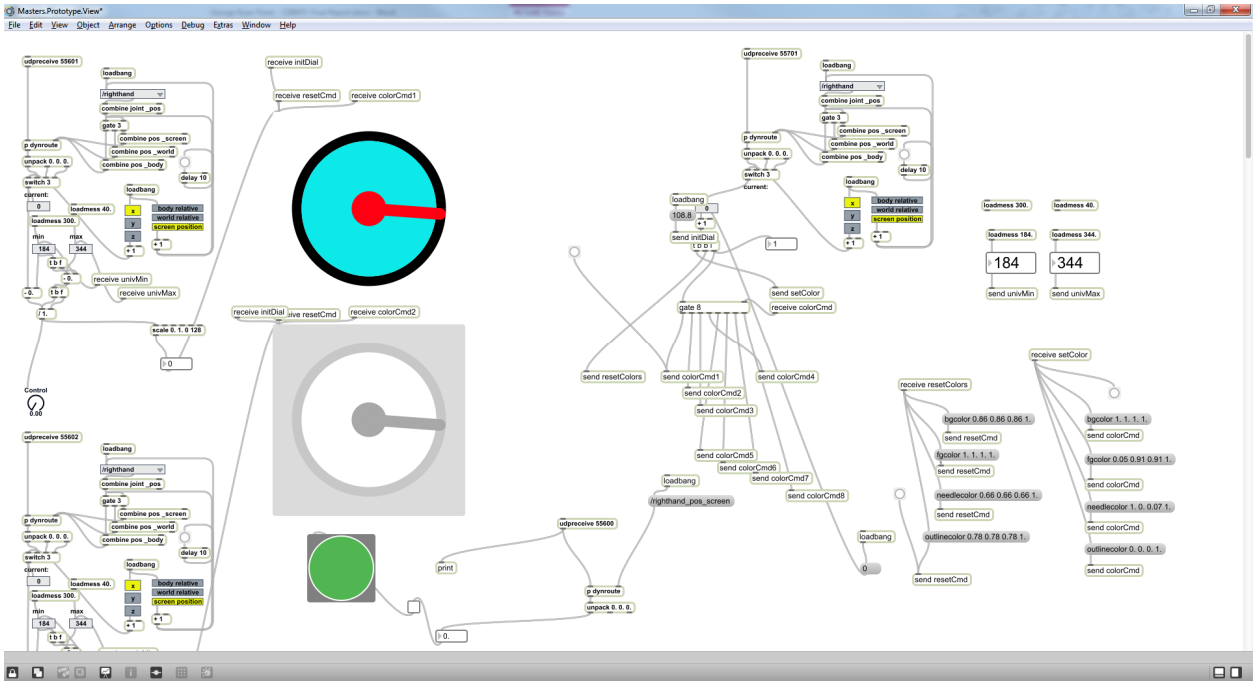


Figure 18. Masters.Prototype.View.maxpat patch

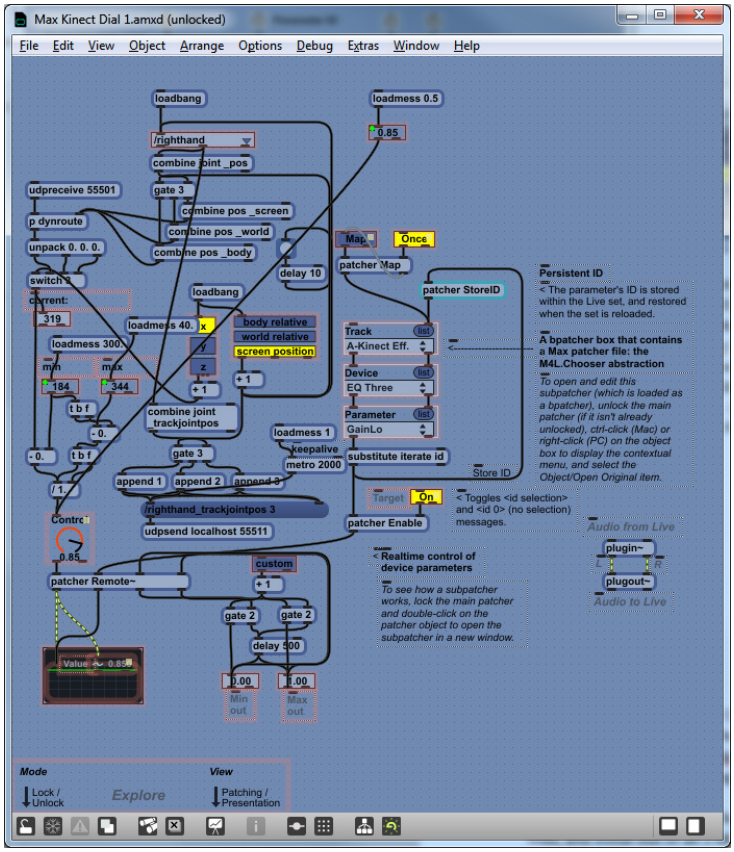


Figure 19. Max Kinect Dial 1 patch, originally taken from Synapse and edited

Post-study Questionnaire

On a scale from 1-7, (1=strongly disagree and 7 = strongly agree), describe each of the items:

1. Using the Kinect interface improves the quality of the work I do
2. Using the Kinect interface gives me greater control over my work
3. The Kinect interface enables me to more facile in my playing
4. The Kinect interface supports external control aspects of my performance well
5. The Kinect interface increases my creative output
6. The Kinect interface improves my concert performance
7. The Kinect interface allows me to be more creative than would otherwise be possible
8. The Kinect interface enhances my effectiveness to express my musical ideas before an audience
9. The Kinect interface makes it easier to interface with external devices
10. Overall, I find the Kinect interface useful in my creative work and performance