Stained Glass Shards: An Exploration of Visual-Reactive Audio

Keene Cheung

University of California San Diego
Department of Music
Encinitas, CA

***Abstract*—In contemporary multimedia art, audiovisual compositions typically rely on audio parameters to drive visual output. This project attempts to explore an inverse paradigm, using visual parameters to control sound generation in real time. By utilizing interactive machine learning and custom digital instrument design, the project explore alternative methods of human-computer interaction and novel approaches to audiovisual composition and performance.**

***Index Terms*—Audiovisual Composition, Physical Interaction Design, Interactive Machine Learning, and Digital Instrument Design.**

I. Introduction

The realm of audiovisual art combines visual and auditory elements to create immersive, multi-sensory experiences. Examples of work within this genre include visual music, abstract film, live performances, interactive experiences, etc. Within the category, most audiovisual works use audio-reactive visuals where musical compositions are performed while their data (volume, frequency, etc.) is tracked and used as input to control visual parameters such as brightness, color, etc, or to trigger various cues. However, it is far less common to see the opposite (visual-reactive audio). One of the reasons this is the case is due to the more variable and dynamic nature of visuals that can make it more difficult to quantify and tokenize as inputs. Looking at raw data, video files can be as much as 100 times larger than an audio file for a given source, making it much more computationally taxing to track, especially in the real-time setting of music performance. Furthermore, human perception tends to correlate sensory stimuli, such as a change in a musical note, to a visual anchor after it is heard, which is why it feels more natural and common to have changes in audio affect visual parameters [1]. This work seeks to challenge this tendency and explore the artistic value of what could potentially be perceptually dissonant art. This project expands on that idea by introducing interactive, non-linear artistic interfaces. These aim to further immerse the user in the augmented audiovisual experience. The goal for this project was to create an interactive audiovisual composition that allows a user to control aspects of a set of visuals that is used as input to control the sonic identity of a soundscape composition live as the user/performer continues to interact with the work, presenting an ever-evolving and amorphous piece. Beyond the scope of the project, the hope for this work is to provide new creative ways in which composers and performers can view and approach audiovisual composition via human-computer interaction and instrument design. To implement this concept in a performance setting, it is crucial to understand the technical limitations of real-time audiovisual processing, which is discussed in the following section.

II. Background

A core difficulty of computer-assisted performance is being able to process large amounts of data in real-time and extract the necessary features with little to no latency as to ensure continuity between computer and performer. Audio data is relatively easier to process in real-time because it is a discrete and one-dimensional stream of data containing frequency, amplitude, and timbre values. To compare these to more musical terms, concepts such as pitch, note length, dynamics, etc., which can all be easily correlated to physical values such as hertz, seconds, decibels, etc. Conversely, video data involves a broader and more complex set of parameters, such as color, brightness, scale, motion, and more, that is tracked over a multi-dimensional scale through a screen. As such, gathering information from these streams is far easier with audio than video. Within the computer music research realm, there exists a field known as music information retrieval that explores the extraction and analysis of data from audio in real-time using audio synthesis and digital signal processing techniques [2]. While it is still an emerging field with new ideas still being researched today, it has significantly more notoriety than any potential visual counterparts, which is highly indicative of the feasibility of live audio-visual analysis. To add to the difficulty of controlling audio with video, mapping visual stimuli to sounds is a subjective task and very difficult to quantify, in turn. For instance, if a sound is intended to represent a bright color, how should its sonic attributes, such as pitch, timbre, or texture, be adjusted to reflect the perceived 'brightness'? The lack of a universal framework for such mappings introduces ambiguity. The correct translation from said color to a sound would be different for nearly every person, and so it becomes impossible to pinpoint the exact qualities that align. While taking a subjective concept like art and turning it into objective categories is already quite difficult to do, it is slightly more straightforward with musical parameters. While the perception of sound via music and the interpretation of whether a work is perhaps "good" or not is a subjective matter, the physical traits that categorize music are fewer and easier to extrapolate and use as parameters for visuals, in turn. In order to find a functional and reasonable way to quantify traits of a visual artwork and use them as control parameters in a live performance, I needed to think about what certain characteristics of a work I can easily identify and calculate. The main one that came to mind was color. Color can be categorized on many different scales, with common ones including RGB (Red, Green, Blue) and HSV (Hue, Saturation, Value). Both these scales provide three-dimensional vectors that can characterize the color of any given part of a visually perceived object. Using these parameters as a starting point, I then worked to see how I could utilize these units and transfer them into a sonic scale that would work both mechanically and musically, which I will discuss later in the next section.

III. Methodology

Multimedia art demands interactions between several different disciplines and tools. For an interdisciplinary artist, beyond knowing these tools, it is important to also understand how different tools can interact with each other. Throughout the course of this project, I have been learning, testing, experimenting with, and troubleshooting countless different programs and software that have led me to settle on what I think works best for the task I set out for myself.

A. Software

For the visual and musical artist, there are countless different software programs used to accelerate and convey artistic intention. Each software has its own strengths catered to the artistic practice that the user leans towards. For an electronic music artist like myself, digital audio workstations (DAWs) will be the most common. These programs are designed to streamline music production, editing, and recording, among other functions. Notable industry standards include Ableton, Audacity, ProTools, Reaper, and more. For a project like this, where computer controlled music composition is a key part, I opted to use Ableton. What makes Ableton good for this work is that it allows for intuitive and creative design of any sound art project while allowing me to automatically control various audio effects using MIDI (Musical Instrument Digital Interface), which is a useful asset I discuss further in the next subsection. For visual artists using computer-generated visuals, TouchDesigner is a popular node-based programming environment that allows for creative development of visuals. Game development is a widely mainstream section of the audio-visual art space, which revolves around software such as Unreal Engine and Unity to program sprites and sound cues. While all these software serve their own purposes very well, I decided not to opt for any of these due to the lack of protocols that allowed it to send video and audio data to and from itself. Instead, I opted for Pd (PureData) to program my visuals. Pd is an open-source music programming language developed by Miller Puckette that is designed for interactive computer music works like this one. It falls under a larger family of music programming languages that include Max/MSP, Supercollider, and more. What made Pd sensible to use in this project was its ability to communicate with Ableton and create visuals both in real time [3]. To create visuals, I had to import the GEM (Graphics Environment for Multimedia) library, which is Pd's real-time graphical programming environment, which would allow me to interactively control the visuals that I create for the work. It also gave me access to the Fluid Corpus Manipulation library for Pd, which is a machine learning toollkit developed for audio programming software like Pd, which allowed me to train the model that I ultimately use to convert the RGB color input into control outputs.

B. Protocols

All these software programs function well in their own unique ways. However, multimedia art seeks to push the boundaries of design interaction. As such, it is important to understand how different programs can interact with each other to enhance a work and/or provide it with direction that it wouldn't be able to accomplish on its own. I have briefly mentioned various protocols that these software use to trigger and modulate various parameters. When it comes to audio, the most common ones are MIDI and OSC (Open Sound Control). MIDI is a standardized protocol that sends musical data. Usually, this is in the form of MIDI CCs (Control changes) that are measured between 0-127 used to signal certain events occurring. These events can include note-on/off, note velocity/dynamic, and more. For example, a value of 1 on channel 64 would usually trigger a damper pedal on most synthesizers. OSC is a protocol that is built to send messages over networks, allowing for wireless communication in real time. What OSC has that MIDI lacks is the ability to send audio signals and lighting messages as well [4]. This makes OSC very common for live theater as it allows easy collaboration between lighting and audio. While many software can use OSC, such as TouchDesigner and QLab, its implementation in music production and programming software isn't as standardized as MIDI. Since I am opting for Ableton to create my work, it is easier to use MIDI, since while Ableton can integrate OSC, it has built in MIDI mapping functions that can allow me to directly control the desired audio effect parameters and levels as long as I establish a link. Furthermore, PureData, while also having OSC implementation, requires third-party objects to navigate, rather than being able to directly output MIDI CCs to whatever source I need. One additional step that I do have to navigate, since I use a Windows computer, is routing MIDI from PureData to Ableton. Mac protocols are built to easily route Pd directly to Ableton, however, with Windows, I have to add in a third-party virtual routing software, loopMIDI, to take the MIDI CCs from Pd as input and route them out to Ableton's MIDI input. This can be thought of as using an analog wire between two different devices, where the software are the two different devices and loopMIDI is the wire, connecting them. While I have opted for a specific choice of software and patching, the presentation of all these resources seeks to acknowledge that this type of interaction can be replicated using a multitude of combinations of any of the programs I have previously mentioned. As such, they can be catered towards each artist's creative strengths with different tools.

C. Workflow

Now that we have established the tools available at the user's disposal, we will discuss the actual logistics that are used to make this project a reality. The first part comes from the visuals. Since this work revolves around the control of color to influence the work, the visuals lean towards a more monochromatic look with slight variations in intensity and brightness of the color to add bits of depth. Since Pd wasn't created specifically with visual artistry in mind, the GEM library relies on very simplistic visuals using mostly geometric objects and particles. While visual design was not the central focus, they serve as proof of concept for the feasibility of this work. The visuals are controlled using three GUI sliders that each correlate to one of the colors on the RGB scale. These sliders can be adjusted live, allowing for interactive creative control over the work. To control audio, the output of the sliders is also sent to a separate set of code that runs a neural network that is trained to convert the 3-dimensional color vector into a 10-dimensional output, both containing values from 0 to 1 for ease of use. This is specifically done through a MLP (Multi-layered perceptron) Regressor utilizing FluCoMa's Pd toolkit [5][6]. The MLP is a supervised learning model that is trained by setting data points on the RGB sliders and then adjusting the output parameters to a desired state that the user believes correlates to the inputs and recording that configuration. Each of these data points is then added to a dataset containing each RGB vector and its corresponding output vector. Using the data from all these discrete data points as the training model, the neural network is then able to make predictions on what the values of the output parameters are based on the given data. This is convenient in this situation because the model is fed discrete data points for several continuous variables. As such, it is able to interpolate values not previously given and provide a more dynamic and broad range of sounds, exploring the unknown ”in-betweens”. For example, an RGB vector that produces red might have output 1 and 2 at .75 while blue might have output 1 and 2 at .25, and output 3 at .5. After training the model, if the user adjusted the color to be purple, it likely would adjust the parameters to be some combination of value in between the two separate colors where output 1 and 2 might be .5 and output 3 is .25, for example. These output parameters are then sent out as MIDI CCs, normalized to the standard MIDI range of 0-127. As discussed in the previous subsection, these MIDI CCs are routed to Ableton using loopMIDI and mapped to selected audio effect parameters, such as the Dry/Wet percentage of a distortion, which can be adjusted based on the user's unique creative vision. With all this set up, the performer can now perform the given composition and control the color of the visuals and timbre of the composition live as if playing an instrument containing only three sliders.

IV. Discussion

The main idea of this project was to explore new methods of audiovisual composition and see how they can interface with current standardized systems to help continue to push the boundaries of computer music. As such, it is also important to discuss the relevant material and other researched works that helped inspire this work, and also provide insight into future iterations of this.

A. Artistic Applications/Similar Works

Throughout this paper, I have mentioned the idea of human-computer interaction in computer music that plays a key role in the thought process of this project. Alberto Boem's paper on non-rigid musical interfaces was especially indicative of the work's development. In the paper, non-rigid musical interfaces are defined as musical interfaces that ”can be physically deformed to generate and/or manipulate sounds or dynamically respond to musicians performance” [7]. Within this definition, the deformable interfaces are those that that shape change in sound through user input like one of the examples in the study that used the deformation of clay to create specific audio signals. On the other hand shape-changing interfaces autonomously shape sounds structures like a heart-rate monitor controlling tempo. While the interface I created wasn't physical and wouldn't exactly fall under the category of ”non-rigid”, the expressive control of a deformable interface greatly aided my perspective on what I needed to focus on in order to create an effective musical interface. For this project, I used color as the primary interface, imagining using a color palette and physically using a paintbrush to mix and select colors as a way to give shape to the sound of the audio visual composition. Within the HCI field, the subsection that many of these papers fall under is referred to as physical interaction design. This field is still emerging, but also contains many well-developed projects that helped to inform my implementation on this work. The most notable example is Rebecca Fiebrink's Wekinator [8]. This is a software that she she developed that uses interactive machine learning to create new systems for gesturally-controlled audio and visuals. It uses gesture analysis to detect whether your hand is close to the screen, for example, and uses that data to send OSC messages to other systems. While my project didn't use gestures to control the art, her approach of integrating interactive machine learning through neural networks like the MLP I used helped guide to me using learning models to create dynamic audio control. When initially conceptualizing the project, I wanted to create a soundscape composition that explored the theme of ”Sands of Time”. As the concept developed and the idea of using color as the controlled input made me think about the connections between sand and color. This is where the title of this project originated from, using color to turn the sands of time into stained glass that would be sonically morphed and broken into numerous pieces throughout the composition as the user controls the color.

B. Future Improvements

In terms of the execution of this project, there are many parts of it that I would like to edit artistically. However, I believe that functionally, it was successful in providing a unique and fun way to approach audiovisual art and human-computer interaction design where a color palette became the instrument. There are still limitations to this though such as the mapping of the specific software I used and the feasibility in which it could be translated to other platforms. As such, if I was to continue to develop this in the future, I would love to explore integration of other software to achieve a similar goal. Perhaps using OSC messages with TouchDesigner and Max/MSP could be an interesting option. To move the project into the physical world, perhaps integrating motion sensors or other ”deformable” interfaces would also be a good idea to explore especially for live performance where a performer's gesture is essential in conveying musicality. Furthermore, I would love to see and experiment with more creatively-driven usage of this design, since functionality took the priority over artistic license in this project. The benefit of this system is that the user can adjust the number of inputs and outputs to their need and retrain the data to map the outputs to any parameters they desire. While I only showed a few examples, there are countless combinations of mapping that can be used depending on the creative vision. For someone who leans more towards the visual computing arts, perhaps exploring generative visuals using these input could also be an interesting continuation of this design. Ultimately, the idea with creating new ways to interface with art is to break free from the limited mediums of expression that the acoustic music world limits the listener to, especially in the world of creative practice, where novelty fuels new research ideas.

REFERENCES

[1] Lavan N, Collins MRN, Miah JFM. Audiovisual identity perception from naturally-varying stimuli is driven by visual information. Br J Psychol. 2022 Feb;113(1):248-263. doi: 10.1111/bjop.12531. Epub 2021 Sep 6. PMID: 34490897.

[2] Markus Schedl; Emilia Gómez; Julián Urbano, Music Information Retrieval: Recent Developments and Applications , now, 2014, doi: 10.1561/1500000042.

[3] Fu, Quanwei. (2021). Research on the Use of Computer Music in Modern Musical Composition. Journal of Physics: Conference Series. 1820. 012153. 10.1088/1742-6596/1820/1/012153.

[4] Wright, Matthew and Freed, Adrian and Momeni, Ali. (2003). OpenSound Control: State of the Art 2003. Proceedings of the 2003 Conference on New Interfaces for Musical Expression (NIME-03).

[5] Tremblay, P.A., Green, O., Roma, G., Bradbury, J., Moore, T., Hart, J., and Harker, A. (2022) The Fluid Corpus Manipulation Toolbox (v.1). Zenodo. doi: /10.5281/zenodo.6834643

[6] Tremblay, P.A., Roma, G., and Green, O. (2022) Enabling Programmatic Data Mining as Musicking: The Fluid Corpus Manipulation Toolkit. Computer Music Journal 2022; 45 (2): 9–23. doi: /10.1162/com_a_00600

[7] Boem, Alberto. (2020). Non-Rigid Musical Interfaces: Exploring Practices, Takes, and Future Perspective. (NIME 2020).

[8] Fiebrink, Rebecca and Cook, Perry. (2010). The Wekinator: A System for Real-time, Interactive Machine Learning in Music. Proceedings of The Eleventh International Society for Music Information Retrieval Conference (ISMIR 2010).