Animation as an Instrument:

Designing a *Visual-Audioizer* Prototype

Thesis

Presented in Partial Fulfillment of the Requirements for the Degree Master of Fine Arts

in the Graduate School of The Ohio State University

By

Taylor J. Olsen

Graduate Program in Design

The Ohio State University

2020

Thesis Committee

Kyoung Lee Swearingen, Advisor

Marc Ainger

Matt Lewis

Abstract

This paper discusses the process of making a "*visual-audioizer"* prototype in which I designed and created as a new method for the computer musician and animator to produce audio. The *visual-audioizer* is a patch I created in Max in which traditional animation techniques, in tandem with basic computer vision tracking methods, can be used as a tool to allow the visual time-based media artist to produce audio and eventually, music. Using this tool with the animated form provides real-time feedback within a dynamic locale constrained to the software and allows the user to hear their visual creations within a sonic setting. For the user unfamiliar with animation techniques and computer music, an experiential media system in which animation, audiovisuals, and exploration are considered synonymous. To the animator, an alternative method/consideration to letting the implicit temporal-knowledge of motion to apply this system as a means of creating audio. For the computer musician, a new way to learn about animation and the role of animation techniques as musical instrument. Through explorations in the realm of graphic aesthetics of scale, movement, patterns, and pictorial ambiguity, aspects of what can be described visually perceptible and within an audibly comparable setting play a key role when manipulating audio creation. Using the *visual-audioizer,* animators and computer musicians will find new ways to experience and create their audiovisual media through means of drawing and interacting with the tool itself.

Dedication

This paper is dedicated to my teachers, advisors, and thesis peers among the Design department and my place of research, ACCAD. It would not be complete without mentioning my parents, friends, and family that have influenced my life with every laugh, inquiry, and moments of peace during these stressful times.

Thank you.

Acknowledgments

For all the time I spent enjoying this research I have many that come to mind deserving acknowledgemen. Thank you to Ryan English, an animator and OSU alum, for convincing me to pursue higher knowledge with graduate school. Thank you Kyoung Swearingen, my head advisor, for direction and motivation through this thesis journey; and for inviting my wife and I to become part of Scott and your daughter's journey through board games and cats. Matt Lewis, for your shared excitement and insight towards anything technical and funny. To Marc Ainger, for pushing my curiosity when working and writing this research.

Thank you to my wife, Taylor, for being the encouragement I needed through times of doubt. For being the meal at the end of the day that made me smile. For your generosity and drive for all my actions and more, you always see the better side of any situation. To my parents and friends who helped raise and support me, I cannot wait to see you again soon. And to my 4 other DAIM graduates, Tori, Abby, Leah, and Joe; cannot wait to see what you make!

There is one that seemed to stand out among others in the end; that one is my piano teacher for about 11 years, Carol Flower. Carol, you were there through most of my upbringing and taught me the discipline, structure, technique, critique, and evaluation of music. I feel a sense of loss knowing I stopped playing for academic reasons, but there was not a day I did not have classical music playing in my headphones when studying. You changed the way I think about life and its endless relationships, thank you.

Vita

2011	SDMTA Senior Classical Piano Competition 2nd Place
2012	Washington High School, Sioux Falls, SD
2016	B.S., Dakota State University Madison, SD
2017 – 2020	M.F.A., The Ohio State University Columbus, OH
2017 – 2019	Graduate Research & Teaching Assistant ACCAD & Department of Design Columbus, OH
2019 – 2020	Graduate Teaching Assistant The Film Studies Program, OSU Columbus, OH

Publications

Olsen, T. "Animation, Sonification, and Fluid-time: A *Visual-Audioizer* Prototype." New Interfaces for Musical Expression: Proceedings Archive. 2020.

Fields of Study

Major Field: Design
Specialization: Digital Animation and Interactive Media

Table of Contents

List of Figures

Chapter 1. Introduction

Animation has always been second nature to me. In a sense it is as much a part of *how* I have learned to observe the natural world, as well as interact with as everyday observation. Growing up with cartoons and animated films permeating my emotional connections to characters and real life – animation, to me, was always about the story and how the story works. While listening, I became more and more insatiable about finding the best stories about the films, the pinnacle of twists and turns in the narrative structure. Feelings of depravity, hope, edge-of-seat occurrences, and analogical concepts flooded my mind about how I could make audiences gasp and shrivel at depictions of motion between

Figure 1: Animated movies from left to right—The Brave Little Toaster (1987), The Land Before Time (1988), Akira (1988), Princess Mononoke (1997)

the brains and the screen.

I will never forget youth-oriented features like *The Brave Little Toaster* (1987), and *The Land Before Time* (1988). Or adult-oriented animated masterpieces like *Akira* (1988) and *Princess Mononoke* (1997). But as I watched more, I began to disregard the visuals, I wanted more from the character development and the plot; I could have the cartoon running, and not even pay attention to the visuals, only listening to the story. This also became prevalent in the future when watching cartoons like *American Dad* (2005), *Adventure Time* (2010), and *Futurama* (2013). Why was I disregarding the hand of the artist, but listening to the audio? Was the sensation of re-watching these films dulled because I had already experienced the visual aspects?

In my adolescent years, my brothers would always wait for my parents to go to sleep to turn on the television to Cartoon Network's late-night animation program, *Adult*

Figure 2: Off the Air and Adult Swim

Figure 3: Off the air Episodes from left to right: Color (2012), Light (2013), Patterns (2019)

Swim. Because of this programming, I was exposed to something other than my normal expectations of animation: visuals, experimentation, motion, and music. Especially, there was a time in the 2011 when the program broadcasted episodes from a series called *Off the Air* (2011), created by Dave Hughes. I describe *Off the Air* as, well, indescribable—it is often shown without an explanation of what content is displayed, only strung together by a loose theme based on a word/phrase. My understandings of story and logical thematic values were discarded and replaced with raw motion, colors, and towering spires of light – stunning, spastic, and unexpected motion had captivated me. Episodes such as "Color" (2012), "Light" (2013), and more recently "Patterns" (2019), didn't ask or expect the viewer to experience a story; it rather considered the act of observing as the experience itself. What was this change that I found myself infatuated with, and why? It was in this moment I felt animation and sound could be something more than storytelling, but an

experience, impossible to put into words.

I spent time attempting to work with traditional 3-act animated stories, but eventually I turned to exploring experimental design process and latched onto non-object animation techniques discovered during my graduate studies. As I explored abstracted methods of drawn motion, I was challenged with the unpredictability of the of the animated form and finding a way to give it deeper meaning than just a visual aspect. This desire to exemplify the experience of watching animation rather than what it should imitate drove me to explore more about past experimental animators, such as Normal McLaren and Oskar Fischinger, that enticed the audience to consider what animated motion *is*, rather than what an animated motion *tells.*

Though animation was a large part of my upbringing, I enjoyed 14 years understanding, practicing, and performing classical music. Ultimately, my practice was put on a hiatus when I stopped taking lessons, but I shared the sentiment towards animation—I wanted to challenge my previous understanding and create a new way to experience the medium. Music was and still is an escape for me, performing was a different experience than watching as an audience member—like animating. Using an instrument required practice to record music, while animating required the same to make a sequence. And while music can be performed live, this is where I found the practice of animating, among all the ceaseless ways to be expressed, was limited. Frame-by-frame animation, until recently, was not a live process to be performed; often it is time-intensive, demanding, and difficult to comprehend. Thinking deeply about what I appreciated about the animated form, digital music creation, and my tacit knowledge as an animator, I holistically decided to combine

these aspects of creative output into a new experience.

The skills that I gained from my previous experiences and throughout my grad school career have become embedded intimately within me and they inherently will carry forward. It was now a challenge for me to consider *how* to move forward—I wanted to consider a psychological grounding for time-based media and audio and compare how the two could be extrapolated from one another into an immersive experience for the creator and the viewer. I recall a quote I stumbled upon from my graduate studies, "Music informs images just as images inform music" (Detheux, 2013). But, which one should come first—and does there necessarily need to be a choice? Within the current age of audiovisual expression, advancements in technological aspects of CPU/GPU architecture allow the elucidation of visuals from digital audio. What if instead we create audio from visuals? The central aim of this paper is to demonstrate the feasibility, prototyping, and usage of a *visual-audioizer*—a software that translates visual information into digital audio and encourages an audience to consider principles of animated motion as an instrument for musical expression.[1]

Impact of this Research

If you are an animator, an artist, a computer musician or sound-designer, someone who likes to tinker with interactive media systems, or have an appreciation for the process of modern data-translation, this is for you. This research and experience will facilitate

[1]Olsen, "Visual-Audioizer Prototype w/ Prerendered Video – 2020": https://www.youtube.com/watch?v=Yi0FtJ771Uk&list=PLzNnI_tCy5c_ETnD576c-bT1kerlXqh9I&index=2

conversations among academics and creatives the like. What are the possibilities of utilizing this media system as a source of inspiration for audiovisual synthesis? How can someone who has never been exposed to concepts of sonified animations gain familiarity and faculty with an open exploration of the animated form?

I want to argue the *visual-audioizer* allows the animator, computer musician, and random user to consider how motion can be observed and created. This exploration of audiovisual synthesis will focus on the deepening of artistic aesthetics and musical expression in the field of arts and humanities. One of my goals for creating this method of visuals to audio is to bring an easily engaged experience to the user, and an observable and reflective method of creative inquiry within the audiovisual realm for the animator and the computer musician. It will foster new opportunities for individuals intimidated by the idea of creating animation and music to explore in an environment that encourages exploration. The translation of code, to visuals, to audio, and finally to interactivity engages a broad range of academics/artists and will allow individuals to become more attuned with how they view the association of visual/perceptual motion. My future hopes for creating the *visual-audioizer* prototype are to facilitate a conversation about utilizing the system experiment with sonification methods in academic, therapeutic, or performative settings.

Considerations for this Research

Please note that a sections of this paper include a modified version of my previous research paper written during my graduate studies, “Animation, Sonification, and Fluid-Time: A *Visual-Audioizer* Prototype” published in the New Interfaces for Musical Expression;

hosting in July, 2020.

A question that often arises when considering using the animated form vs. live-action form is this: why choose one over the other? In the case of this research, both methods happened to be used to inform design decisions later—animation was eventually decided upon as the source of generated audio from the *visual-audioizer*. With live-action video, the ability to quickly generate content has its advantages, but this is where I also found it limiting. Animation, on the other hand, has the innate ability to quickly change position, morph, and holds the perceived expression from the hand of the artist. To add, animation can be easily changed/edited to the desire of the artist, while live-action needs to often be completely re-recorded. Animation, on the other hand, does not necessarily have to hold a specific meaning and allows the artist to explore inherent qualities within their practice. With film there is a need for a general direction, as well as a suitable setting to work with—there is often a longer period of preparation needed to begin creating suitable content. This content, as well, is often hard to control. If I were to consider animation as a musical instrument there was a need for clearly defined control over the practice and the sound created. Much like a regular instrument, a piano for example, every note is laid out in front of the musician and allows complete control over what note is played—with practice. In the same sense, the canvas on which animator uses can be considered the keys on a piano. Frames allow precision over timing considerations, much like rhythm within music; and the practice of animation has similarities to that of practicing a musical instrument.

For myself, or ironically when learning, one should generally have a sense of failure

about the task. Without this fear of failure, we have no anxiety to create a learning environment. To animators, sometimes the greatest and simultaneously worst tool is not one of a pencil, a computer, or technical skill: it is anxiety of the mind. The anxiety translates to an animator's process: "Where do I start? Who is my audience? What is the story about? How do I convey anything at all?" Coupled with the process of approaching animated content as a top-down process can cause hesitation in the animator.[2] And while this process is used often, it can also be associated with critiquing animation. As the critic analyzes animated content, within the standpoint of a narrative structure, they often search

[2] Top-down process – meaning I would view animation as the motion, story, and technical requirements all at once, focusing less on the motion of animation and more about the big-picture.

for story first, and focus on the motion of the characters last.

During my time of research, I created a 12-frame loop (Figure 4) that had my peers staring at it for a least a couple minutes. To me, although it may be considered minimal in the sense of how much content was created, this concept of maintaining gaze through a process of layering interweaving paths of motion in a small amount of time demonstrates that it deserves more attention than is shown in the end output. In relation to film, a 'once

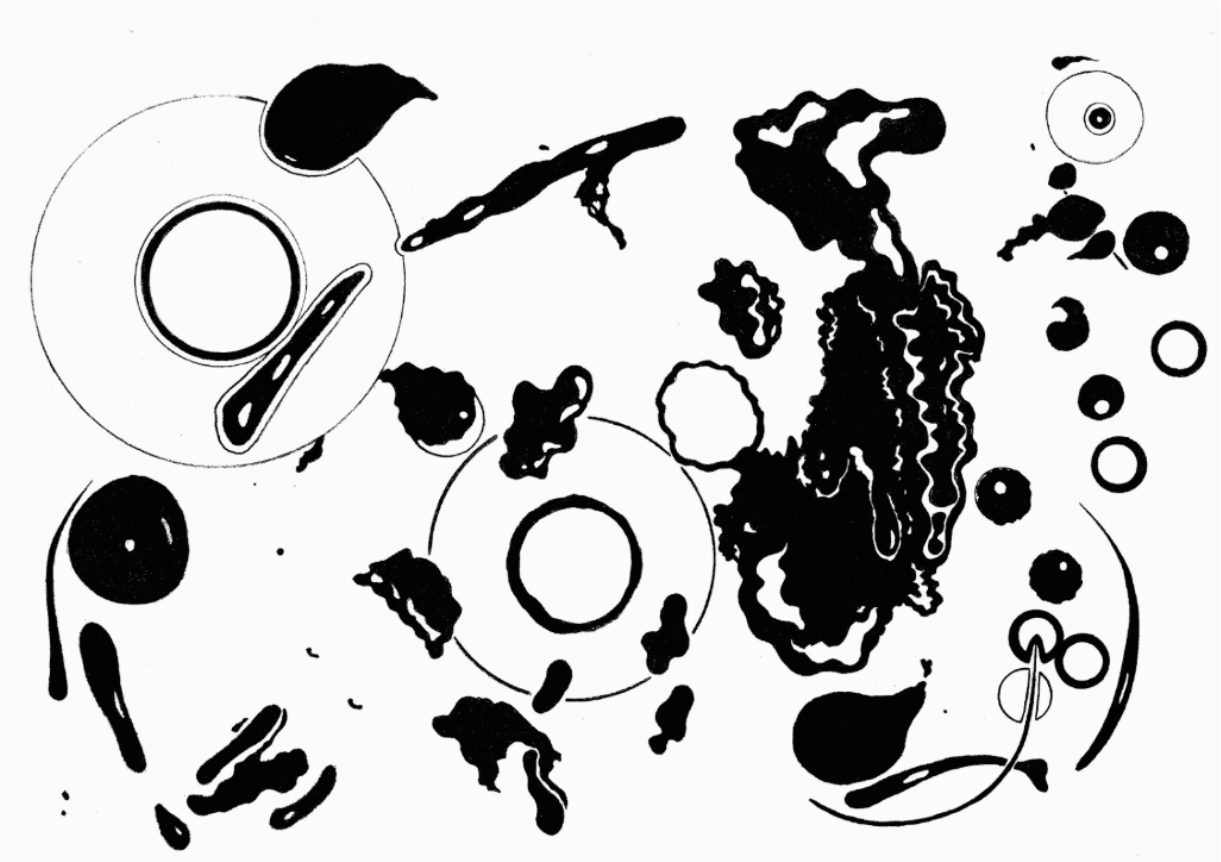


Figure 4: Looping Animation Frame

through' of the content is typically what the director is envisioning, getting the audience in/out in a timely matter.

For animated loops, there is no limit. Creating these animated pieces aided in my beginning thoughts of animation instrumentation and pushed me into the realm of experimental animation. This led to a further exploration of how animation can take the passive gaze of a viewer and pull them in an ultimately more immersive setting that is not concerned with narrative or objective structure. Considering loops and the alternative approach to appreciating animation, I reconciled about what I considered music to be. I as the listener, and often the audience that also engages, repeat the music that we enjoy. The processes of observing, learning, and performing music begins with understanding notes on sheet music; how loudly and long they are played; and the tacit to implicit knowledge that comes with muscle-memory. Having such a breadth of understanding for notation and types of ways to observe and 'read' sound, I was never taught the basics of how to *create* outside of the realm of audio, rather than how it can be *read*. Similarly, for my focus in animation: how the story can be *read* is what I was taught, rather than what animation can *create* beyond the visual realm. My desire was to explore the reverse processes of these two mediums and juxtapose them together, while letting animation take the lead in the creative process.

Practice-Based Design Research

This research is practice-based and is motivated by a series of questions surrounding animation, audiovisual synthesis, and interactive media. Practice-based research is, "an original investigation in order to gain new knowledge partly by means of

practice and the outcomes of that practice" (Candy, 2006). This research includes a solid base of writing but is not complete without a "direct reference to those outcomes"; the outcomes for this paper come in the form of an interactive media system and an example of it in use.[3] From an animator and computer musician's perspective, these questions should feel familiar, but not without creative inquiry. For others, certain concepts and theories may seem foreign but are briefly explained via footnotes, and in further detail during Chapter 2. My goal for these design questions is to question their validity and whether the outcomes of the research has achieved their goals. The first requires skill-building in the area of computer vision, the second requires practicing animation and logical comparisons between its principles and computer music, the third needs both the first and second question to be completed in order to assess how the outcome and animation practices work together.

Research Questions

To begin, how can computer vision aid in the data translation of visuals-to-audio?[4] With the development and innovations of computer vision software, visually quantifiable elements such as position, scale, orientation, and elongation can be used to the discretion of an animator's ability to generate audio. To elaborate a bit further: methods of generating these visuals has progressed, allowing a gestural animation process to have the ability to be a considered a "live performance". This question also carries historical implications of

[3]Candy

[4]Computer vision is a technique in which a computer "sees" the environment around it. The programmer can tell the computer what to "look" for in imagery. Modern uses (within the early 21st century) include self-autonomous machines, facial recognition, motion-capture techniques, and agricultural operations.

previous animator's attempts at eliciting audio from visuals and will be addressed later.

From an animator's perspective, how do we turn frame-by-frame animation practices into a real-time instrument for musical expression? While sound designers typically face the challenge of attributing musical attributes to visual manifestations, such as an audio visualizer, what principles of animated forms could dictate the creation of digital audio?[5] Data has become an accessible method in translating audio into visual data, showing new ways to perceive visual time-based media. Per the skill of an animator to create nuanced motion and complex forms within this time-based medium, similar sounds to those created by sound designers could be replicated and altered based on animation principles when utilizing the *visual-audioizer* interface.

What creative effect does real-time user manipulation of data within the translation of visual-to-audio synthesis demonstrate? The ability of modern tracking methods allows the *visual-audioizer* interface to observe and react synchronously; this question can also pose as a space for an audience to learn about the animated form as a method for musical expression. If a method for a musician is to have their audio visualized, why couldn't the animator have their visuals sonified? To add to this consideration: why couldn't an animator utilize a similar toolset?

Design Process

The process in which I based my research was mediated via my initial research

[5]Audio-visualizer (or music visualization): a software or animated piece in which animated imagery is based on musical qualities; often rendered in real-time. The visualizer is generally synchronized with the audio as it plays, whether this is by rhythm, pitch, clarity, etc.

questions. Though the questions took time to facilitate review and more in-depth exploration on my part, the outcomes from past audio-visual creatives aided in conceptualizing the prototype, and informed how I wanted to approach my research. It should be mentioned that this process is not linear and is of course open to other's interpretation; there is always room for improvement. During times of working on the prototype I was simultaneously creating projects that would inform how the *visual-audioizer* would function, how it reinforced and assessed my proposed design questions, and how animation methodologies and previous works validated the usability of the *visual-audioizer*.

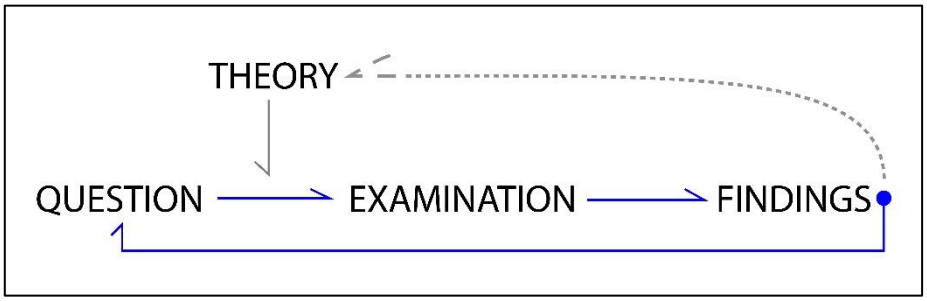


Figure 5: Theory as a Contextual Tool (Beck & Stolterman)

The process I created for my research is inspired by an article, from Jordan Beck and Erik Stolterman, regarding the practicality of theory within design research. The aspect of my research which resembles their examined methods uses "theory" as a contextual tool. "In this model, researchers start with a research question. Theory is used as a tool for contextualizing or situating the research question within a particular discourse" (Beck & Stolterman 2016, 131). In the case of my research, there is multiple questions with the basis of creating projects and examining the outcomes; these outcomes are then assessed and return to the proposed design questions for review. "Theory does not in any significant way

change the question, but it does result in “position taking” relative to other questions and existing research.” Contradictory to Beck and Stolterman, I did not see my initial questions as perfect and felt the necessity over time to evolve and shape the questions to aid in validating my research and contribution of knowledge. “In many cases, the knowledge object that the researchers’ reference may be called “theories” or “models” or even in some cases just “ideas” “concepts” or “perspectives,” among others. Once the question has been contextualized, the examination proceeds.”

For my design research methods, I posed different concepts, and historical backgrounds, that contextualize my work within a comparable history of audiovisual media. Continuing on, “The examination yields findings and, in this model, findings can (1) talk back to the original framing question, and (2) either talk back to the contextualizing theory, or not”.[6] The projects themselves would improve my technical skills—these skills would then inform the next project I decided to tackle. Outcomes from these projects would also alter my initial design questions, allowing a better understanding from individuals who are not familiar with the material. Through this method, I eventually found a custom cyclical process that worked to my advantage. A shortened version is as follows: formulating questions, creating projects, and conducting background research, reviewing outcomes, and interrogating the initial research questions for validity and achievement. In the next few paragraphs, I will provide a visual overview of this process. During Chapter 3, the projects in a general chronological order along with their impact on the design questions and the prototype.

[6]Beck and Stolterman

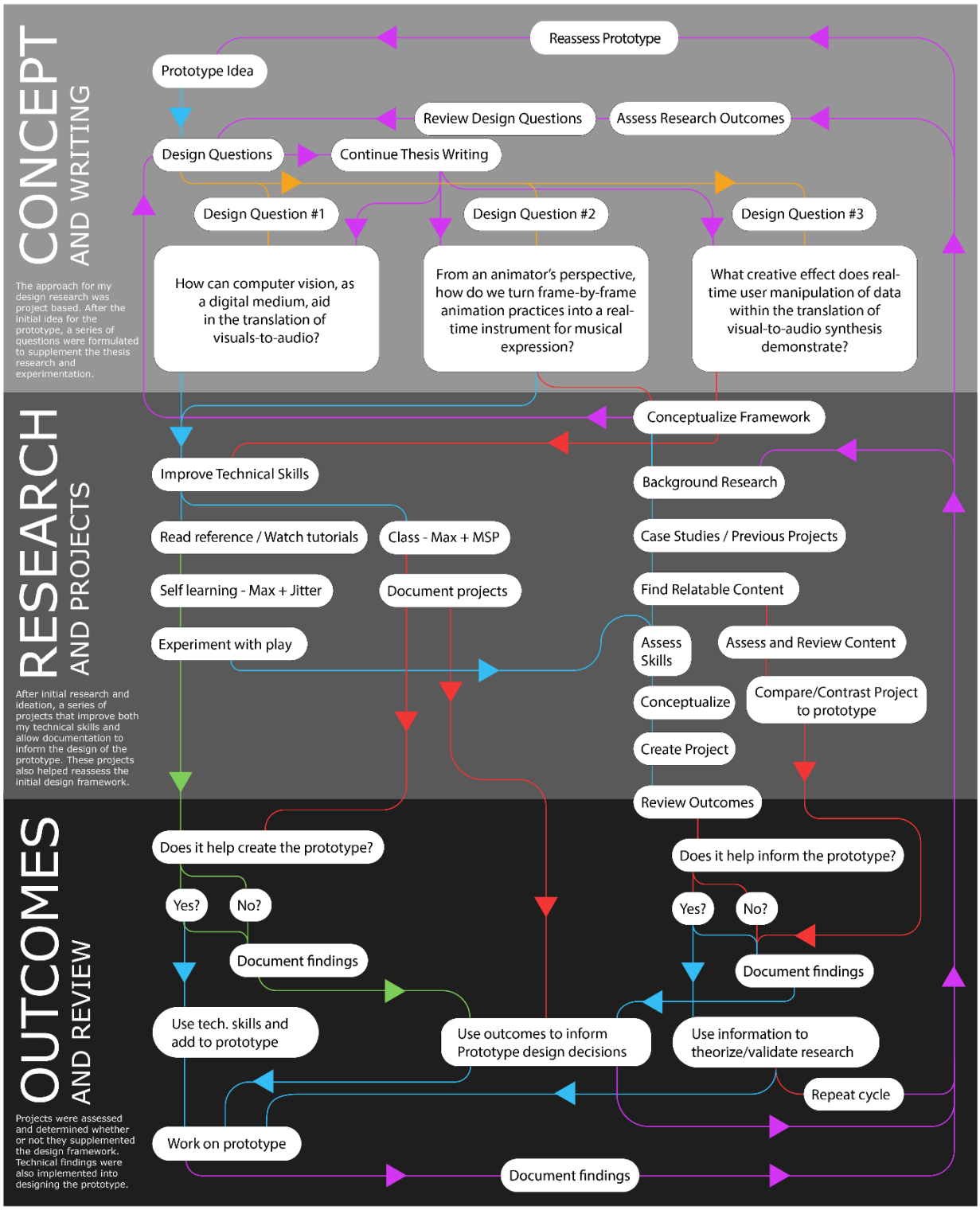


Figure 6: Design Process Flowchart

A flowchart of the cyclical process I designed can be seen in Figure 6. You will see separated sections in this flowchart; lines that are connected between each are considered input/output to each path. Arrows are used for the sake of organization and understanding how the information flows from piece to piece; colors are only present to help discern different paths and don't hold a particular significance. To break it down, I used 3 separate sections.

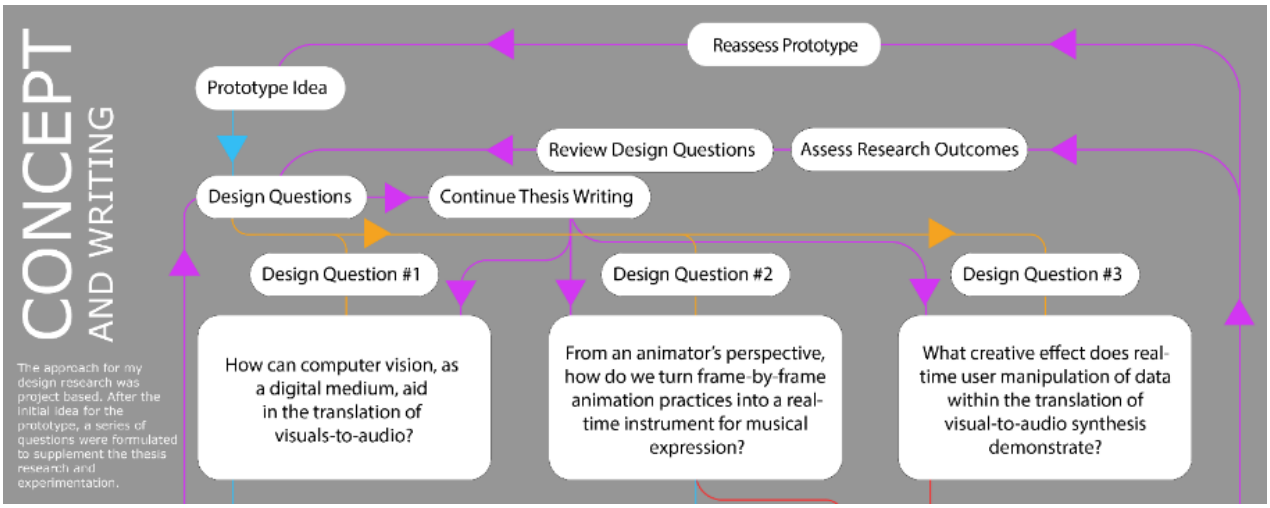


Figure 7: Design Process Section 1—Concept and Writing

The first is "Concept and Writing"; this section includes the conceptualization for the prototype, the design questions that were formed along with the dispersal to respected technical/research frameworks, and the assessment and revision of design questions derived from section three, "Outcomes and Review". Moving into the second section, "Research and Projects", I found that each question was focused on either technical skills or theoretical frameworks. The question involving the use of computer vision as a digital medium was aimed towards technical skill building. Regarding animation as a real-time process, this is targeted towards both technical skill building and conceptual framework

research. The question regarding the creative effect of data manipulation was almost always associated with the conceptual framework; parts of this question would be used as a necessity for designing the interface of the *visual-audioizer*.

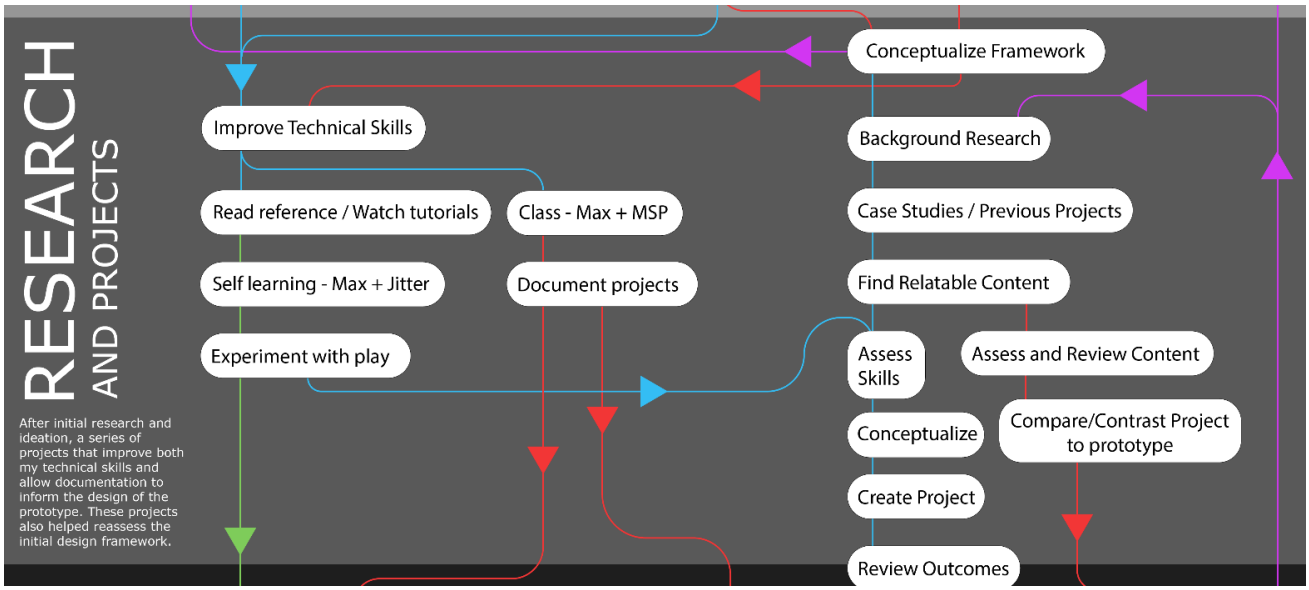


Figure 8: Design Process Section 2—Research and Projects

The second section is “Research and Projects”, as seen in Figure 8. This section is where the questions have been split into the two categories: technical skills and conceptual framework. On the technical side, the skill building that is mentioned is aimed towards understanding the Max coding environment. Projects on the right side either inadvertently informed my understanding of the Max environment or provided a reviewable outcome that might inform the prototype. When experimenting within Max, I realized there was a connection to the skills I would assess before each project. On the conceptual side, case studies and previous projects from past and present audiovisual creatives were reviewed.All the projects and within this section were documented for reference as well—as there was a need to return for assessment later in the design process.

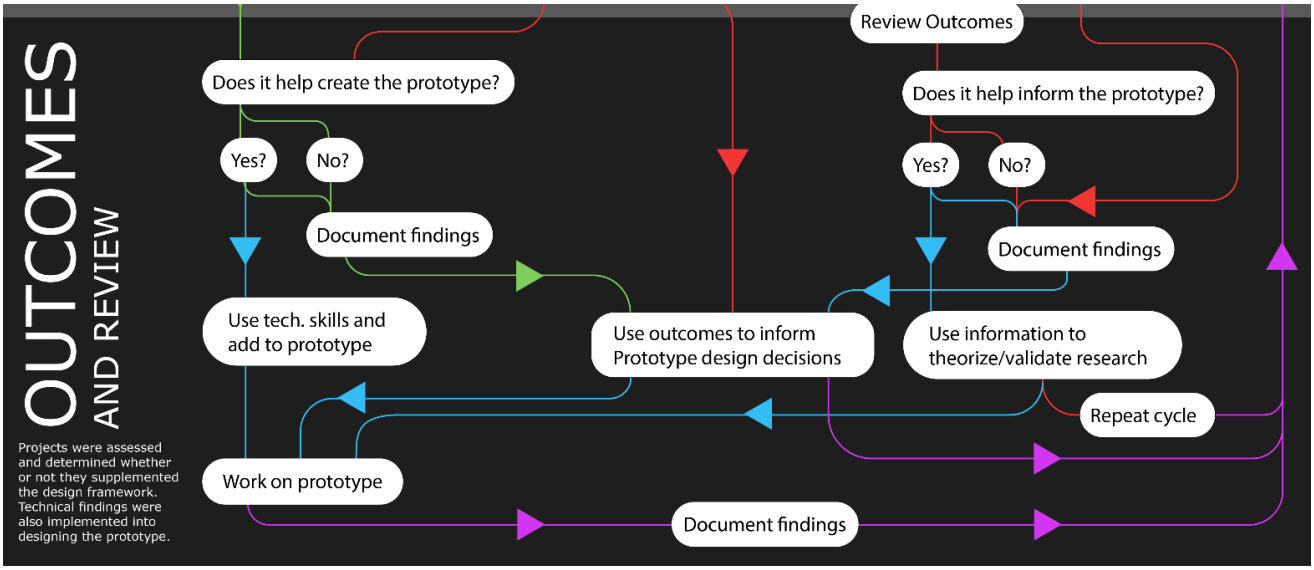


Figure 9: Design Process Section 3—Outcomes and Review

Section 3 is Outcomes and Review, taking all the information from the previous projects, studies, and experimentation, into assessment. As seen in Figure 9, this section also uses findings to inform how the prototype is designed; if there is a skill learned from experimentation and side projects within Max, it was most likely added to the prototype. When designing the prototype, outcomes from other projects would inform how the prototype would be designed—as well as information from research. When working on the prototype, there is the method of documenting the process; this is also done when using information to validate the research. After, I would repeat the cycle from the beginning. If you look at Figure 6 on the right-hand side, there are purple lines that flow from bottom to top. You may notice that one returns to section 2 (Research and Projects) to background research; this is to inform the next project that would be created that wasn't directed towards working on the prototype.

Concepts

As mentioned in my design process, there is a necessity to explain my research using different concepts, theories, and definitions that lend themselves as a contextualizing toolset. In the next few paragraphs, I will define animation in the experimental realm, a new method of animating called "fluid-time animation", computer vision, sonification, timing & spacing within animation practices, and "cooperative interaction". As a preface, story-driven animation/film is not what I will be approaching; and in the previous paragraphs the advantages of animation vs. film in this paper have been approached. Rather, in this research, we are focused more on the process of animating and the creation of a tool that allows said medium to be used as a musical instrument.

Concepts: Experimental Animation

In the realm of experimental animation, theorist Paul Wells states: "[Abstract] Experimental animation either redefines 'the body' or resists using it as an illustrative image. Abstract films are more concerned with rhythm and movement as opposed to the rhythm and movement of a character. [Experimental] animation prioritizes abstract forms in motion, liberating the artist to concentrate on the vocabulary he/she is using without the imperative of giving it a specific function or meaning" (Wells 1998: 43). In the case of my research, having a defined form (human, animal, anthropomorphized character) coerce the creation of sound is not necessary. Regarding Wells' statement, if I were to consider the ways in which the motion of an object affects the sound, it was imperative to work within abstract forms. This in turn let the proposed research to focus more on the motion within a

specific object(s), rather than entire character interacting within a scene, allowing for faster testing methods during the process of creating of the prototype. Giving the object the meaning of motion first, and letting the program interpret the movements of said object second, allows the extrapolation of audio in a purer sense. Using a character as a means of creating audio immediately sets a precedent, and an expectation, that a character will 'sound' a certain way. In the case of the ability of experimental animation to convey rhythm, this is where the medium shines. Having the hand of the artist play a role in the shape, opacity, and strobing of an object from the experimental standpoint allows the user to foresee what tempo an animation will have before letting it loose within the program. With a character, this is somewhat abandoned and becomes a trial/error game of: 'what number of frames will my character need to get from point-A to point-B and look natural?' The other which I am proposing (experimental animation): 'how can I break up point-A to point-B in a non-linear fashion?' I am not disregarding the effort of character animation, as most of the process required of such dictates a level of understanding within experimental animation.

Concepts: Fluid-Time Animation

Fluid-time animation, a concept in which I am observing and investigate in this paper, is where the process of creating animated loops removes the notion of a starting/ending frame. This concept also relies on creating hand-drawn animated forms as a real-time gestural process, rather than a frame-by-frame one. As an analogy, think of a blank piece of film tape—if the beginning and the end of the tape were somehow

connected, the notion of where the beginning/end of the film frame sequence is absent; if manipulated, this freedom allows additive manipulation and expression to evolve over time as the film repeats.

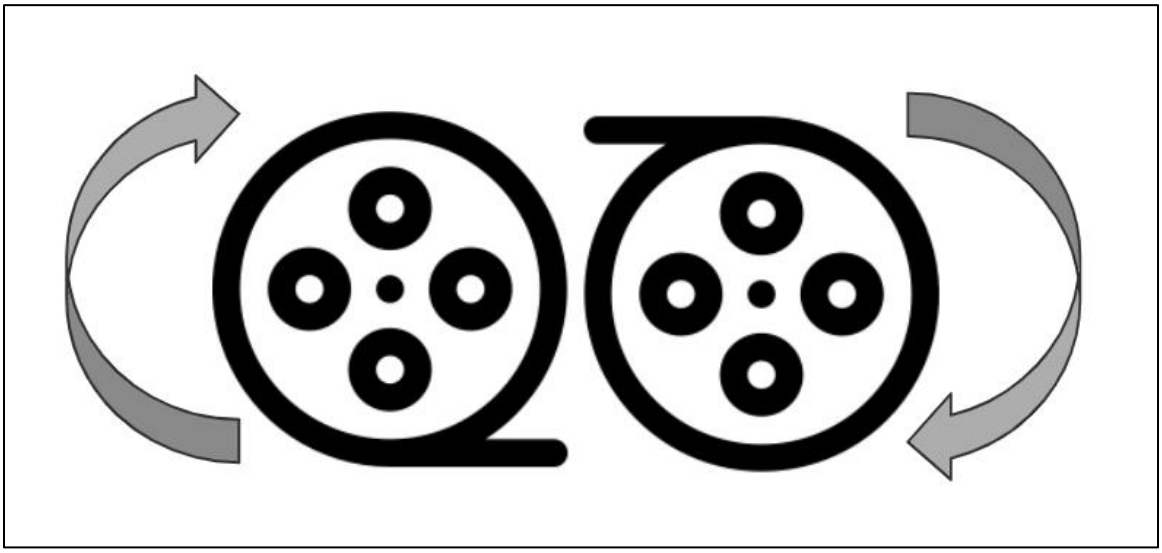


Figure 10: Endless Film Analogy

You as the user might experiment with the pacing of the playback, duplicate/cut sections, or rip it up and start anew. What makes utilizing the *visual-audioizer* unique is the ability of processing visuals in real-time and creating sound, allowing a non-objective animation workflow, and fluid-time, as a new method for musical expression. The concept was experimented with utilizing both the fluid-time animation software called *Looom* and the *visual-audioizer*.[7] *Looom* employs the use of a vector-based animation engine, a fluid-time frame looping system, the gestural ability from the user and a human interaction device (iPad and Apple pencil for example), and individualized layers techniques to create varied/editable animated loops.

[7]The app *Looom* from Iorama Studio: https://iorama.studio/

Concepts: Computer Vision

The *visual-audioizer*, a software prototype created withing a programming environment called "Max", partially relies upon computer vision externals within Max from Jean-Marc Pelletier.[8, 9] "Computer vision (CV) is the field of study surrounding how computers see and understand digital images and videos," as defined from DeepAi.org.[10] The purpose of utilizing the techniques of computer vision within the patch is to extract the positional x/y data, as well as the scale, orientation, elongation, and number of recognized forms. One of the numerous ways of utilizing CV, especially in this approach, is to convert imagery/video to pure black & white (no grey). These black and white values, when interpreted by CV methods, can be considered as the values 0 and 1 respectively. The CV method then finds groups of either the white (1) or black

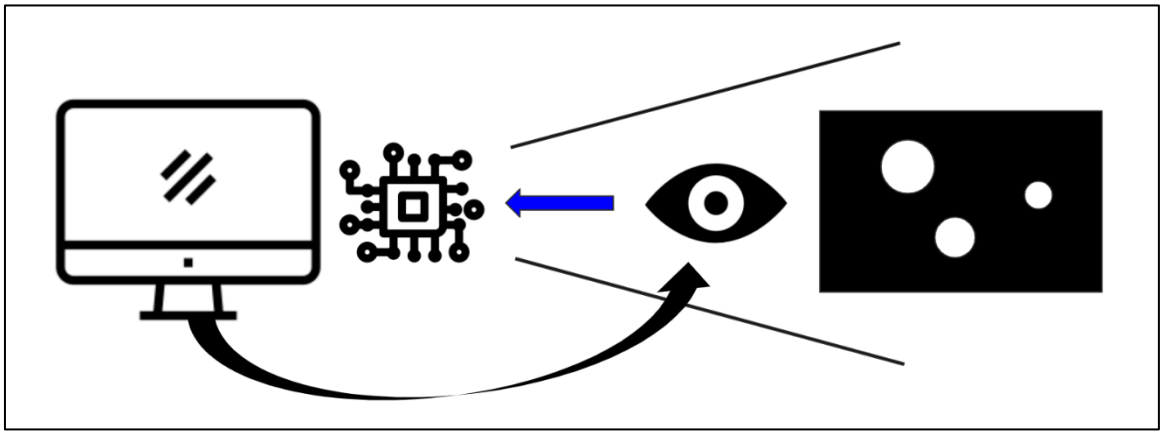


Figure 11: Computer Vision Example

(0) values based on a determined distance threshold and is considered as an "object" with a centralized position in an x/y coordinate space. Beyond position values derived from the CV system, considerations of scale (size of the form), elongation (how thin a form is), and orientation

[8]Max/Msp: a programming environment for "Sound, Graphics, and Interactivity": https://cycling74.com/
[9]Pelletier's cv.jit library notes: https://jmpelletier.com/cvjit/:
[10]https://deepai.org/machine-learning-glossary-and terms/computer-vision

(degrees of rotation) of a form can be observed and digitized. Considering the use of animation, this makes content creation a straightforward process by animating the form as a white object against a black background. The advantage of knowing how the system interprets data, in relation to the ability of the animator, allows the artistry (and the motion) of the form to create varying degrees of sound.

Concepts: Sonification

"Sonification is the use of nonspeech audio to convey information…the transformation of data relations into perceived relations in an acoustic signal for the purposes of facilitating communication or interpretation" (Bargar, Kramer, Walker, 1999). Forms, that do not move and are within our field of view, are generally silent. Only when the form collides, or moves, does the compression of air allow the creation of sound. Advantages of considering sonification to make inaudible data have perceptible means include our auditory ability to distinguish pitch, sound localization (position), and loudness

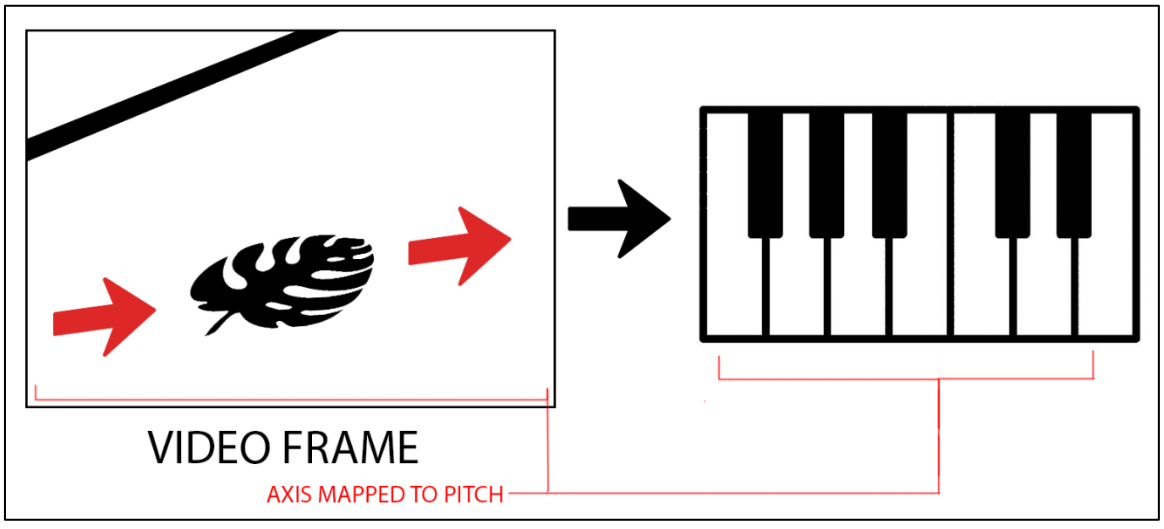


Figure 12: Sonification Example

(amplitude). Though sonification is generally a means of turning data into something audibly perceptual, much like data-visualization, the use of sonifying visuals within the field of animation is to provide another layer of depth to the animated form. The animated form, much like a still object, does not make sound on its own. Once audio is produced/recorded, we as creators can sync the visuals to the audio, or vice versa.

An example of visual sonification can be observed from Jean-Marc Pelletier in which the horizontal and vertical axis of a video have been mapped to pitch.[11] The video itself is a still shot of a river with a still branch penetrating the surface but remaining generally within the center of the image. As time goes on, small leaves float down the stream, cascading from left to right and creating a "glissando" from a low to high pitch.[12] We as the audience can follow the motion, as well as make the association to which object has created the sound. And though the sound might not be what an audience expects, it is the consideration that notable changes in visual information is what caused the audio to be generated in the first place. Pelletier states, "Since there is no single correct way to sonify an image artistically, the choice of the precise type of sound to use is left to the creator" (Pelletier, 2009). The *visual-audioizer* interface specializes in this consideration—allowing the user to experiment with the sonification of pre-made animated forms, or through a streamed source of visual input.

[11] An example of Pelletier's sonification experiments: https://www.youtube.com/watch?v=Z7btudUVT4E

[12] Glissando: a musical slide, either upwards or downwards, from one note to another.

Concepts: Cooperative Interaction

The consideration of utilizing the *visual-audioizer* and techniques of the animator as an instrument of musical expression can be supplemented from music psychologist Shinichiro Iwamiya; specifically the interaction between auditory and visual processing. When comparing the complementary aspects of audio and visual cooperative enhancement he proposes a concept called *cooperative interaction,* in which "each modality contributes to the evaluation of the other. In audiovisual communication, both modalities work together to make the product more effective." (Iwamiya, 1995) Iwamiya found that working with audio and visual spectrums almost always complemented one another but noted that if there was a clear association between the causation/timing of the audio and visual stimuli that an enhancement was noticed. For the case of using the *visual-audioizer,* all sound is derived directly from visuals – directly complementing one another on a 1:1 basis.

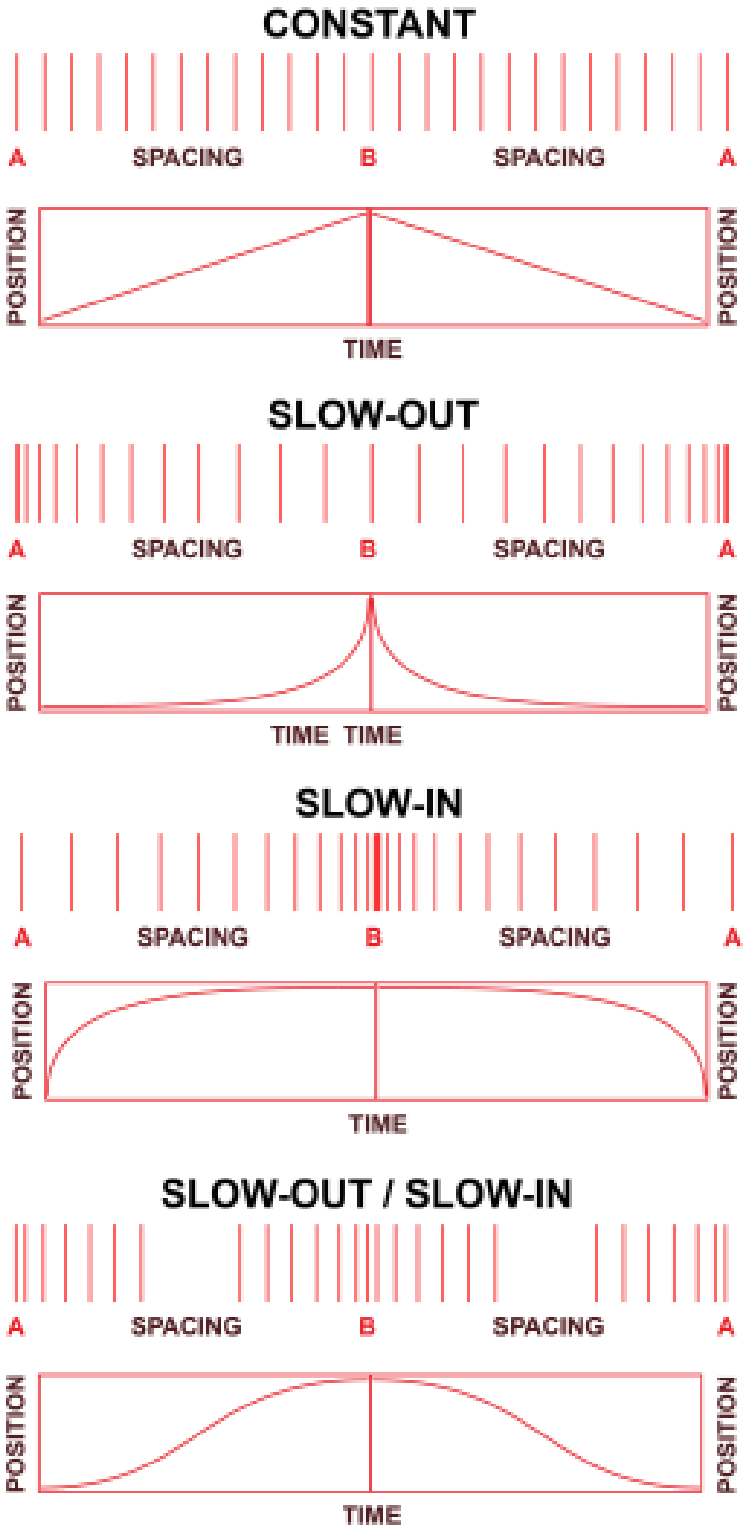


Figure 13: Timing and Spacing

Concepts: Timing & Spacing

While the main aspect of using this patch as an animator is to allow animated motion to be the sole determinate of digital audio making, the consideration of two basic animation principles make a dramatic difference in the elucidation of audio. As mentioned by Richard Williams, the creative mind behind *Who Framed Roger Rabbit*, animation is all about the timing and spacing of forms.[13] Using the *visual-audioizer*, these principles in consideration to Williams comment provide audio-driven insight into the differences in positional coordinates tracked by the patch. With this in mind, we can imagine two points: A and B. If a form was to translate from point A to point B (over a set time period) while the patch simultaneously tracks and outputs audio, the many ways in which the timing & spacing of the animated objects position can be creatively altered allow the animator to create more expressive sound with a the consideration of a glissando sound. An example of different degrees of motion are shown in Figure 13. Beyond frame-by-frame animating the translation of forms, it is up to the animator during a fluid-time performance to complement the *visual-audioizer* with considerations of timing, spacing, framerate, and visual ambiguity to create music.

Chapter Overview

In Chapter 2, I will begin by observing and reviewing past creatives and theorists who have had a hand within the realm of audiovisual synthesis. Their contributions were reviewed and used to inform the creation of the prototype. Chapter 3, initial considerations for the process of where to begin translating visuals into audio, the conceptual development

[13] Williams's "Who Framed Roger Rabbit" (1988): https://www.imdb.com/title/tt0096438/

process of creating separate projects along with the practice of animation as an instrument, and the documented process of creating the prototype. In Chapter 4, a synopsis of the information presented, with research assessments to the proposed motivations for the design questions. In Chapter 5, considerations and examples of using animation as an instrument, and possible uses of the *visual*-audioizer in multiple settings. The final chapter will also cover the current state of the prototype and expectations for the future.

Chapter 2. Background

Preface:

This section provides an overview of different attempts of synonymously joining these two mediums, from animators and computer musicians, with a focus in audiovisual experimentation, and compares/contrasts the current position of this research. Inspiration was often drawn from the projects discussed within the rest of this chapter.

The relationship between sound and audio has been an area of fascination to artists and composers throughout history. Most considerations for the use of the *visual-audioizer* comes from multiple sources of audiovisual artists. If we recall from Pelletier about there being no correct way to sonify a visual, there will never be a correct way to visualize a sound. The theoreticians and artists behind the works mentioned henceforth are ones who consider the manifestation of visuals to be complimentary to their works and have influenced the considerations of using animation techniques as a means of musical expression. Oskar Fischinger, an abstract experimental animator, used his own interpretation of sound as an expressive translation into forms by means of replicating what he would loosely describe as “mental imagery” that could be associated with the audio at hand. Norman McLaren, a famed Canadian experimental animator, once created an entire piece utilizing photographs of rectangular shapes fed into an analog optical soundtrack

(storing sound onto a filmstrip) that simultaneously linked with his visuals. Though effective, the amount of drawings it took to create a sound vs a single drawn frame was at least greater than 6:1 – a time intensive process.

The attempt at mirroring audiovisual synthesis settings had become an experimental film language within itself. Moving into the modern era, the ability of computers to give seamless real-time feedback, and direct translation of digitized audio via a digital-audio-converter, allow more intricacies in the realm of audiovisual expression. This concept coupled with computer graphics, means that the two have become more and more synonymous; using the computer, the artist and musician can become a single entity. And with the continuous rise of artistic expression in technological settings, this constitutes further exploration and a consideration that motion design in the realm of audiovisual synthesis can become a strong influence on musical expression. As I have explored and exposed myself to this realm, I have become infatuated with the idea that the techniques/toolsets of an animator can be used to create music.

Daphne Oram: The Godmother of Graphical Electronic Music

As a pioneer to the consideration that musicality is inherent within all of us, even if this does not include playing an instrument, Daphne Oram is a founding figure in electronic musical instrumentation. The project I am highlighting, an invention of Oram's (circa 1960s), much to her name's likeness is called the "Oramics Machine". A sound-drawing device that can read the drawn waveform gestures from a human on 35mm film; these drawings are then fed into a scanner. The scanner reads the ink from the drawings as

an electromagnetic signal, and outputs audio. In the purest sense, and from what my research has led me through, this method was a first step to creating a 'visual-audioizer'. Unfortunately, the machine is not easily available, and I wanted to make this project/concept more readily mobile and widespread than what 'Oramics' had to offer. Regardless, I found that the machine was not the most important aspect of this background development, but what Daphne had to offer when speaking about the human condition and musicality—with all the ability of our hands, Daphne wondered why shouldn't we consider drawing as an acoustic medium? And though she was not determined to find the 'perfect sound', her intent was more on the technological and creative aspects. What *could* be done and the exploration that followed was more important—rather than what, in her time, the market demanded.

Figure 14: Daphne Oram and her Oramics Machine (1957~1962) Image downloaded from http://daphneoram.org/daphne/

Though it was highly experimental at the time (1960), and gender-biased norms were still quite prevalent in society, Daphne was sometimes criticized for not following 'rules' of standard music; this dealt with patterns and planned compositions, harmonics, cyclical beats, and "motivic phrases within a defined structure…Oram favored the lucid flow of sound over time, the elasticity of rhythm and tone, allowing the sound freedom and space to take its course" (Hutton 2004). Oram herself states, "Maybe…by pursuing analogies between electronic circuits and the composing of music, we will be able to gain a little insight into what lies between and beyond the notes; we may be able to glimpse forces at work within the composer" (Oram, 1971: 5-6). Her work and usability of the 'Oramics' became "redundant", and unfortunately ceased for a long period following the 1960s, but Daphne continued to work with what she learned from her machine. Ironically, towards the end of her life, Daphne mentioned she was working on a computer software "that would incorporate drawn sound. This was not complete although it is hoped that her ideas inspired later developments in computer software for drawn sound" (Jo, 2003). This more mobile method, seeing as the machine was sizably comparable to a large office photocopier, would eventually be deemed the 'Mini-Oramic' machine. The machine was recreated in 2018 by PhD student Tom Richards; he mentions that utilizing this method of music-making allows us to consider what Oram saw and wanted others to experience with graphical music.[14]

[14]An interview/overview from PhD Tom Richards and the "mini-Oramics" machine: https://www.gold.ac.uk/news/mini-oramics/

Fischinger: Visual Poetry

Oskar Fischinger (1900-1967), a German-American animator, was often opposed to representational imagery. He strayed from the 3-act narrative structure that Disney was dominating at the time (even though he worked for him on a few different films as a cartoon effects animator) and focused on the mental imagery that became an association from the auditory rhythm that music held. The connection that abstracted animation holds in tandem with the rhythm and pitch of a song has an emotional appeal. "Fischinger…was perhaps more than the others committed to preserving film as art, that is to say, in Kandinskyesque terms, as pure form and colour, as a spiritual and emotional experience with the artist as prophet." Considering Fischinger was ahead of his time in the explorative mental imagery that is transposing audio to visual media, his dynamic relationships with music and animated imagery shape and change the way we associate our preliminary viewing and/or listening of the material. His visuals provided another layer of sustainability among past musicians whom already dedicated their life to the shaping of musical expression.

In works such as *An Optical Poem* (1938), Fischinger explored the concept that his works might have been unconditional experiences in and of themselves, much like the music would provide emotional/representational appeal when played alone. [15] The visuals synced up to Franz Liszt's "Hungarian Rhapsody No. 2" with pieces of cut out paper-

[15]Oskar Fischinger's film, "An Optical Poem" (1938): https://archive.org/details/1937OSKARFISCHINGERANOPTICALPOEM

circles hung delicately by wires.[16] The circles cascade, flow, appear/disappear, and move in fashions that only animated imagery and nuanced motion can capture. With the intentionality of being deliberate in the visual interpretation of the audio, this allows the visuals to establish a deeper connection to the timing, rhythm, and flow of the music. Fischinger's choice colors, shapes, and non-representational imagery attempt to visualize that which only our ears can discern as identifiable – but restrain themselves only to that of what the music has to offer. In a broad sense, he was interested in using the identifiable

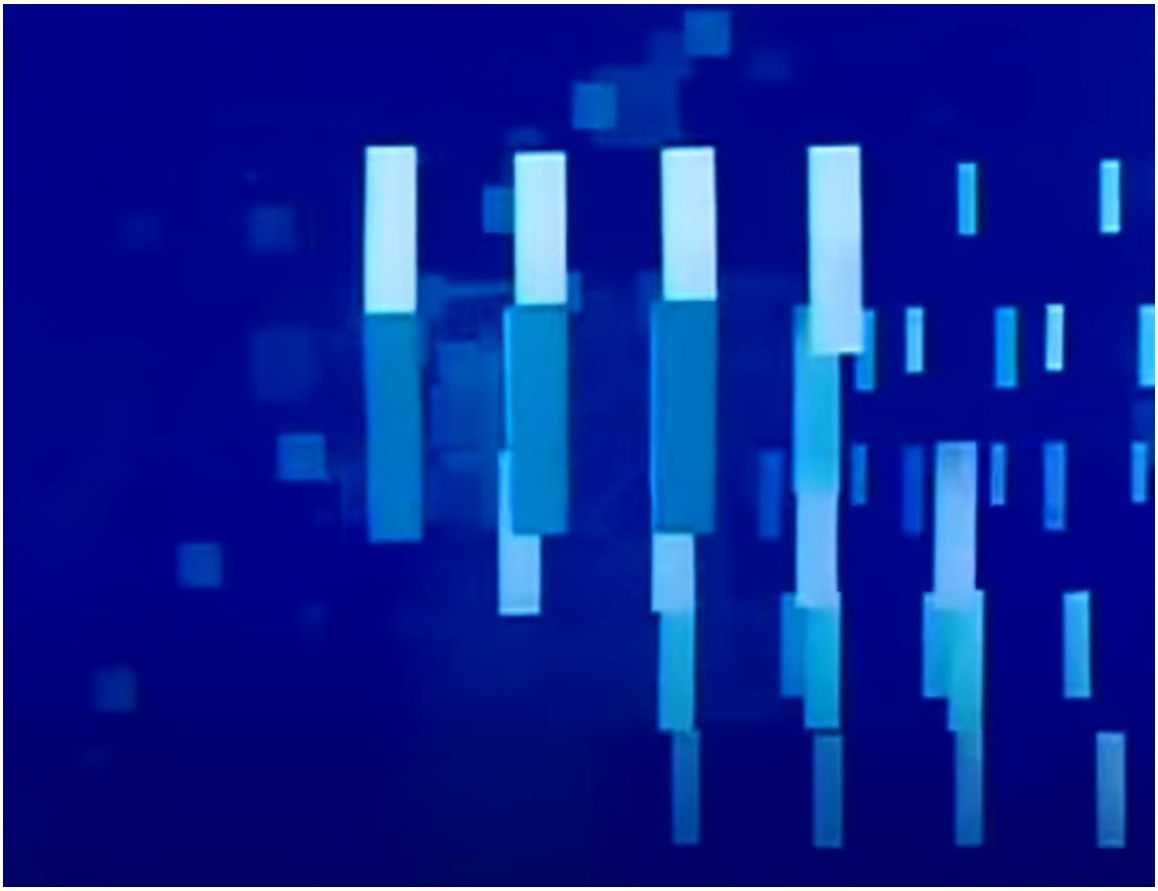


Figure 15: Still from Fischinger's "An Optical Poem". Video-image snapshot downloaded from https://www.youtube.com/watch?v=6Xc4g00FFLk

[16]Franz Liszt's song, "Hungarian Rhapsody No. 2" (1847): https://archive.org/details/LisztHungarianRhapsodyNo.2_689

traits of musical "laws" (rhythm, tone, envelope, harmony, timbre, etc.) as a means of visual expression. His works influenced the prototyping of the *visual-audioizer* by considering the establishment of mental imagery as a direct influence on the audio. It is deliberate, controlled, and explorative, allowing a seamless connection between the desired sounds and the realized motion. The "layer of sustainability" the visuals bring to the audio, as mentioned in the last paragraph, can be considered less heavily – as the *visual-audioizer* provides the animated form with the proposed layer of musical expression.

McLaren: Synchronicity of Visual-Driven Audio

Norman McLaren (1914-1987), an animation titan on the National Film Board in Canada for 40+ years, would create sounds in his animations to compliment his visuals. McLaren would "draw" the sounds onto the piece of the filmstrip itself that coincided with the imagery on the same frame – meaning he would manually put in the marks for a specific frame on the films sound strip.[17] An example of this is visually elongated shapes would sound shrill and high, while large shapes that take up space on the screen would be loud, low, and resonant. McLaren was not using the animated imagery as the sounds that would be made, but rather using this imagery as a basis to what his mind interpreted the sound a certain shape would create. Though his technique allowed him complete control over his audio, the amount of time that it would take to both draw the visuals and the sounds

[17]Norman McLaren's film, "Pen Point Percussion" (1951): https://archive.org/details/1951penpointpercussionbynorman_mclarennfb1080mp4

themselves was at least doubled.

There was a limitation of not being able to create sounds for each individual shape, but for the temporal moment at hand. His sounds had to achieve an overarching tone if there were many objects on the screen—or would have to prolong themselves to provide more information that a visual could not. Another limitation to consider is that as soon as the mark-making was present, the ability to readdress the pitch or tempo of the audio meant

Figure 16: Norman McLaren drawing sounds (1951) Video-image snapshot downloaded from https://www.nfb.ca/film/pen_point_percussion/

having to re-draw entire sections. Though McLaren could have just recorded different sound strips, this was not in the spirit of the artistry surrounding the simultaneity of the

visuals. Most of the time it was just small brush strokes that would create little blips of sound for each frame, but later went on to have long connected strokes on the sound strip for stricter control over the length of a sound. This provided a sense of depth within the animations, allowing the sounds to compliment the change in motion rather than be spontaneous and off-screen.

In an article from Kuihara Utako, an audiovisual theorist, he recalls a quote from McLaren, "What happens between each frame is more important than what happens on each frame. How it moved is more important than what moved". McLaren believed that the still of the frame was more about being a part of whole experience rather than singular. He stressed that animation has an ability to be used as an inquisitive medium rather than a source of reaction; he explored this intention through his film *Rythmetic* (1955).[18] As examined by Utako the said film, "was classified as a film not for "aesthetic pleasure" but for "information and education". I found that McLaren was attempting to express a universal language through his animations—meaning his use of animation and numerical values would be recognized in most (if not all) countries and would provide a shared connection between cultures. Using sounds, visuals and timing, Utako says, "we could be struck with wonder at the well-regulated placement of the figures and symbols and numerous calculations on the wall-to-wall screen, in addition to the continuously moving and changing nature of animation…In *Rythmetic*, the orderly enchantment and the ornamental one are organized from confrontation to integration". McLaren knew his visuals and audio would not work without one-another. If someone were to solely watch

[18]Norman McLaren's film, "Rhythmetic" (1956): https://www.nfb.ca/film/rythmetic/

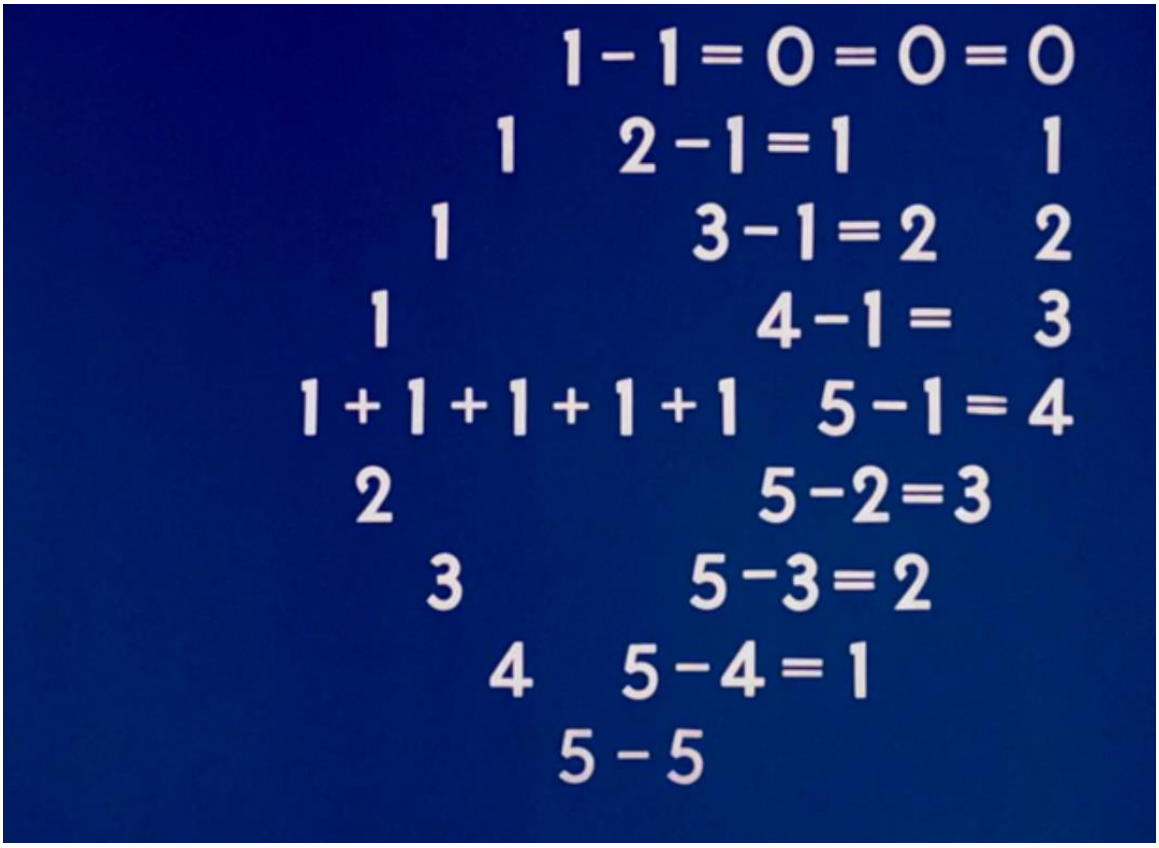


Figure 17: "Rythmetic" Film Still (1956) Video-image snapshot downloaded from https://www.nfb.ca/film/rythmetic/

the visuals of *Rythmetic*, temporal sense of timing and auditory expression would be lost. The counter-part, however, shows that while the temporal aspects of audio are notated with greater esteem, the visuals bolstered and gave meaning to the sounds attempting to mimic the motion. Utako closes his observations with, "*Rythmetic* is an invitation to the aesthetic confusion of order and disorder by figures and symbols; and besides, it is a temporally designed artwork both in the visual and auditory aspects, more so than an educational film or arithmetic lecture".

What is it about animation synced with audio that provides an innately deeper connection to synchronous happenings? Something to consider with McLaren's technique

is during his time of animated cinema the filmstrip is limited to only one soundtrack. This disconnect that he had to consider when creating his visuals meant that he could not draw multiple sounds for multiple shapes. With today's tech, the ability to parse out individual digital sounds from sonified imagery and map them to different sources means a wider array of options, rather than the limitations found in filmstrips.

Chapter 3. Concept Development

The ability to manipulate the data translation between audio and visual became part of my research focus during my first year in the DAIM program. I created projects that were synced to foley sounds, beats of music, and dialogue. I explored animated looping practices and considered how the creative process for creating these loops finds a correlation to creating music. Using these loops, I found software from other designers that extrapolated audio from a picture; I created an animated sequence with the derived audio, syncing up the audio/visuals to each frame of an animated piece. Moving forward, I then utilized sound and visuals as an interactive game that relied on one another; this provided solutions to feedback response and made a positive case for the synchronous nature of audiovisual media mapping methods. Considering the multitude of ways that a system would sonify visuals, I decided upon utilizing practice-based research as my primary design method in working through these projects; and how they ultimately informed my final decisions towards building a *visual-audioizer* prototype.

Previous Explorations and Projects

Using a practice-based research process, I undertook a series of projects that helped build my technical skills while simultaneously examining the methods for creating and the outcomes of each project. Because of this, I found that the projects themselves continuously informed the initial concept of a *visual-audioizer* prototype. Each one drew inspiration from other animators, programmers, and audiovisual theorists—while working through similar practices, each provided its own challenges to overcome and considerations for the

prototype. Some of the different topics for these projects include: audiovisual music, animated loops, rigged character animation,

Animated Music Video

The first time I had thought about creating this interactive media system was during my first semester at ACCAD; the project assigned was to create an animated music video. In the 2-weeks allotted our group (of 3 people) created, rendered, compiled, and edited the footage to sync up to the audio.[19] After reviewing all the methods used to create visuals, I saw a pattern of having difficulty for our separate animations to sync up without time-remapping. This was solved eventually, but during that moment I took a step back and considered what the animation would "sound" like on its own. Turning the sound off, and just watching the visuals, there was in a sense a rhythm to the animation. The completed

Figure 18: Animated music video snippet

[19] Group Music Video: https://vimeo.com/333549068

animation itself was strengthened by the audio, encapsulating a pulsating rhythm and shifts in color, playing with a loose focus on the story of the short and a stronger focus influenced by the beat of the music. Some lessons learned from creating the video included the difficulty of coordinating everyone's efforts towards a singular goal, compiling the footage in a conducive way to aid itself during editing, and finding a method to sync up the visuals to the beat of the audio. Also, because of the limited time and resources during its development (two weeks), we created/rendered footage from Maya and compiled everything into After Effects.[20] To sync the visuals to the audio a few constants had to be initially considered. First, we were aware of the frames-per-second of our overall composition, as well as the beats-per-minute of our song; 24 and 140, respectively. Using an online "FPS to FPB to BPM" calculator, our group found that every ~10 frames our visuals should "shift" in some way to reinforce their purposeful syncopation with the song.

The resultant video felt effective in its final iteration; but, this was short-lived going back for review. I felt a connection between my mental capacity to translate visuals for audio; but to consider the process in reverse proved difficult. The video felt too gimmicky– like it was too deliberate in the execution. An issue I take with creating visuals for an already-made musical piece, one that also has no dialogue, it all too often takes the turn of animating to the "beat" of the music. This method makes it easy to know when to transition to a new visual but placates any tension from attempting to come up with a narrative.[21] This can be both a blessing and a curse–I noticed our visuals were filled with harsh edges

[20]After Effects: a software from Adobe for video compositing, animation, and motion graphics.
[21] It should be determined beforehand whether the visuals are supposed to portray a narrative or be experimental in a professional setting. In this case, due to time restraints and working within an academic setting, we chose experimental.

and different loops of animated footage that didn't hold any sort of artistic weight on its own, until it was heavily edited and placed into the musical sequence. Through this realization, I felt the desire to what an animated visual would hold on its own—I wanted to strip away all the extra layers from narrative-driven stories and find innate connection between animation and audio.

Looping Animations

Part of this "stripping away" included a creative project in which I explored animated loops as a method of maintaining attention and inviting contemplation beyond the scope of frames that the piece utilized. For example a single second (let's say 12 frames) animated loop has the ability to maintain the human gaze dependent upon the way its visuals weave and wind through one another. Victoria Hart (ViHart on Youtube), a self - described "recreational mathemusician". Her videos typically entail the connections drawn between movement, repetition, music, and math. In a certain video, Victoria takes the weaving of lines and draws a squiggle—this squiggle has two basic rules: end where it started, and make sure any crossings are distinct and noticeable. After this, start putting over/under bridges on all of the lines that cross one another; notice how it always weaves perfectly.[22] For example, if I were to start doodling and had a continuous line that also ended up at the same spot as when it started, I would be able to weave an "over-under" pattern throughout, and it would ultimately end in its correct orientation. The pattern will never have an "over-over" or "under-under" section if the criteria is maintained and will

[22] Doodling by Vi Hart: https://vimeo.com/147913560

always end with the opposite layered section thus making the shape a perfect loop. Something else to note: when you complete a squiggle loop, if you place another on top of it and repeat the over-under process, it will still be a perfectly closed loop. Finding inspiration and a connection to maintaining the human gaze and the complexity these patterns could take, I decided to implement these into an experiment with utilizing motion to provide another layer of depth.

Initially I created different shapes within my notebook and made each one a closed path based on Victoria's videos, and then took the path and drew shapes on top of it using a light-box at ACCAD. To accomplish the method of imbuing motion into these squiggles, I transitioned into utilizing an animation technique to translate the paths into animation motion. The technique to accomplish this, coined as the "weaving-loop" technique (by Caleb Wood), is the way in which you utilize the final frame of your content as the starting "onion-skinned" frame to the beginning of your animation timeframe.[23,24] Caleb found that if you create a specific number of frames for an animation beforehand, you can go directly back to the beginning and keep adding more drawings than before, while still maintaining the same number of frames; allowing you to create animations that are endless in nature, and let the audience ponder upon where it was started. Next, I took those shapes into photoshop and found that if I just used shape-after-shape (butted up right next to each other) the animation was stale and boring. It almost looked like a string of Christmas lights, so I had to reference a technique from *The Animators Survival Kit* (Richard Williams)

[23] Weaving Loops: Caleb Wood https://vimeo.com/208751134
[24] Onion-skinning: an animation concept where frames of content are transparent and on-top of one another. In the digital realm it acts as a piece of paper, almost like a thin piece of vellum or an "onion-skin".

based on “overlap” in animation to create a more cohesion between the frames.[25] I considered the audio for the difference in tangential forms vs. overlapped ones—how could this method create/alter an audio signal? I then started to lengthen certain sections of shapes to give a clearer sense that they were overlapping each other and were part of the same continuous path; and pushed it to the limit in a certain corner of the initial shape that had about 5-6 overlapping lines. This resultant visual was cluttered and hard to follow. I imagined if this section took up a large portion of the screen (perhaps over half), what the sound design would entail. Flowing and repetitive? Or noisy and uncoordinated? In terms of creating a short looping animation consisting of about 12 frames and didn’t need any post-processing for it to “work”, I succeeded. But it felt empty without a soundtrack, sounds, or any sort of diegetic source.

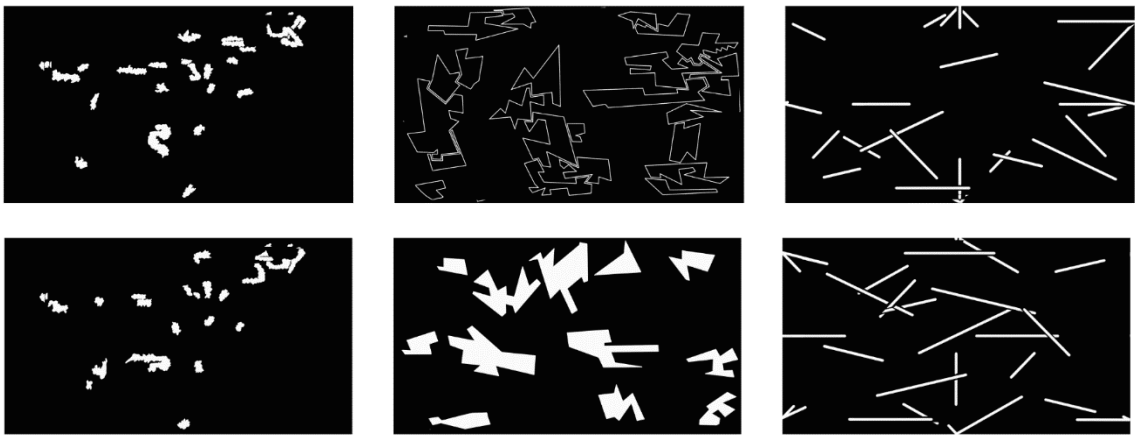


Figure 19: Looping animation frames example

After working through a few other looping animations, 8 and 6 frames respectively still encompassing a second of content, I decided to overlap and juxtapose the different pieces together. This eventual cacophony of all the animations together appeared as

[25] Overlapping Animation Example: https://www.youtube.com/watch?v=4OxphYV8W3E&vl=en

something of a “breathing” mass; the other frame rates making a visual polyrhythm to the timing of the frames in relation to one another. This “breathing” inherently held a rhythm, as the animation had been organized in such a way; much like a piece of music is organized and performed/replayed. There was an inkling of the idea at the time, considering this connection between animation and music (while being cognizant of animation as an instrument), considering how to make animation as an “improvised” form. To understand how “improvised” works, consider the musician able to play music instantaneously—the animator has to plan and coordinate. Much like improvised music, jazz for example, how can a form that relies on the sequence of (traditionally) planned images be transformed into real-time audio producing instrument? This question would arise later but began as a starting place for the prototype ahead.

2D Characters and Timing Considerations

To aid in understanding the complexity of visual forms and the translation to audio, the next project that I worked through was creating a 2D rigged original character and utilize such in an animated short. The character itself was an anthropomorphized bird, complete with rigged body controllers and mouth shaped phonemes; the name of this character is “Chuck”. Regarding my research, the technical aspect of this project was to practice matching the mouth shapes to the spoken text; the conceptual goal was to personify this bird character utilizing existential philosophy and humor. The conceptual goal held less of a significance than that of the technical goal. Because of this, I will focus more on the explanation of creating the character and matching the phonemes. Using practical

drawings, I designed Chuck while attempting to be cognizant of body parts that would be moving, like the arm, legs, and neck portion of the body. This process included understanding how inverse/forward-kinematic rigs on a 2D character would affect the joints that use rotational motion, while attempting to maintain the profile of the character to match that of the original design as much as possible. In the realm of creating the phonemes, the same process of practical drawings came into place. Making the phoneme shapes of the mouths to match that of the spoken narration became an issue as the mouth of this character was a beak. From what we as humans can observe from the beak of a realistic bird, the most prominent visual we see is that of the mouth open and closed. Getting the beak to simulate that of a human mouth was challenging; I ended up taking some artistic liberties and adding in visible teeth and a tongue to certain mouth shapes to emphasize the spoken words.

Using After Effects, Chuck was rigged using the "DUIK" toolset, from Rainbox Labs, and the "Joysticks and Sliders" plugin, from Mike Overbeck. [26, 27] The DUIK toolset is used to create animatable character rigs, utilizing similar functionality to that of a 3D character skeleton rig. The plugin from Overbeck was used as a function to control pre-drawn facial expressions, and the animated timing of mouth shapes. Let us consider the visual and auditory connection between mouth shapes and speech: an aspect to be cognizant of while recording for an animated character is the desired frame-rate of the animation. I say this towards the notion of how fast one speaks, as having 12fps vs 24fps

[26] Rainbox Labs "DUIK" toolset. https://rainboxlab.org/tools/duik/
[27] Mike Overbeck's After Effects plugin, "Joysticks and Sliders". https://aescripts.com/joysticks-n-sliders/

regarding the dynamism of changing mouth shapes will be altered. Talking at an advanced rate, let's say 5-7 syllables per second, with a 12fps frame-rate lends itself to be more difficult to work with than 24fps. The reason for this is the visual and auditory connection that is made if the speech is diegetic during viewing; vococentrism can also be a factor in this, as a perceptible hierarchy is established when compared to other sonic elements. [28, 29] If we hear speech and can see that the syllables are not matching the mouth shapes of the animated character, our mind loses the connection between the two aspects. Rather than feeling the unification of audio and visuals as a simultaneous media, our brain is left to stitch together the broken connection and fill in the gaps. This consideration can also be linked to the ability of our ears to process information faster than our eyes.

A problem arose when animating Chuck's facial shapes, as I like to speak quickly when recording myself. Rather than synchronize speech and visuals down to the millisecond, it is good practice to give a 2-frame buffer for the visuals before the audio (2 for 24fps, 1 for 12fps); this allows our eyes to process information and lead to a better feeling of synchronization between the two. I ended up using 24fps for the animated short, and slowing down my rate of speaking, giving myself more flexibility to work within a larger supply of frames to match the audio. That being said, using every frame (24 per second) as a different mouth shape ends up feeling aimless and unrealistic—its visual density becomes overbearing and uncomfortable to watch; the brain ends up focusing more

[28] Diegetic sound: a source of a noise that is on-screen and matches our visual/audible reference.
[29] Vococentrism: a term coined by audio-visual theorist Michel Chion that considers the human voice as the pinnacle of perceptible audio even when combined with other sources. If a voice is introduced, it becomes the top of the hierarchy and is usually regarded as the most important/recognizable sonic element.

on the audio since it can keep up, and discards the visual.

Developing the project with Chuck, and experimenting with mouth phonemes, led to some considerations for the prototype of my final project:

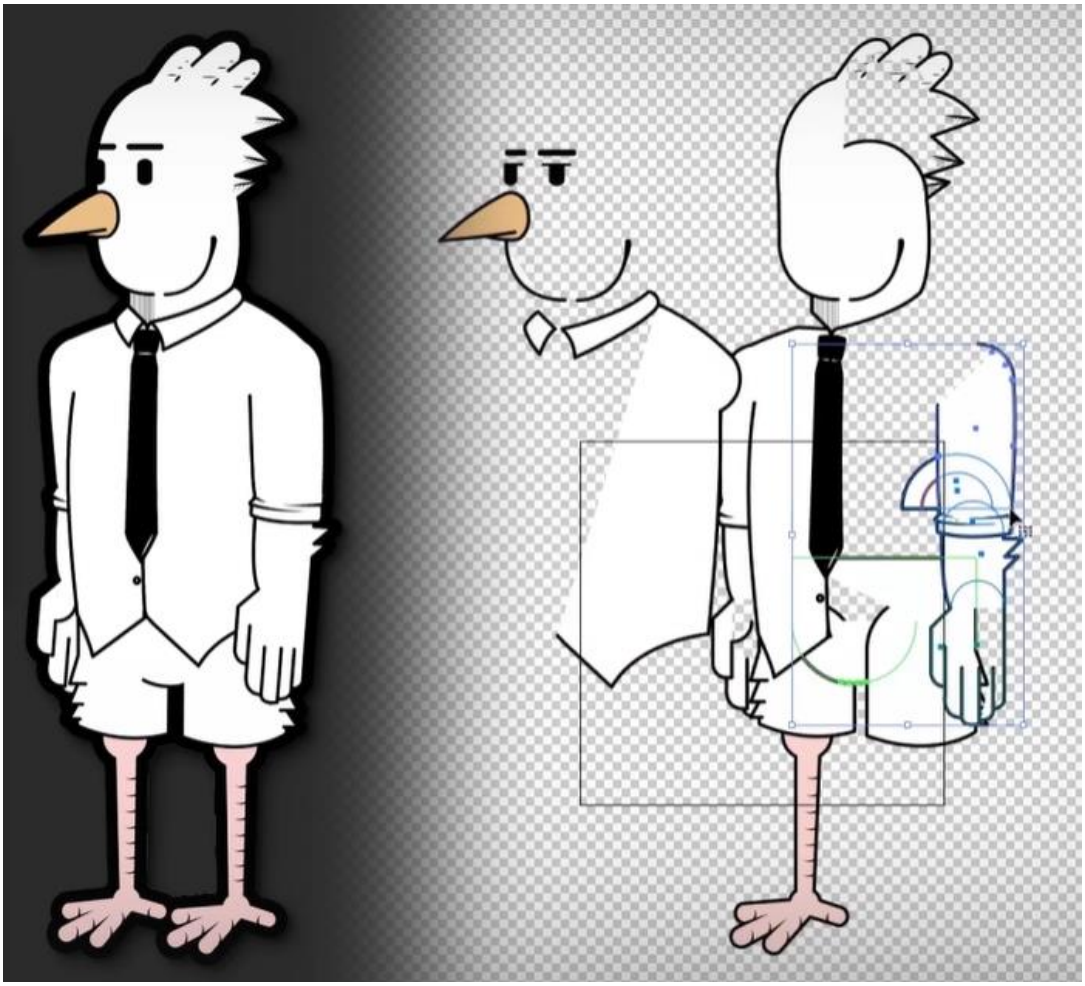


Figure 20: Chuck character example

1.) Since I wanted the audio to be derived from the visuals, I needed to consider how to make the delay between the visual processing and the sonic output within a reasonable time of synchronization. The first consideration was processing the image beforehand and outputting it to a soundtrack, then layering the two together and playing it back. How to process the imagery in the first

place was another caveat, as I had no prior experience to understand CV methods. Because of my limited knowledge in the area, I had to determine whether to simplify the meth od of visual processing and consider non-objective forms as the basis of CV processing.

2.) During the introduction of the video with Chuck, he asks the audience to read aloud the words on the screen. An attempt was made to have an audience be more involved in an animated film. In this case it did not work, in my opinion, seeing as the animation was already pre-determined and would not change depending on the audience reaction. If I were to have the audience feel like their interaction mattered, that would require a signal or an obvious change due to their contribution. And though part of animation includes watching/observing, I was motivated to allowing a user/audience creating the experience of animating for themselves, rather than be a passive participant.

3.) I wondered how to make the process real-time, possibly allowing the method for deriving audio to be dynamic and interactive—letting the users/creators animation methods to inform their generated audio. Having the process be real-time felt important as it how animation is observed in the first place—while one could certainly watch the process of someone animating frame-by-frame, the outcome of animation itself is focused on a series of images presented over a length of time.

Non-Objective Animation and Infographic

During this time, I took an experimental approach to considering the ways, and this grammar is redundant, in which an animator animates non-objective forms. This was step in a different direction compared to animating Chuck, as the sounds would not be driven by the metaphysical connection between Chuck's observed mouth movement and the audible speech. This time, rather than creating a narrative and attempting to follow along, I decided upon breaking down animation principles to their basics; eventually combining the methods into their own categories. These basics include a total of 12 principles of character animation; originally created by Disney animators Frank Thomas and Ollie Johnston. The principles of animation served as a guide for the artform of imitating life as we know it. [30] The "12 Principles of Animation" (alphabetical order) and an explanation are as follows:

1.) Anticipation: this has to do with the moments before an initial movement. This can also include our expectation of an outcome occurring after this motion has played out.
2.) Appeal: deals with the 'quality' of motion. This could be derived from a character trait, or from our expectations of the physical world. Framerate comes to mind as well; more frames for a slow-motion sequence vs less for a quick action are good examples.
3.) Arcs: rather than draw a motion from start to finish in a straight line it is always more dynamic to reinterpret the motion as an arc in some way. It can be subtle

[30] Frank Thomas and Ollie Johnston's animation book, "The Illusion of Life: Disney Animation" (published in 1981) is utilized in academic institutions as an introductory step in understanding the complexities, and the effect of one's artistic decisions, of character animation.

or apparent and exaggerated.

4.) Exaggeration: this principle deals with overextending our perception of a path of motion. This could include how the ‘appeal’ of an object’s movement.

5.) Follow-through/Overlap: understanding if the drawings ‘overlap’ one another there is an easier perceived motion. Speed of motion can also be compared in this case, as quick moving objects (edge to edge outline) shouldn’t overlap one another to give a stronger sense of fast momentum. Follow-through is a technique in which until the moment a motion is completed, it should remain visual within the frame.

6.) Secondary Action: this means having another source of motion within the frame rather than that of just the main character/object. This includes how the main action can affect the space around it.

7.) Slow-in/Slow-out: this is a technique in allowing the frames to have some overlapping at the beginning of an action or as an action comes to a halt. This is more often called ‘ease-in/ease-out’. It gives more weight to an action rather than instantaneously moving or moving at a constant rate.

8.) Solid Drawing: understanding aspects of maintaining volumetric qualities of a drawn form. Or, how to manipulate them for more expression within animated forms. Ties in with Follow-through/Overlap, giving weight and density to the animated form.

9.) Squash & Stretch: ties into ‘exaggeration’ and ‘solid drawing’ principle. Can include aspects of drawing disproportionate volumes in the animated form for

more expression during an action/reaction.

10.) Staging: to make decisions based upon the space utilized within the frame. Where does the motion move to? How does the animated form play/interact with its surroundings? Does the motion inform the cinematography of a scene?

11.) Straight-ahead / Post-to-pose: an animation technique in which the motion of an action is either drawn in either fashion. Straight-ahead means to draw frame-by-frame, to what you the animator determine, to be the beginning/middle/end of a motion. Pose-to-pose means to draw the 'extremes' of a pose, the most important looking silhouettes of a sequence, and draw (how much is to the animator's discretion) the 'in-between' frames.

12.) Timing: can be considered in the sense of framerate. But it is more important to associate this principle with the number of frames that make up a motion; as well as the spacing of the frames when creating.

Through these principles, I considered if each would be valuable to influence audio creation. This was an interesting dilemma, as the animation principles founded by Disney animators were aimed towards character animation, and not necessarily mapped to that of non-objective techniques and audio generation. I decided to utilize these principles but had to simplify them in order to consider an animation process that could match the speed of a real-time output. Considering the process of characterizing Chuck, drawing and animating characters is a complex and time-consuming method. So much so, this method would not necessarily work within a real-time instrumentation atmosphere. The next project included a guide in which to simplify and utilize the 12 principles into a streamlined way.

To begin, I first determined which principles I thought were the most essential to grasp when the animated form is non-objective. I grouped these principles into four separate categories. First, the ‘Action’, or when the motion begins. Second, what aspects to consider when a form is in ‘Motion’. Third, how this affects the things around it, as well as the form itself, the ‘Reaction’. Fourth, how the ‘Layout’ of the motion translates across the frame, how an artistic flair comes across, etc. To reiterate, I used these four categories in the end: Action, Motion, Reaction, Layout. Each category would have 3 principles; the means in which I also associated principles to categories, from least to most difficult (in my opinion), were as follows:

1.) Action	**2.) Motion**	**3.) Reaction**	**4.) Layout**
Anticipation	Arcs	Exaggeration	Staging
Timing	Follow-through	Squash & Stretch	Solid Drawing
Straight-A / PTP	Slow-in / Slow-out	Secondary Action	Appeal

The next project I decided to pursue was towards an animated infographic, showcasing my own take from the categories above. As one reads through the infographic, the complexity is raised to match that of the principles associated with the categories. The ‘Action’ section shows just a dot moving from left to right, up/down in a zig-zag motion. The action is constant, and one has an expectation that the path will change as the dot approaches the edge. The spacing of the dot per individual frames is also constant. In the ‘Motion’ piece, the dot’s path becomes arced; as well as having a variation in the timing of the motion, causing a slow-in/out of the motion. The form appears/disappears in the

bottom left/right based on scale, also signifying the follow-through principle as we see the complete motion. For the ‘Reaction’ panel, it shows more advanced motion and forms. The dot is now a morphing shape, bending to its motion, stretching, and dramatically

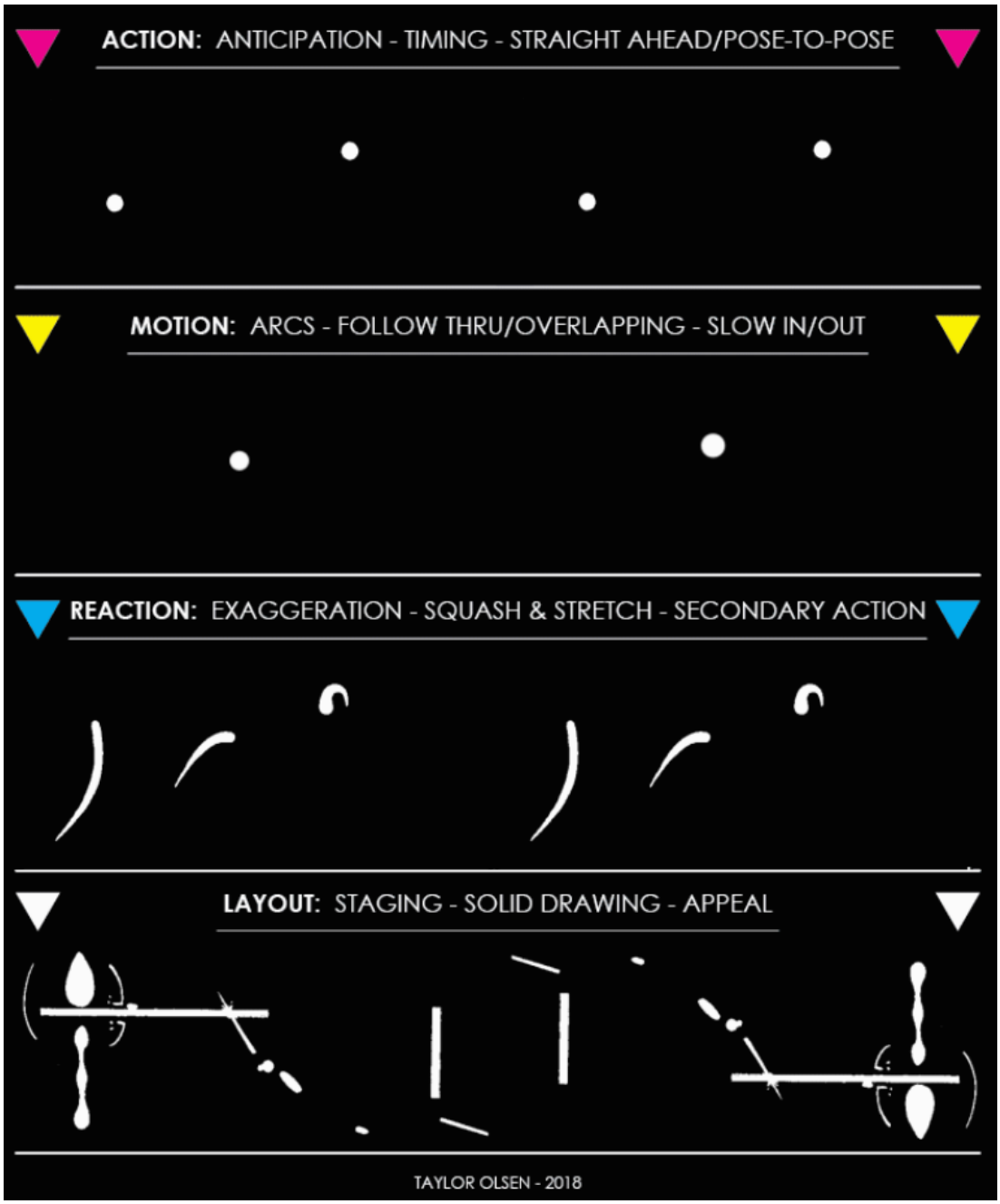


Figure 21: Still from animated infographic

exaggerating it's spacing between frames. 'Layout' includes taking all these principles and playing with the space around them; allowing the animation to loop on itself, have different shapes, and utilize the space in an interesting way. The infographic is also playing on a loop, allowing one to review the categories/principles and consider the differences between each step.

Creating this piece helped with my technical skills, especially considering the non-objective techniques. If I were to consider using this infographic for the visual-audioizer, I needed a reference for how the changes in the way I was animating would have an influence in the data being read from the visuals.

Though, I considered whether more complex imagery would produce different results, I decided it mattered more about the overall shape of the form, rather than the detail. This idea coupled with how the forms timing/spacing alters our perceived motion played a key role during the process of the final project. If animation were to become a closer to real-time process, and interpreted as an instrument simultaneously, having the tacit knowledge of animated motion becomes a cyclical process. The animator draws, the system reads, the animator hears, then reinterprets how they want their animation to play out. There are other factors to consider, such as orientation of the screen and how the data is mapped to the frame, but those are considered later in the development process of the prototype.

Sound-Splitting and Frame Manipulation

Moving on from these last few projects, I created a prototype of ways that visuals can be translated to audio, rather than how audio transcribes into visuals. This project was decisive in the conceptual relationship to audiovisual synthesis, as well as my own personal research. The interaction between them is crucial to the exploration of animation language; finding a common ground that is intuitive, and possibly interactive, is the goal I had be looking for. I began with crude explorations from individuals who have previously attempted to create a software that explores this notion of translating visuals into audio. Victor Khashchanskiy, an audio-software professional, created a program called “Bitmaps & Waves” that scans static imagery and is transcribed into black and white numerical values. [31, 32] The program takes these values and attributes them to frequencies along the audible spectrum of human hearing; the resulting output is a direct correlation to what the programs scans to an audio file.

Figure 22: Hand drawn animated loop example

[31] Victor Khashchanskiy, "Bitmaps & Waves," Bitmaps & Waves, accessed March 15, 2019, http://victorx.eu/BitmapPlayer.htm.

[32] Mila Vasileva, "Images To Sound": https://www.youtube.com/watch?v=WgZ01bAOMMU.

In my first attempt, I began with utilizing the sound produced from the software and splitting the tracks into separate pieces, play through each image (a 12-frame single-second loop over 24 frames-per-second) for 2 frames with each respective audio track, and repeat once the 12 frame cycle is completed. [33] This interpretation of visuals to sound is not what I was necessarily looking for but was critical in my own research and the development of the prototype. The process of creating this piece was not nearly real-time, and the way the piece was 'read' was scanned like a piece of music. The project was not the 1:1 relationship I was not hoping for. The imagery had to be made, scanned, processed, sonified, coordinated and edited within After Effects, and exported. This was unnervingly time-consuming. I wanted the process to rather be as soon as I begin 'animating', not as an afterthought.

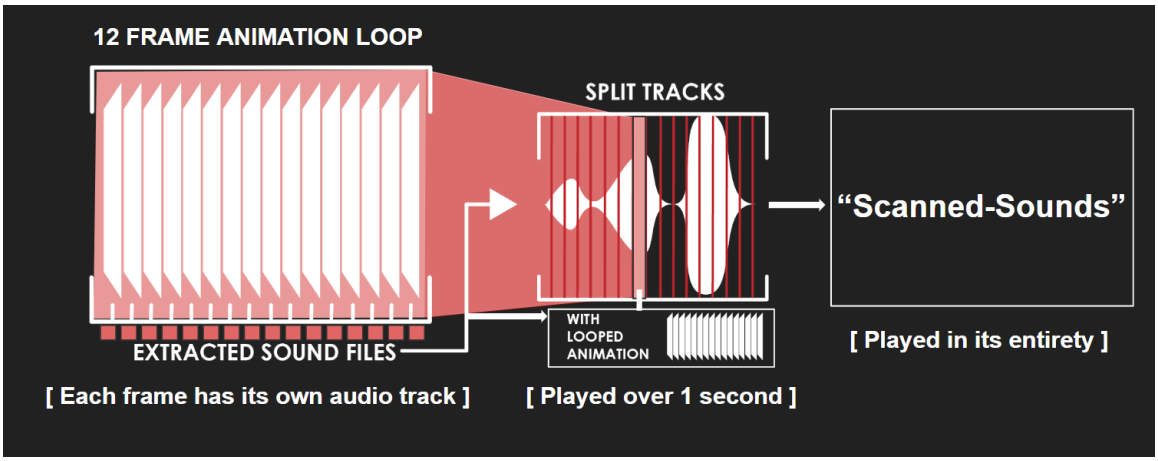


Figure 23: Scanned sounds flowchart

Part of the point of my research is finding the validity in utilizing animation as an instrument. If I were to consider how I wanted the visuals to be 'read' by the patch, I needed to remove the idea that a scanned 'line' would become the only method to work with. I saw

[33] Taylor Olsen, ""Hearing Visuals – Testing – 2018": https://vimeo.com/311110400.

the scan-line as a limitation rather than expressive. It was only the animation itself that provided that expression in the first place; but, reducing the elicited audio to only a fraction of the animated imagery, shown at a given time, meant the animation and the 'canvas' of the frame had to adhere to where the position of the line moved. Another issue with this method is this: hypothetically let us say we play frame 11 out of 12 at timecode 10seconds. Inherently, each frame holds its own sound bite for the entire image, but if we delegate a specific frame at a specific time we are not utilizing the full animated imagery. Only a piece of this imagery, in this case $1/12^{th}$, is being utilized. The line does not stop and play each of the 12 frames. It plays one, moves on, and plays another. In the spirit of animation, the line is not working 'with' the imagery or showing the whole frame, only what it sees at a given time. During this, it is reducing the imagery down to a thin sliver of pixel information to elicit audio; when we watch the line moving across the frame, this is where our focus lies: the sound from the line.

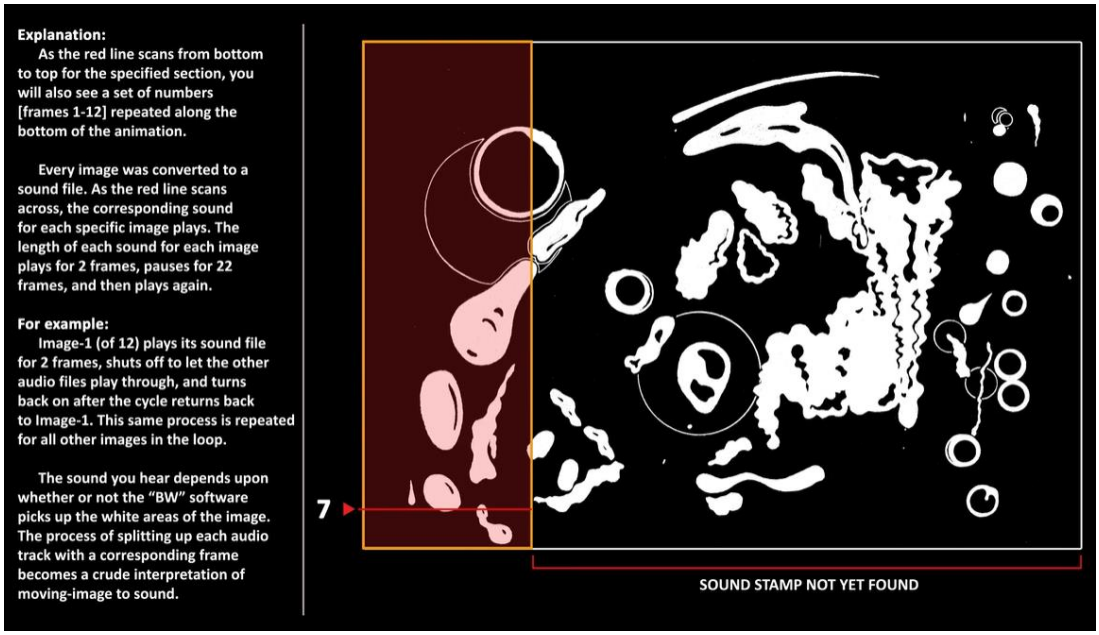


Figure 24: Still from my example, "Hearing Visuals" (2018)

I asked others if they noticed the animation during the same time the audio was playing, and they only focused on where line was in the piece, rather than the animated loop. I found that I, and others, were observing 'parts' of the visual experiment, rather than the 'whole'. This could also be due to my own part, seeing as the process took so long to only sonify a small portion of the entire frame, and I deliberately explained via text in the video what the line represented. I did keep the entire animation present during the experiment, and the information provided via text was a quick read. As a result, I felt this project was the first instance in my process of sonifying animated content.

Exploring "Max"

Onward from the exploration of Victor's software, and in the field of audiovisual synthesis, the program Max/Msp is an industry standard among computer scientists, computer musicians, software engineers, and academics the like—graphic artists can explore within this programming environment as well. [34] The first aspect of Max that I learned was the ability to control how fast information was processed. The lingo for this information processing in Max is called a "bang"; every bang is a separate message. If you wanted to send a list of the first 10 numbers out of a list of 100, this is feasible. You can also control how fast this information is sent—among the myriad of other possibilities.

Investigating further, I found the program is a sandbox for interactive media, data manipulation, and signal processing. For this aspect I spent time (about a semester) teaching myself this program's data manipulation before moving into signal processing,

[34] Max Mathews, "Max Software Tools for Media | Cycling '74," https://cycling74.com/products/max/.

the audio portion of the software; this came later during a class with one of my committee members, Marc Ainger. Through the understanding of both methods (data manipulation and signal processing), and utilizing the Max environment, there was still a need to analyze visuals and output desired information. I would later attempt to explore other interactive programs, such as Troikatronix's Isadora, further into my studies; but at the time found that Max was the most forgiving and well documented in how to utilize its node-based programming environment.

The class began with easy-to-grasp explanations of the science behind sound and how to utilize those concepts when generating audio within Max/Msp. As a preface: sound is in fact the expansion and compression in a physical medium such as air. The vibrations from the air are perceived via our human eardrums, which the brain can then interpret as sounds. Molecules bump into one another to create these vibrations—it should not be mistaken as flowing from one place to the next, this would be considered 'wind'. It is important to remember the expansion and compression aspect.

For sound waves, the class examined a few different concepts: waveforms, amplitude, wavelength, and pitch. The easiest way to represent this is plotted to an x/y graph; imagine a smooth rollercoaster going up and down. From the top to the bottom, this can be considered amplitude, or the "loudness" of a sound. If the peak and the trough are increased respectively, the sound will become louder. If we looked at a plotted x/y graph of a waveform, and examined the peaks, we would find air molecules packed closely together, vice versa for the troughs. These are the vibrations that create sound. Wavelength is quite literally how long a waveform is—from one peak/trough/middle to another.

Wavelength also influences the frequency of a sound; frequency can be considered as a note on an instrument—like a ‘C’ or an ‘A’ on a piano. From the analogy, the number of times the rollercoaster moves from top to bottom within a set time is considered the frequency or the “pitch” of the sound.

What made using Max/Msp vital to the prototype is the ability of the software to sequentially read in data, interpret, then outputs data/audio. The data side deemed the ‘Max’ portion of its likeness, is all about data manipulation and reading. The interface is a visual-programming environment, making it easier to use than my current understanding of text-based coding environments. For myself, I know that I find it difficult to code, but one of the features of Max is “objects”—considered by other visual-programming environments as “nodes”. These objects already hold a specific set of instructions, a coded ‘function’ per say, with reference material as a guide. These different objects can then be connected via an input/output fashion; the connections are called ‘patch cords’. This allows for a workflow that can be seen from beginning to end. Max also allows the ability to place ‘watchpoints’ on every patch cord, making it easy to find errors throughout your patch. The digital audio side, Msp (also known as Max signal processing), has the ability to change data into a waveform. Say for example there is a patch that works like a simple addition calculator. We could change the resultant number into a sound. Say ‘2 + 2 = 4’, the resultant ‘4’ could be scaled to 400 and changed to a frequency—400Hz. You could also change one of the 2’s to be a dynamic input, allowing you to interactively change the

resultant pitch.

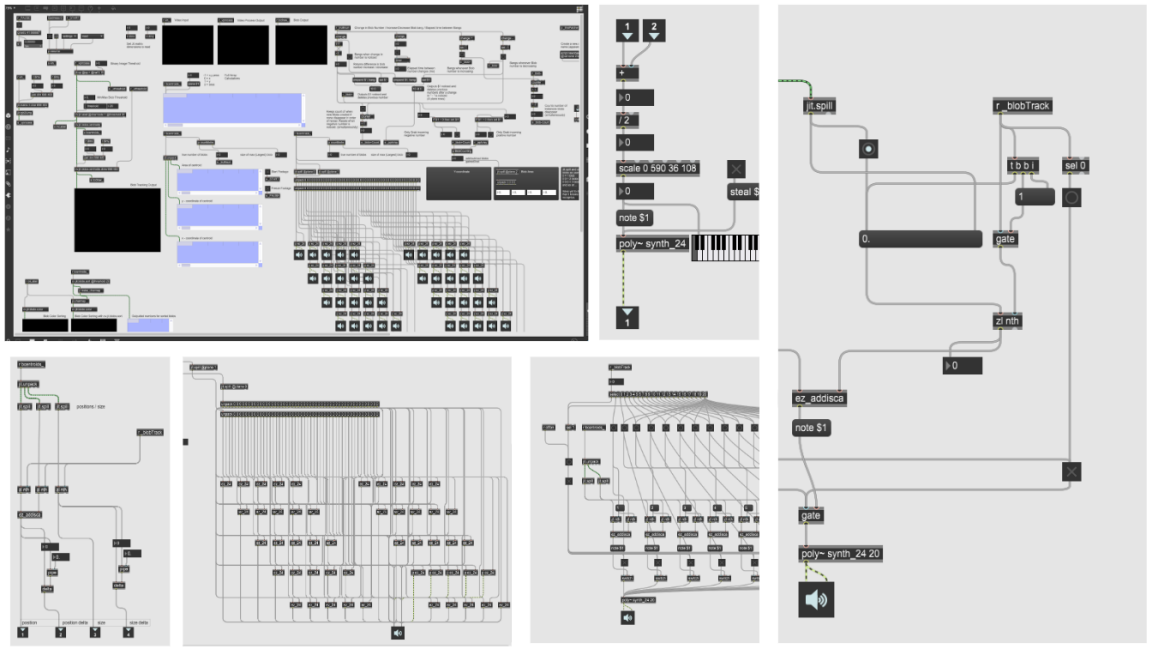


Figure 25: Max interface examples

A significant aspect of max that makes the patching environment expansive is the ability to "encapsulate" patches. This means to take a selection of objects at your own discretion and compress them into its own separate function within a patch. This encapsulated patch can also receive data and electronic signals. This ability to compress patches becomes invaluable if one begins to work with large patches. One step further, and you can save the encapsulated patch as a separate patch—this patch can then be called upon depending on the users' needs. Organization becomes a key factor in recalling how one of your patching environments work; knowing where to return and make fixes makes organization even more necessary. Labeling via comments, and version-saving, makes the workflow of Max like an iterative design notebook.

Grasping these concepts and the patching environment of Max, I prematurely dove into a realm of video-graphics and data processing; I decided to reverse-engineer the

method of audiovisualization to my advantage. I say prematurely because learning curve, from the straightforward additive synths we had been creating to working with data arrays and matrices, was drastic. I found myself in a rabbit-hole of information; I eventually had to keep a separate folder of each patching attempt so I wouldn't forget certain methods. Much like coding, node-based programming is very logic oriented. Meaning the deployment of messages being sent at one time, certain events happening before others, "if this then that" considerations, and a myriad of other processes.[35] I ended up finding what I need from the class: create a synthesizer for the *audiovisualizer* that would derive it's signal information from the computer vision data. The synth portion would change over time, but I achieved what I was theoretically looking for. Further into my understanding of Max was the ability to communicate to other programs—regardless if this is coming to, or leaving from, Max. This ability is achieved through "Open Sound Control" messages, otherwise known as OSC.[36]

There was a brief period where most of my time was spent learning more technical aspects of Max, but the next project supplemented the idea of user interaction and real-time audio feedback. This related back to the design dilemma of making animation a real-time process for the sake of 'instrumentation'. I wanted to use the OSC messaging system to interact between different programs. Because most of the initial work for the *visual-audioizer* was being able to properly ready and digitize image data, and more time was spent to flourish design decisions using project-based research examinations to inform the

[35] "If this then that", a programming concept where an event happens due to the cause of another. "If this" 2 is greater than 'x', "then that" will cause something to happen. If not, another event will be triggered.
[36] Open Sound Control: a communication method between different multimedia devices. Can be used for live shows due the ability to continuously stream data and/or flexibly control multiple programs at once.

prototype.

Isadora and Interactive Audio

During my Fall 2019 semester in a class called "Devising Experiential Media Systems", I spent time attempting to develop an experiential space that allows an audience member to experience animations in an ambisonic and spatialized environment.[37] This would include utilizing part of the work I developed from my Max patch as an interface while the audience member experiences the animations in a new way. To aid in the creation of this project, I learned how to work with the program Isadora. This program is a queue-based object-oriented software aimed at utilizing live-video techniques. Though Max/Msp is similar in nature, it is not necessarily known to be versatile with video applications, but it has a place among audiovisual users. Learning Isadora happened faster than expected, about less than a month, considering the similarities between the two programs. Some of the only differences included the flow of data from left-to-right (Isadora), and top-to-bottom (Max/Msp)—it should be noted that there are more in terms of layout, organization, and interface options, but the fundamentals remain the same. Isadora is also more upfront about it's input/output information, labeling each parameter visibly for the user to see—within Max you must hover your mouse cursor over the input/output of the object and wait for a small caption to appear.

As for the project, the animations were projected onto the ground, and the speaker

[37] Ambisonics is the science of understanding how to spatialize sounds in a 360-degree setting. It can be under headphones, or through speakers. But, the effect of using this system gives a feeling of atmosphere from being 'within' the soundscape—being able to notate and pinpoint where the sound is coming from.

setup would utilize the positional data from the CV techniques and attribute the data to the corresponding speaker. This means as animations playback, the source of the sound dynamically changes with it. This project, the final for the class, was part 1 of 3 total "Cycles". During Cycle 1, I devoted my time to this system, though it ultimately changed. I learned that the amount of time I would need to develop this was too long, and I would have to develop a different set of skills regarding projection, ambisonic science, and advanced mapping techniques. Another reason why I did not go with this project, is that I had already found the consideration for a similar experience that could come from a screen rather than be projected. Being under headphones was an option and hearing the panning from left to right could provide a somewhat similar experience rather than unnecessary space/equipment. Finally, the only point of interaction within the space was to change the videos around—this was less than desired as I wanted the interaction from the user/audience to bolster the experience, rather than just 'viewing' it.

During Cycle 2, the project developed from being an experiential space, to an interactive game that utilized human touch and audio. Since most of my research included finding methods to elicit audio from visuals, I wanted to return to the idea of the audience member to consider how animation could be an instrument, and how the use can manipulate the visuals in this way to create music. This project informed why I wanted the creativity of the user's gestural movement and animation techniques to influence the sound, rather than solely focusing on the user to make the connection from pre-rendered videos with the elicited audio. The idea for the game was an interactive piece that uses the body as a "human tuning-fork", almost like a game of hot and cold. The method would create two

separate points on an x/y plane, and each point would have an individual pitch. One would be controlled by the player, and the other would be randomized and generate a constant tone. For the goals of the game, it was left to the player to "match" the tones together. To do this, I used Pythagorean's theorem [$a^2 + b^2 = c^2$] and some simple math to determine the distance between the two points. This distance was then scaled to a dynamically changing note influenced by the players movements; as the player came closer to the position of the constant tone, the two pitches would completely even out, and the player would gain a point.

I used Isadora and Max/MSP to communicate with one another; the data from skeleton information via a Kinect, a body tracking system, would transmit from within Isadora. The program would then send the messages via OSC to Max and would therein transmit a sound based on those OSC messages. As it would turn out, correctly mapping the data from the Kinect skeleton in multiple settings proved unwieldy and somewhat buggy. The data from the skeleton was easily retrieved but was not always consistent. This is not to say that it does not have its uses, as I used the skeleton data and a "drawing" actor within Isadora to create lines on a "projector" actor. There is also the ability to read multiple skeletons, though it might have worked better if the height of the Kinect was raised above 7 feet, angled downward toward the audience. The Kinect also had the ability to use the depth information using an infrared sensor; this could be used to limit the field of information streamed from the Kinect. Ex: rather than observe everything within a 0-10ft range, we could observe only 2-6ft. Though the Kinect had its advantages, I struggled to find the validity of making a game that was not as responsive as I had hoped.

Cycle 3 of this project used Isadora to have the game be interactive via touch. In a similar fashion, Isadora has its own object-based programming interface. Rather, they are called “actors”. Within Isadora, there is an actor called the ‘Mouse Watcher’; this actor tracks and outputs positional x/y information according to the computer screen resolution, though these. These values can be scaled to how one wishes; as such, I would transmit the data to Max and create an audio signal. Anywhere the mouse was moved the pitch of the sound would change. It was enjoyable to play around with this interaction but felt shallow considering it was not necessarily leaving behind a mark, or a trail to work with. I ended up creating different modes of difficulty within the game to test out the use of a visual cue. The visual cues informed the player of whether they were closer or farther away. Some examples included placing a dot where the cursor was, and dynamically changing the size of the dot depending on the distance from the goal (farther = larger, closer = smaller). There was also the addition of a trail that would show the player’s previous movement, change size, and better inform them where to go next.

After getting the functionality of the system to work, I moved it to a large ~40” touch screen. I found that using a computer mouse allowed test players to ‘cheat’ by just shaking the mouse around the screen until they gained a point. This method of playing eliminated the idea of listening and reacting to the sounds and ended up as erratic movements—one could play and beat the game without any audio at all. The larger screen made the interaction take more time; the participant was no longer using a mouse either, rather their finger or a dedicated touch-pen. I in turn shifted the amount of time for each difficulty to a smaller amount to hopefully make them use both the visual and audio cues

to pass the game. Also, or the sake of someone who was not adept at audibly ‘tuning’ in the game, the visual cues provided necessary insight to win. One could easily beat easy/medium difficulty with just the visuals. But those who did use the visual cues more than others eventually found the use within listening during hard mode. This mode had no visual cues other than the cursor dot which was not scaling dynamically. Each player was then “forced” to listen in order to win. Though I did not record this in a quantifiable way, asking the participants about this was often met with similar responses like, “I was only watching at first, I couldn’t beat Hard mode. Over time I started listening and it made playing the game easier.” This informed me that as the player used the system more over time, the association between the motion of their movements and the output of audio became stronger. Participants had no desire to return to the easy/medium modes, as their understanding through experiencing the game made them realize the validity in ‘listening’ to their movements. For myself, and playing multiple times, I would close my eyes and still be able to complete the game; the experience eventually felt natural.

The Visual-Audioizer Prototype: Intro

With the knowledge leading up to the prototype from previous projects, and a better understanding of the Max environment, I found there were methods of live tracking with visual input. Most of the methods lead me to methods utilizing computer-vision, but most were also plagued with caveats of coding skills I currently did not have. Most of this dealt with attempting to learn a completely new coding language—and yet Max is technically considered a coding language. Max has a wonderful mode called “Presentation Mode”,

where you can dynamically change and readjust interactive systems. This mode also allows you to “hide” all of the clutter underneath of your patch, and view only the aspects that you’d want someone to interact with—like buttons, dials, switches, lists, etc.

I was wary of attempting to layout all this content in a coherent way, I doubted my ability considering all the options. This almost made me turn away from the project entirely. What made me come back is that the system, in theory, could be completed. The systems in theory are usually easier said than done; I could not comprehend the layout within the current timeframe. I took a step back and recollected on what I felt my prototype should accomplish:

- Create a Master switch
    - Turns the entire prototype on/off
- Allow full control of video choice and playback:
    - Forward, stop, reverse, playback speed, etc.
- Allow the ability to switch between different modes of visual input:
    - Video, Desktop stream
- Full control of audio switches:
    - On/off
    - Gain, frequency, vibrato, modulation, waveform, mapping interface
- Option to record the interaction
    - Limited recording times (10seconds as a test)

With more investigation, there was a community of users dedicated to using a package from Jean Marc-Pelletier regarding computer vision methods within Max. The

package would use the “Jitter” matrix, the visual side of Max, to digitize visual media into a “matrix” of information. This matrix of information is comparable to a spreadsheet—every row and column correlate to a certain pixel in comparison to the entire frame. Depending on how the information is altered and read this pixel can hold information regarding color, matrix position information (where it is located on the ‘spreadsheet’), transparency, hue, saturation, brightness, black/white, etc. All this information was not necessarily utilized in the prototype, but easily could be mapped to different values when manipulating audio. There is also the ability to read the mean, median, mode, minimum/maximum values of the matrix; this is a regular feature within Max, but the ability to control these values via the Jitter interface allows for precision and numerous creative possibilities.

So, to being working with the patch, I needed to read a movie. This meant creating a video player within the patch and streaming it into the Jitter matrix. This matrix did not have any external parameters and was playing at full speed and resolution—this would be dealt with later; parameters often come as an on/off switch, or as a list of information. For the moment, the movie was also playing sound (unintentional, but fortunate to catch)—within Max you can target certain parameters with a “message” box. In this case a “mute” message was needed, or a “vol 0” parameter. There is a need to “activate” the video via a “trigger” option. With this movie playing, there is the ability to only output new matrix information if there is a change in at least one of the pixels—this was achieved using a parameter called “unique”; this prevents extraneous frame output from being sent to the Jitter matrix. This would be necessary due to optimizing the information from each frame.

After reading in the movie, within the Jitter matrix, there is the ability to then manipulate the information of each pixel. This is accomplished via “rgb2luma”, an object that takes the information of each pixel and changes each RGB value to an attributed lightness.[38] This could be controlled via the total amount of information from each color value—a threshold. If you wanted a certain total sum to be considered enough (via the threshold), this would be converted to a pure white pixel, or a “1”. If not, a black pixel, or “0”. This information is then sent back into another matrix before it is sent into the computer vision objects. So, at this point every pixel of information is either a zero or one; when using the package from Pelletier, CV.jit, the information is read into computer vision algorithm. [39] This algorithm can determine the distance between each activated, “1”, pixel of information within the matrix.

If there is a grouping of these pixels, this is then read as an object. If there are multiple objects, these are assigned a number depending on a certain mode. This object can have a few different pieces of information: an assigned number, a size (or scale), position, orientation, and an elongation valuer. This information can then be used to the user’s advantage—my own mapping experiments followed this consideration.

The assigned number is determined in two different modes: largest to smallest object, or position-based naming (from top left to bottom right of the screen-space). To have an object with an assigned number, the object must have a minimum scale compared

[38]Luma: = luminance. In the case of the Jitter matrix, this deals with the determined brightness of a pixel. A value of “255” for each R (red), G (green), and B (blue) value is pure white, “0” is black. The sum of all the values, and a threshold that determines if the value is enough, will attribute a “1” or a “0” to the matrix information of the pixel. Activated/Not-activated respectively.
[39]Adrien Kaiser, "What Is Computer Vision?" https://hayo.io/computer-vision/.

to the rest of the screen. The scale of an object is determined by how pixels are within a close enough distance of one another to be considered a grouping. This grouping is also proportional to the number of total pixels within the resolution of the video input. Position could be determined via two methods: picking upon the corners of a bounding box for a cluster of pixels or using the centroid position of the dot. The value of the position is determined via the same method as the assigned number. For the centroid, it uses the information from the bounding box corners, finds the distance between each, and determines a central point. This central point can then be used as a position for a stereo speaker output. If the recognized object within an animation is playing on the left/right, the direction in which the sound is received will change. Under headphones, the effect is even more noticeable. This effect provided an even deeper understanding to the visuals being played with and eventually created—much like the relation to the effect of interacting with the audio game.

Orientation of the object needs to have an object with an end that is larger than the other to consider it as 'pointing' in a certain direction; the attribute is then assigned a number via a radian, or a degree.[40] The elongation value of a shape is found by how "stretched" a grouping of pixels appears. If the shape of the objects is compact, a value of "0" is given. A shape that becomes more line-like will raise dramatically in number. I have had values over 200 in my experiments; this number was a little dramatic to use within the context of the patch, though. It was an easy fix using an object called "scale", which

[40] Radians are determined by [x* π]. For this use, it is a 0 to 6.28 comes from 2*π, or 360 degrees. In this case 2π = 6.28. If desired, there is an option to switch these values to degrees—a single π is 180 degrees, 2π equals 360 degrees.

proportionally adjusts the range of numbers to a different set. Like when using fractions, “100 out of 200” would be ½. But in this case, we can set a specific range, even non-zero. One could use the scale object to shift all numbers between [0, 10] to [-23.45, 134.88]. In this example, number 7 would be equivalent to 87.38; an example of this is shown below in Figure 26.

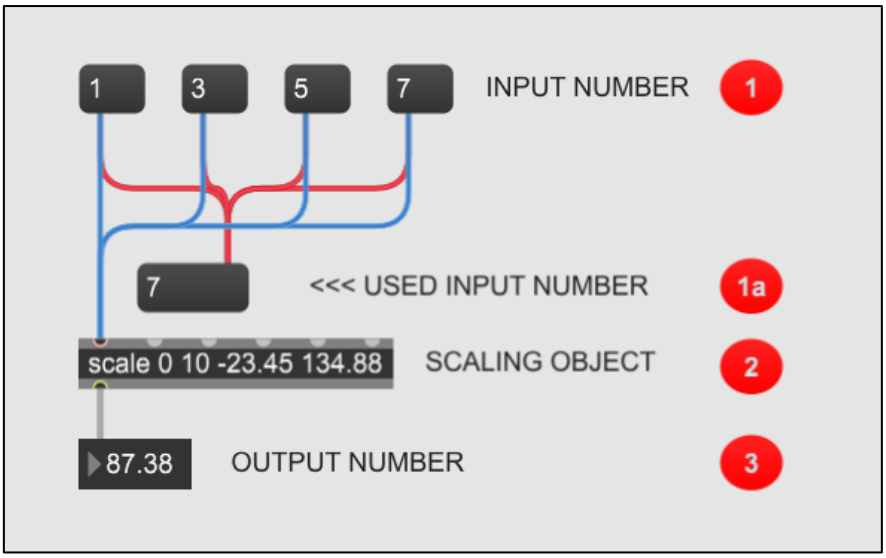


Figure 26: From the patch: (1) Input Number (1a) Display number input (2) Scaling the input (3) Output Number. Patched within the Max environment.

After reading all the separate values of information (size, position, elongation, orientation, assigned number), there becomes the issue of attempting to unpack and evaluate the data per each assigned object. In the case of the prototype, I wanted this to be simultaneous and as close to real-time as possible. The data that is read from the cv.jit objects output a list of numbers all at once. These numbers were in the order of which the objects were being read (whether it was determined from scale or position), preparing the numbers sequentially was not the issue—it was to split all the data out simultaneously, and have each number read, packed, and distributed to its own sound synthesis.

The Visual-Audioizer Prototype: Light Boxes

Before attempting to work with multiple sonified forms, I created a version of the patch that only recognized one object at a time. The object in its current state had output values of position and scale. The values were also stereo-panning depending on the x-position within the frame. As an experiment for real-life applications, I placed myself in a dark room. I then used the camera feed from a webcam as the input to the patch and used a flashlight function on my phone. The intensity, or size (proximity to the camera as well), of the light was making the system react. If the light was shut off, the sound disappeared.

I used this test during an ACCAD Open House in 2019. Within the sound-room, I turned off the lights and adjust the speakers within for the advantage of panning. I also used reflective tape on the ground for ambient light to illuminate the area to stay within. Within the space, there were small white illuminated boxes. When you picked one up from the wall, it would be recognized as an illuminated object and create a sound. The movement of the user's manipulation of the box altered the pitch of the sound. Some would interact with the light box by tossing it up and down, running left to right, or tossing it between one another. Others would move the light with one hand and cover the illumination with the other, making blinking noises along the way. As everyone noticed, though, there was only the function to recognize one at a time—I would demonstrate to them that the patch was recognizing all the shapes, but my current limited knowledge in splitting all the sounds equally was where the process had halted. A layout of the space is seen in Figure 27. [41]

[41] Light Boxes example from Taylor Olsen: https://vimeo.com/326931873

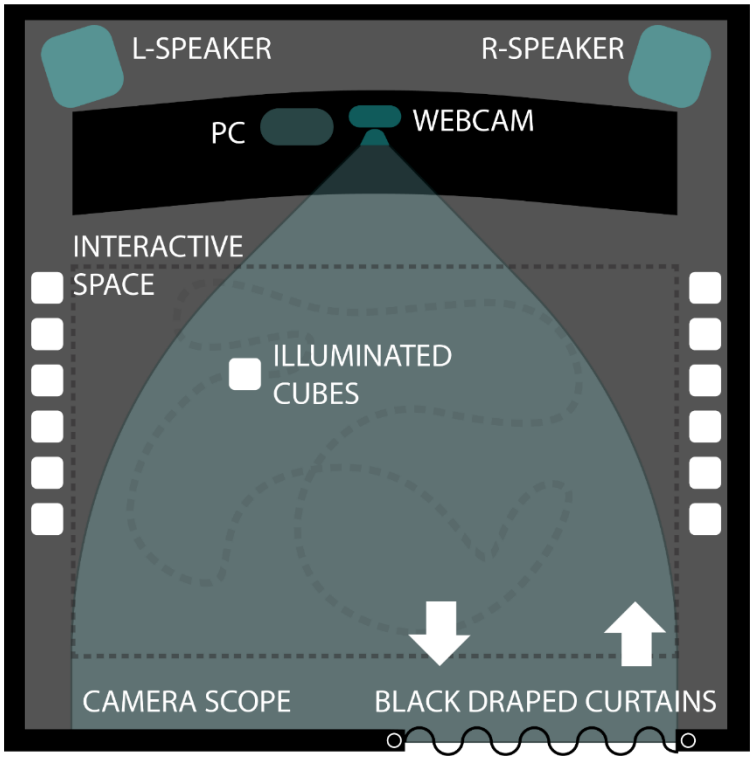


Figure 27: Open House Audible Illumination Room

During the Open House, I realized there was nothing beyond interacting with the cubes that the audience could utilize. I wanted to grant them the ability to interact with the patch itself, allowing more play rather than granting someone the ability to easily replicate the sound of someone else. Another issue to note is the layout of the room was not in favor of the lighting situation; the patch was reading the brightest object within the room, and each time the curtain was opened the incoming light would disrupt the sound. Though I am glad it worked, it occurred to me the loudness of the objects was also a factor that did not seem obvious when just the cubes were in use. In fact, the volume of the patch was turned up over the level threshold—the size of the cubes was not large enough to emit a discernable sound. I could have scaled the numbers to make them between 30%-85% but it did not occur to me at the time.

The Visual-Audioizer Prototype: Multiple Form Solution

To begin, there are complications with using this data at the same time—it makes having a simultaneous input/output of video/sound difficult to achieve. The complication of finding a method to output the sound simultaneously synchronized with the visuals was also relevant—do we create a sound that outputs as fast as the system can handle, or only when a new frame of animation is read? How do you take each piece of information from

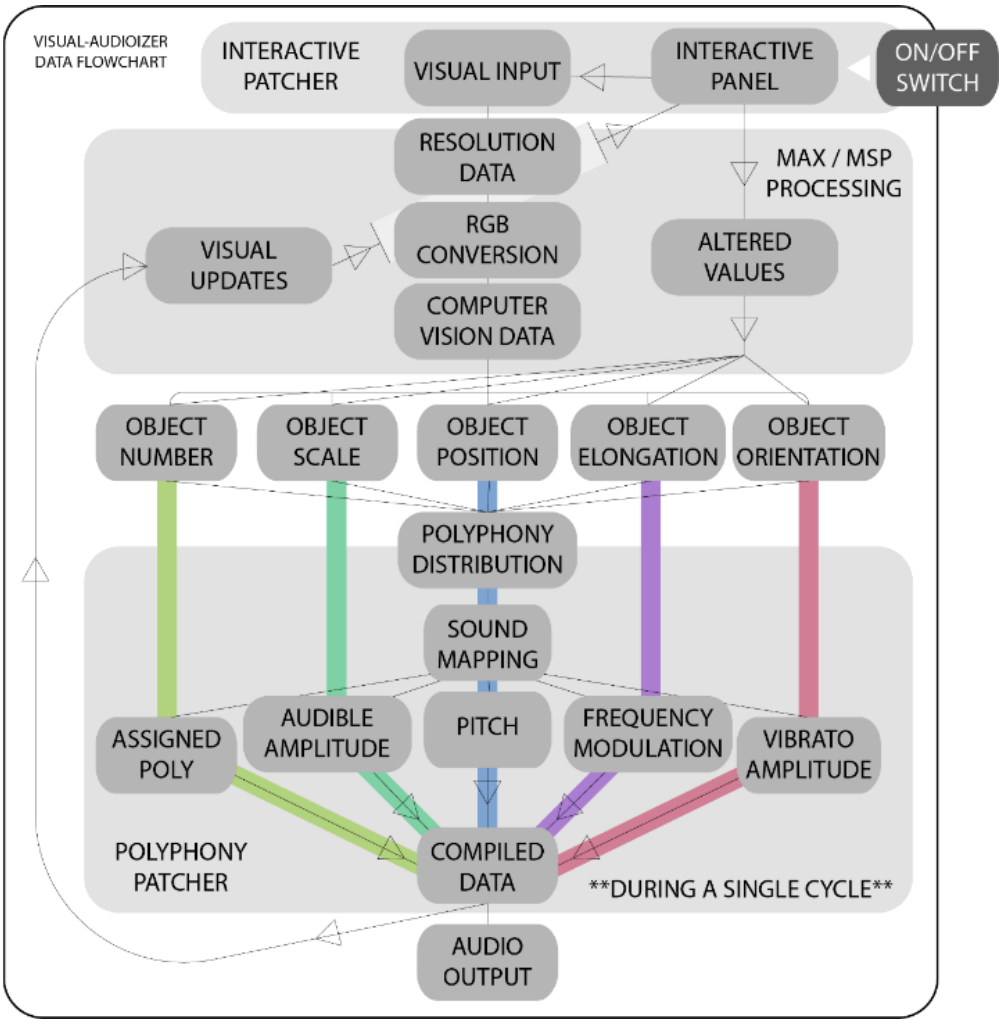


Figure 28: Visual Data Flow Example

each object and redistribute it accordingly to multiple sources? To answer, we need to look at the information that is read from the computer vision data and understand the flow of information starting from the beginning to end (see Figure 28).

Initially, the system is turned on and begins sending out messages waiting for a source to pick them up. The source after these messages is either used to read a pre-animated video, or from the streamed desktop information. After reading one frame from either source, applying the resolution data, and converting the pixel information, the computer vision output data and number example is as follows:

- Number of objects, and the number assigned to each object.
    - This number can range from 0 – 255 (objects).
    - The number of objects affects the framerate of the patch. Higher number of objects equals lower framerate, and vice versa.
- The scale of each object.
    - This number, once adjust later, ranges from 0 – 90. This is then sent into the amplitude. I clip the number at 90 to prevent overly loud output.
- The position of each object.
    - This number depends upon the x/y resolution data.
- The elongation of each object.
    - How line-like an object appears. More compact objects, like squares or circles, will have a value of 0. Resolution data can also affect line-like-ness data.

- The orientation of each object.
    - Objects with larger ends will be considered point in that direction (towards thicker end). This data is either in radians or degrees.

So, the patch has 5 attributes that are needed create a sound. Indefinitely there is a need for scale to create any audio in the first place. The 5 attributes are object number, scale, position, elongation, and orientation. While it would be nice to have each neatly packaged and sent on its own way to be sonified, this is not the case. Each attribute is contained together. That means if there were 10 objects the values for position, and others, are all contained in one message. This may seem unnecessary, seeing as we want each value on its own—but this has its advantages, one being it is more efficient to have all the data contained in one message rather than separate ones. In this case, we have a total of 5 messages for 10 objects, eventually fanned out to 50. In the case of separate messages (5 messages x 10 separate objects), we would have 50 messages for 10 objects, or an eventual number of 500 separate messages at once. Distributing this data beforehand would also be a pain; early in my studies I subjected myself to this method. When working more with the prototype and how to analyze these long lists of information, I found the "zl" object group. Since all the computer-vision information is compiled into lists, it is necessary to pick out each element. This can be done with the object zl.nth; this takes the "nth" number of a list, say third or fourteenth, and extracts it from the list. An example of this setup can be seen in Figure 31. Perfect—now how do I get this to send to multiple outputs simultaneously?

Initially, I experimented with just duplicating the lists and placing the same connection cords to each of them. This was a time intensive process, that yielded little

control if I wanted to change every single object at once. It was also limiting to how many times I had duplicated each, in this case, digital signal generators. You can see an example in Figure 29; the synths are the objects named "ez_24". In the figure there is a total of 50 different instances. This method was a step in the right direction, but proved itself to be time-intensive and was not optimized in any way. There's also the issue of controlling each synth without having to make a connection to every single one. When the system was turned on, each object also output sound as loud as it could, there was no determinate to

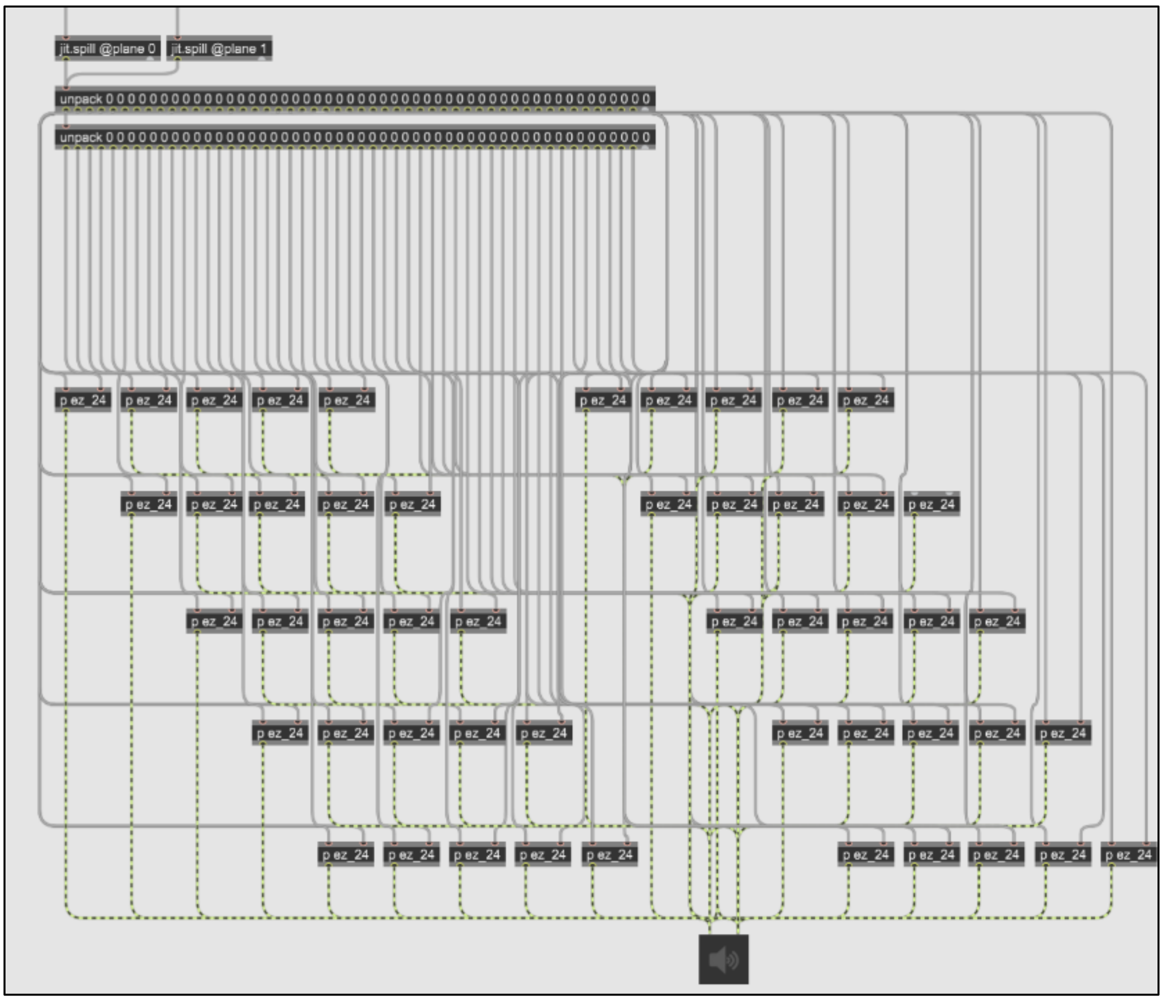


Figure 29: Attempt to create a sound for each separate recognized form

how loud a sonified form should sound.

The solution was the use of an object within Max called “poly~”. This object, poly~, can utilize encapsulated patches and create multiple instances of a patch at one time.[42] Poly is often used in the signal processing side of Max but can be utilized to perform specific functions repeatedly. The use of this object was the solution to the multiple forms problem. For the sound side, I used the poly~ object to create multiple synths at once for each object. These poly-synths, in order to work correctly, need to know how many of the encapsulated patches can or will be used. A second aspect is targeting the specific voice number that the sound will be creating. For example, the object number derived from the

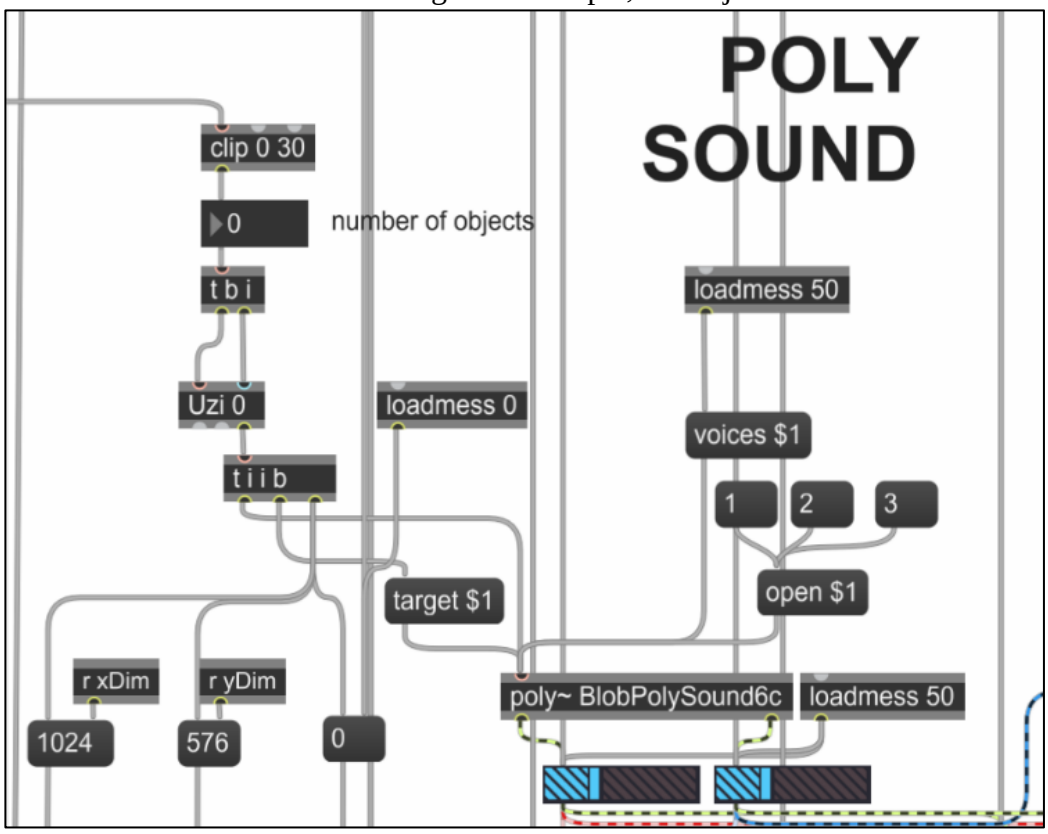


Figure 30: Example of Max objects: uzi, trigger, poly~.

[42] Polyphony: multiple parts playing together at once. Often used as a musical term—can refer to the combination of independent melodies playing simultaneously.

computer vision data will play as the voice number in the poly patch; object number 5 is voice number 5.

After placing all the zl grouping messages (derived from the computer vision data) within the poly patch, I then used two objects, "uzi" and "trigger", to send out the messages simultaneously—this can be seen in Figure 30.[43, 44] It should be noted that the number of objects, derived from the CV data, is how the poly system is initially triggered. This entire process happens during just one cycle of audible sound output—and this cycle only

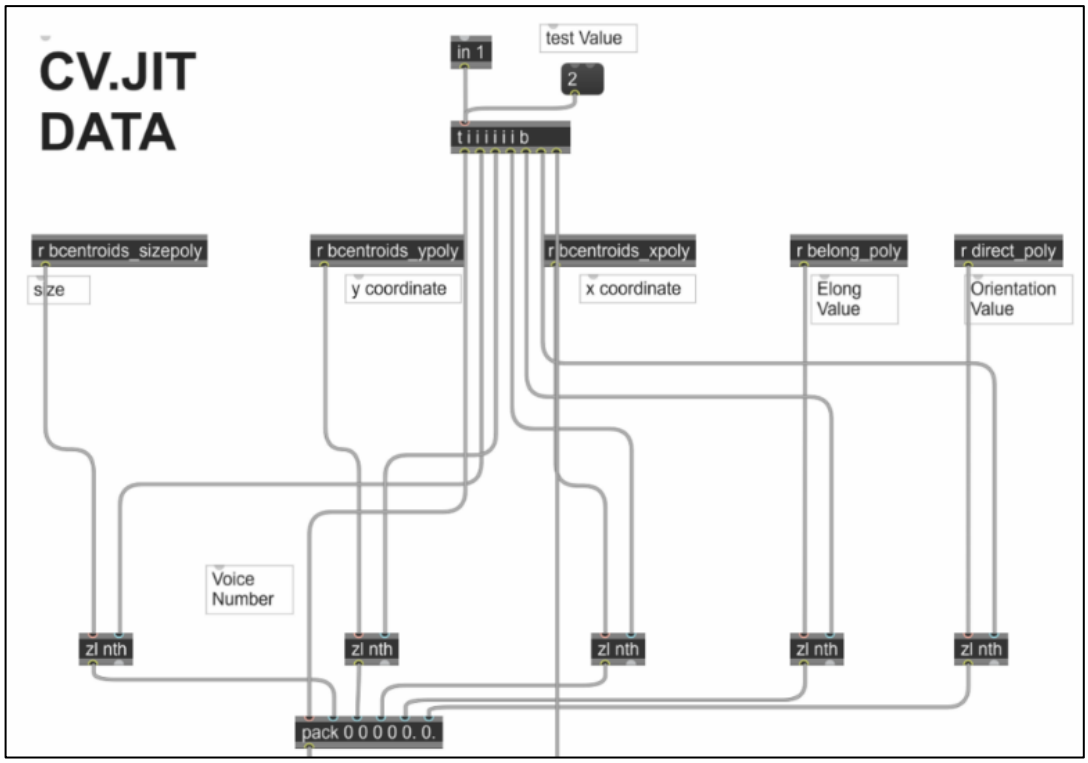


Figure 31: Zl.nth Max object example

[43]Uzi: Max object that sends out multiple messages all at once. This can be done in a sequence—let us say you send the number '50' to an Uzi object, it will then send out "1, 2, 3, 4, etc.", until it hits 50, into separate messages.

[44]Trigger or (t): this Max object can send out specific commands in a certain order. In the case of the *visual-audioizer*, and in Figure 30 as the object "t i i b", it first sends out a bang message. This is then followed by two integer messages, the first sent into the "Target $1" message sent to prime the specific poly voice number, the second being the number sent into the poly patch to parse out the lists of computer vision data within.

changes depending on the changes in the animated form. Having all the information distributed to each poly patch (from Figure 9: BlobPolySound6c) I had to consider that each piece of information might not come out in the order as desired. To solve this issue, Max has an object called “pack” and “unpack”. This object takes in and stores messages, hopefully from a source of triggered objects. In this case, and within my own poly patch, I had the messages from the computer vision data sequenced and sent into the pack object (see Figure 32). This message then contained the 5 determined pieces of information needed for the audio. Next, after knowing the message was unpacked and sent to each audio data inlet, it was onto exploring how to map the audio.

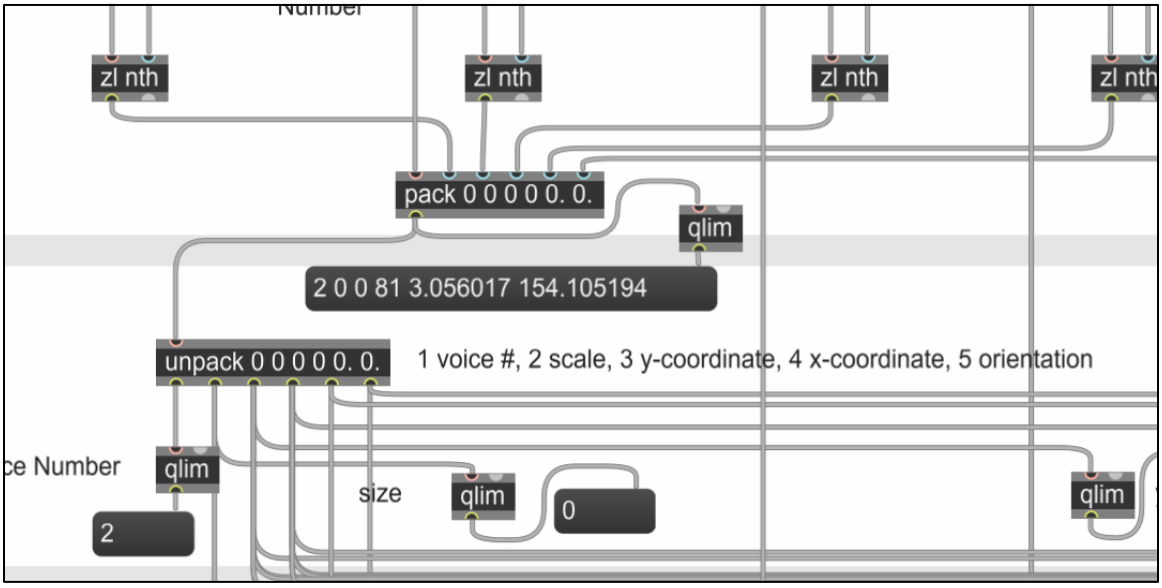


Figure 32: Pack and Unpack example

The Visual-Audioizer Prototype: Mapping Audio

It was a particularly interesting dilemma the amount of times mapping caused a headache over how it should be used. Initially it began as a comparison to existing methods of reading music. Within musical notation the pitch of the note is from top to bottom, in an equivocal sense this could be assumed as a y-axis. This, as well as the x-axis, was the first option that I went with seeing as it was easily obtainable. This meant using position information of the object to determine the pitch. The configuration of the x/y audio can be seen in Figure 33. The y-axis option takes the y position, and the current resolution data, and scales the number to a set frequency. I used the range of numbers between 80, somewhere in C2-C3 range, and 2093, a C7, for the range of frequencies that would be the default output. This would the default for the rest of the mappings—there was no reason for why I chose these numbers beyond the desire to hear a range of pitches during testing.

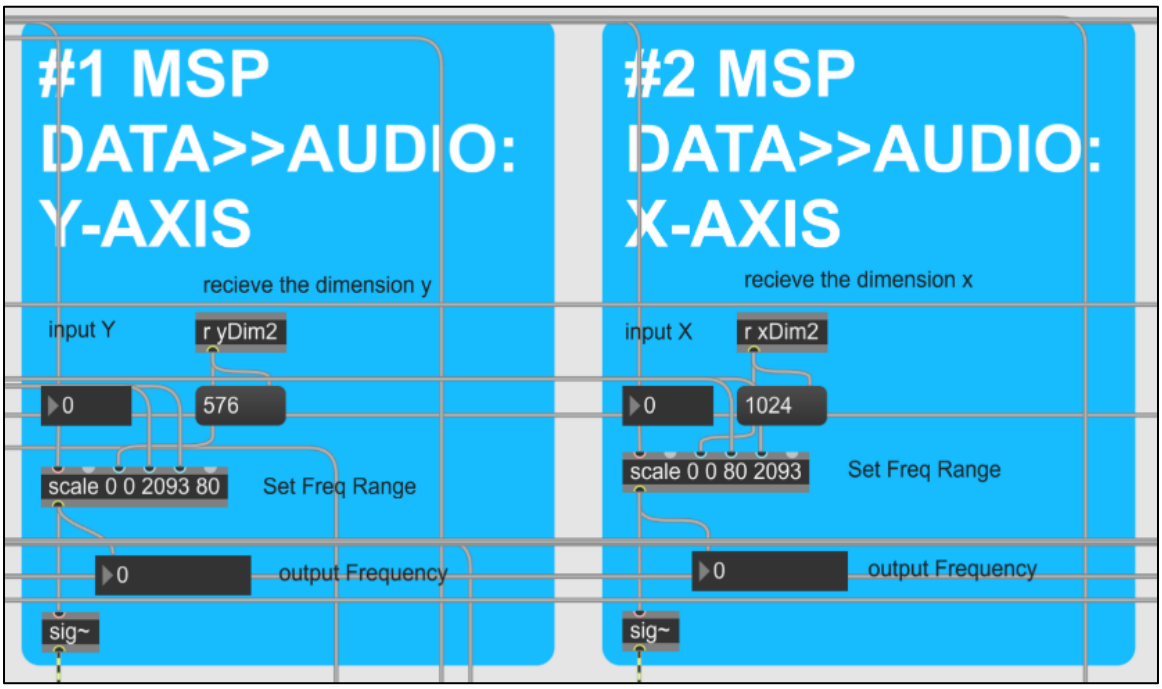


Figure 33: X and Y-axis audio mapping setup

This scaled number was then sent to a “signal” object—this converts the number into signal data, eventually being translated into audio when passed into the digital audio converter

After completing these two mappings, I decided to move beyond just the linear orientations of the x/y axis and ended up combining the two into new ways of mapping the audio. The reason for not sticking only with these two mappings, though one could become an expert in a single mapping type, is the ambiguous and ever-changing nature of animated forms. While one might find themselves sticking with a simple x or y layout, the ability to create a similar sound while completely and simultaneously changing the visuals allows an animator to explore a variety of motion within their canvas. This consideration, and the ability to switch between mappings, allows the animator to hear how their drawn motion changes depending on their mapping choice.

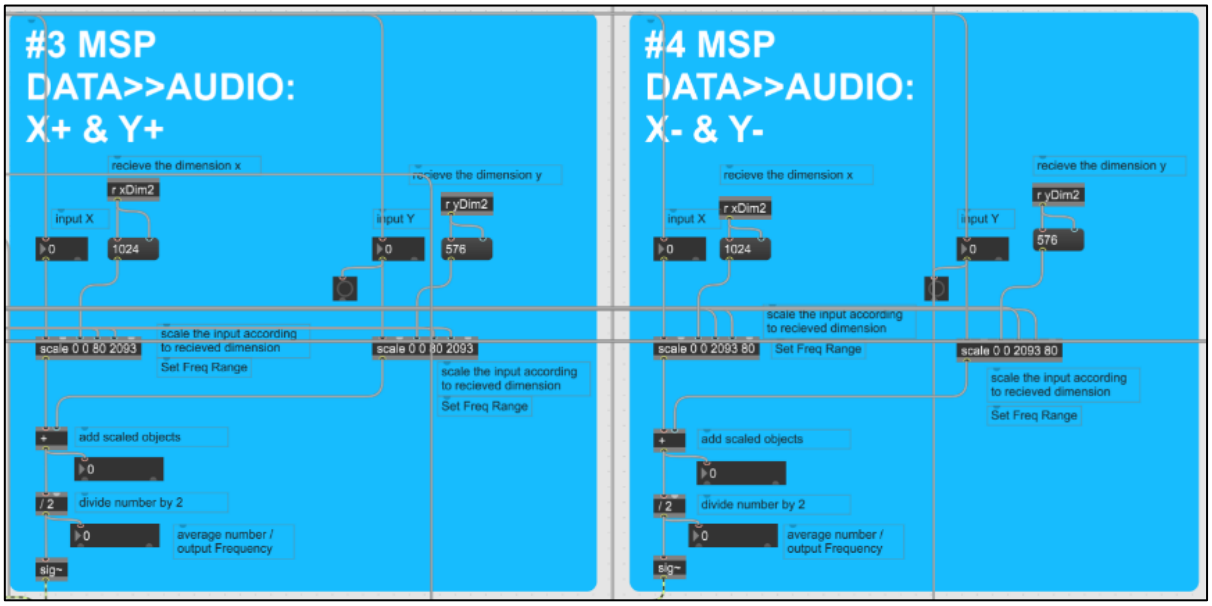


Figure 34: X and Y combination axis setup

Moving forward, the next two mapping choices included adding together the x and y position values, then dividing by 2. The lowest pitch values are in the top-left corner, and

the highest is in the bottom-right. This makes a diagonal representation of pitches, meaning if one were to move a form from the bottom-left to top-right, the pitch would remain the same. Moving from top-left to bottom-right, the pitch progressively gets higher. For the sake of having the opposite direction, I reversed the numerical values of the scaled object. This time the lowest pitch is in the bottom-right, and the highest in the top-left. Moving in a diagonal direction, from bottom-left to top-right, will still produce the same results as the initial setup.

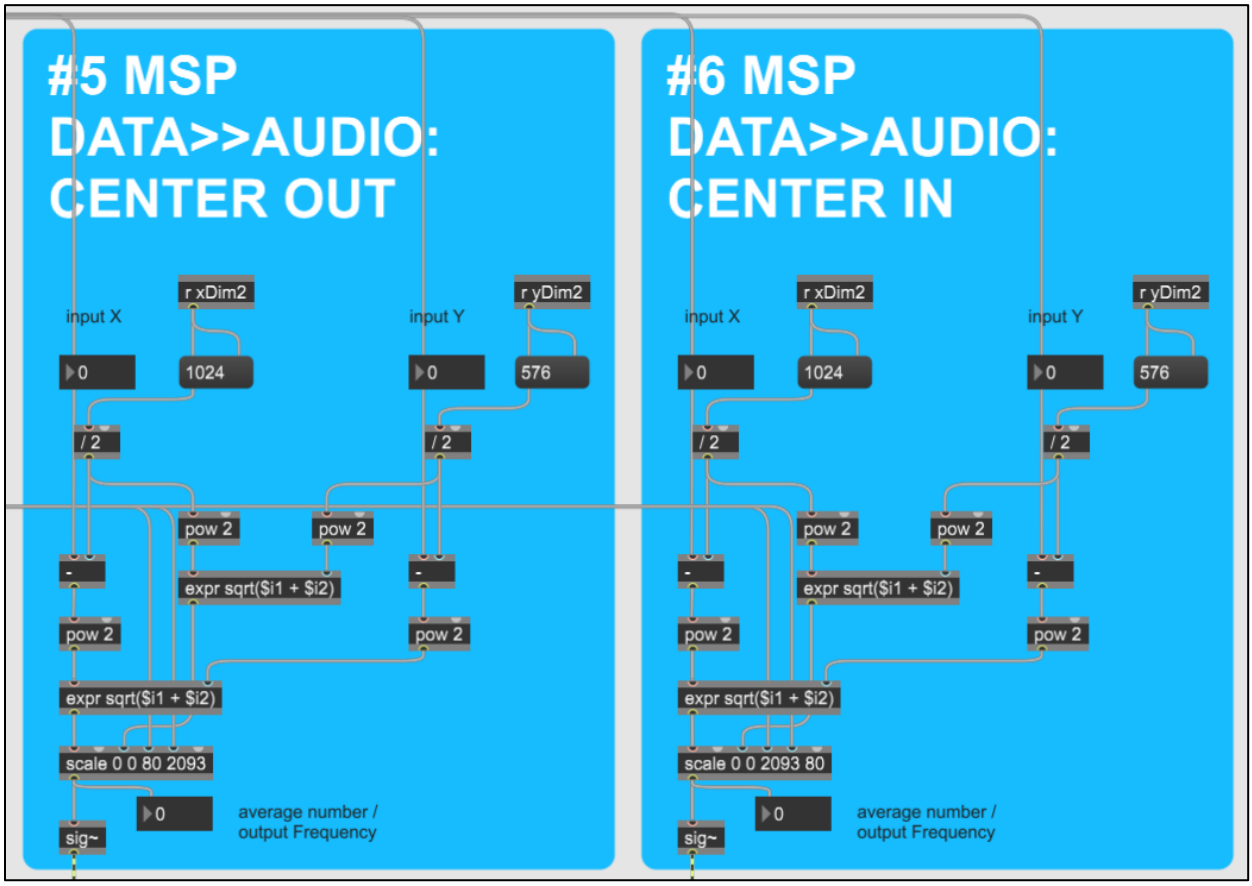


Figure 35: Central Audio Mapping

The next consideration for mapping I wanted was the ability to utilize the center of the canvas as a starting point. This meant I wanted the lowest/highest pitch of the form to

originate from the center of the canvas. I used a similar function to that of the audio game, except the initial point was always the center of the canvas. To accomplish this, I initially took the resolution data of both x and y and divided each individual value by 2. This value was the determined center of the canvas. In case the resolution of the canvas ever changed, this value would be dynamically adjusted and always find the center. After obtaining this value, the mapping then uses the Pythagorean theorem to determine how far the position of the animated form was from the center of the canvas. After a distance value is determined, it is scaled to a value and changed into a signal. With this mapping, the farther away the form is from the center, the higher the pitch; if one is looking for the highest pitch, go towards the corners of the canvas. The lowest pitch would be located within the center of the screen. An example of this mapping is shown in Figure 35. Once functional, I duplicated the mapping layout and reversed the pitch values. Center is highest, outwards to corners are lowest.

The last mapping experiment I wanted to use was the individual x and y-axis as a split canvas. If in the x-split, any derived position would take the x-value and determine the distance away from the center of the canvas in the x-orientation. The same was implemented for the y-axis. If one were to use the x-split, the canvas was vertically separated into two, or a left and a right half. The x-value would then be subtracted from the determined central axis and passed through an “absolute” object—this takes any number and changes the number into an absolute value (always positive). Any position value near the center would the be lowest value, while any towards the left/right edge would be higher. Forms moving up and down would produce the same pitch. Using the y-split,

the canvas is horizontally split, a top and bottom half. Again, forms near the central split of this orientation would produce the lowest sound, while towards the top/bottom create higher pitches. Forms moving left/right would also produce the same pitch. An example of this layout is seen in Figure 36.

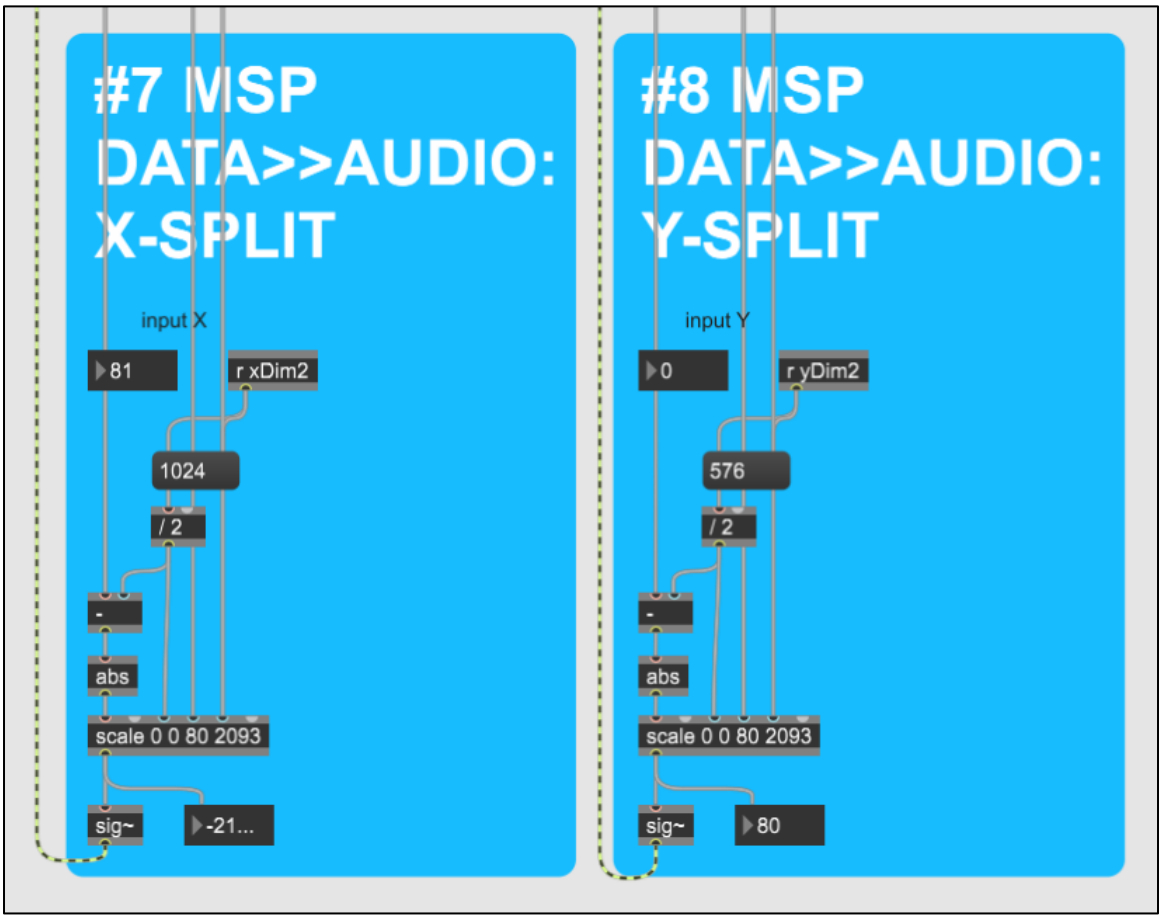


Figure 36: X-split and Y-split mapping example

You may notice at the end of each mapping layout is a “signal” object. This takes any number and transforms it into a signal value, allowing it to be seamlessly passed into a digital audio converter without disrupting the flow of data. Using these signal values, I passed each one into an object that allows the simultaneous switch and choice of which to use. This is discussed in the proceeding section.

The Visual-Audioizer Prototype: Creating and Panning Audio

With all the different mapping options in place, the next step was to create the audio. As an experiment, I had each mapping option output audio at the same time. The result was chaotic and unorganized, and made little to no sense when attempting to listen and associate the animated form with each specific sound. The solve this problem there is an object called "selector~" that allows one to choose from a variety of inputs, in this case the mapped values to a signal (eventually a pitch), and decide which one is allowed to pass through. An example of this object is seen at the top of Figure 37, with the "selector~ 8".

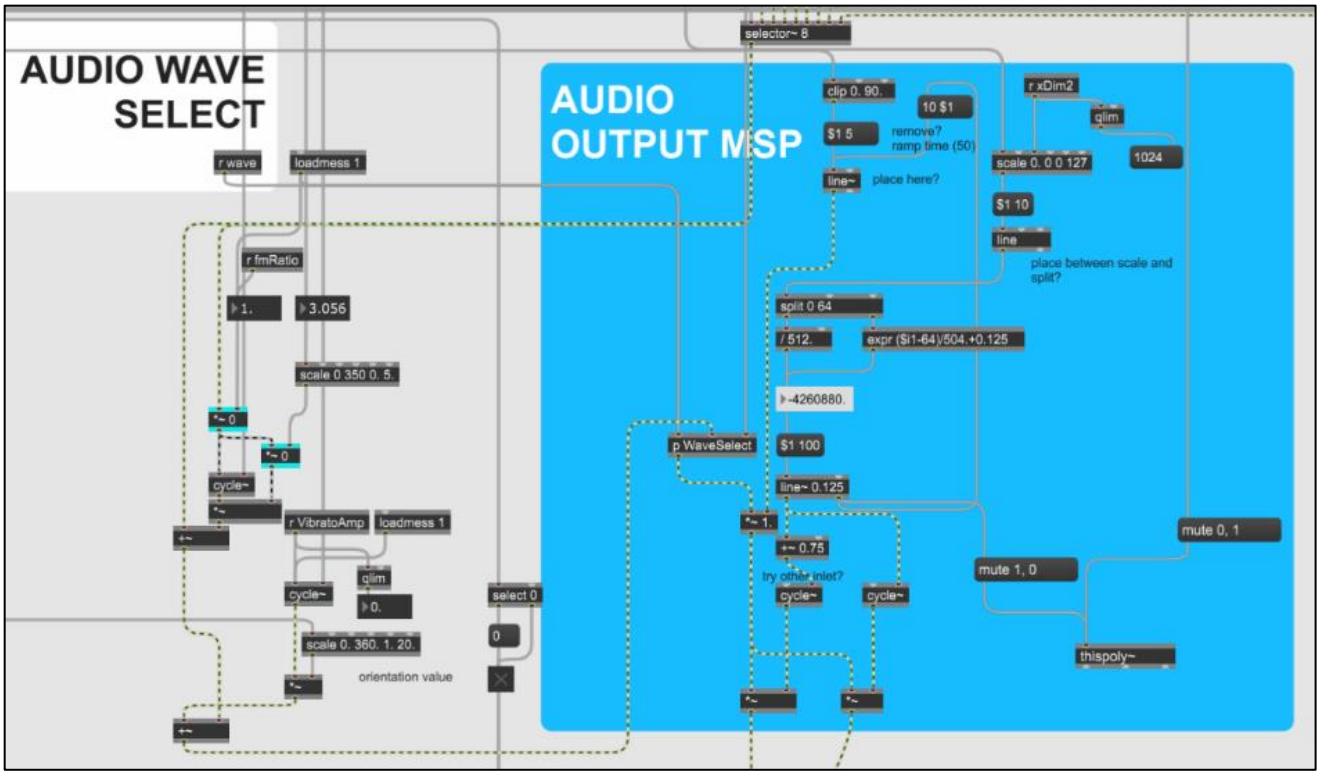


Figure 37: Audio creation and panning

The number after the selector represents the number of inputs being passed through. Loading in each input into the selector~ then allows an individual to tell the object which

option to use. This mean if I gave the selector~ a value of “2”, it would only output the second input into the object. I would advise anyone to only use integer numbers (no decimals) as the selector~ can become buggy if one sends a float value into the object. After a mapping signal is chosen, it is then up to the user to decide what kind of waveform to use. This option was created within a sub-patch called “WaveSelect”—as seen Figure 38. Within this encapsulated patch there is 4 different waveforms to choose from; they include: cycle, saw, tri, rect. Each has their own unique sound if chosen. It should be noted that I used the selector~ object once again within this sub-patch to allow the user to choose.

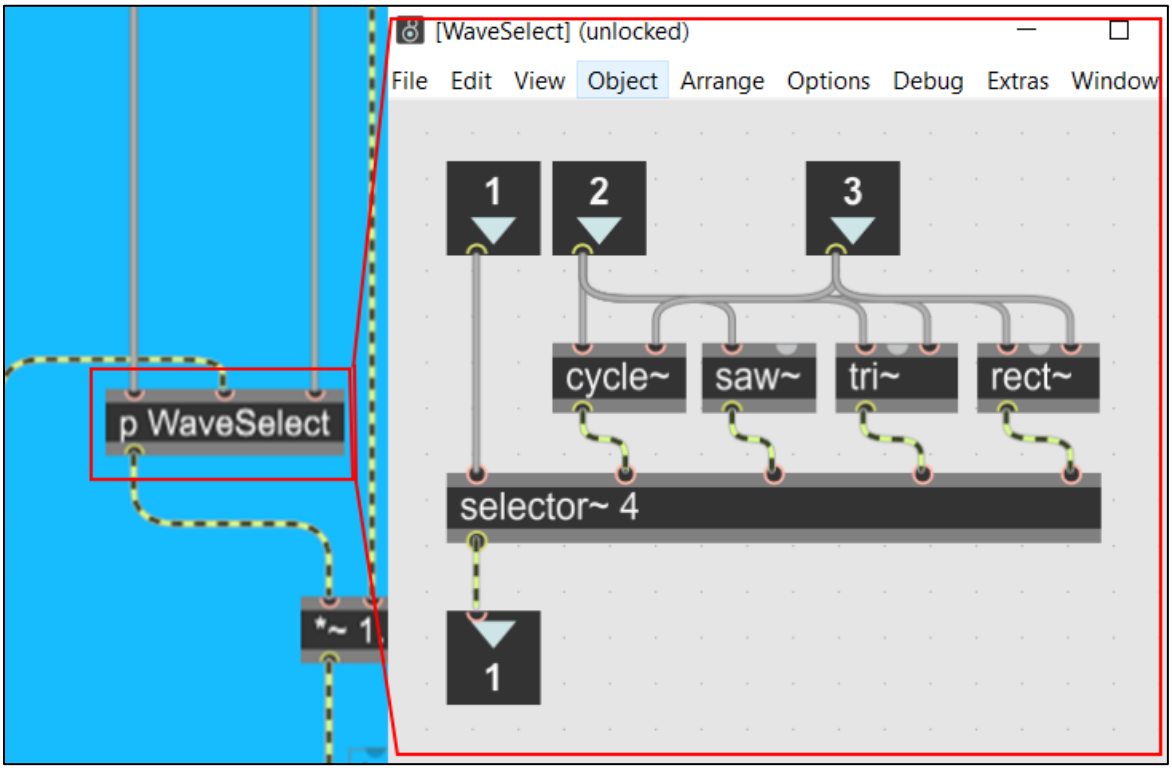


Figure 38: Inlet 1: Wave Select, Inlet 2: Signal, Inlet 3: Phase Reset.

To give the illusion that the animated forms were panning, I used a technique found within an example path from the Max tutorials. This technique takes the x-position and scale of the form and tricks each speaker (left and right) to amplify the sound according to

how far left or right the form is within the resolution of the canvas. The source of a sound from an animated form sounds louder in the right speaker if it is near the right side of the resolution space, and vice versa. An example of this is found in Figure 39.

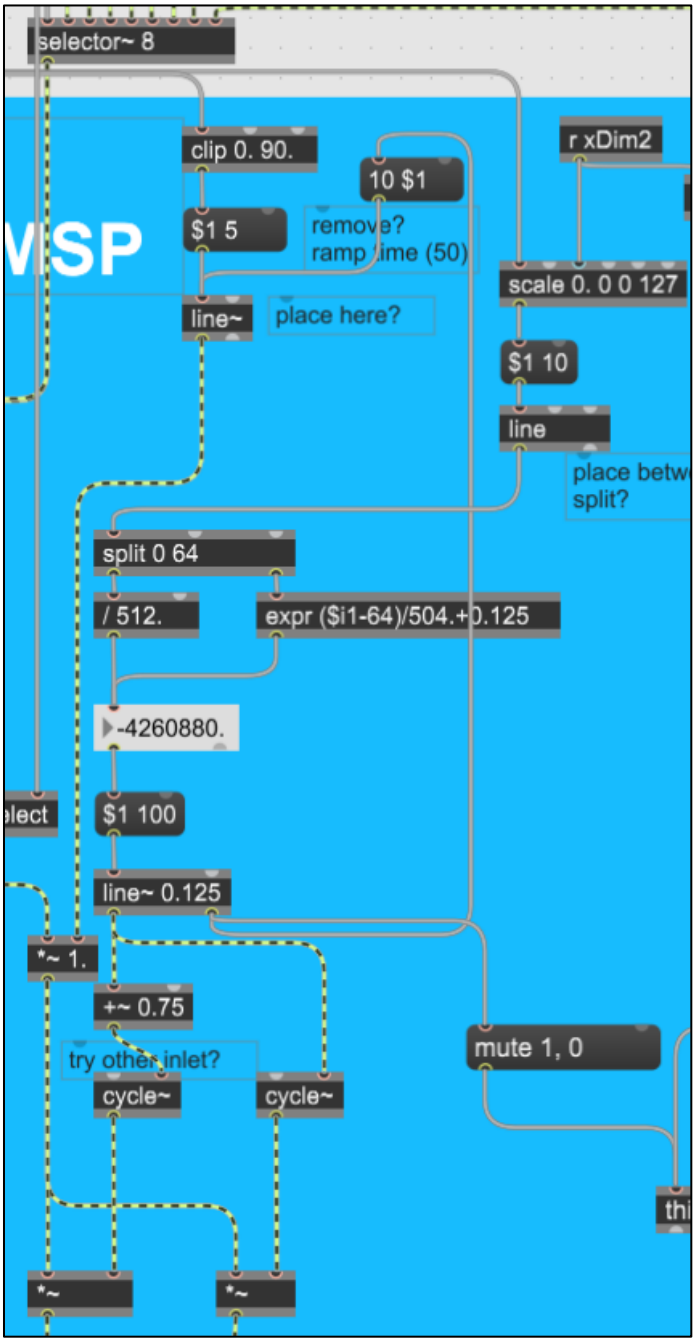


Figure 39: Panning layout

I found it necessary to have panning as an option, as moving a form from the left to the right side of the screen allows a better connection to where the source of the sound is coming from. This panning can also influence how one would animate when using the *visual-audioizer.*

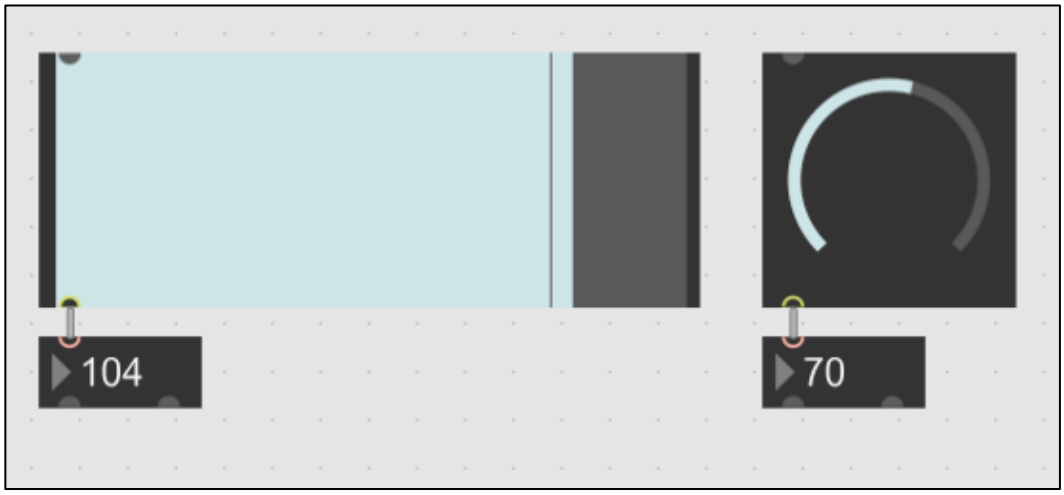


Figure 40: Max interactive objects example

The Visual-Audioizer Prototype: Interface Design

Part of the complications when developing the *visual-audioizer* included the ability to control most, if not all, aspects of the data flow. This meant having interactivity to alter the sound once the system was running. I knew the act of animating is its own time-intensive process to work with and I did not see the need to improve upon techniques that were already available; I wanted to focus more on the ability of the *visual-audioizer* to create and control these sounds while simultaneously letting the role of animation be unimpeded. By no means do I consider UX design to be my strength, but easily enough, Max has a myriad of objects often referred to as sliders, dials, buttons, and the like to aid in the process of creating quick iterations of interactive designs. These interactive objects

can then be controlled via the click/drag of a computer mouse, or, by means of an external controller. With deeper interactivity, one can also control what range of numbers can be outputted from the object. In Figure 18, we can see a “slider” and a “dial” object. The slider, when interacted with and based on the range of numbers, can send out anything from 0 to 127; the dial can do the same. Notice how only part of the dial and the slider are filled, this is where the interactivity can let the user define what number is outputted.

When I first began creating the interactive portion of the *visual-audioizer*, I had placed these interactive objects within the same space as the entirety of the patch. There were my proposed options within the interface: read a file, start the system, pause the system, and turn on the sound. This version, being in the early stage of development, was made solely to declutter the space around it and observe the values working with the

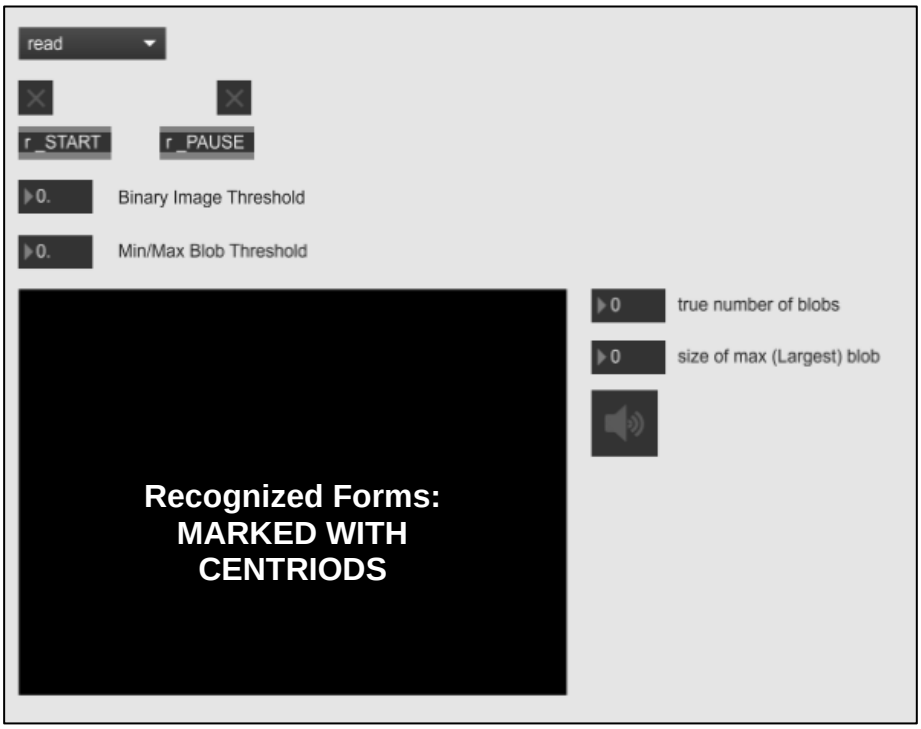


Figure 41: *Visual-Audioizer* Test – Layout #1

computer vision data. The video data was displayed on the large black square in Figure 41, and additional data included the number of identified forms, size of forms, and video tracking-threshold options.

As my technical skills working with Max developed (and continues to), so too did the design of the interface. My second working design for the interface was like the first, but with added sliders and some brief comments for the interactive elements. An example is the interactive threshold, of how many pixels are required in a recognized form, changed from a number to a slider object. Shown in Figure 42, this layout had two screens; one displays the original video (left screen— either pre-rendered video or a webcam stream), while the other is the post-analyzed video stream (right screen—recognized forms with

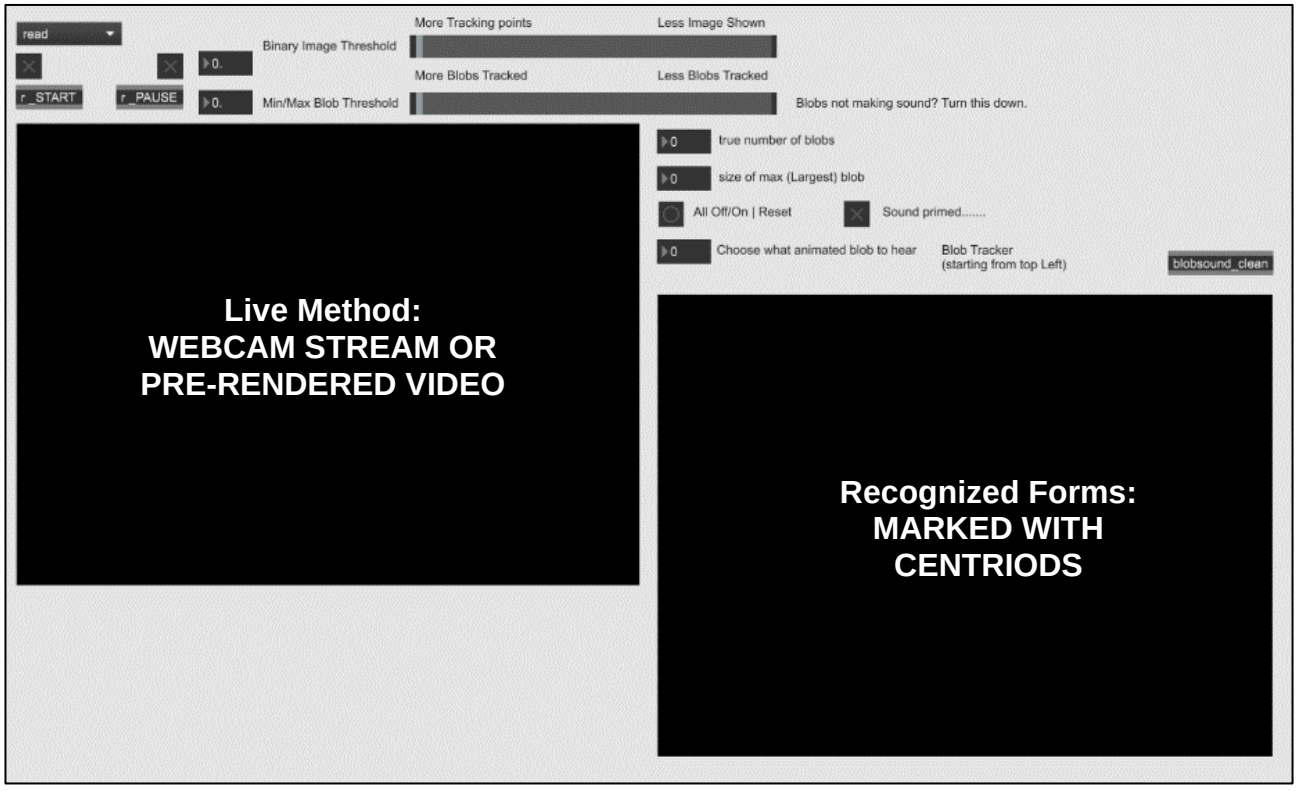


Figure 42: *Visual-Audioizer* Test – Layout #2

centroids).

The next version was the method used during the 2019 ACCAD Open House, as seen in Figure 43. The interface became dark because of the setting the room was in, having a bright screen would trigger multiple sources to become illuminated; this method reduced the brightness of my interaction with the Max patch. When listening to suggestions from the guests at Open House, most suggested the interactive portion of the patch felt ambiguous to what the interface even meant to the random user.

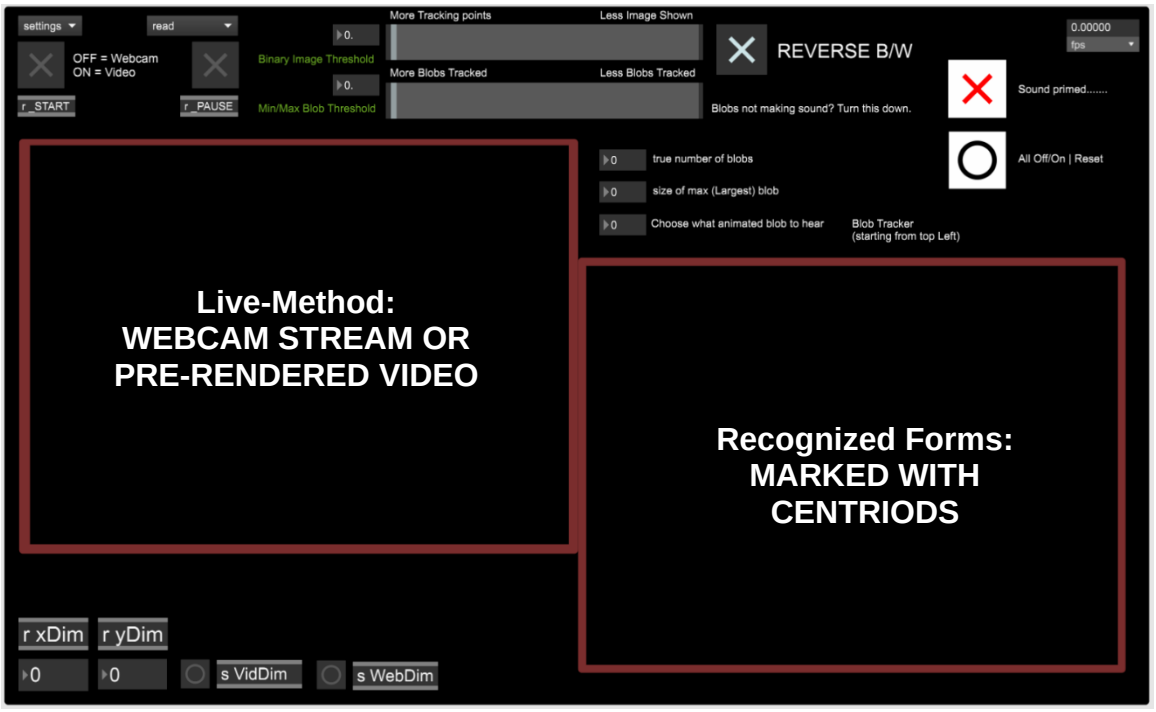


Figure 43: *Visual-Audioizer* Test – Layout #3

After receiving feedback from the Open House, most if not all, folk mentioned the purpose of the layout was ambiguous unless explained. There was also the confusing of using a file vs using a webcam and making it more user-friendly. I decided to change the

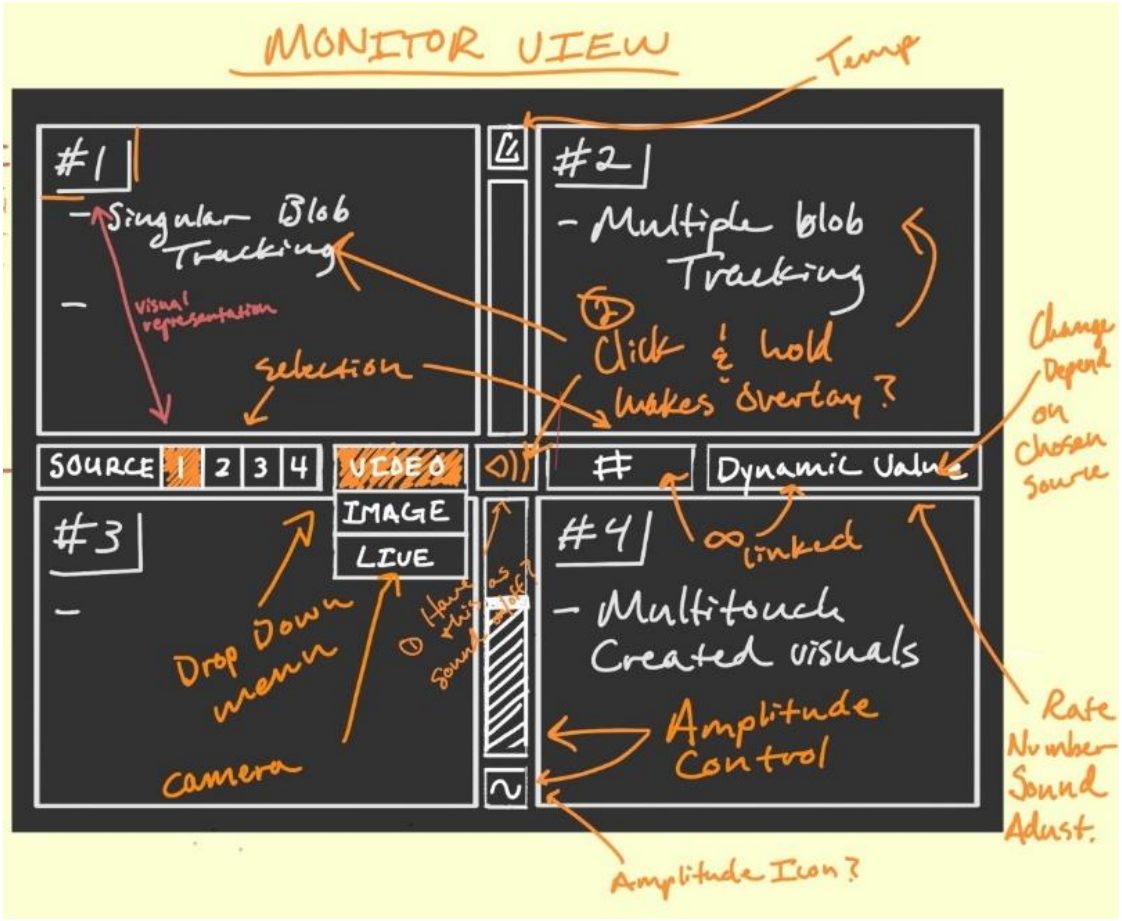


Figure 44: 4-panel Layout Concept

layout of the workflow entirelay and attempt to separate the content to 4 separate uses. This can be seen in Figure 44; the four different uses would be videos, still images, live (webcam), and multi-touch (touch-enabled sounds). This layout was a good change from the previous, making use of all the space in a smaller space. The 4 different sources would be visible on the computer, and the portion in white would be on a separate device, like an IPad. Max has an app, Mira, and a special object within the program that can mirror the layout of a Max patch to a mobile device. Unfortunately, many of the interface options I had used were not displaying correctly on the IPad—this might be due to my using of a

Windows operating system and the IPad, a non-native operating system, having compatibility issues. This was not investigated on my part but should be considered if someone wants to experiment with Mira and Max.[45]

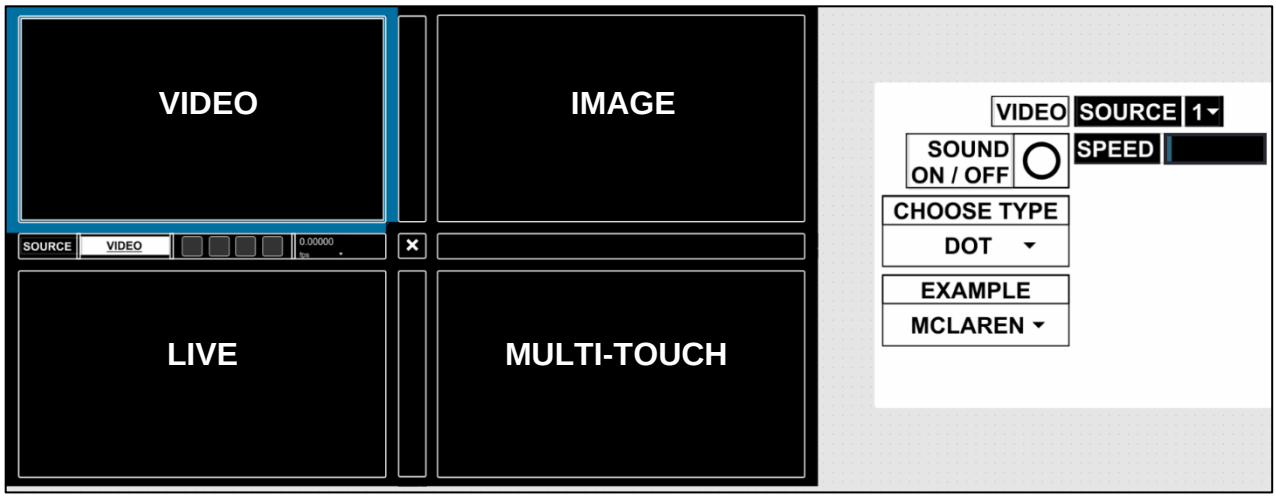


Figure 45*: Visual-Audioizer* Test – Layout #4

I spent some time away from working with the interface, and more with the technical aspects behind the work—this included some basic optimization to the patch, removing unnecessary objects, etc. I investigated into some modern musical interface designs—most stemmed from the company Teenage Engineering.[46] Their portable synthesizers are purposefully designed to be compact. At first glance it is hard to tell what to interact with, but because of the consistency and the use of repetition, every placement of a button feels purposeful. This is further proven as the layout of each interactive piece on the hardware lands within the use of a grid. Color schemes for both like to use solid hues to pop against the white/black backgrounds. The use of intermixing white/black text

[45] Mira (for IPad): An app to connect to your Max interface and mirror to your screen. https://cycling74.com/products/mira

[46] A company of audio engineers and designers who focus on creating portable synths known as "operators". Their website: https://teenage.engineering/products

for respective design models is also a good choice—it is clear, and it stands out. From my own analysis, I decided to experiment with the layout of the *visual-audioizer* while keeping these two pieces of hardware in mind.

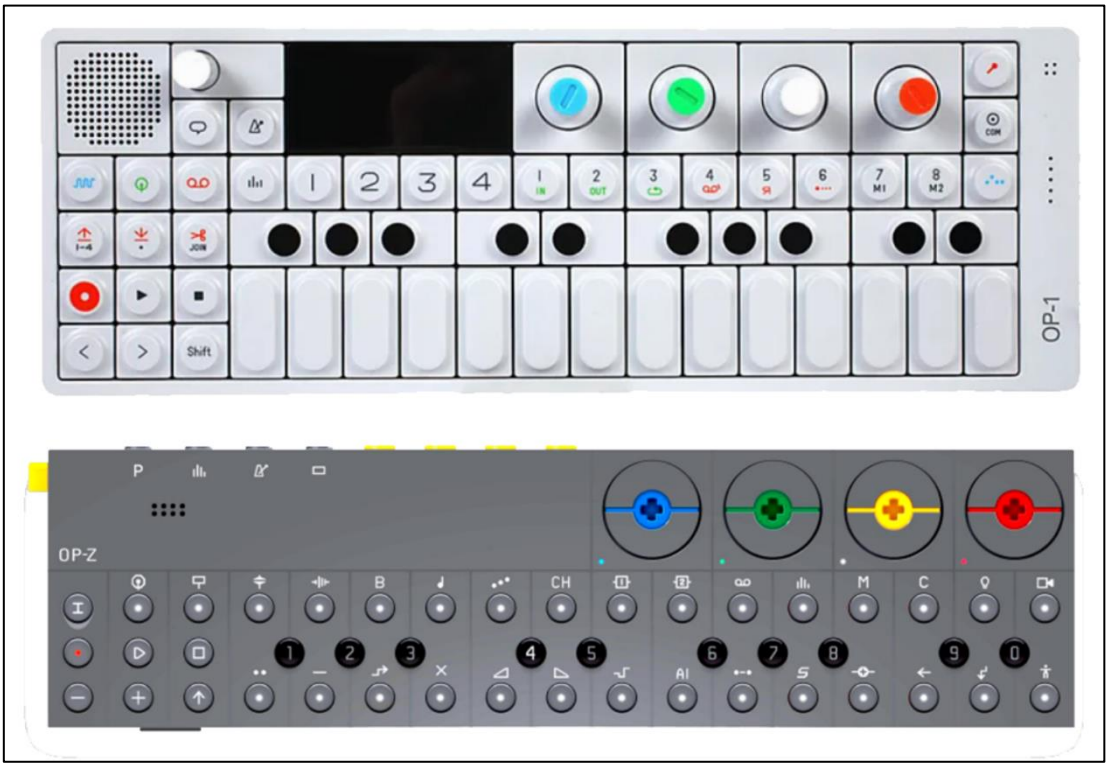


Figure 46: Teenage Engineering Portable Synths:
OP-1 (top), OP-Z (bottom)

In the Spring of 2020, ACCAD announced it would host a playtest day, I decided to act upon this as an opportunity to utilize a general public of people for feedback. The transition was quick when re-evaluating how my patch would look. I chose to use pure white backgrounds with black text. The interactive portions would be kept to the left side in some varying degree, and font sizes would have a hierarchy to their title/explanation. I chose to have a black bar with white text for the portions that included using slides to adjust the audio aspects of the patch as well. I removed the 3 different screens and the options for

the multi-touch, image, and live interactions. Using only the video portion made it easier to create video tests and provide the user with a limited degree of freedom, rather than an overload of buttons with little information. One screen vs. 4 was better optimized as well, making the interaction more responsive.

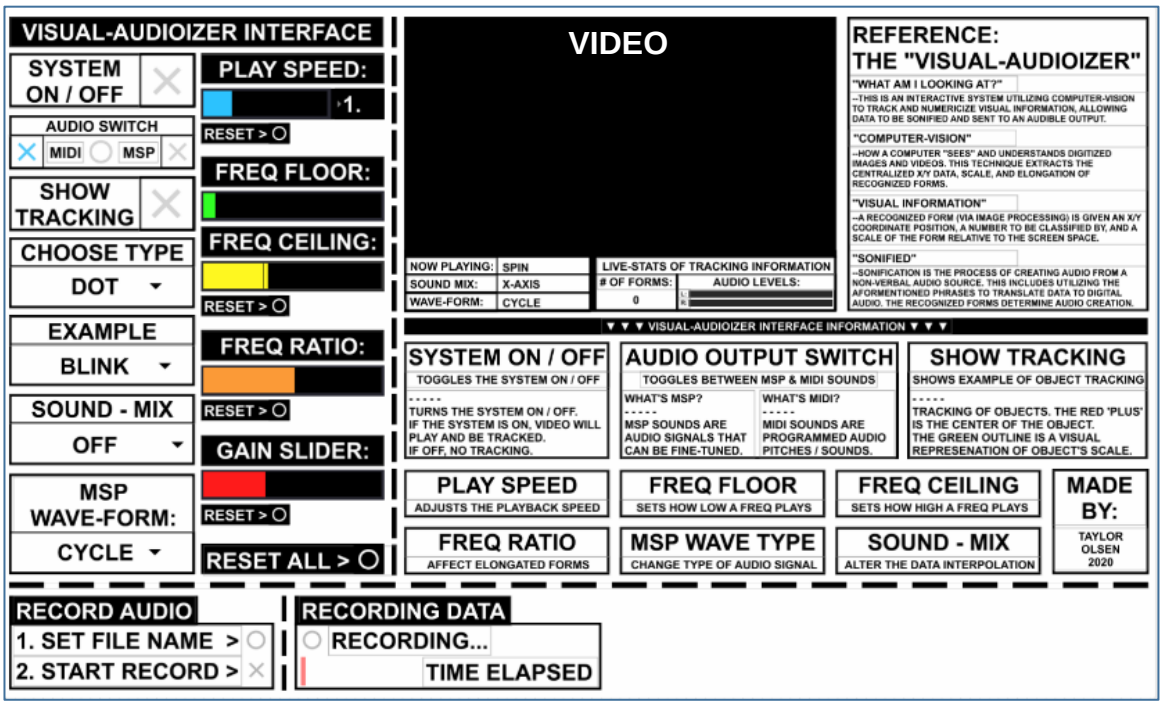


Figure 47: *Visual-Audioizer* Test– Layout #5

In Figure 47 we can see the change in the layout based on previous feedback. This layout had more intention behind its placement of interactive objects and information. The interface options were placed within their own section on the left, while the right held an explanation to what the patch does. Underneath, interface information for using the *visual-audioizer*. On the bottom left I also worked into the ability to record the audio during a session with the *visual-audioizer*. In the center, the video player of the current content, and a toggleable switch to change the video to the tracked objects. Underneath the player, there

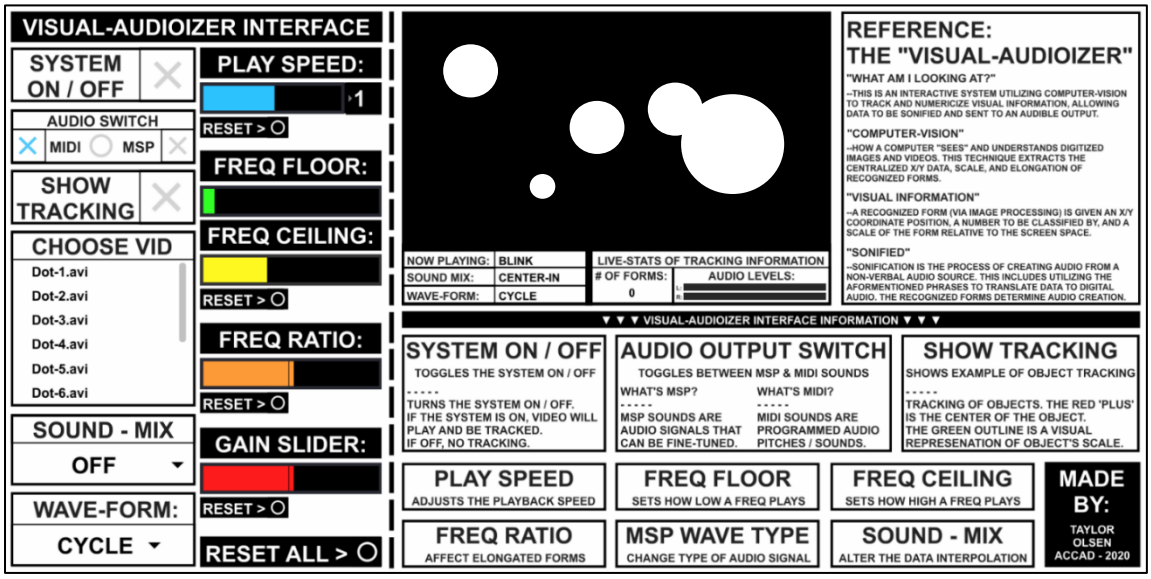


Figure 48: *Visual-Audioizer* Prototype – Layout Option #1

is live stats of the sound mix, wave form, and the current video playing. This info also includes CV information on the number of forms, and the audio levels when outputting. I then removed such elements such as the ability to record during playtest day and reworked the interactivity of selecting a video to play. I considered this layout to be an actual 'prototype' of what I was looking for. There is room for improvements of course, but this layout proved to be useful and straightforward. One of the notable changes is the "Choose Vid" tab; rather than two separate lists ("Choose type" and "Example"), I used a "list" object within Max and connected a folder of any desired video files I needed to play. The feedback from the playtest day went well, and most were happy to find the association between animation and visuals. I was happy to hear feedback about the design of the *visual-audioizer*, as when working with a protype there is always room for improvement. Most did comment on the layout of the entire software and wondered if there was room to organize the content a little more. The hierarchy of information was somewhat lost; I

wanted the user to choose a video, listen, and observe. Most participants focused on the solid hues of different color, rather than the large swath of text in the rest of the prototype.

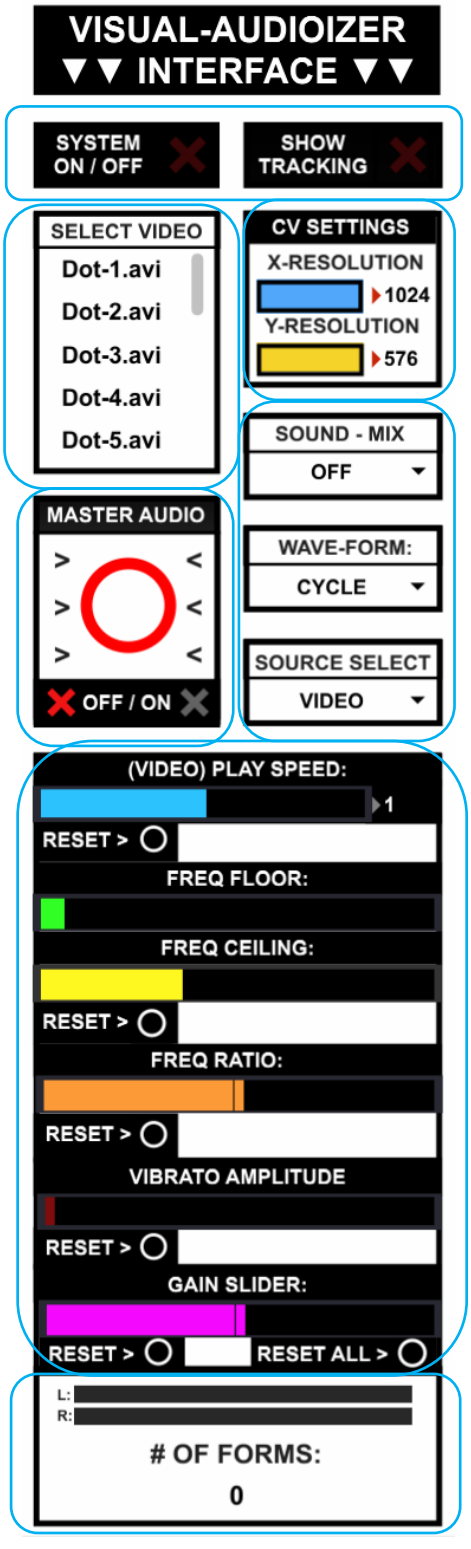
-Turn the System ON and OFF.
-Show the tracking methods on the chosen video/desktop stream

-List of videos to choose from

-Master Audio ON/OFF Button

Interactive Audio Options:
-Video Playback speed
-Frequency Floor
(how low is the pitch range?)
-Frequency Ceiling
(how high is the pitch range?)
-Frequency Ratio
(normal to shrill sounds – influenced by elongation value)

Live updates:
-Loudness in each speaker (L and R)
-Number of recognized animated forms



-Computer Vision Resolution Settings:
(lower = more fps, less clarity in pitch dynamic range)
(Higher = vice versa)

-Sound Mix Options (mapping)
-Wave Form Choice
-Source Selection
(Video or Desktop)

Interactive Audio Options cont:
-Vibrato Amplitude
(how "wavy" a pitch sounds – partially influenced by orientation)
-Gain Slider
(make all sounds louder)
-Reset Buttons
(reset individual values – or – reset all Interactive Audio options)

Figure 49: *Visual-Audioizer* Prototype - Layout Option #2

The Visual-Audioizer Prototype: Fluid-Time Animation Testing

After creating the prototype, as seen in Figure 48, I spent time experimenting with fluid-time animation coupled with the *visual-audioizer*. As mentioned in Chapter 2, the application *Looom* specializes in this concept, allowing a user to explore the animated form in a playful environment. A minimalist version of the original prototype, with some modifications, was made for using the app *Looom.* This version only used the interactive portions as a sidebar on the screen, seen in Figure 49. This modified version streamlined the interaction for a user, like me, who is familiar with the way the *visual-audioizer* systems works. The method in which I used this layout is as follows:

> I would load the *visual-audioizer* patch and place it on the left side of my screen. Using my IPad, *Looom*, and a streaming application called *AirServer*, I streamed the IPad screen to my desktop. Within Max, there is an object that can use your desktop screen as the source of video information. The amount of resolution information streamed into Max can also be controlled. I used a separate screen within Max (jit.pwindow) attached to the desktop stream to find the correct dimensions, and used the *Looom* application to draw a guide to animate within. Because my laptop screen was also touch-sensitive, I could place my IPad onto the keyboard portion—making the two in proximity of one another.[47] I created a video showcasing this interaction, an example of the layout can be seen in Figure 50.[48]

[47] Note that using a laptop device that has a detachable keyboard/screen makes this interaction setup easier.
[48] Visual-Audioizer Live Example from Taylor Olsen: https://www.youtube.com/watch?v=wWBe3RSUuOA&list=PLzNnI_tCy5c_ETnD576c-bT1kerlXqh9I

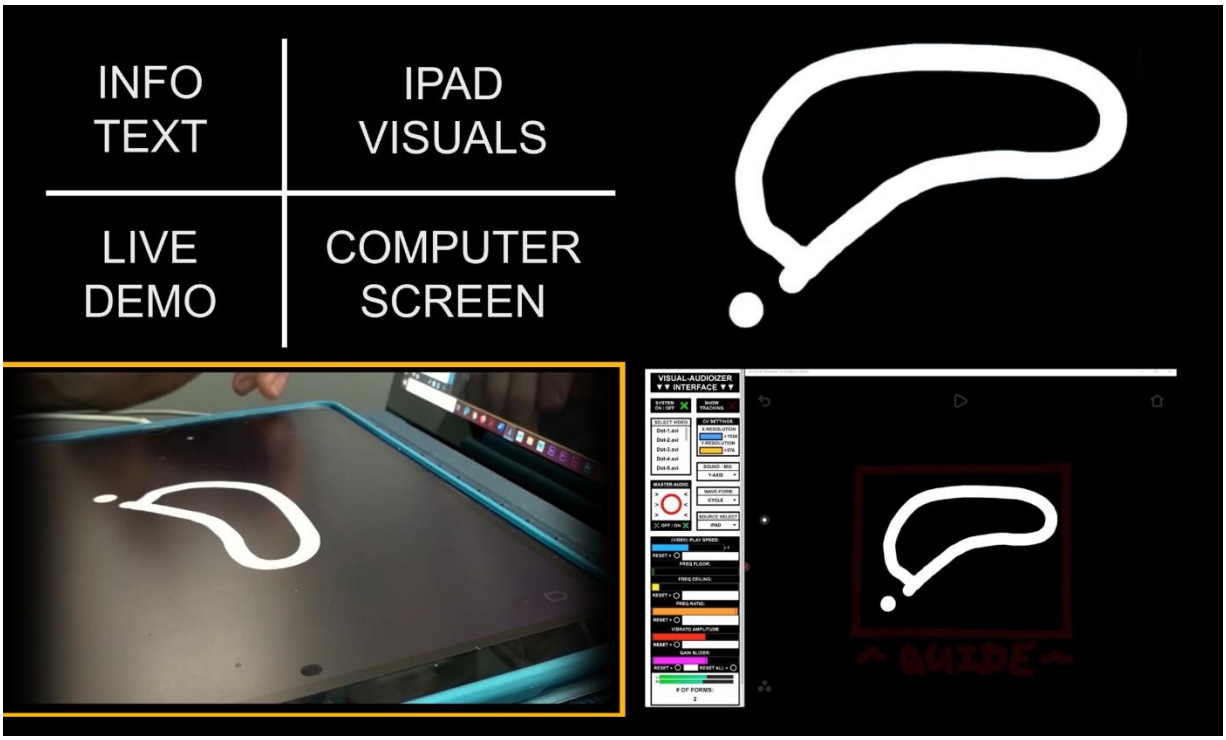


Figure 50: *Visual-Audioizer* Live Demo Example

The Visual-Audioizer Prototype: Data Size and "Blips"

Another issue, still occasionally to this day, is a small noise 'blip' that comes with every new frame of content. This was noticeable during the time with the white cubes, as this was sometimes accompanied by a low-banging drum sound. I found there is a setting within Max to determine the amount of in/out information being used in a single processing loop; I/O is where the system stores the audio before sending it out to other places within the computer. If you increase this I/O vector size within the audio setup in Max, this allows more information to pass through before it jumps to the next processing loop. Initially, I was not aware that this would be a needed change—it did not occur to me the visual information, process, and audio output would need a larger data pool size per cycle. This

makes sense, seeing as the default setting of the patch is aimed towards signal processing. The *visual-audioizer* must read data, process, and provide a signal for each object it processes—the number of objects could also range anywhere from 1 – 50 objects. Again, we can see at least 255 objects, but the processing time and the sound is almost unusable. I found based on using a few different machines (a Windows laptop vs. a Mac), the sweet spot was around 25-35 objects maximum if I wanted to maintain at least 24fps.

From the Max documentation, "With an I/O vector size of 256, and a sampling rate of 44.1 kHz, MSP calculates about 5.8 milliseconds of audio data at a time." This finding became important to consider. If we consider the animation standard of 24 frames-per-second, this equates to 41.66 milliseconds per frame. If we increased the I/O setting to 512, the time takes 11.8 milliseconds of audio at a time. The vector size that I ended up using was 2048; this number equated to 23.6 milliseconds. I was not worried about this affecting the interactive portion of the *visual-audioizer* patch, as I wanted the audio to be clear over the interactive slider values. This time was just over half of the 41.66, and meant it gave the patch enough time to read all the data from each frame and output accordingly. Asking the patch to output more audio than needed, seeing as audio was only going to change if the frame changed, wasted processing power and made the annoying data-overload "click" that plagued the interaction. Changing these values does not ultimately get rid of theses clicks but reduces their amount. Using different audio drivers can also improve this latency and makes better use of your hardware. [49]

[49] For more information about audio drivers, I/O and signal sizes, and a general overview of the Msp audio settings within Max: https://docs.cycling74.com/max5/tutorials/msp-tut/mspaudioio.html

Chapter 4. Review and Evaluation

As my design process mentions, after creating a prototype, I would return and interrogate the validity of my design questions. My goal for these questions, mentioned in the Introduction, are to prove their validity and determine if the outcome of the prototype has achieved their goals. The first, second, and third questions for review:

1. How can computer vision aid in the translation of visuals to audio?
2. From an animator's perspective, how would one turn frame-by-frame animation practices into a real-time instrument for musical expression?
3. What creative effect does real-time user manipulation of data within the translation of visual-to-audio synthesis demonstrate?

For the first question, I turn towards the process of utilizing computer vision to derive information from any source of streamed content, such as videos, webcams, or animations. Using computer vision as a real-time tracking device allows for the instantaneous retrieval and manipulation of data. With the use of computer vision, there also comes the ability to train and tweak the system to recognize forms within a certain visual spectrum of your own parameters; and of the varying level of complexities within computer vision systems, the methods described in this research (number, position, scale, orientation, elongation) using Pelletier's Max patches are approachable and reproduceable by someone who has a familiarity of Max or visual programming languages.

The *visual-audioizer* prototype serves of a proof-of-concept in this regard. While creating animation in any non-objective form can be its own time-intensive burden to tackle, such as the case with Norman McLaren, using computer vision to continuously

identify and calculate data removes the burden of having to attribute sounds to the animated form on frame-by-frame basis. Having multiple forms on screen would also prove a difficult task to track and attribute sound depending on the complexity of the forms' motion. If one were to attempt to play a note for each form on the screen as a live-performance, this would be possible—but upwards of 5 or more forms at a time, with no telling of when the form might disappear/reappear, becomes difficult to handle. With repeated exposure to the animated sequence, one could practice in this way of attempting to match the audio to the motion, but the *visual-audioizer* can analyze and output on the first playthrough of content. The necessity for the system to be real-time proved, using computer vision as a visual aid, it would be able to keep up with fluid-time animation practices as seen within *Looom.* Without it, and if processed over a period, the system would have no need for this type of animation practice and would remove the notion of using animation as a real-time musical instrument.

The validation of the second question is influenced from answering the previous. With traditional frame-by-frame animation practices used as a common method for animating, the process itself is time-intensive and would not necessarily be considered musical live-performance material unless viewed alongside visuals as an end result; this end result is the case from the animated music video project as described in Chapter 3. Also, the practice of hand-drawn animation is traditionally done frame-by-frame; there has nary been a method in which one can animate instantaneously and have control over those frames' afterword, along with an audio output. This is not to say one could not practice frame-by-frame animation techniques and use the output as a tool for creative inquiry.

Using the *visual-audioizer* in fact encourages this method of animating and allows one to take their practice and sonify the outcome simultaneously. But this is still not real-time performance material and does not satisfy this research question being interrogated. In the case of the Oramics machine the process was real-time, and an artist would draw their sounds. But one would only hear them after the strip was fed into the machine. The gesture from the hand of the artist may have made the sounds, but the images themselves were viewed and analyzed as waveforms, as thousands of frames per second, rather than as traditional frame-by-frame animated content. Rather than allowing the drawings to be expansive and dynamic upon their canvas, the drawings were focused on a lateral motion; the strips were all moving in the same direction.

Though Oramics limited itself with this lateral motion and treated the visuals as waveform analysis, some of the principles used within the machine are comparable to the fluid-time animation techniques used within the IPad app, *Looom.* For example, *Looom*'s frame-by-frame animation techniques use tracks. These tracks are like filmstrips, allowing one to instantaneously change the length of the track, anywhere from 1 to 32 frames, up to a total of 5 tracks. This can be limiting if one wants more frames to work with but encourages the animator to use frames strategically (as animation practices look to lessen the amount of frames needed for a motion to feel complete) when creating audio. There is also the ability to change the rate of playback per track, meaning I could have a track use 6 drawings, but only play at 3 frames-per-second, over the course of two seconds. Like the filmstrip, one can "play" and view the content within the track on a loop—but this is where fluid-time animation comes into the animation process. While the looping track is playing

and a user begins drawing, the gesture from drawing, over the period the track is playing, becomes the animated content within each frame. This content is immediately recognized by the *visual-audioizer* and elicits a deeper response to the animated content made by the animator or computer musician. I as the user can also go back and specifically target these frames and change the animated content, as if I were to edit content on a filmstrip; this digital method allows one to add, edit, and erase content from a frame easily and allows the animator and computer musician to quickly return to their practice. Timing and spacing of animated forms, as well as the animation principles, can become a useful tool when sonifying, as the change in motion (translation, scale, orientation, and so on) creates different sounds. To conclude, the immediate association of the drawn form when using the *visual-audioizer* and *Loom*, turns frame-by-frame animation practices into a real-time instrument for musical performance.

For the third question, I return to the outcomes of the first and second, supplementing how real-time user manipulation of the visual data demonstrates a cyclical creative inquiry into the act of animating and performative music. Without computer vision and Max, there would be no association between these visuals and no data to manipulate. Without real-time techniques like fluid-time animation, there would be no need for the practice of animating nor a desire from the real-time performative artist to consider the animated form as a musical instrument. If I was to consider animation as an instrument, I wanted the ability of the interaction in which I animated to be controlled, much like how I can control what note is played along with the awareness of how many notes, as well as the range of sounds, I can play.

A new advantage becomes apparent within the animated form when using the *visual*-audioizer and *Looom*, as the proposed keys and fingers on a piano become the canvas and the animated form, respectively. *Looom*'s canvas is vectorized, meaning I can infinitely stretch/shrink on my own volition. This ability to change the canvas lets the animator and computer musician to literally see and hear their compositions from new angles. Coupling this change with the audio-mapping options used within the *visual-audioizer,* discussed in Chapter 3 and Chapter 5, the animator and computer musician can draw a single motion and experiment with mapping options for a large degree of sonified output possibilities.

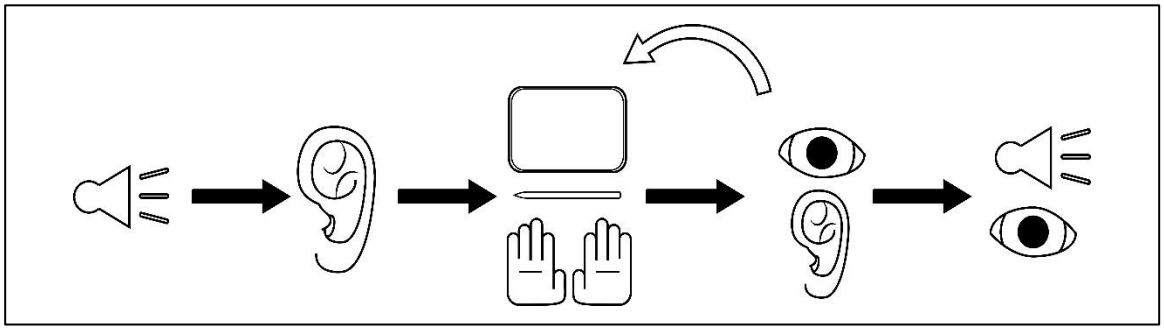


Figure 51: General workflow when animating to audio.
Outcome is watching music and visuals as a synonymous experience.

If someone were still attempting to recreate the effects like McLaren and/or Fischinger, it would require a substantial amount of planning and busy-work to create the desired outcomes. For the animator, this process can be observed as seen in Figure 51. This process is dependent upon having a piece of audio to animate to. The animator listens to the audio, animates to synchronize with the audio in some way, watches their animation and listens for synchronized cues, and the result is the animated visuals with the original audio. There is room for changes, but only in the visual process. To change the audio would

require rerecording from an external source. If the desired outcome is to create an animated work to compliment a source of audio, this process is fundamentally usable, but linear in its outcome.

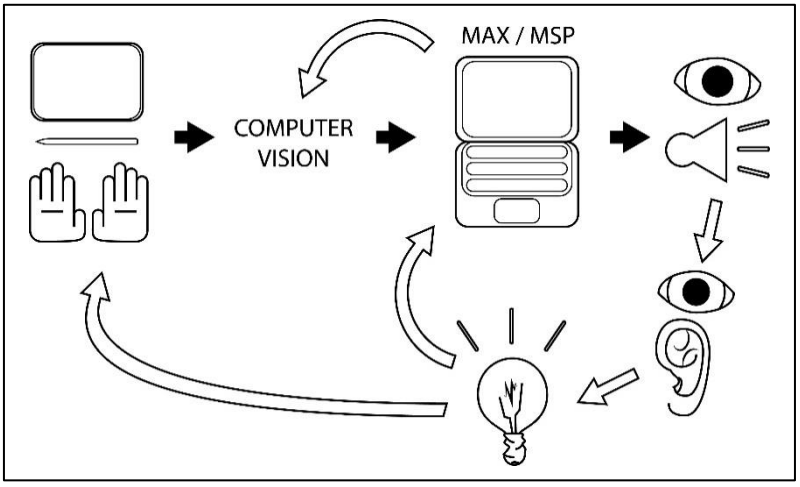


Figure 52: Animation as a musical instrument workflow.

The implementation, process, and outcome of using the *visual-audioizer* is ultimately up to the animator and/or computer musician, but a generalized workflow can be surmised from having the ability to manipulate the visual data (seen in Figure 52). Under the assumptions and goals of this research, it is the visuals that create the audio and its resultant outcome becomes cyclical. Rather than letting a piece of audio be the constant variable and the catalyst for the resultant output as seen in the previous workflow, using the *visual-audioizer* prototype, and *Looom*, allows the visuals to begin the process and output audio. But it does not stop there, if a user is more attuned to making a different sound, this can be changed done by changing the animation. Alternatively, if one is happy with their animation, changing how the computer vision data is interpreted is another route. This encourages one to return to the animated motion, or to the data manipulation, to edit the resultant audio.

The user will ultimately have to begin with visuals, but because of the cyclical workflow that comes the moment a visual is recognized, the effect of manipulating the data in real-time demonstrates multiple avenues of creative inquiry in both the audio and visual realm. While one avenue (visuals) is needed to start the cycle, the two mediums become unified and a completely new form of audio-visual expression is created. In the case of Iwamiya's cooperative enhancement, as described in Chapter 1, having both mediums simultaneously work together with clear causality bolsters the experience—throw in the addition of being the one to create these visuals and sounds and the causality is immediate, allowing the user to focus on the creative aspects of manipulating the audio and the visuals and worry less about synchronization. With the *visual-audioizer* interface, having the ability and interface to change how the digitized motion is interpreted allows instantaneous feedback and insight into the motion of the animated form; allowing the creator the ability to experiment, edit, record, and create musical pieces from their synchronized visuals.

Chapter 5. Results & Future Direction

Interface, Mapping, and Sound

The interface for using the *visual-audioizer* is within a max patch. For the tracking of the visuals the CV.jit objects from Pelletier are utilized on a two-dimensional axis. Within the interface there are multiple aspects of control for the audio output. The patch gives the user the ability (in real-time) to read fluid-time animated forms from a streamed source, or pre-animated files; allowing the user and an audience to witness the tracking of the animated forms coupled with digital audio output. The ability to speed-up/slow-down the visuals is available both in the *Looom* application, and within the *visual-audioizer* patch via pre-rendered files. Within the pre-animated sequences, considerations of timing & spacing, (as well as squash and stretch—a supplementary animation principle) are represented. There is also the ability to switch between MIDI and Msp sounds – allowing the user to consider the implementation of digitized instruments vs. that of generated audio signals. The Msp objects such as cycle~, rect~, saw~, and tri~ are utilized and easily interchangeable via the interface. The ability to alter the pitch is also within the interface, allowing the user to adjust the frequency range of the audio.

The method to interpret the visual data into the various options of visual information include utilizing the position, scale, and elongation of animated forms. Pitch is mapped to the position of the object, the scale is to amplitude (the percentage of screen space the form takes up is proportional to amplitude), and the elongation of the form is mapped to the frequency modulation index of the audio output. The position of the form also dictates how the panning of the pitch is interpreted; meaning forms on the L/R side of the screen directly correspond to the output of audio in a L/R speaker setup. To review, this flow of data is shown Figure 28, found in

Pitch-mapping of the visuals include various options of reinterpreting the translation of the x/y data into digital audio signals via the interface. From low to high pitch, the various methods of data mapping to pitch within the system can be seen in Figure 53. The methods of reading the visuals via the CV.jit objects include beginning the analysis from the top-left to bottom-right of the screen space. These include x-axis (left to right), y-axis (bottom to top), x- & y- (top-left to bottom-right), x+ & y+ (bottom-right to top-left), x-split (lower pitch in the center, higher on the left/right edges), y-split (lower pitch in center, higher on bottom/top edges), center-out (lower pitch in center, higher in corners), and center-in (higher pitch in center, lower in corners). Positional coordinates for panning remain constant; to reiterate, objects on the L/R side of the screen space dynamically adjust and correspond to the L/R speaker output.

For the moment, it is important to remember that the prime method of correctly utilizing the *visual-audioizer* is to animate with black and white forms; future uses of color tracking will be utilized, but the current status of the patch works best with this intentionality. While there is no correct way to map and record the audio from the patch, the considerations of animated motion and the ability of the animator to control these motions in pre-rendered visuals, as well as fluid-time performance, becomes an overarching discussion in the use of CV methods to aid in the sonification of animation. I consider the *visual-audioizer* as a working proof-of-concept; meaning there is room in the future for considerations of color, pictorial ambiguity, and deep-learning techniques to allow visual elements within complex settings to be sonified.[50]

[50] https://www.nationalgeographic.com/news/2017/04/worlds-first-cyborg-human-evolution-science/

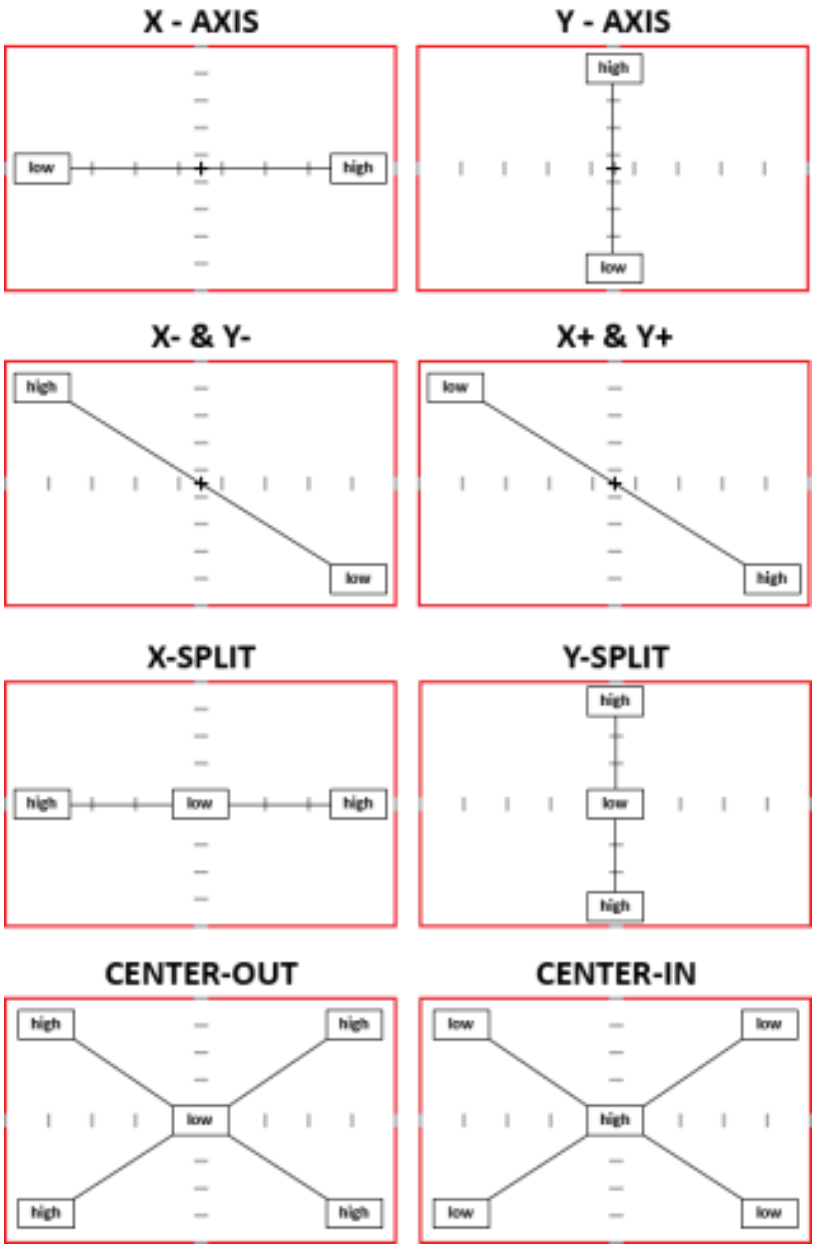


Figure 53: Pitch-mapping examples

Multiple Forms

A caveat for the animator when using the *visual-audioizer* is to consider the use of

multiple forms. The patch parses out the multiple objects within the scene and sends each piece of information to a poly~ object. A max of 255 forms can be recognized. The *Looom* app specializes in the ability to create editable framerates as separate layers; this is great considering how complex imagery and patterns could be digitized and sent to multiple outputs; allowing the simultaneity of polyrhythms to be sonified and interpreted henceforth.[51] Depending on the amount of recognizable forms, an 'uzi' object sends out the grouped data to the poly~ object. And though the *visual-audioizer* can discern multiple objects, it can also be the cause of straining the patch and not being as synchronous as hoped. When playtesting with the number of objects in a scene, I noticed when the number of discernable forms jumped back and forth from 1 to 100, the patch had a hard time interpreting the data quickly enough for the poly~ audio to be simultaneous with the visuals. This could be solved by having the data interpreted, digitized, recorded, and played back on its own—but ultimately removes the notion of the real-time feedback produced by CV methods.

Form Proximity

Another consideration for the animator when creating content for the *visual-audioizer* is the proximity of forms. If some forms were too close to one another, the CV method groups them and creates a unified sound, rather than separate pieces of audio. Though this can be solved by applying a threshold to the proximity-grouping method in the CV patch, it often became a headache of tweaking the settings until a 'perfect' scenario

[51] https://www.britannica.com/art/polyrhythm

was produced. Being flexible with the animated form is a goal of utilizing this patch and knowing how to work within the limitations of the CV methods will yield more results for the creative who enjoys experimentation.

Frame-rate and Codec

Another caveat for the animator to consider is the framerate of the video input. Considering the range of human hearing can lie anywhere from 20 – 20,000Hz, the human eye is only able to identify framerates anywhere from 1 – 150 frames-per-second (fps), or 150Hz; the standard framerate that an animator works with is 24fps (a visual 24Hz).[52] This in turn can cause some limitations in the process of providing enough visual "depth" within the motion of the animated form to create a desired envelope with audible nuances. In relation to this visual "depth", specific video codecs should be utilized for the flow of visual information, data extrapolation, and audio output to be as synchronous as possible. Visually dense codecs that utilize "lossless" quality will slow the system, while codecs such as HAP will substantially reduce the amount of CPU usage and allow the patch to produce real-time audio with more synchronicity to the visual input. When animating in fluid-time, the *Looom* app can vary the speed of these framerates on the fly, allowing the framerate to be edited and varied for the consideration of polyrhythms. To retain visual complexity and audible synchronization, the user should lower the resolution of the streamed content into the *visual-audioizer* and stream the full resolution fluid-time visual performance to a separate source.

[52] http://www.cochlea.org/en/hear/human-auditory-range

Scale and Envelope

When designing the scale of the animated form using the *visual-audioizer*, it is intriguing to note that the rate of scale increase can draw parallels to that of synthesized envelopes. For example, if the form dramatically increases in scale over a short period of time this is comparable to the "attack" of the sound. Similarly, if the animator were to diminish the scale of the form over a shorter/longer period, or fall to zero, this is the "decay", "sustain", and/or release of the sound.[53] Experimenting with increase/decrease in scale can lead to dramatically different results in the ADSR spectrum; and coupled with the ability of the animator to control scale over time, this can lead to complex techniques with holds and atypical modulations in the envelope of a sound. This allows the animator to consider the size of their forms as an audible dynamic range. A limitation of envelope "attack" is to consider the previous paragraph speaking about the framerate of an animation; while the animator can control the scale of the form, the shortest amount of "attack" is 1/24th of a second (41.66ms) if the animator is following that of standard animation practices.

Conclusion

The *visual-audioizer* prototype is designed to contribute the conversation of utilizing visual information as a tool for creating audio. Animation plays a critical role in the usage of the prototype and demonstrates the practice of animating frame-by-frame, and within in a fluid-time

[53] https://blog.landr.com/adsr-envelopes-infographic/

animation environment, provides a new experience of creating and manipulating digital audio. Though there is a myriad of ways to consider visual complexity and CV methods to create music, utilizing animation principles provides a clear direction in the creative process when using the *visual-audioizer* as a tool for musical expression. I hope for a surge of CV technologies to be incorporated into the creative process of sonifying visual information to its full extent. For example, there is always room for improvement in the methods of reading visual data; including RGBA values, hue/saturation/opacity, and 3D imagery. Coupling this with the ability of the *Looom* app, colors could then be mapped as different ways of altering digital audio derived from animated motion.

Beyond the realm of animation and computer music, I want to note the considerations in which this system could be used. One method I have thought about is in the case of music therapy. While some people might be musically inclined when receiving music therapy, using the interactive method with the iPad allows a natural connection between the user's gestural movements and the sound being generated. Another use would be in elementary music settings—I have considered what creative effect there would be on child's development towards music if their first interactions with an instrument was through their drawings.

For the artists and designers who intend to push the envelopes of animation making as a visual and rhythmic artform will find the *visual-audioizer*, as a new way to push CV methods further into the hands of those willing to create music with the animated form. With the rise of new technology and tracking methods, I propose both artist and audience will see a growth of fundamental CV aspects be incorporated into the creative process of musical expression—and should welcome these technological/conceptual advancements not as a limitation in the field of

musical expression, but as ever-expansive insights into coexisting with modern/future creative techniques. Furthermore, I believe the *visual-audioizer* will inspire others to explore the layers of depth the animated form can hold within the spectrum of audio creation, experiment and explore visual-programming, and welcome those who have not explored visual-programming or considered CV methods to elicit audio and create music.

Bibliography

"Teenage Engineering." OP-1. Accessed February 15, 2020. https://teenage.engineering/products/op-1.

"Teenage Engineering." OP-Z. Accessed February 15, 2020. https://teenage.engineering/products/op-z.

Akira. VHS. Japan: Tokyo Movie Shinsha, 1988.

Beck, Jordan, and Erik Stolterman. "Examining Practical, Everyday Theory Use in Design Research." *She Ji: The Journal of Design, Economics, and Innovation* 2, no. 2 (2016): 125–40. https://doi.org/10.1016/j.sheji.2016.01.010.

Candy, Linda. "Practice Based Research: A Guide - Creativity and Cognition." Creativity & Cognition Studios. University of Technology, Australia, November 2006. https://www.creativityandcognition.com/resources/PBR%20Guide-1.1-2006.pdf.

Chion, Michel, *Audio-Vision: Sound on Screen*. New York: Columbia University Press, 1994.

Detheux, Jean. "Neither Fischinger nor McLaren, Visual Music in a Different Key," Animationstudies 2.0, accessed March 15, 2019, https://blog.animationstudies.org/?p=346.

Faber and Faber.

Fenderson, Jerobeam. "How To Draw Mushrooms On An Oscilloscope With Sound," Youtube, March 25, 2014, accessed March 15, 2019, https://www.youtube.com/watch?v=rtR63-ecUNo.

Groening, Matt. Whole. *Futurama,* Season 1 to 7. Fox / Comedy Central, 1999 - 2013

Hughes, Dave. Whole. *Off the Air*. Adult Swim, 2011 – 2020

Hutton, J. Daphne Oram: Innovator, writer and composer. *Organised Sound, 8*(1), 49-56. doi:10.1017/S1355771803001055, 2003.

Iwamiya, Shinichiro. Interactions between auditory and visual processing when listening to music in an audiovisual context: 1. Matching 2. Audio quality. *Psychomusicology : A Journal of Research in Music Cognition* 13, 1-2 (March 1995), 143, DOI:http://dx.doi.org/10.1037/h0094098

MacFarlane, Seth. Whole. *American Dad,* Season 1 to 17. Fox / TBS, 2005 - 2020

Max Mathews, "Max Software Tools for Media | Cycling '74," Max Software Tools for Media | Cycling '74, accessed March 15, 2019, https://cycling74.com/products/max/.

McLaren, Norman, "Synchromy - Norman McLaren," YouTube, November 24, 2011, accessed March 15, 2019, https://www.youtube.com/watch?v=UmSzc8mBJCM.

McLaren, Norman, “Norman McLaren: Pen Point Percussion,” Youtube, April 28, 2008, accessed March 15, 2019, https://www.youtube.com/watch?time_continue=340&v=Q0vgZv_JWfM.

Oram, D. *An Individual Note.* Norfolk: Gaillard, 1971.

Pelletier, Jean-Marc. 2008. Perceptually Motivated Video Sonification. (2020). Retrieved March 11, 2019 from hrrps://jmpelletier.com/sonification-demos/

Pendleton, Ward. Whole. *Adventure Time*, Season 1 to 10. Cartoon Network, 2010 – 2018

Princess Mononoke. VHS. Japan: Studio Ghibli, 1997.

The Brave Little Toaster. VHS. United States: Hyperion Pictures, 1987.

The Land before Time. VHS. United States: Universal, 1988.

Utako, Kurihara "Norman McLaren's animated Film *Rythmetic* as Temporal Art". The Japanese Society for Aesthetics Sienan Gakuin University, Fukuoka, 2011.

Vasileva, Mila, "Images To Sound," YouTube, September 17, 2016, accessed March 15, 2019, https://www.youtube.com/watch?v=WgZ01bAOMMU.

Victor Khashchanskiy, "Bitmaps & Waves," Bitmaps & Waves, accessed March 15, 2019, http://victorx.eu/BitmapPlayer.htm.

Wells, Paul. 2013. *Understanding Animation*. London, Routledge.

Williams, Richard. 2009. *The animator's survival kit*. London: Faber and Faber.