MAKER: Painting Pitches

Emily M. Meuer
Erin A. Kern, University of St. Thomas
Michaela Andrews
Amanda Tenhoff, University of St. Thomas

Amanda Tenhoff is an undergraduate student at the University of St. Thomas, majoring in mechanical engineering. She is a research student in the Playful Learning Lab.

Ms. Kristen Andrews, University of St. Thomas
Ms. Paige Huschka, Playful Learning Lab
Elena M Ryan
Mr. Luke Tozour
Dr. AnnMarie Polsenberg Thomas, University of St. Thomas

AnnMarie Thomas is a professor in the School of Engineering and the College of Business at the University of St. Thomas where she is the co-founder of the UST Center for Engineering Education. Her research group, the Playful Learning Lab, focuses on engineering and design education for learners of all ages.

Painting Pitches: Connecting Sound and Sight with Processing

Abstract

The following paper presents a system for creating real-time visuals based on multiple simultaneous vocal inputs. The goal of this work is to augment live musical performances by visually conveying the spirit and structure of the piece. Using the Processing programming language, sounds are analyzed and return pitch and amplitude as numerical values. Visuals representing pitch and amplitude for each of the musicians are created in real-time and are projected concurrently with the live musical performance. This process is demonstrated by work presented here with a professional 8-voice ensemble. As part of this project, a library of functions is being created and shared to allow others to implement similar productions.

Introduction

The initial goal of this work is to create a visual counterpart to the on-tour performances of Cantus, an *a cappella* men's vocal ensemble. The artistic goal of the work is to explore the connections between sight and sound and to add musically relevant and engaging visuals that would react to the singers' voices in real time. A secondary goal of the efforts is to create these functionalities in a way that can easily be modified and shared by other users.

The Processing language is widely used because of its substantial range of powerful yet easy-to-learn artistic libraries[1]. Processing was our choice for this project because of the appeal of its visuals and the strong support community surrounding it[2]. Processing also has several libraries devoted to sound generation and analysis, which we knew would be crucial to our project. However, we realized early on in our work with Processing that it did not include the capability to detect pitch, a vital part of our project. We wrote our own class for this and eventually found the Beads library[3], which allowed us to get pitches for multiple different inputs simultaneously.

For the first workshop performance using "Painting Pitches", we were able to use these tools to construct four very different visual programs to accompany stylistically different songs. After performing for a live audience and receiving useful feedback, our team continues to work on the functionality of our library and making it accessible to any Processing user.

Background

There is a long, rich history of work exploring the visualization of sound. For example, Smith et.al[4], Balandra et. al[5], and Malinowski[6] visualized musical scores by changing the conventional staff notation into something that those with limited music composition experience could view and understand. In many such projects, a geometric shape is used to represent each tone in the music evaluated. The tones' pitches and corresponding place in time are mapped on an X and Y axis. Other elements are added in for representation of dynamics and timbre. For instance, in Hiraga et.al's[7] visualization program, Chernoff faces were used to symbolize notes with their

facial features determining dynamics: if the nose was shaped like ">," that note would be played softer than the previous note, whereas if it was shaped like "<," it would be played louder. Because the score is displayed in a way that is more comprehensible for listeners who have not studied music, many aspects of the music can now be more easily observed, and the listener can get a better grasp on the piece as a whole.

Other works have focused on the emotion displayed by the visualization of music rather than just representing the score. This is specifically done in research regarding deaf and hard of hearing individuals. Fourney and Fels'[8] investigated what type of graphic visualizer conveyed the most emotion of popular pop songs based on their different characteristics to deaf and hard of hearing participants. Out of the three programs tested (Music Animation Machine, iTunes, and Motion Pixels of Music), almost all participants enjoyed iTunes better because of its color-changing, swirling graphics that pulsed with the beat and became more intense with rising volume. Although participants could view more information about the song in the other visualizers, like the movement of notes and dynamics, it did not communicate the emotion of the piece as well - the visuals were not captivating and exaggerated enough.

Live coding, the performance of changing code in front of an audience to produce different sounds and effects, has been explored on multiple occasions. In such performances, the audience can see both the code and the performer altering it, giving them the opportunity to understand how the code directly correlates to the sound being output[9]. Another form of this, called live writing[10], inputs works of poetry instead of code. A program was developed to turn a qwerty keyboard into an input device, assigning each key a note or sound to be produced through a speaker when pressed. When a performer types their poem, the words and phrases display on a screen for the audience while music is produced. Performers can control the sounds they produce by purposefully creating poems that use letters and phrases specially corresponding to the sounds they desire.

Methods

Before any sound could be analyzed, it had to be converted from an analog to a digital signal. As shown in Figure 1, the singer(s) interacting with the program sang into a microphone connected directly into a digital audio interface (we used a Behringer Firepower FCA1616 for this project). Since the interface only has four inputs with microphone preamplifiers, we supplemented the interface with an analog mixing console. Any additional microphones were connected to the preamplifiers on this console, sent to discrete busses and connected to the interface. Since the interface can take no more than eight analog line inputs, if a ninth input is needed, the eighth input on the interface will be a mix of the eighth and ninth microphone outputs. The interface was then connected to the computer via USB.

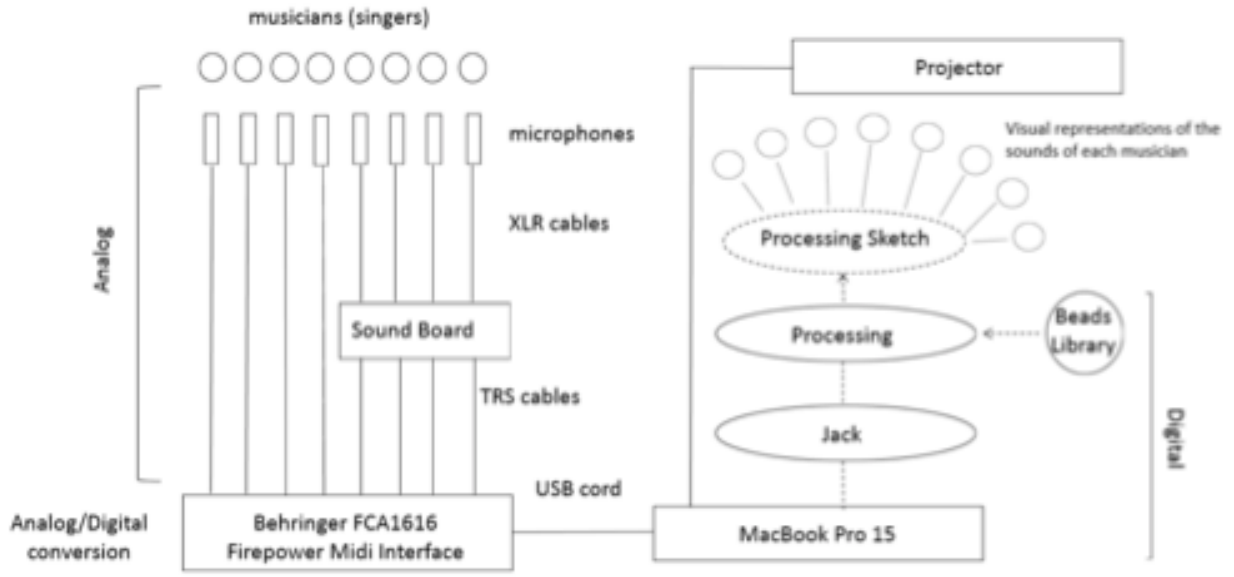

Figure 1. Flow chart showing the elements of the Painting Pitches system.

Unfortunately, the Processing language is not equipped to handle multiple inputs, unlike the Java Mixer[11] on which much of its sound functionality is built. Processing can access the Mixer and view the multiple inputs, but cannot access their data. This difficulty, however, can be overcome by using the Beads library, the JACK Audio Connection Kit (JACK), and a digital interface[12].

Beads, a library created by Ollie Bown, Ben Crawford, and Benito in Java and with a version for use in Processing, is structured around creating "chains" of audio signals[13]. Its main components are Unit Generators (UGens). UGen itself is an abstract class, and its subclasses are specialized for certain tasks, such as filtering a signal or creating a waveform; examples include Compressor, Envelope, Gain, and SamplePlayer. Every Beads project has an AudioContext object, to which are connected chains of UGens.

Although it is mainly focused on sound generation, Beads has some preliminary sound analysis functionality, which we gratefully found and used. By chaining UGens, Gains, ShortFrameSegementers, FFTs, PowerSpectrums, and, finally, Frequency objects, we were able to use Beads' pre-built pitch-detection algorithms. Since they provide no way to access amplitude, we adjusted the class slightly, creating a version that allowed access to amplitude. We wrote our own class, the "Input class" to simplify this functionality.

JACK is an inter-application audio routing program capable of sending sound from analog inputs or a particular program to either the computer's output or another program. Beads comes ready to communicate with JACK, so we set JACK to connect to the interface and Beads to connect to

JACK. JACK can handle multiple lines of audio, and Beads gets each of the lines from JACK as a UGen, which solves the multiple inputs problem.

One of the appeals of Processing is the ease with which it can be used to make visuals. The Input class allowed easy access to numbers representing each input line's pitch or volume, so we wrote our visuals with these values as function parameters.

Implementation

Working with Cantus, the previously mentioned professional men's choir, we chose four pieces of music from their repertoire representing four very different styles of music in the hopes of demonstrating the widest possible array of visuals.

We decided to visually focus each piece on a theme: one on color/shape, one on images, one on shape, and one on light. The first piece drew lines to create a geometric shape each time a pitch crossed a threshold then caused these shapes to grow and rotate when certain voices sang more loudly and in a higher range. The second piece represented each input line as a circle of light that illuminated a background picture, simulating the feel of candlelight; each circle moved or grew as the singers line changed. The third showed childhood pictures of the singers coming into focus as the singers sang loudly or quietly. The final piece used the idea of an equalizer and several different colors to represent which singer was singing at any given time.

The first piece, Zikr, composed by A. R. Rahman, had visuals based on patterns common in Islamic art. The visuals had multiple scenes for different parts of the piece. The first scene drew one line of a star shape each time a new pitch threshold was crossed. The next added more stars around the first one in response to higher pitches from the singers. The colors of each drawing also changed; louder sounds from the bass parts added more red and louder sounds from the tenors added more blue. The third and generally favorite scene (among viewers) caused the stars to spin with speed determined by the tenor voices: the higher they sang, the faster the stars spun, coming to rest when the singers ceased singing. Finally, the piece ended with an adaptation of Conway's Game of Life (Figure 3), triggered by a Makey Makey-enabled tambourine.

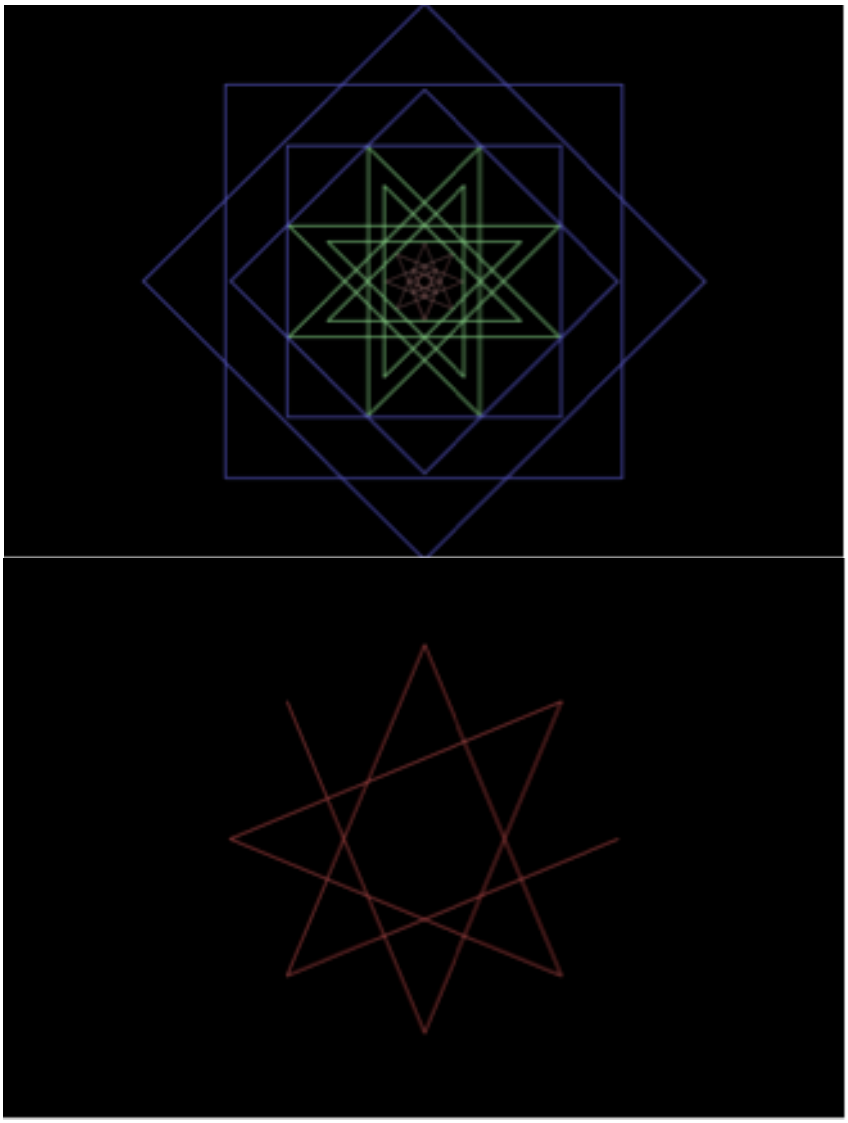

Figure 2. Screen captures of output from the Zikr visual treatment.

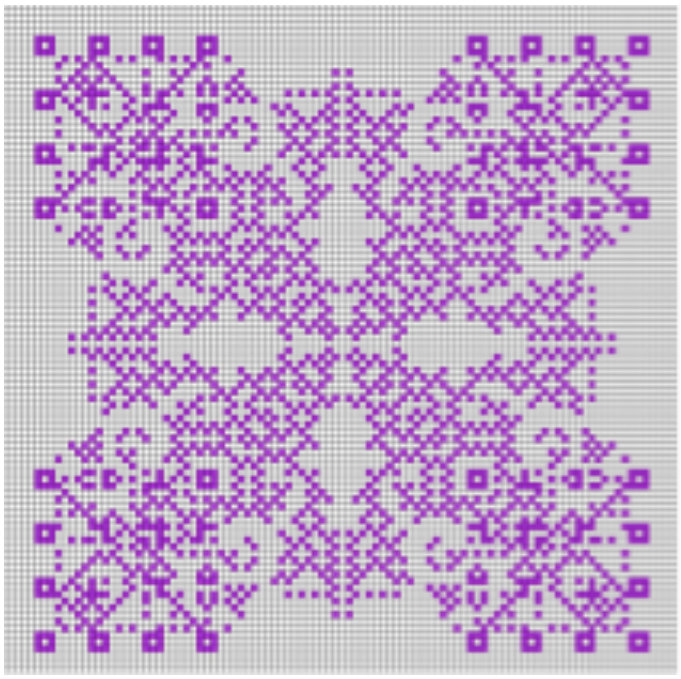

Figure 3. Screen capture of the tambourine triggered portion of the Zikr treatment.

Lux Aurumque, Latin for "light and gold", is a subtle and contemplative Christmas piece by Eric Whitacre (text by Edward Esch). The treatment of this piece, shown in Figure 4, reflected both the text and the music through multiple scenes where the voice of each singer controlled a light illuminating the background image. The first scene depicted light "warm and heavy" as the amplitude of the singing controls lights falling like rain. The soloist in the beginning controlled the Christmas star, rising as the other lights fell. The second scene drew attention to the star with the radial position of each light controlled by pitch and the size of each light controlled by amplitude. In the third and fourth scenes, angels were illuminated by the singers' lights. Pitch controlled y position and amplitude controlled size. The fifth scene occurred when the music began to reflect the beginning again; the lower parts became falling light and the upper parts highlighted an angel kneeling. The final scene was the Nativity and occurred at the words "modo natum" or "new-born [babe]". The soli tenor notes highlighted the child as the other lights, with radial position controlled by pitch and size controlled by amplitude, highlighted the rest of the scene including the mother.

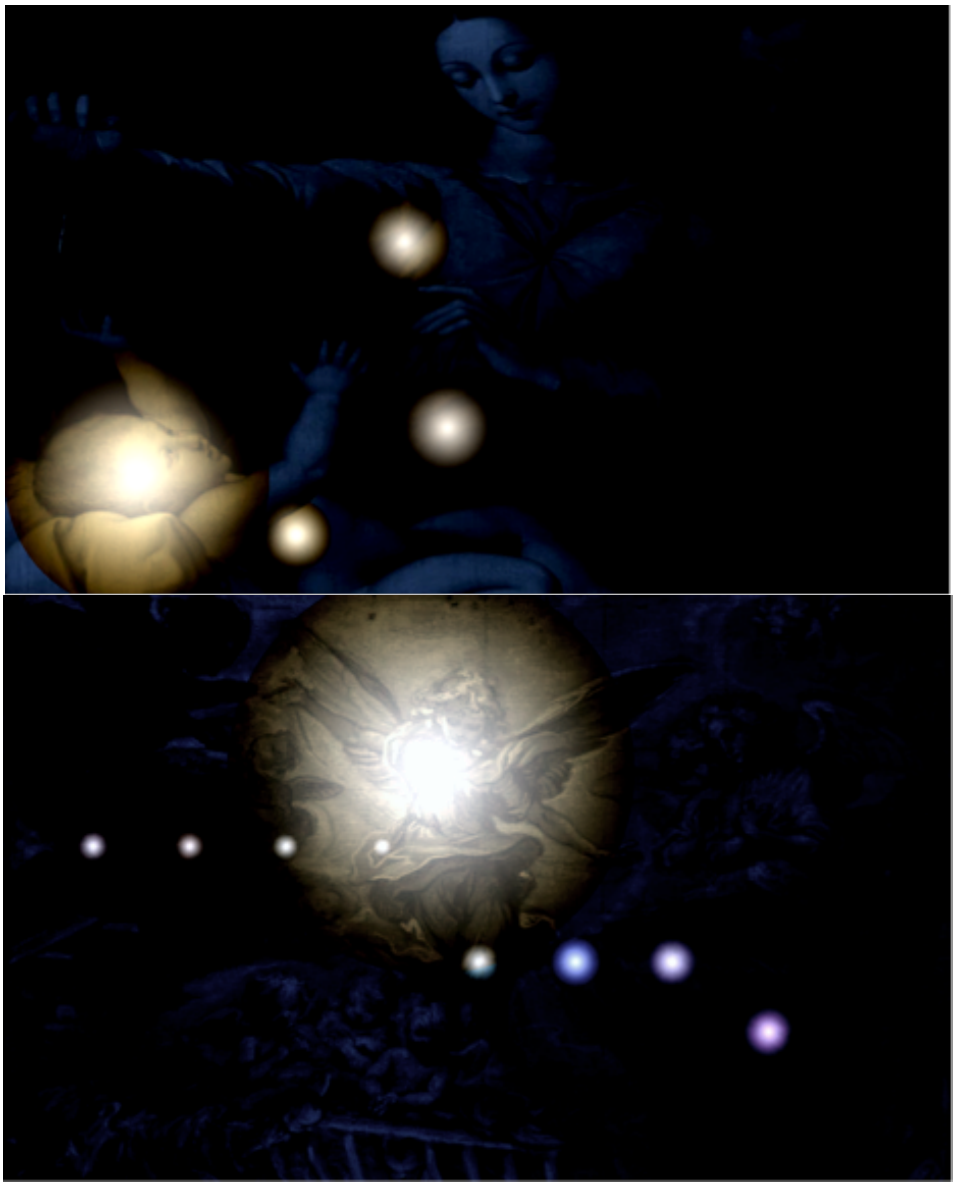

Figure 4. Screen captures from the performance of Lux Aurumque, in which a pre-set background image is illuminated by voice-controlled balls of light.

Wanting Memories, a piece composed by Yasaye M. Barnwe, focused on images. Using images of the singers as children to emphasize the nostalgic feeling of the piece, the images were arranged around the screen in a collage format. The images blurred in and out while the singers sang: the louder a singer was, the more clear the image was. Throughout the song, the images were changed one at a time at regular intervals to keep the scene interesting and different. There was also a picture of the singers at their current age in the center of the collage of pictures. This picture gradually came into focus throughout the song as small dots were drawn over the middle and largest square. These small dots created a pointillism effect, and the picture was fully visible by the end of the song.

Figure 5. Screen capture of the collage portion of Wanting Memories.

The fourth piece, a medley of Daft Punk songs, had nine very different vocal parts. The visuals created to accompany the piece consisted of three distinct parts: nine equalizer bars, colored balls moving across the screen at different heights, and a long bar which lit up on a key press. For the equalizer portion, the code took frequency data from the singers and drew stacks of bars with heights varying according to volume. The color hues of the stacks also changed according to pitch; higher pitch meant lighter hues. The balls in the background took audio input data from only one singer, and the balls would travel faster across the screen for higher volumes and changed height on the screen depending on the pitch. To fashion a key press mechanism, conductive tape and a Makey Makey kit were fitted onto a drum used by one of the singers. When the drum was hit, the tape on the singer's hand connected with the tape on the drum, and the Makey Makey kit registered this hit as a key press. This key press action was used to light up a white bar on the bottom of the screen whenever the drum was hit.

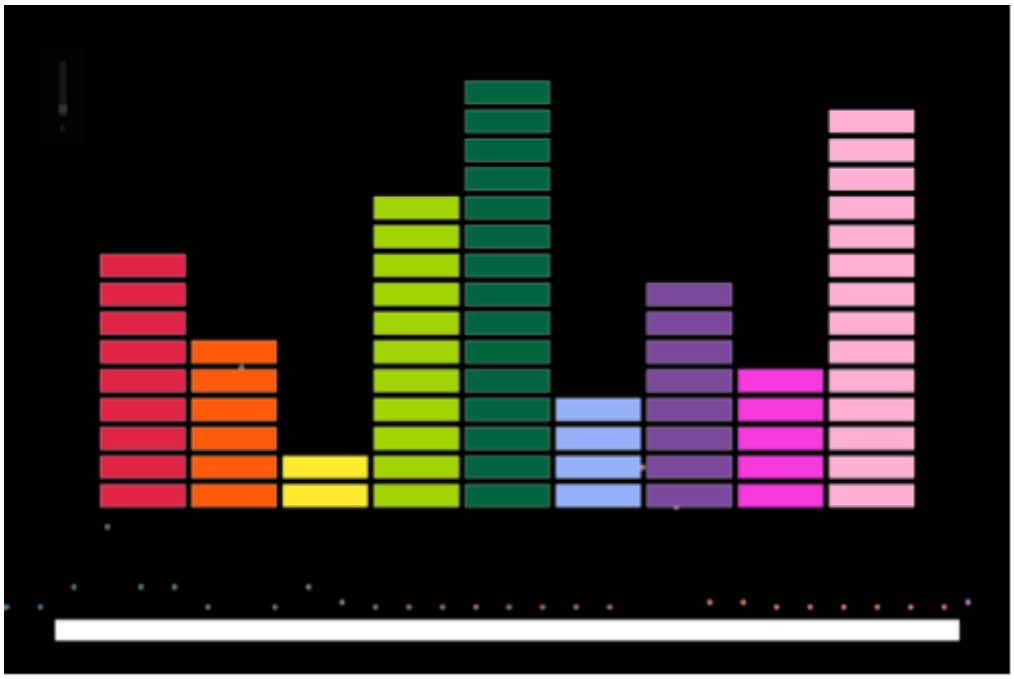

Table 6. Screen capture from a performance using the Daft Punk treatment.

Evaluation

To assess how audiences would react to the Painting Pitches work, a workshop concert of the four pieces described above was presented to an audience of nearly 200 people. After the presentations, audience members were encouraged to ask questions and give feedback on what they had just seen. The responses were overwhelmingly positive and encouraging with several questions about next steps.

Some audience members voiced confusion about the connection between the audio and visual aspects of the performance. Comments mentioned that while the audience generally felt that these aspects supplemented each other well, it was unclear the visuals were responding in direct correlation with the live audio. Audience members were also confused by the input system we used. The microphones used by the singers were used only to transmit sound into our A/D system and not to amplify their voices, and the audience found it odd for the singers to be holding microphones without hearing their voices being amplified.

Future Work

In response to feedback from our audience, we are looking to make our visuals more clearly interactive to demonstrate that the visuals depend completely on real-time audio input. We have a wide array of ideas for interactivity ranging from continued collaboration with Cantus on full-length songs to short and simple demos showing the distinct correlation between audio input and visual output. The focus of our efforts is the musicality of the visual. Our goal is to have a visual that feels like the music without distracting the audience from the nuance of the performance and composition. To achieve this goal, we have begun to develop tools to make it easier for the

performer to calibrate the visual to the timing and mood of the music. For example, one of our current demos allows the performer to experience a solid color visual assigned to each pitch. Musicality is applied when the performer is able to adjust the speed at which the colors evolve with the feeling of the music. The performer can also choose the colors she feels are the best for any given piece. Colors for minor keys could be a simple transition from blue to red through out the scale, while a more playful piece can explore the rainbow. As an audience member, these nuances become profound when you get to see and associate the different colors with the different structures within the piece. With multiple performers, it's possible to see the connection and contrast of each note of the piece of music in a new and exciting way without ever losing focus of the themes and details of the performance.

We are also supplementing our work with related libraries for Processing and keeping all of our code open-source, so anyone can access it and play with it in any way they might like. Through collaboration with others, we hope to widely expand our program's capabilities and uses.

References:

1. "Processing 2.x and 3.x Forum." *Processing 2.0 Forum*. N.p., n.d. Web. 20 Sept. 2016. <https://forum.processing.org/two/>.
2. Fry, Ben, and Casey Reas. "Processing.org." *Processing.org*. N.p., n.d. Web. 21 Sept. 2016. <https://processing.org/>.
3. Merz, Evan. "Sonifying Process: The Beads Tutorial." (n.d.): n. pag. 2011. Web. 20 Sept. 2016. <http://www.computermusicblog.com/SonifyingProcessing/Sonifying_Processing_The_Beads_Tutorial.pdf>.
4. Smith, S.m., and G.n. Williams. "A Visualization of Music." *Proceedings. Visualization '97 (Cat. No. 97CB36155)* (1997): n. pag. *IEEE*. Web. 18 Jan. 2017.
5. Balandra, Alfonso, Hironori Mitake, and Shoichi Hasegawa. "Haptic Music Player - Synthetic Audio-tactile Stimuli Generation Based on the Notes' Pitch and Instruments' Envelope Mapping." *International Conference on New Interfaces for Musical Expression, 2016. Proceedings.* (2016): n. pag.*NIME*. Web. 18 Jan. 2017.
6. Malinowski, Stephen. "Music Animation Machine." *Music Animation Machine*. N.p., n.d. Web. 18 Jan. 2017. <http://musanim.com/>.
7. Hiraga, R., F. Watanabe, and I. Fujishiro. "Music Learning through Visualization." *Second International Conference on Web Delivering of Music, 2002. WEDELMUSIC 2002. Proceedings.* (2002): n. pag. *IEEE Xplore Digital Library*. Web. 18 Jan. 2017.
8. Fourney, David W., and Deborah I. Fels. "Creating Access to Music through Visualization." *2009 IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)* (2009): n. pag. Web.
9. Sorensen, Andrew, and Andrew Brown. "Aa-cell IN PRACTICE: AN APPROACH TO MUSICAL LIVE CODING." (2007): n. pag. Web. 18 Jan. 2017.

10. Won Lee, Sang, Georg Essl, and Mari Martinez. "Live Writing: Writing as a Real-time Audiovisual Performance." *International Conference on New Interfaces for Musical Expression, 2016. Proceedings.* (2016): n. pag. Web.
11. "Mixer (Java Platform SE 7)." *Mixer (Java Platform SE 7)*. N.p., n.d. Web. 20 Sept. 2016. <http://docs.oracle.com/javase/7/docs/api/javax/sound/sampled/Mixer.html>.
12. "JACK Audio Connection Kit|Home." *JACK Audio Connection Kit|Home*. N.p., n.d. Web. 20 Sept. 2016. <http://jackaudio.org/>.
13. Profenza, George. "Getting Multiple Audio Inputs in Processing." *Stack Overflow*. N.p., 23 June 2016. Web. 25 June 2016. <http://stackoverflow.com/questions/37944418/getting-multiple-audio-inputs-in-processing>.