q[[1]](#footnote-0)

MUSI-3770: Group 4

Project

Davis House, Jordan Chatfield, Terrence Graham

# INTRODUCTION

There are various programs meant for automatic music generation as well as those that can generate a video based on incoming audio. However, there don’t seem to be any applications that can generate ambient music using real-time data derived from an input video. Many music generators use an AI model that is trained on a database of existing compositions and then plays the resultant composition using a set of predefined samples. Our project was initially focused on allowing the user to design their own sounds using simple oscillators, filters, effects, and LFOs. We wanted to have a system that provided the user with plenty of timbral variety so that someone with almost no knowledge of music theory could still create custom soundscapes for their video projects. However, we faced many challenges with the incorporation of the wavetable component in our system and had to settle for a much simpler implementation due to time constraints.

# Motivation and Problem Statement

Sight and sound are intrinsically connected, and as art seeks to more fully engage viewers through their senses, the need for audio to be paired with visuals becomes more apparent. Even silent films needed musical backing tracks. Based on our preliminary research we were unable to find systems made to generate ambient music based on video input. We have also yet to find a program that can seamlessly integrate video with generated sound. As such, we would like to fill that niche by creating an application that generates original sound in time with a video and outputs the original video with the new score attached. We feel this combination could be useful to the world of music and would lend itself to many creative applications. Along with original videos, this program could also breathe new life into existing video recordings.

# Applications

The intended use for this program is to give users the ability to generate sounds that may serve as an accompaniment to any video. This could be useful in many content creation scenarios; for example, YouTube videos and independent films are always in need of suitable scores. We also believe that our system could add more intrigue to existing visuals. For example, one could take a scene from their favorite movie and run it through our program to hear an alternative score based on concrete visual information. Given that human-composed scores are vital in creating a viewer’s subjective experience of a given scene, our generated scores would inevitably recontextualize what the viewer sees and perhaps prompt them to appreciate the video from a different perspective. In addition, the pattern of sound output by our system would be unique to each video making it great fodder for sampling in the user’s own compositions.

# Related Work

There are a few programs that can automatically generate music, such as Soundraw.io or Chrome Music Lab, but they don’t have any video analysis element. We discovered some programs that can create and edit a video in response to incoming audio, such as the Max for Live device, RokVid. We also found plenty of applications that are capable of video feature extraction (like OpenCV).

# Goals/Research Statement

We had three main hurdles to overcome in building this project. The first was to successfully create an algorithm that could analyze a video and extract discrete image characteristics that could be mapped to synth controls. This was accomplished by analyzing the video through OpenCV. Our second goal was to actually build a robust wavetable synthesizer with several options for waveforms and plenty of mappable parameters that could be customized as the user prefered. Finally, the third goal was to find a suitable and efficient logic for mapping the image data streams so that the connection between video and audio would be apparent to the user.

Our interest lies in creating a music generator that does not rely on other compositions in order to derive its output. The sonification of a video object is something that we believe will produce original and unexpected results.

# Proposed Method

This project was split into three major tasks based on the aforementioned goals: video feature extraction, sound synthesis, and feature mappings. Davis was largely focused on the portion of the system that takes a video file as input, analyzes each frame of video, and then extracts 4 image features that vary over time. Computed once per frame, these features are the mean red/blue/green values and the brightness value. The data stream from these features act as control signals that will be sent to three separate oscillators in the synthesizer. The frequencies of oscillators 1, 2 and 3 are determined by the mean red, green and blue respectively. Amplitude is derived from the mean brightness. Each value ranges from 0 to 255. Terrence focused on the sound synthesis itself and the implementation of basic sound design tools (filters, FM, LFO, reverb, etc). Jordan was responsible for coming up with a method to decide how each video characteristic should be mapped onto the synthesizer. By the final presentation we planned on having a more robust video analysis system that could extract even more video features, a system for mapping that works and is flexible enough to give the user sufficient control, and a more sophisticated synthesizer capable of outputting a greater variety of polyphonic sounds utilizing several of the modular effects mentioned previously. We were also considering the option to allow the user to reroute the mappings themselves and to have a method of loading one's own samples into the synthesizer, either as background noise or as a spectral envelope to be applied to the synthesized sound. Ultimately, these expansions of synth functionality did not make it into the final product.

*A. Midterm*

As we approached the midterm presentation, we had an almost fully working system for video analysis and a synthesizer that could only output very basic sound. The two systems were not yet integrated with one another.

We had succeeded in generating wavetables as arrays and reading those tables into the audio buffer such that the system would output sound, but we were unable to accommodate polyphony in order to create basic harmonies. Frequency control was nonexistent at this stage which meant that we really didn’t have much of a synth at all. The sound we were generating sounded like jumbled noise instead of a coherent tonal piece.

At this point the feature extractor was mostly done and we decided to put most of our focus toward troubleshooting our synthesizer. Unfortunately, we ended up running into more problems than solutions and discovered several mistakes in the construction of the synth that made it computationally impractical. We essentially had to go back to the drawing board to re-write some of the foundational elements of our synth.

# Challenges

We discovered that Python has some less-than-convenient limitations when it comes to real-time synthesis and audio output. For every synthesis problem we fixed we found that it created more problems elsewhere. It took a long time just to get our frequency control implemented, and then it took even longer to get the system to reliably output pitched audio. This was our first time trying to build a wavetable synth of this kind and our lack of a comprehensive understanding of Python and wavetable synthesis proved to be a major hurdle. Poor time management definitely hampered our ability to fully realize our initial plan. By the time we decided to try combining the video analysis and synthesis objects, we had very little time left to fix the numerous kinks that inevitably appeared. Most of these problems were so fundamental that it was very difficult to test any individual parts of the program without addressing them. When it came time to think about adding audio effects we realized we had no idea how to apply these effects to an array and output the effect in real time. As a result, audio effects had to be forgone in the final product.

The division of tasks among our group was not very efficient. It may have been helpful to have several people work on the synthesizer first to ensure that it was functional and ready to receive control data. Without a fully functional synth we weren’t able to implement the user customization elements as originally intended.

# Evaluation/Measure of Success

We succeeded in creating a system that generates real time audio which accurately reflects the action of the video. Our mappings work properly and it’s apparent to the user that the image is having a direct effect on what’s being heard. In that regard we have a solid proof-of-concept. However, we weren’t able to refine the audio enough that it can really be called “ambient” music. The timbre of the sound is simple and dry because audio effects did not make it into the final product. Our synthesizer has multiple oscillators operating simultaneously. but we did not add a way to define harmonic relationships between them or create chords. The user experience is not the smoothest as we were not able to create a GUI for the program. For the final product we were forced to make some significant compromises from our original vision due to poor planning, team management, and lack of experience creating systems of this kind.

# Resources

All video analysis and characterization was done in Python using the OpenCV library and related functions. Additional libraries that were used include Numpy, Scipy, and PyACA. We used Python’s PyAudio module to process and format the arrays for real-time playback.

# Deliverables

The final deliverables of this project include the full python code, commented with notes and keys, and a sample video with the corresponding generated sound.

# Novelty

This is intended to be a tool for creating scores for your personal video recordings. It could also be used simply for enjoyment by being applied to your favorite pieces of existing video. Our system is essentially a synthesizer that has its parameters controlled by variable video features. The process is semi-automated and allows for some user control over the textures being output. The nature of the music being created means that the accidental generation of copyrighted music will not be an issue (the output will be simple drones and noise).

From our research we found that not many people are creating a program or application in the style that we are. There are ambient music generators and video feature extractors, but seemingly nothing that combines the two of them. We hope to bring this project to full completion in order to open up this new product to the world of music and video creation.

1. [↑](#footnote-ref-0)