Winter 2023

An Interactive System for Generating Music from Moving Images

Hanlin Wang
Dartmouth College, hanlin.z.wang.gr@dartmouth.edu

An Interactive System for Generating Music from Moving Images

A Thesis
Submitted to the Faculty
in partial fulfillment of the requirements for the
degree of

Master of Science

in

Computer Science

by HANLIN WANG

Guarini School of Graduate and Advanced Studies
Dartmouth College
Hanover, New Hampshire

November 2022

Examining Committee:

_____
Michael Casey, Chair

_____
Lorie Loeb

_____
James Mahoney

_____
F. Jon Kull, Ph.D.
Dean of the Guarini School of Graduate and Advanced Studies

Abstract

Moving images contain a wealth of information pertaining to motion. Motivated by the interconnectedness of music and movement, we present a framework for transforming the kinetic qualities of moving images into music. We developed an interactive software system that takes video as input and maps its motion attributes into the musical dimension based on perceptually grounded principles. The system combines existing sonification frameworks with theories and techniques of generative music. To evaluate the system, we conducted a two-part experiment. First, we asked participants to make judgements on video-audio correspondence from clips generated by the system. Second, we asked participants to give ratings for audiovisual works created using the system. These experiments revealed that 1) the system is able to generate music with a significant level of perceptual correspondence to the source video's motion and 2) the system can effectively be used as an artistic tool for generative composition.

Acknowledgements

I am deeply grateful for my thesis committee and the invaluable contributions they have made to the research and writing of this work. I want to thank my advisor Michael Casey for making this work possible, from filling gaps in my music theory knowledge to pointing me to SuperCollider, algorithmic music, and countless other ideas vital to my thesis. I also want to thank James Mahoney, especially for giving me the idea of mapping optical flow fields to music and consequently digging me out of a brainstorming rut, and Lorie Loeb for providing useful ideas for experiment design and taking care of countless logistical matters.

This work is also the product of years of engaging with visual media and music. I want to thank Jodie Mack for introducing me to experimental animation, visual music, and optical sound. I also want to thank Miki Sophia Cloud for helping me explore works by contemporary composers (like Philip Glass), and Marcia Cassidy for keeping me grounded in the musical community during my entire time at Dartmouth.

Finally, I want to thank my friends and family for their continued support and encouragement throughout this process.

Contents

List of Tables

List of Figures

1 Introduction

Moving images are a fixture in the present-day media landscape. We view them as a whole and register what we see (birds flying, people dancing). The perceived motion arising from each frame moving into the next is easily taken for granted. If we take a closer look at the pixel level, moving images provide a unique wealth of information pertaining to motion.

This thesis explores the possibility of mining the kinetic features of moving images to transform its structural and emotional qualities into music. Cognitively, we are motivated by the interconnectedness of music and movement, including empirical evidence of the two sharing a common dynamic structure. Aesthetically, we seek to combine existing sonification frameworks with theories and techniques of generative music.

We designed a computer program that generates music from video data. The program takes video as input and uses optical flow to capture the perceived motion of feature points. It maps the flow vectors into the musical dimension according to perceptual and symbolic relations, then pipes that data into a sound synthesis engine to generate music.

This program is an example of a generative system: the outputted music is determined by the image frames given as input; it progresses and evolves without the artist's control. The goal is to generate music that a) has a level of perceptual or affective correspondence to the moving images, and b) has aesthetic potential for creating audiovisual works. To test the program against both goals, we designed a two-part experiment. In the first experiment, users decide whether several pairings of video and generated music are a match. In the second, participants give ratings for audiovisual works created using the program.

More broadly, we aim to set the foundation for an engaging, artist-centric tool suitable for creating unique sonic possibilities that are outside the realm of traditional musical composition, both in terms of imagination and complexity.

2 Survey of Related Work

2.1 Sonification of Motion and Moving Images

Sonification, or conveying information with sound, has been used in various scientific and industrial contexts, from increasing information access for pilots in the cockpit to rendering sound from data sources as diverse as seismology and electrocardiograms [1]. Less commonly, it has been used in artistic contexts. For example, methods have been devised to creating music from still images [2]. The information stored in still images, however, lacks a temporal dimension. Since music and movement are both dynamic phenomena that exist only in time, there is arguably a stronger case for sonifying moving images. In the sections below we will discuss a few different methods of creatively sonifying moving images that have been explored in the past.

2.1.1 Optical Sound

Early experimental animators such as Norman McLaren conducted experiments using "optical sound" in which marks were directly scratched or painted onto the soundtrack area of film strips [3]. When played through a projector, the sequence of marks would produce sound that he described as "a small orchestra of clicking, thudding, buzzing and drum-like timbres."

Figure 1: Norman McLaren drawing synthetic sounds directly onto film [4]

Timbre could be controlled by the shape of the marks, and pitch by the distance between lines. Harmony could be achieved by juxtaposing different patterns along the same section of film. In this way, McLaren combined music and moving images at a fundamental structural level via a "score" that was inherently audiovisual.

2.1.2 Video Brightness as a Control Signal

In 1971, Erkki Kurenniemi designed the "Dimi-O," a video-controlled synthesizer [5]. The instrument could receive optical input from a television or video camera and use the video feed as a control signal to produce notes. Besides being used to play back graphically represented music, the Dimi-O facilitated some more avant-garde applications: controlling the synthesizer with the performances of ballet dancers or even experimental animations to create music.

Kurenniemi's instrument sonifies changes in brightness, which is indeed an indicator of perceived motion, albeit a restrictive one. Using the video feed locations as a set of on-off switches allows for nearly complete freedom in the instrument's input but compresses the input information into a single dimension, thus limiting the musical variety of its output.

2.1.3 Perceptually Motivated Sonification Using Optical Flow Fields

Pelletier devised a framework for sonifying moving images using optical flow estimations [6] [7]. In this methodology, salient image features are first identified using a corner detection algorithm. These features coordinates are then passed into an optical flow estimation algorithm, which describes the apparent motion at a point. The optical flow estimation returns a flow field, a set of $(\Delta x, \Delta y)$ motion vectors at each feature point describing how far the point has moved. Features are re-computed at each frame to account for objects entering or leaving the image bounds.

Pelletier's framework is "perceptually motivated"—taking into consideration the way sounds and images are perceived, in the hopes of creating music that "sounds like what it looks." It is grounded in perceptual and psychological principles, primarily Gestalt principles [7].

Gestalt principles start to come into play in the corner detection stage, where the original image is reduced into a sparse set of features. Even though this set of features contains vastly less data than the original image, it is still possible to identify the makeup of the image because of perceptual grouping.

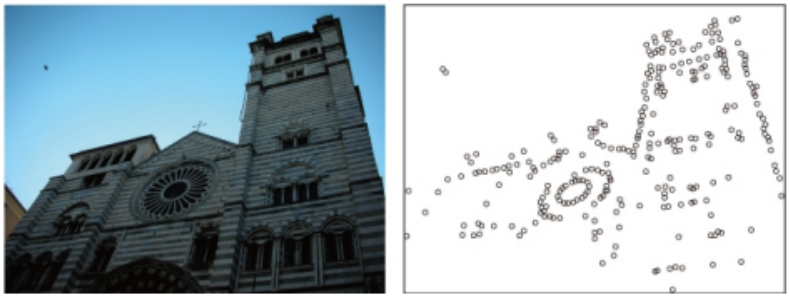

Figure 2: 283 corners detected in an image

In addition to visual gestalts, Pelletier describes how *sonic* gestalts come into play in this framework. Gestalt principles can describe and predict groupings in auditory perception analogously to visual perception [8].

To perform the conversion of motion into sound, Pelletier devised the following mapping, each motivated by different perceptual considerations:

Space: Assign the normalized x value of each motion vector to the stereo pan position of the sonic element it corresponds to. This is motivated by the inherent spatial nature of images. Mapping visual location to stereo pan position preserves the relationships that existed between visual features in the sonic dimension. Sonic components are clustered primarily through the principles of *proximity* (nearby features are perceptually grouped) and *common fate* (features that move in the same direction are grouped).

Amplitude: Assign the length of the motion vector (directly related to velocity) to amplitude. Mapping velocity of motion to amplitude establishes a relationship between the dynamic contours of both the moving images and the produced sound. Motionless features remain silent; moving features produce sound proportional to their perceived energy. An additional metaphorical motivation is: since sound can be generated from friction, faster movements lead to louder sounds.

Frequency: Rather than define a singular mapping for frequency, a variety of approaches are suggested. Pitch could be determined by a given image axis (most intuitive), distance from origin, displacement direction, or displacement amplitude. Pelletier argues that the most appropriate mapping would depend on the nature of the input image, the type of sound synthesis technique used, and the intent of the artist. However, the law of common fate plays an important role—visual objects tend to be rigid and have feature points that move in correlated trajectories, lending itself to mapping to correlated pitch trajectories that are likely to be perceived as a single melodic line.

Timbre: Like frequency, the mapping depends on the context. Approaches suggested include image complexity (number of feature points, determined either by the image itself or the parameters of the feature detector), as well as indirect control via the superposition of different sound components. There are cases where features moving similarly in very different parts of the image can be mapped to the same frequencies, and thus be perceived as a single entity. Pelletier suggests remedying this by mapping one of the two dimensions to timbre.

Pelletier's framework is implemented in Cycling '74's Jitter system. For corner detection, either the Shi-Tomasi method or the FAST method are used, depending on whether GPU processing is available. Optical flow is estimated using the pyramidal Lucas-Kanade algorithm.

2.1.4 Color-related Sonification

The installation *sound/tracks* [9] aims to capture the visual experience of looking out the window of a train by translating it into a musical composition. Here the passing scenery can metaphorically be considered the "score" of the musical composition.

This sonification framework is musically motivated by composer and synesthete Alexander Scriabin's mapping between tone and color. Every 7 frames, the middle column of pixels in the frame are analyzed: they are split into 4 sections to generate notes of different octaves. The pixels in each section are mapped to a pitch by computing the cosine distance of each pixel's HSV-value to all twelve colors of Scriabin's Clavier à lumières, and choosing the note with the least cumulative distance [10].

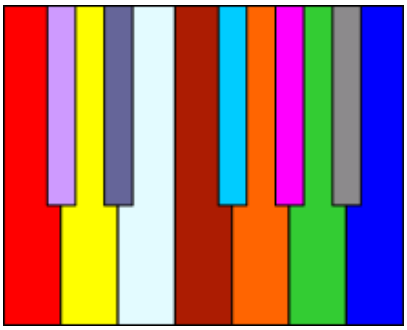

Figure 3: Scriabin's tone-to-color mapping

2.2 Music-Movement Connection

Since our thesis aims to devise a method for generating music *from* movement, it falls upon us to review what has already been researched about the connection between the two. As it turns out, music and movement are intuitively related to each other; this connection has been demonstrated empirically in a number of studies.

2.2.1 Expressing Emotion with a Shared Dynamic Structure

In one study [11], Researchers created a computer program that produced isomorphic samples of music (simple melody) and movement (bouncing ball); both the music and movement were controllable via five parameters that controlled analogous attributes in the music and movement samples.

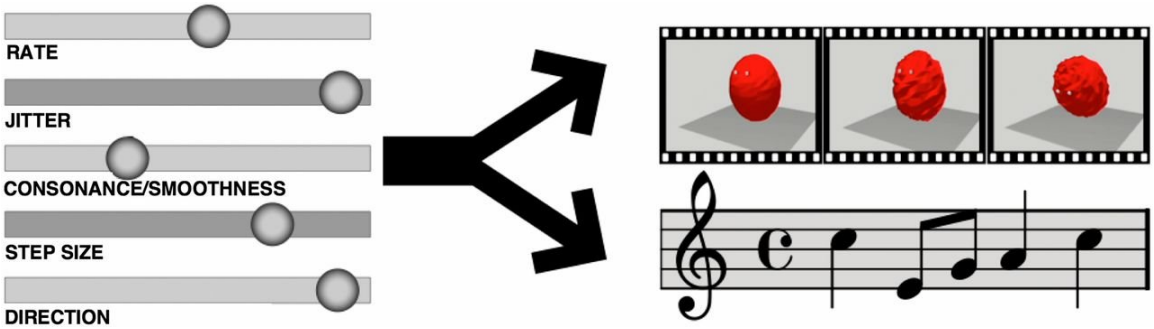

Figure 4: Slider Paradigm for the Sievers et al. Study

After learning to use the program, participants were instructed to express five different emotions by moving the sliders. Half of the participants were to express these emotions by shaping the bouncing ball and the other half by shaping the melody. The experiment was conducted in the United States and in a rural Cambodian village.

The results suggested that expressions of emotion are cross-culturally universal, with similar dynamic contours in music and movement. For example, to express the emotion "angry," participants in both cultures positioned the slider bars in similar configurations for both the bouncing ball and the music.

2.2.2 Music and Body Movement

A number of studies have bridged music and motion specifically with respect to the human body. For example, motion analysis using optical flow has been shown to have potential in quantitatively evaluating active music therapy for disabled children [12]. Rhythm and timbre-related characteristics of music have also been shown to correlate with different types of body movements [13]. One study investigated the effect of music on involuntary body motion when the subject was instructed to stand as still as possible, finding quantifiable patterns linking features of the pulse, rhythmic pattern, brightness, and loudness to the *micromotions* detected in the subject [14].

In one study exploring how people perceive correspondences between music and body movement, participants were asked to create both 'sound-tracings' on a digital tablet and 'free dance movements' that they thought matched a short clip of music [15]. The study found a certain level of consistency in the way different participants interpreted the music in the form of drawings and dance movements.

2.3 Brian Eno and Generative Music

A term coined by Brian Eno in 1996, generative music refers to music that is "ever-different & changing, created by a system" [16]. Since this thesis attempts to design a kind of generative music system, a discussion of important figures, theories, and techniques surrounding generative music is warranted.

2.3.1 Generative Work of Brian Eno and its Influences

Drawing influence from John Cage, Terry Riley, and Steve Reich, Brian Eno was a pioneering figure in generative music. Before formalizing the term "generative music," Eno was already exploring the idea of generative processes in works such as *Discreet Music* (1975), where a group of performers follow a set of instructions which undergo various

permutations with often surprising results [17]. In *Ambient 1: Music for Airports* (1978), Eno creates seven different tape loops playing a single vocal note truncated by arbitrary lengths of silence and starts them playing, creating a soundscape of overlapping notes and silence with a very low chance of repeating in the same way. Eno was heavily inspired by the work of minimalist composer Steve Reich, who was the first to utilize a tape loop system.

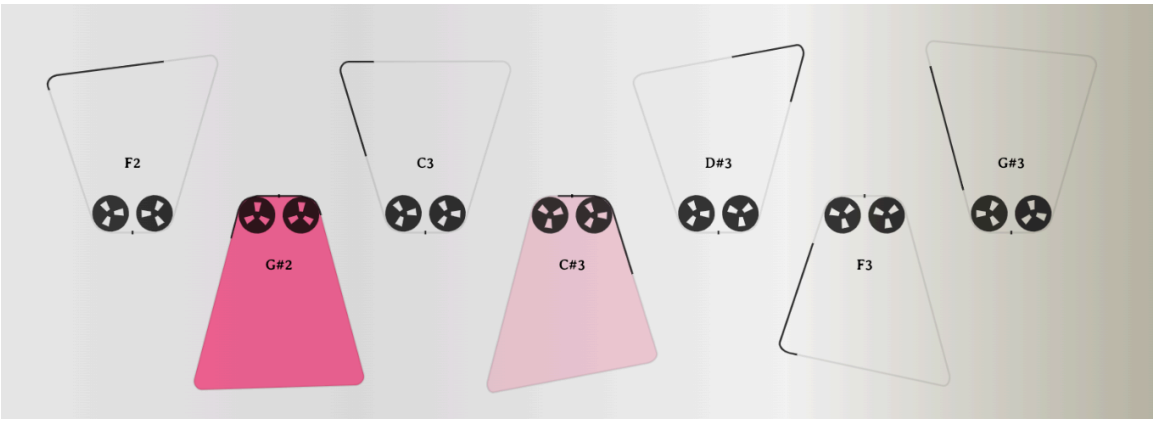

Figure 5: Visualization of the tape loop system in Music for Airports [15]

Discreet Music and *Music for Airports*, though produced through generative systems, are in the end still only a static sample of what the system can produce. The listener is only hearing a recording of the process, not witnessing or involved in the process itself. Only in 1994 did Eno finally release *Generative Music 1*, a software system which took guiding parameters as "seeds" and subsequently create probabilistic musical developments that would never repeat in the same way. At this time, Eno articulated his conception of generative music in relation to the types of music already in existence:

> "From now on there are three alternatives: live music, recorded music and generative music. Generative music enjoys some of the benefits of both its ancestors. Like live music, it is always different. Like recorded music, it is free of time-and-place limitations - you can hear it when you want and where you want" [18]

2.3.2 Interactive Generative Music Programs

Particularly relevant to this thesis, Brian Eno also helped create several interactive applications for computer generated music. [17] Collaborating with software designer Peter

Chilvers, Eno created Bloom for the iPhone in 2008. Bloom allows the user to tap the screen, causing circles to appear on the screen and produce different pitches. The user can either choose to listen or to actively create music.

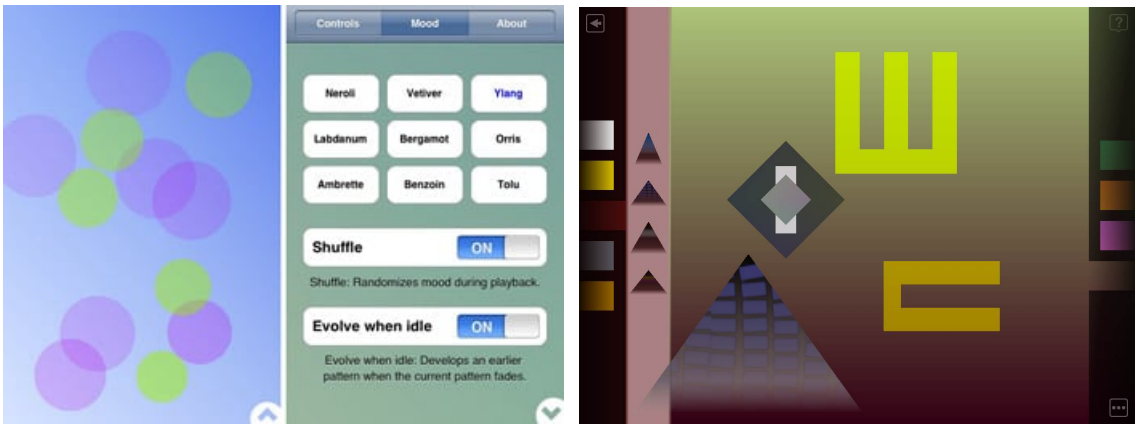

Figure 6: Screenshots from Bloom (2008) [19] and Scape (2012) [20]

After *Bloom*, Eno and Chilvers developed more apps, including *Trope*, *Scape*, and *Reflection* that extended the original idea of *Bloom*. These all operate on a similar philosophy of a user interacting with a generative music system but differed aesthetically as well as in the interaction and composition methods. *Trope* allows drawing of shapes that produce sound upon dragging one's finger across the screen. In *Scape*, the user selects and arrange shapes to influence the nature and evolution of the piece of music, in a way that rewards exploration and play. In all cases, the mobile device helped Eno achieve the goal of finally allowing his listeners to own the process rather than the results of the process.

3 System Design and Implementation

In this chapter we present the design and implementation of our generative music software system. First, we outline the system framework and tools used. Then we discuss details of the algorithms, motion-to-music mapping, and user interface.

3.1 Overview

3.1.1 System Framework

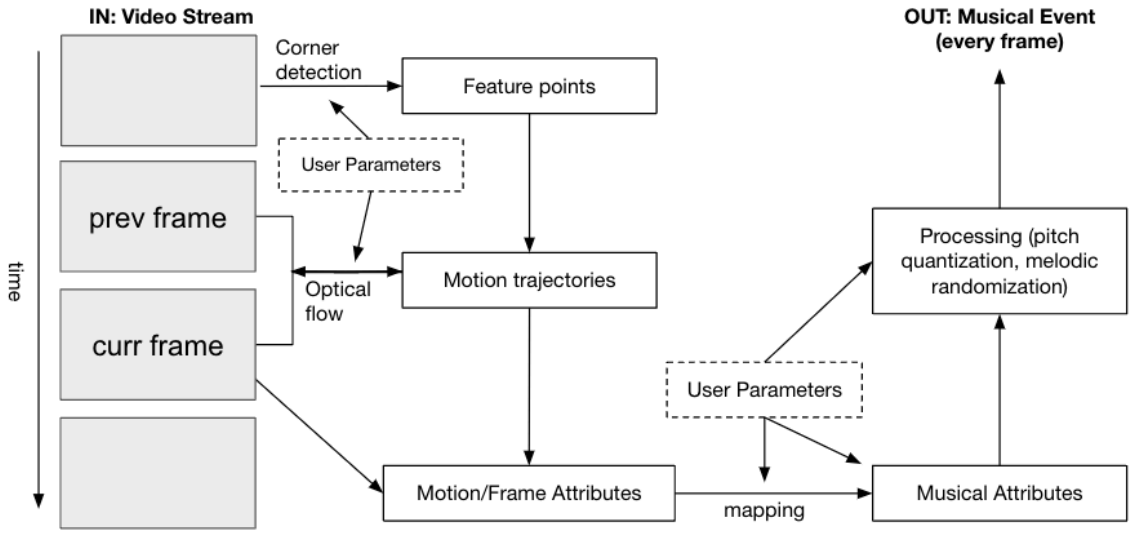

Figure 7: Video to Music Pipeline

The system takes video stream (either pre-recorded or from a connected camera) as input and processes each frame to produce a musical event. These musical events happen in rapid succession over time, producing a musical track.

1. Feature points are detected at given frame intervals (default is 3 frames but can be changed by the user). Once these features are detected, we mark them as the first point of a new "trajectory."

2. Optical flow is used to compute the pixel motion and update every trajectory in the scene with its next step.
3. From the trajectories, we extract local and global motion attributes
    a. Local motion (note-specific): the most prominent trajectories (exact number depends on user settings) are assigned as "players" to play either a melody or harmony note. The speed, position, and pixel contrast at these trajectories a particular trajectory controls how that note sounds.
    b. Global motion: we analyze all trajectories in the flow field as a whole and compute values such as average speed and directional variation. These values affect how every note sounds.
4. To ensure musicality of the output, the result of mapping motion to music undergoes further processing.
    a. Pitches are quantized to the nearest note in the chosen scale (melody) or chord (harmony).
    b. Random pitch offsets with different probability weightings are applied to reduce repetition and create melodic variation.
5. All active players produce their assigned sound.

The system operates within a user interface, where the user can change settings pertaining to feature detection, optical flow, and additional musical parameters not included in the motion-to-music mapping.

3.1.2 Technologies Used

The system is written in Python 3 within a virtual environment. Several libraries provide functionality for computer vision, image processing, and a graphical user interface (GUI). Music generation is achieved by sending data to SuperCollider, a sound synthesis engine.

1. OpenCV provides algorithms for Shi-Tomasi corner detection and sparse optical flow (Lucas-Kanade). We use these two in tandem to compute a flow field at each frame of the video input. Additionally, we utilize OpenCV drawing functions to visualize the flow field on top of the image frame
2. Tkinter provides a toolkit for creating a simple GUI.
3. Pillow configures each frame of the video stream to be displayed within the GUI.

4. FoxDot provides an API to define and schedule musical events before sending them to SuperCollider. This creates music in real time.

3.2 Motion Tracking

3.2.1 Optical Flow

Optical flow describes the apparent motion of objects in a scene. Apparent motion can be attributed to the objects themselves moving or the camera moving. We can quantify optical flow as a field of 2D vectors, where each vector represents the displacement of a point from one frame to the next. Algorithms exist to compute optical flow both for every pixel in the frame (dense optical flow) as well as for a sparse feature set [21].

Dense optical flow computes optical flow for every pixel in the frame. The output is visually descriptive and lends itself well to the idea of mapping motion to sound. The initial iteration of the program used dense optical flow as a preliminary proof of concept of sending motion data into a music generator and producing responsive results. For a simple, discrete mapping scheme, we divided the y-axis into three bands, each with a different pitch, and divided the x-axis into two halves, each with a different timbre. The average length of flow vector over the flow field was assigned to intensity. When running the interface with the webcam, the user was able produce rudimentary music by moving around the screen. Faster motions produced louder sounds, and "higher" movements produced a higher pitch. The user could also creatively choreograph motion between left and right hands to produce sound of each timbre independently or in unison.

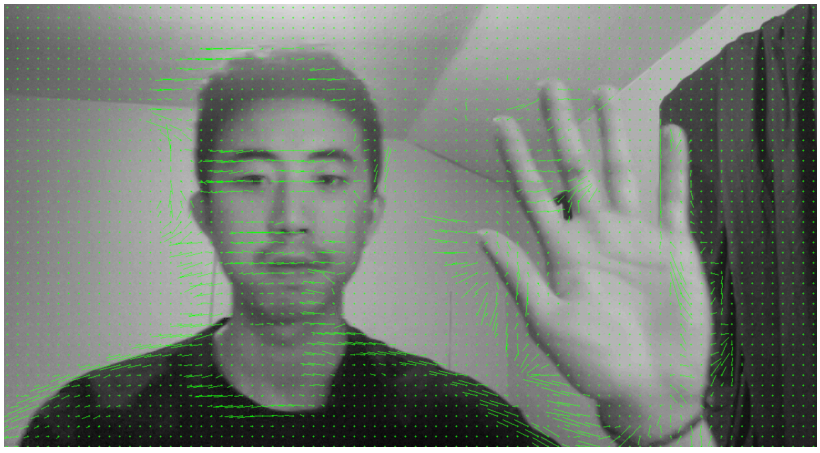

Figure 8: Visualizing Dense Optical Flow Field on Webcam Stream

Dense optical flow provides a rich amount of data about movement globally across the frame. At the same time, it is computationally intensive; frames were unable to be played back in real time on a MacBook Pro with a 3.2 GHz M1 chip. For the purposes of generating music, the dense flow field contains a high level of redundancy. It also proved to be difficult to interpret the data, that is, how to distill all the flow vectors into a small number of relevant metrics describing the current motion. Discretizing the flow field into bands is not a very flexible approach.

The problems of redundancy and interpretation with dense optical flow can be solved by looking at a sparse feature set instead, as originally explored by Pelletier [6].

3.2.2 Sparse Optical Flow with Corner Detection

The final system implementation utilizes sparse optical flow to extract motion information from videos. By reducing each frame from hundreds of thousands of pixels to a few hundred (or less) feature points, we can compute flow information much faster. In order to decide which points to track, we use the Shi-Tomasi method for corner detection provided by OpenCV. The algorithm examines spatial differences in pixel intensity to find the most visually distinct points in an image. At given frame intervals i (i set by user, default $i=3$), the system detects at most k Shi-Tomasi corner points (k set by user, default k=12).

Once a new set of corner points is detected, they are initialized as the first point in a *trajectory*, and this trajectory is added to a running list of *trajectories*. We will refer to the set of all latest points in every trajectory as the *frontier*. At each new frame, we pass in the frontier, previous frame, and current frame into a sparse optical flow algorithm (Lucas-Kanade method provided by OpenCV). The Lucas-Kanade method computes the motion flow of each point in the frontier, thereby updating every trajectory with a new point representing where it moved to.

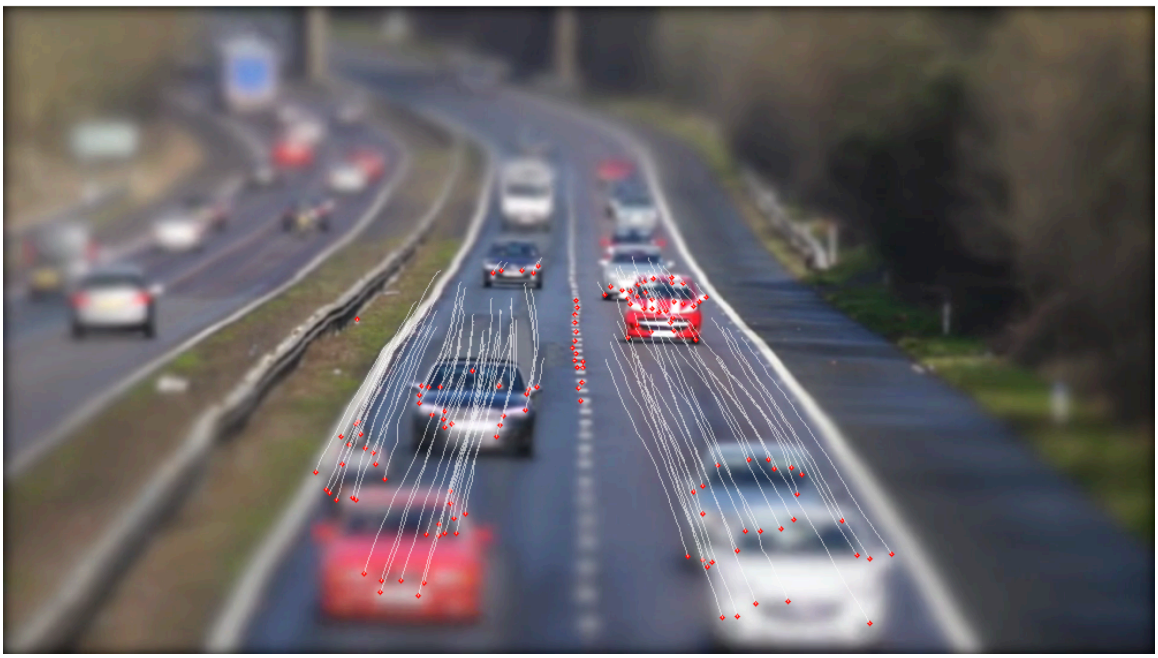

Figure 9: Optical Flow Field on a Sparse Feature Set

Using this technique, we were able to achieve real-time playback on a Macbook Pro with a 3.2 GHz M1 chip. In addition, the flow field is easier to interpret and use for generating music. We can take the most prominent motion vectors (greatest magnitude) and directly convert them into individual sonic entities. This precludes the need for potentially having to do neighborhood search and clustering operations in a dense flow field.

Having a set of salient flow vectors facilitates an "ensemble" setup, where each vector can act as a musical "player," producing a specific sound based on its own attributes. This lends itself to generating music with both melodic and harmonic elements, which will be discussed in the next section.

3.2 Algorithmic Music Generation with FoxDot

A project of Ryan Kirkbride, FoxDot was created primarily for the purpose of "live coding," an emerging style of performance where a programmer/artist/composer codes in front of an audience, "executing, editing, and re-executing blocks of code to generate music [22]". In practice, in can be used both as an interactive live coding environment or simply as Python library that provides an API for sending commands to Supercollider, a sound synthesis engine.

For our system to generate music, we use FoxDot as a Supercollider API. We specifically take advantage of FoxDot's ability to define player objects—objects that play a sound based on instructions given. Relevant instructions for our system include duration, pitch, stereo pan, synthesizer style, timbre, (for individual notes), as well as scale, key, and tempo (globally for all player objects).

```
p1 >> pluck((0, 2, 4, 6), dur=1/2, amp=0.4, pan=0.6, sus=2, lpf=900)
Clock.bpm = 105
Scale.default.set("minor")
Root.default.set("E")
```

Figure 10: Example FoxDot code generating a repeating E minor seventh chord

Having a set of player objects streamlines the mapping process: individual motion vectors can be mapped to individual player objects. As the video stream progresses in time, the player objects evolve with it, generating a stream of music. How we map the video attributes into musical attributes is the subject of the following sections.

3.3 Motion-to-music Mapping

As a generative system is governed by a set of rules, we developed a set of rules in the form of a mapping from video motion to music. As mentioned previously, some of these mappings are local (trajectory-specific) and some are global (based on entire flow field).

Video Data (Local)	**Musical Parameter**
A feature point	A "player" object which plays a note.
Speed of flow at a feature point (length of flow vector) / (number of steps)	• Amplitude of the player • Dampening (low pass filter)
Y-position of the feature point	Pitch of the player
X-position of the feature point	Panning (left/right) of the note
Pixel contrast of feature point	Dampening

Video Data (Global)	**Musical Parameter**
Average speed of flow field	Tempo (beats per minute)

Circular standard deviation (modified) of flow field directions	Degree of variation in note length
Average hue (from HSV) of pixels at flow frontier	Chord quantization

Table 1: Summary of Mapping

Each individual mapping is described in detail below.

3.3.1 Players (Notes)

The system produces musical elements by assigning motion vectors to FoxDot player objects. We set a maximum of *m* melody players and *h* harmony players; both *m* and *h* can be customized by the user. Adding player objects beyond this limit tends to produce sonically congested results and can quickly overwhelm CPU memory via SuperCollider.

We assign the longest *m* trajectories to melody players and the next longest *h* trajectories to harmony players. Although all player objects undergo the same mapping scheme, the way we compute pitch, intensity, and duration is different for the melody and harmony players.

3.3.2 Pitch

Pitch is an essential component of music, describing the auditory sensation caused by the frequency of a sound's vibrations [23]. Pitches heard in succession and simultaneously produce what we perceive as melody and harmony. Our system maps the y-position of a player (relative to frame height) to pitch. The decision of using the y-axis stems from the intuitive associations of "low" and "high" across movement and music. Physical locations such as the ground and the sky often set lower and upper bounds of observable motion. In most forms of music notation, where higher notes are printed higher up. Higher frequencies are associated with higher pitches.

An important element of producing coherent-sounding music is that pitches should be selected to fit a scale. To achieve this, we must *quantize* the raw computed pitches. For melody players, each raw pitch is rounded to the nearest note in the scale (for C Major, that would be the C, D, E, F, G, A, B). For harmony players, each raw pitch is rounded to the

nearest note in the current *chord*, for example the tonic (I) or the dominant seventh (V7) chord. The default chord is the tonic (for C Major, that is C, E, G), but mapping to different chords will be discussed in Section 3.3.8.

Finally, melody and harmony players are mapped to a different range of chords. In the default key setting of C Major, melody players are linearly mapped to a range from one octave below middle C to three octaves above it. Harmony players are mapped to the range of two octaves below to one octave above. This lower range helps the harmonic elements server as more of a backdrop to the music overall.

3.3.3 Intensity

We map speed to the intensity (loudness) of a player. Speed is defined as the total distance along the trajectory divided by the number of points in the trajectory (equal to the number of frames that have passed). We map this speed value to an intensity value on a logarithmic scale. We chose a log scale after performing tests on a wide range of stock video footage and finding that variations between slow and moderately fast motion are more commonly seen (and more perceptually salient) than variations between moderately fast and very fast motion. In turn, we want to emphasize these dynamic contrasts in the generated music.

3.3.4 Space

Following the perceptually motivated framework of Pelletier [6], we translate the inherent spatial nature of images into the sonic dimension. Our system maps the x-position of a player to the stereo pan position of the player, in the range of [-1, 1]. A player in the leftmost pixel column of the frame is panned completely to the left, while a player in the centermost pixel column produces sound that is equally distributed between both sides. Objects moving on either side of the screen or horizontally across the screen thereby generate clear sonic positionings and trajectories.

3.3.5 Tempo

We map the average speed of the flow field to tempo. At each frame, we compute the average speed over all trajectories, map it to a tempo marking (in BPM, beats per minute) using a log scale, and set the global BPM to that value using FoxDot. We chose a log scale

for a similar reason as for the intensity mapping: it emphasizes slow to fast tempo changes and prevents very fast motion from producing excessively fast music (to the point of sounding unmusical).

Because most pieces of music follow consistent periods of a set tempo, the decision to map average speed to tempo was not initially perceived as sensible. Doing so can produce instantaneous (frame-wise) tempo change, and therefore jittery music. However, we can reduce this effect by rounding the computed value to the nearest 10 BPM. With the rounding, successive frames that are close in average flow field velocity will not change the tempo. The results are perceptually convincing—videos with repetitive motion tend to produce music that moves at a constant pace, whereas videos with high variation in velocity can still generate music that emulates the jittery quality.

3.3.6 Note Length

We map the directional variation of the flow field to the degree of variation in note lengths (duration of notes relative to the tempo). Directional variation is computed using circular standard deviation (so that a direction of 5 degrees and 355 degrees are measured as 10 degrees apart rather than 350). The baseline note length has a value of ¼ (quarter notes). We define a series of ascending thresholds at which different notes lengths are added to the set of possible note lengths. The final chosen note length for the current player object is randomly sampled from this set.

Certain videos have motion that we perceive to be moving in a single direction, yet due to perspective, the pixel trajectories go in a variety of directions (for example, videos where the camera dollies forward). To counteract this effect, we split the video into four quadrants, and compute the directional variation for each quadrant. (If a quadrant has less than 2 trajectories, we skip it since standard deviation cannot be computed). We then weight the value for each quadrant by the fraction of the total trajectories present in that quadrant before adding them together.

$$directional\ var = \sum_{i=1}^{4} \frac{num\_trajectories(quad_i)}{total\_trajectories} circstdev(quad_i)$$

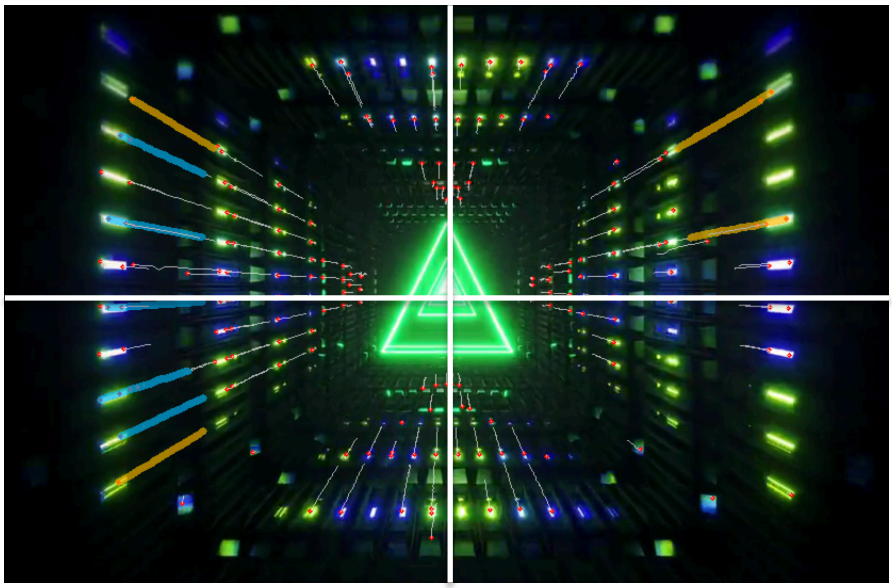

Figure 11: Computing Directional Variation Per Quadrant

The result of this mapping is that videos where objects are moving in the same direction overall produce repetitive rhythmic patterns, while videos with plenty of directional variation produce varied rhythmic patterns.

3.3.7 Dampening

We map both pixel contrast and speed to the level of dampening, or softness in timbre, of a trajectory's player. Dampening is controlled using the low-pass filter (LPF) attribute in FoxDot. The LPF passes signals below a certain threshold and attenuates signals above the threshold. A lower threshold permits lower frequencies to pass through, therefore creating a more dampened timbre.

The trajectory's speed is first linearly mapped to an initial LPF threshold, with LPF increasing with speed. Thus, slower trajectories have more dampening than fast trajectories. (Note that speed is already mapped to intensity; we found that mapping it to dampening makes perceptual sense as well).

We then compute an additional attenuation factor, based on pixel contrast, to add more dampening. To obtain a rough measure of pixel contrast, we define a 10x10 pixel patch around the latest point in the trajectory and compute *luminance* (perceived brightness) values for each pixel. Pixel patches that would go beyond frame boundaries are

simply cropped. Given the BGR value of a pixel, its luminance is estimated as 0.0722*B + 0.7152*G + 0.2126*R. We calculate the standard deviation of luminance over this pixel patch, and map that value to an attenuation factor using a square root scale. This attenuation factor is multiplied with the initial LPF threshold for the player. Lower standard deviation, i.e. lower pixel contrast, therefore leads to additional dampening.

Figure 12: Trajectory with high contrast (left) and low contrast (right)

Assuming speed is similar, the brighter fireworks map to a high LPF threshold and generate a brighter sound. Correspondingly, the faded, blurry fireworks map to a much lower LPF threshold and sound much more muted.

3.3.8 Chords

To incorporate evolving patterns in the harmony via chord progressions, we map *hue* to *chord*. Hue is extracted from the HSV (hue, saturation, value) of a pixel. We take inspiration from Scriabin's color to tone mapping but instead use the HSV color wheel, split into 4 sections rather than 12. Each section corresponds to a different chord index.

To define the current chord index, we compute the HSV at each pixel in the frontier and add each hue value to the nearest bin in a four-binned histogram, unless the saturation or value are below a threshold (too light or too dark). We evaluate and reset the histogram every 10 frames, assigning the chord index to the fullest bin. The initial (default) index is 1.

Figure 13: Hue to Chord Index Mapping

The set of chords is pre-defined and can be changed by the user (e.g., I IV V I). The chosen chord is simply the i^{th} index of the set of chords. Harmony pitches will be quantized to fit the nearest note in the chosen chord. Thus, it is not the colors themselves that hold any particular meaning, but rather the color changes in relation to the set of pre-defined chords.

3.4 Additional Musical Processing

3.4.1 Melodic Offsets

Inspired by musical and perceptual grouping principles based on proximity [24], we probabilistically add small random offsets to the current pitch if it is the same as the previous pitch. In conjunction with player trajectories, these random offset help achieve an effect similar to the stochastic melody generation technique of a *random walk* [24]. The player trajectories create general melodic trajectories, while the offsets add small fluctuations to the melodic trajectories. Doing this enhances melodic interest by reducing the presence of constantly repeated pitches (from horizontal motion) and monotonously ascending or descending scales (from vertical/diagonal motion).

3.4.2 Pitch Refinement

To reduce the amount of dissonance between different layers of melody (mainly due to adjacent pitches being played simultaneously, we probabilistically "refine" the melody players at each frame so that their pitches are *most likely* either all even or all odd. We also allow a smaller likelihood for mixed parity of pitches, as a small amount of adjacent pitch intervals can add melodic interest.

3.4.3 Synth Volume Calibration

We found that different FoxDot synthesizers produce widely varying intensities given the same amplitude setting, so we normalize volumes by individually multiplying intensity by a factor based on synthesizer type.

3.5 User Interface

An important part of the system's design is a GUI enabling the user to take part in the generative process as an artist as well as visualize the motion being transformed into music.

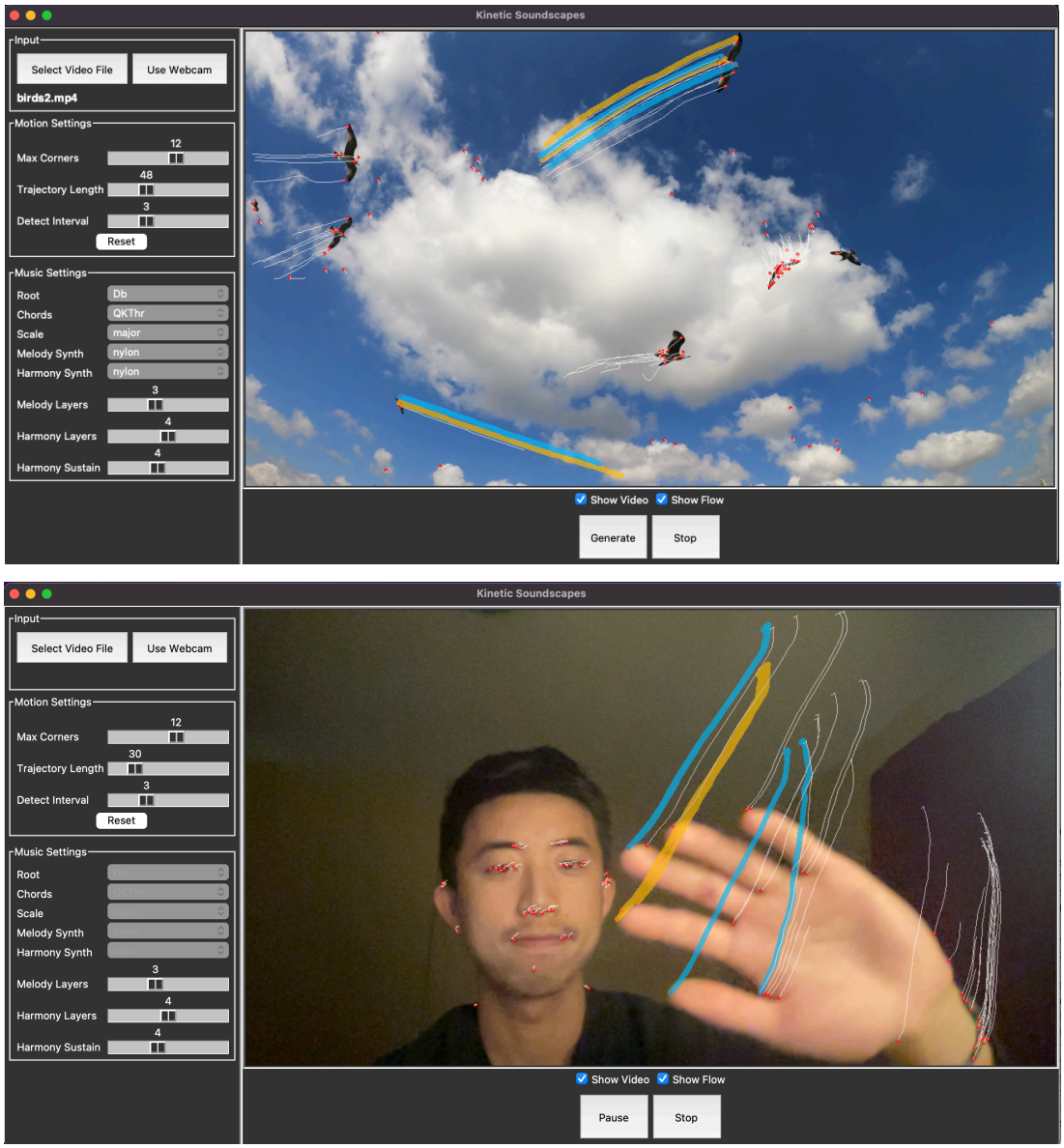

Figure 14: The GUI

3.5.1 Playback Visualization

To begin using the system, the user is able to select an input video from the local machine or opt to use their webcam. The selected video or webcam stream will be loaded into the player window. The user can then press "Generate" to begin playback and music generation, and pause or stop as desired.

The system provides visualization of the motion flow field by marking tracked feature points with a red dot and tracing each trajectory (up to "trajectory length" frames) with white lines. Additionally, trajectories that are being converted to music are highlighted in yellow (melody) and blue (harmony). The user is able to toggle the visibility of the video and flow field. Depending on the situation, one may want to hide the flow field, hide the video, or overlay the two.

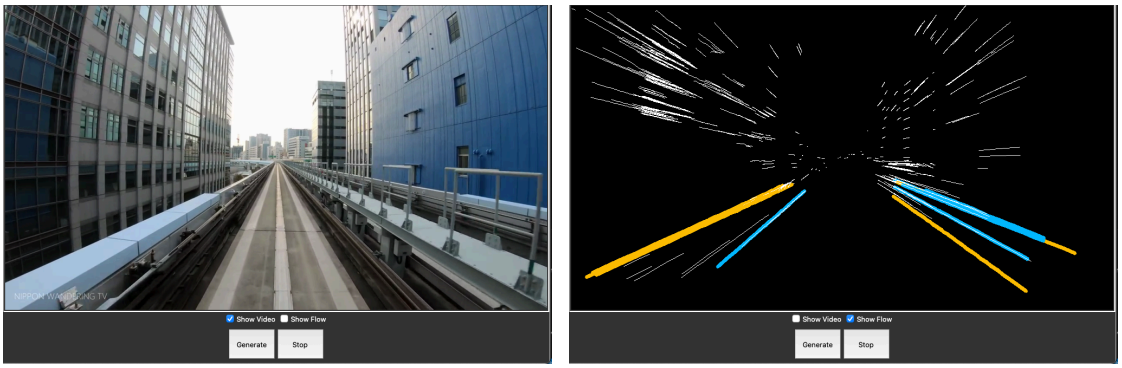

Figure 15: Toggle Options for Showing Only Video or Motion Flow

3.5.2 Motion Settings

The user can alter parameters for motion detection depending on the nature of the video. "Max Corners" specifies the maximum number of new feature points that the corner detection algorithm will generate each time it runs. For visually dense scenes with "busy" motion, it may be useful to turn this down for performance, since many of the feature points will be redundant.

"Trajectory Length" specifies the number of frames a feature point will be tracked for. Longer trajectory lengths will tend to produce more continuous lines of melody, since a player object will persist for longer. Trajectory length will also slightly influence the amount of directional variation.

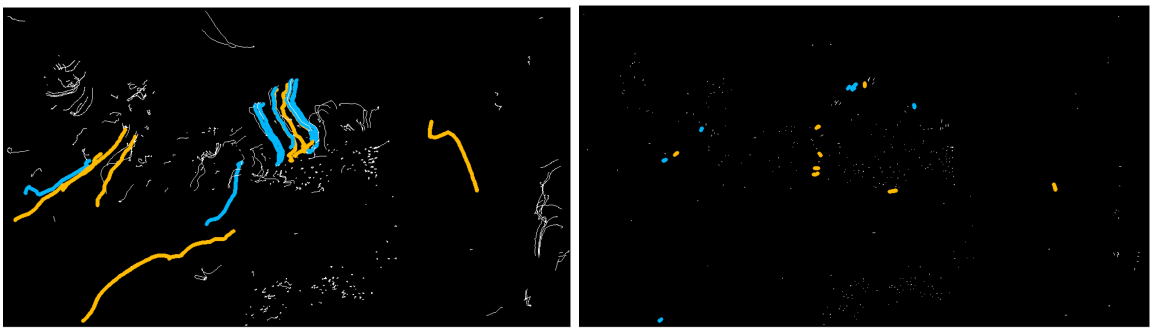

Figure 16: Same scene, with trajectory length of 160 on the left vs 3 on the right

Finally, "Detect Interval" specifies the number of frames that passes before corner detection is performed again. For scenes with objects moving in and out of frame rapidly, it is useful to have a lower detect interval to keep up with the visible feature points.

3.5.3 Music Settings

The user can alter parameters for music generation, elevating the program as a compositional tool. They can change the key, scale, chord progression, synthesizer type, maximum number of melody or harmony players, and level of sustain for harmony.

3.5.4 Feature Spec

Feature	**UI Element**	**Functionality**
Select video	button	Opens file dialog for user to select input video
Use webcam	button	Starts the webcam to use webcam stream as input
Set max corners	slider	Sets the max number of corners for corner detection
Set trajectory length	slider	Sets max number of frames that motion of a feature point is tracked for
Set detect interval	slider	Sets the frame interval at which new feature points are detected
Select root	dropdown	Sets the root of the scale (C, C#, D,…,B#)
Select chords	dropdown	Set the possible chords to be used (from a set of predefined chord

		progressions)
Select scale	dropdown	Sets the scale (major, minor, pentatonic, etc.)
Select melody synth	dropdown	Sets the synth for playing melodies
Select harmony synth	dropdown	Sets the synth for playing harmonies
Set number of melody layers	slider	Sets the maximum number of synths playing melody at the same time
Set number of harmony layers	slider	Sets the maximum number of synths playing harmony at the same time
Set harmony sustain	slider	Sets the sustain level of harmony players
Generate/Pause Music	Toggle button	Start/Pause video playback and music generation
Stop	Button	Stops video playback (reset to first frame) and music generation
Show/Hide Video	Toggle	Shows/hides the video stream
Show/Hide Flow Field	Toggle	Shows/hides the flow field

Table 2: GUI Feature Spec

4 Testing Methodology

To test the system, we conducted an empirical study of the system's output from both cognitive and artistic points of view. 20 participants, aged 19 to 30, were recruited to each take part in a two-part experiment. Participants were asked to rate their musical ability and video editing/videography background from 1 to 5, though no significant differences were found between music/video background levels at the conclusion of the study.

4.1 Experiment One: Judgement of Video-Music Input/Output Correspondence

In the first experiment, we tested the efficacy of our video-to-music mapping. Specifically, we wanted to investigate whether subjects (with no knowledge of the system) would be able to correctly judge if the generated music matches the source video, and to what extent.

4.1.1 Energy Scores

To prepare for data collection, we gathered 20 video clips with a wide range of "energy" (slow-moving jellyfish, rapidly blowing wind, etc.) and fed them into our system to generate music. All video clips were edited to be 18 seconds in length. Music and motion settings were kept as default to ensure that the only factor influencing how the music sounded was the source videos themselves.

In order to obtain more descriptive results with regards to motion and music connections, we devised and computed several metrics to describe the motion of a video: velocity, velocity variation, directional variation, and hue variation. These are the average velocity, standard deviation of velocity, standard deviation of direction, and circular standard deviation of hue, respectively, of the flow field over all frames of the video. Velocity is measured in pixels moved per frame, direction ranges from 0 to 2π, and hue ranges from 0 to 4.

Using these 4 values we created a composite metric called "energy," defined as

$$E = v + v\_var \bullet v + d\_var \bullet v + h\_var ,$$

where *v* is velocity, *v_var* is velocity variation, *d_var* is directional variation, and *h_var* is hue variation. We give precedence to velocity since it appears to carry the most influence in our subjective interpretation of energy. Velocity variation and directional variation are also perceptually important, so we decided to weight their contributions with velocity. Color changes in the video have a much more subtle perceptual influence, so we add that as a constant.

video	**velocity**	**velocity var**	**direction var**	**hue var**	**energy**
fireworks	1.96	0.9	0.85	0.46	5.85
blooming	0.31	0.26	0.72	0.12	0.73

Table 3: Example motion metrics

We then converted each of the 20 energy values into an energy score:

$$E\_score = \frac{E}{E_{max}} \bullet 100$$

The video with the highest energy is given a score of 100, and all other videos are given a score relative to that. We categorized the videos into two classes: high energy (energy score >= 40) and low energy (energy score < 40).

Video	**Energy Score**
pendulum	100
dolphins	89
coaster	76
DVD	64
fish	60
rays	53
dancer	50
wind	50
escalator	49
fireworks	46

Video	**Energy Score**
birds	35
cars	34
train	27
waves	18
candle	15
puppy	14
jellyfish	13
aquarium	7
blossoms	6
blooming	6

Table 4: Energy Scores (high energy on left, low energy on right)

4.1.2 Trial Generation

Before meeting with each participant, we systematically generated a set of 10 trials, each containing a video clip and a music clip. First, we randomly sampled 5 high energy videos and 5 low energy videos from the set of all videos and shuffled them in a random order. Then we randomly assign 5 of the 10 videos to be "match" trials and the other 5 to be "mismatch" trials. For the "match" trials, we paired each video with the music clip it generated. For the mismatch trials, we paired each video with a randomly selected music clip, making sure no music clip appeared twice in the entire set of 10 trials.

Video	**Music**	**Energy Difference**
fireworks	birds	11
waves	waves	
birds	jellyfish	22
candle	blossoms	9
blossoms	fireworks	40
blooming	blooming	
wind	wind	
coaster	coaster	
escalator	DVD	15
aquarium	aquarium	

Table 5: Example set of trials (highlighted are mismatch)

For each mismatched trial, we took note of the difference in energy score between the video and music. We consider the energy score of a music clip to be that of its source video.

4.1.3 Participant Judgement Sessions

Sessions were conducted mainly over Zoom using screen and audio sharing, but several were conducted in person. In both cases, participants were using headphones. At the beginning of each session, participants were told that they would watch short video clips, listen to short music/audio clips, and make judgements about whether or not they match. (We use "music/audio" here due to inconsistent wording between earlier and later sessions.)

Because out system *first* takes in video and *then* generates music, we adhered to this directionality when testing correlational judgements. Using a psychological priming technique, we *first* showed the video, *then* played the audio, before asking for the participant's judgement.

With a set of 10 trials as described above, the participant first watched the video clip, then immediately listened to the audio clip. Then they were asked to give a yes or no answer to the question, "does the music/audio match the video?" If the answer was yes, they were asked to rate how well it matched on a scale of 1-5, 5 being a perfect match and 1 being only a slight match.

Because our system aims to generate music with a level of perceptual correspondence to the source video, the goal was for participants to correctly answer "yes" or "no" more often than not. The null hypothesis was that the accuracy distribution of participants' judgements will be statistically no different from that of randomly guessing "yes" or "no". The alternative hypothesis was that the accuracy distribution will be greater than that of randomly guessing. We set $\propto = 0.05$.

At the end of the 10 trials, participants were asked a follow-up question: "how did you judge whether the music/audio matched the video?"

4.2 Experiment Two: Subjective Ratings of Audiovisual Pieces

In the second experiment, we evaluated the artistic potential of the system as a tool for generative composition, specifically in the case of creating soundtracks and soundscapes that accompany video.

4.2.1 Creating Audiovisual Pieces

We created 10 different musical pieces meant to be played back along with their source videos. 5 high energy and 5 low energy video clips were used. We will refer to these creations as "audiovisual pieces."

Rather than restricting the settings to default as in the first experiment, we granted ourselves complete freedom to change the motion and music parameters. The goal was to create music with aesthetic appeal while complementing the source video. Depending on

the video, we changed the key, scale, synthesizers, melody/harmony density, as well as motion settings to widely different configurations. Each audiovisual piece took no more than several minutes to craft using our GUI.

4.2.2 Trial Generation

Before each session, we generated a set of trials by randomizing the order of the 10 audio-visual pieces to show to the participant.

4.2.3 Subjective Ratings

Participants were told they would watch 10 more video clips, this time accompanied by music. Subjects were instructed to rate each audiovisual piece from 0-10 based on how well the music "fits" the video.

At the end of the 10 trials, participants were asked: "what relationship, if any, do you think there is between the videos and the music in this part of the experiment?"

5 Results

5.1 Judgement of Video-Music Correspondence

5.1.1 Judgement Accuracy

To analyze the accuracy of participant judgements, we create a confusion matrix categorizing the true positives, false positives, true negatives, and false negatives.

		Actual Match	
		yes	no
Perceived Match	yes	76 TP	57 FP
	no	24 FN	43 TN

Table 6: Confusion Matrix for Participant Judgements

From that, we compute the rate of each type of correct and incorrect judgement, as well as the overall accuracy (rate of correct judgements).

True positive rate	0.76
False negative rate	0.24
True negative rate	0.43
False positive rate	0.57
Accuracy	0.6

Table 7: Judgement Statistics

The true positive rate of 0.76 is high, indicating that when the music actually matched video, participants indeed said it matched most of the time. At the same time, the

false positive rate of 0.57 is quite high, indicating that even when the music did not actually match the video, the participants said it did half of the time.

The overall accuracy of 0.6 suggests that participants performed better than randomly guessing.

0.8	0.7	0.5	0.6	0.5	0.7	0.4	0.4	0.8	0.4
0.9	0.4	0.4	0.6	0.6	0.6	0.7	0.6	0.7	0.6

Table 8: Accuracy of each participant's judgements

To test our null hypothesis and assess the statistical significance of the accuracy distribution, we performed a one-sample t-test (one-tailed, $n = 20$, $s = 0.15$, $\mu_0 = 0.5$). Here n is the sample size, s is the sample standard deviation, and μ_0 is the theoretical mean accuracy of randomly guessing. The test yields a t-statistic of 2.83 and p-value of 0.0054. Thus, we reject our null hypothesis and find evidence to support our alternative hypothesis. Participants performing better than randomly guessing is statistically significant.

5.1.2 Ratings for "yes" Judgements

Recall that when a participant made a "yes" judgement, we asked them to rate how well, from 1-5, they thought the music/audio matched the video. We found a very slightly higher mean rating for true positives than for false positives.

judgement	count	mean rating	sample standard deviation
TP	76	3.67	1.06
FP	57	3.49	1.18

Table 9: Participant Ratings for "yes" Judgements

A simple t-test reveals that the difference in ratings between true positives and false positives is not statistically significant.

5.1.3 Mismatch Trial Judgements and Energy Difference

Due to the high rate of false positives, it is worth taking a closer look at participant judgements where the music did not actually match the video. To do this, we analyzed false positives and true negatives in relation to energy differences between video/audio pairings.

judgement	count	mean energy difference	portion of pairings at different energy levels (“low” vs “high”)
FP	57	28.44	0.19
TN	43	42.3	0.56

Table 10: Overall Energy Difference for Mismatch trials

Overall, there was a noticeably higher mean energy score difference for true negative judgements than for false positive judgements. In addition, over half of true negative judgements occurred when a “high-energy” video was paired with a “low-energy” music (or vice versa), while only about a fifth of false positive judgements occurred under these circumstances. This strongly suggests that video-music pairings that are closer in energy are more likely to warrant false positives. Likewise, video-music pairings that are farther apart in energy are more conducive to true negatives.

We visualized all mismatch trials with respect to their energy score differences in the scatter plot below. The x and y axes represent the energy score for the video and music clip, respectively, in a given mismatch trial. The line $y = x$ indicates all hypothetical pairings with equal energy between video and music.

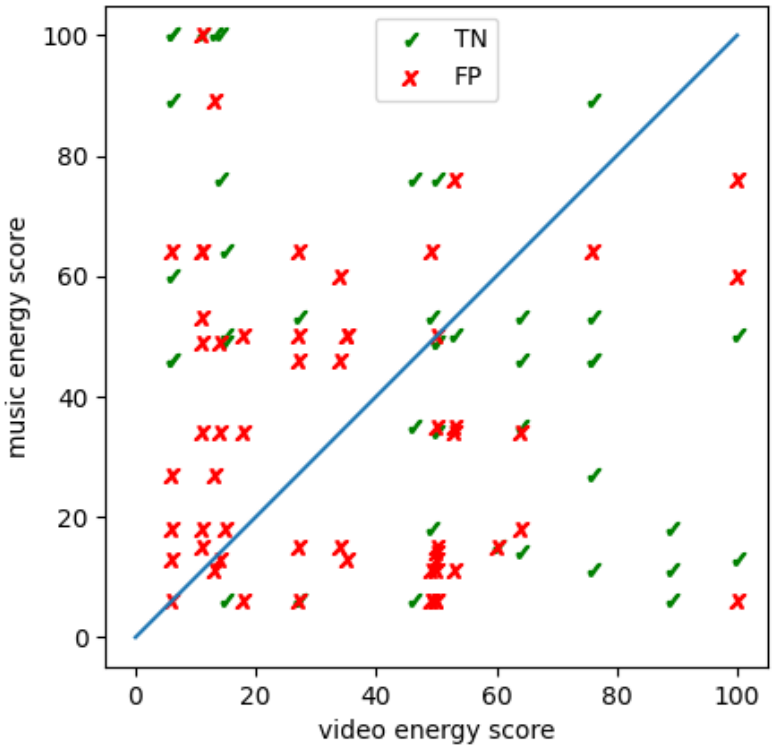

Figure 17: Scatter Plot of Mismatch Trials with respect to Energy Score

From the scatter plot we see that true negatives are rather spread out and false positives are comparatively clustered more closely around $y = \mathrm{x}$. This suggests that false positives are more common at lower energy differences and less common at higher energy differences. Intuitively, lower energy differences between video and music may often be mistakenly perceived as a match, while high energy differences may register as more obviously mismatched.

5.1.4 Participants' Rationale

Participants gave a variety of answers to the question, "how did you judge whether the music/audio matched the video?" Emotion and energy were two main factors. Out of 20 participants, 12 mentioned a combination of "emotion," "mood," "ambience," or "vibe." These participants often used specific words like "peaceful," "solemn," "excitement," "intense," or "chaotic" to describe both video and audio. 10 participants pointed to speed, tempo, pacing, and rhythmic changes/correspondences between video and music, using words like "slow" "fast," "moderate," "repetitive." 4 participants directly mentioned motion (as opposed to indirectly mentioning it through speed of things moving), either in terms of the amount of motion or how the motion of the video matched the qualities of the music. One participant made a correct "yes" judgement based on how the stereo panning matched the motion of the video (for the DVD clip). A few participants noted that timbre was a difficult factor to consider, since all the music samples used the same default synthesizer. One participant was caught off guard by the default synthesizer's timbre, saying that it evoked 8-bit style video game music in their mind which to them did not match the videos.

Overall, participants seemed to base judgements on their expectations for the music after watching the video. Their "yes" or "no" answer often reflected whether or not the music aligned with those expectations. One participant made a false negative judgement of the "dolphins" clip because it felt too frenzied and dissonant, contrary to their expectation for something more harmonious. In addition, short-term recall may have been an influence, as some participants mentioned trying to replay the video in their head while listening to the music.

5.2 Subjective Ratings of Audiovisual Pieces

With regards to how well the music "fit" the video for each audiovisual piece, participants gave a mean score of 7.8 and median score of 8. Several audiovisual pieces consistently score 9 or above. The lowest median score for a piece was 6.5 ("rays"). Lower energy pieces performed slightly better than high energy pieces, except "pendulum."

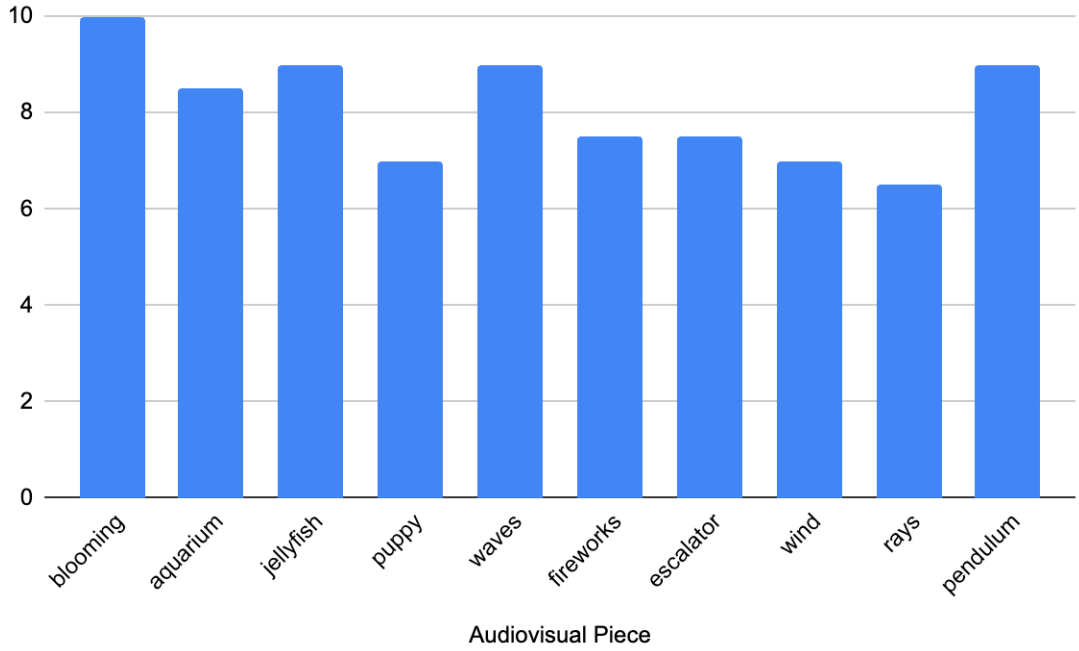

Figure 18: Median Ratings of Audiovisual Pieces (ordered from low to high energy)

With regards to the question, "what relationship, if any, do you think there is between the videos and the music in this part of the experiment?", most participants articulated, in some form, that the "emotion," "tone," or "mood" of the music matched what they saw in the video. In addition, about half of participants directly mentioned similarities in "motion" or "movement." Several pointed out that the timing of musical changes was synced with changes in the video; this was especially apparent in "fireworks." Participants who referred back to the first experiment all stated that the connections were much stronger in the second experiment.

A few participants said that they could picture the audiovisual piece as a scene from a movie. One participant mentioned an instance where the music did not match their expectations based on the video ("escalator"), but they still found it to be a good fit in an artistic sense.

6 Discussion

In this chapter we interpret our findings, contextualizing the quantitative data with participants' qualitative responses. We also discuss limitations of our system and avenues for future work.

At this point it is worth addressing an inconsistent word choice in the first experiment. (We used "audio" in earlier sessions and "music" in later sessions). It is possible that the word "music" may have led to a greater expectation for "artistic" correspondence with the video clip, whereas "audio" may have led an interpretation closer to "does the audio sound like how the video looks?"

6.1 Success in Generating Music with Similarity to Source Video Motion

Recall the first experiment: judgement of video-music correspondence. To evaluate our generative music system based on participants' judgements, it is useful to articulate what each of the four types of judgements might imply:

- *True positives* may indicate cases where the system generated music that successfully evoked some degree of similarity to the source video. Conversely, *false negatives* may indicate cases where the system generated music that failed to evoke sufficient similarity to the source video.
- *False positives* may indicate cases where the generated music evokes similarities to an arbitrary video. The qualities of the music may be broad enough to subjectively match many different kinds of videos. Conversely, *true negatives* may indicate cases where the generated music does not resemble an arbitrary video.

Since participant judgements revealed a high rate of true positives, we infer that, under default settings without artistic control, the system is usually able to generate music with a level of perceptual similarity to the source video.

At the same time, as evidenced from the considerable rate of false positives, the music is often perceived as similar to other videos as well. Our analysis of mismatch trials with respect to energy difference revealed that false positives are associated with lower energy differences than true negatives. In other words, participants were more likely to correctly identify a video/audio mismatch when the energy difference was higher versus when the energy difference was lower. Since our metric of energy is based on motion qualities, this suggests that motion is being mapped to music in a relatively consistent manner.

Considering that 1) overall judgements had a statistically significant mean accuracy of 0.6, 2) participants judged mismatch trials more accurately when energy difference was higher, and 3) these results were achieved by the system itself without any artist's involvement, it appears that the system *is* able to take a video and generate music with a level of affective/perceptual correspondence to its motion. This is as we originally intended.

6.2 Limitations of Motion-to-music Mapping

The rate of false positives (0.57) and false negatives (0.24) may indicate that there may be certain limitations our motion-to-music mapping.

Based on comments from participants, false negatives occurred when the music did not align with participants' expectations after watching the video This often occurred because the music did not match certain symbolic or semantic associations evoked by the video (e.g., dolphins with serenity and harmony). In an initial test run of the experiment, participants unanimously gave a false negative judgement to a video of soldiers marching. Because the level of motion in the video was quite small (it was filmed at a distance), the system generated music that was slow and soft, albeit rhythmic. Participants were probably expecting faster and/or stronger music. In another case with a video of a candle flame flickering and then blowing out, one participant expected the music to also die down at the end. However, the smoke blowing from the extinguished candle has plenty motion, which our mapping converts into a loud and fast sequence of notes.

Likewise, false positives may also have occurred in cases where an arbitrary music clip matched the participants' notions of a video clip. For example, the videos clips "train" and "wind" evoked feelings of "peacefulness" that aligned with a number of different

music clips with much lower energy scores.

Indeed, our system strictly generates music based on the pixel data of moving images. Symbolic and semantic associations between the video and the viewer are therefore overlooked. With regard to the goal of generating music with affective/perceptual correspondence to the source video, we may infer that directly mapping motion attributes to music does not always produce the intended output from an emotion perspective.

Thought we might be able to improve these results by making changes to the mapping, it would not be appropriate to fine tune the mapping so much that it becomes tailored to specific types of videos. There should be a balance between emotional correspondence and adaptability to various source videos. There may even be the possibility of integrating symbolic meaning into our system by using techniques such as semantic segmentation, but doing so would likely introduce severe performance limitations and is outside the scope of this thesis.

6.3 Potential as an Artistic Tool

In the case of artistic use, the substantial rate of false positives from our first experiment may actually be a feature rather than a bug. The fact that our system is able to generate music that matches a video more often than not lays an aesthetic foundation for fine tuning the settings as an artist. It is important to note that the choice of source video does not need to match the desired emotion in the music. Our tests of video-audio correspondence and correctness of judgements do not imply that the video-to-music mapping should be taken literally. A video of a march will not necessarily translate to march-like music. As discussed, there may be symbolic and semantic meanings inherent in different types of videos that are independent of their motion qualities.

As shown in the results of our second experiment, audiovisual pieces created using our system (in the span of minutes) were perceived to demonstrate a very good fit overall between video and music. Having the freedom to change attributes such as the key, scale, timbre, and melodic/harmony density was very conducive to creating music that strongly resonated with its source video.

Based on the results found of the two different experiments, we can potentially infer

that motion and music have more of a synergistic relationship than a directly mappable relationship. Played separately, their meanings can often become vaguely interchangeable or even distorted to fit one another (false positives), or sometimes simply fail to resonate with each other (false negatives). When played in tandem there is more potential for strong emotional resonance with one another.

This is arguably where the artist comes in and takes advantage of the synergistic relationship. It is the artist's task to choose a source video and set the audiovisual parameters to generate a desired sound. It is also up the artist to decide if they want to avoid videos with strong symbolic associations, or to use them anyway and generate unexpected results. We can imagine what it might be like to practice using the system much like one would practice playing a musical instrument—trying out different source videos and configurations of settings to understand which types of visuals translate into which types of sound. There may be potential for use of this type of system as a mode of live performance or installation, perhaps as an "audiovisual jockey" who curates and sequences video material (possibly abstract) to generate an ever-evolving stream of music. There may also be potential for use as a tool for filmmakers or video-creators with limited musical background to generate soundscapes or soundtracks (possibly temp tracks) based on emotions they want to evoke in certain scenes.

6.4 Future Work

6.4.1 More Advanced Tools and Software

The system currently generates music through SuperCollider via the FoxDot API in Python. This choice of tools was made based on convenience. More sophisticated programming environments and music synthesis engines can be explored, such as Max/Jitter, which is optimized for real-time audiovisual work [25]. This would require a rewrite a of the system but could open up more flexibility in our motion-to-music mapping and algorithmic music manipulation.

6.4.2 Improvements to Motion Tracking

The system currently employs corner detection without considering the "usefulness" of each corner to our generative system. This leads to cases where it misses out on the actual

motion present in the scene. In the frame below, the stationary buildings are tracked because they represent areas of high contrast, but the moving clouds are not tracked because they are visually "weaker" corners (less contrast).

Figure 19: Example where Inactive Feature Points are Tracked

To remedy this, future improvements might be made to the system framework so that corner points are given a "motion rating" before deciding whether or not to track them. Corner points below a certain motion rating threshold could then be discarded to make room for other corner points that may be less visually salient but are actually moving.

Another (considerably more advanced) improvement to motion tracking would be to counteract the effect of camera movement, so that the movement of objects in the video can be isolated from the movement of the camera.

6.4.3 Improvements to Trajectory-Player Mapping

Recall that the system produces musical elements by assigning motion vectors to FoxDot player objects. Specifically, we assign the longest *m* trajectories to melody players and the next longest *h* trajectories to harmony players. Doing so poses a slight problem for melodic continuity: the longest *m* and *h* trajectories may change rapidly, sometimes every frame. This can cause melody and harmony players to jump between different pitches rather than follow a single feature point through its trajectory. Visually, we can see this when the melody and harmony trajectories are "flickering" in the GUI player window.

In the next iteration of the system, it would be valuable to devise a method to ensure that when a trajectory is paired with a player object, this pairing persists until the trajectory length falls below a certain threshold (and thus is not perceptually salient enough to warrant

its use). Doing so may create more continuous lines of melody.

6.4.4 Additional GUI Features

There are numerous ways that our system can possibly be enhanced as a piece of creative software. Here we list a few:

- Supporting multiple "layers" of video to create additional sonic possibilities from overlaying multiple scenes
- Creating a looping system to save and replay generated sonic fragments. This can be a tool in live performance.
- Packaging the system into a mobile application (in the tradition of Brian Eno) with the ability for live recording and generation

6.4.5 Conducting Video-Music Correspondence Experiments with Reaction Time

In retrospect, for our experiment on judgements of video-music correspondence, one important metric was left out: reaction time. In future studies it may be valuable to measure how long it takes participants to come up with a judgement of "yes" or "no," making sure that this reaction time is measured entirely without their knowledge. Reaction time data may provide further insight into how participants respond to video and audio correspondences at various energy levels and energy differences. This data has potential to be highly informative in cognitive and psychological spheres.

7 Conclusions

In this thesis, we presented and evaluated a framework for generating music from moving images. We began by contextualizing a desired framework in terms of existing literature around artistic sonification, generative music, and cognitive/emotive links between motion and music. Next, we detailed our implementation of a software system that maps motion data to music using optical flow, corner detection, and a perceptually grounded mapping scheme. Then, we outlined the design of two experiments for our empirical study: one to test the robustness of our motion-to-music mapping by having participants make judgements on video-audio correspondences, the other to evaluate the system's potential as an audiovisual compositional tool. We followed by presenting the results of each experiment, noting that 1) participants correctly judged the correspondences at a rate better than randomly guessing, and 2) participants gave high ratings to the audiovisual works generated using the system.

In conclusion, we developed a system that is able to take moving images and generate music with a level of affective/perceptual correspondence to their motion qualities. The system also functions effectively as an artistic tool for generative composition, which can be explored in a wide variety of avenues from soundtrack generation to live performance. Future work should be done to improve the motion tracking and mapping scheme, enhance the functionality of the GUI, migrate the system to a more advanced audiovisual programming environment such as Jitter, and also potentially take *reaction time* into account for future experiments on video-audio correspondence.

Appendix

Experiment 1 Media: Video Only

https://www.youtube.com/playlist?list=PLjn8JyHMp4TJF6Ua7FMO_NKMrW6FcT_y6

Alternate Link: https://tinyurl.com/48h2ka3f

Experiment 1 Media: Audio Only

https://www.youtube.com/playlist?list=PLjn8JyHMp4TI7wM1hTx9w35m4diTn_EdQ

Alternate Link: https://tinyurl.com/4y5d4hw9

Experiment 2 Media: Audiovisual Pieces

https://youtube.com/playlist?list=PLjn8JyHMp4TIaw2b3jiEo47SWd7ZpiKFg

Alternate link: https://tinyurl.com/e3bmzhtm

References

[1] T. Hermann, A. Hunt and J. G. Neuhoff, The Sonification Handbook, Berlin: Logos Publishing House, 2011.

[2] S. C. Gwenaelle, R. Mallipeddi, J.-S. Kang and M. Lee, "Generating Music from an Image," in *SIGCHI*, Seoul, 2015.

[3] H. Rogers, "The Musical Script: Norman McLaren, Animated Sound and Audiovisuality," *Animation Journal,* vol. 22, pp. 68-84, 2014.

[4] "The Kid Should See This," [Online]. Available: https://thekidshouldseethis.com/post/pen-point-percussion-norman-mclaren. [Accessed 1 Nov 2022].

[5] M. Ojanen, J. Suominen, T. Kallio and K. Lassfolk, "Design Principles and User Interfaces of Erkki Kurenniemi's Electronic Musical Instruments of the 1960's and 1970's," in *New Interfaces for Musical Expression*, New York, 2007.

[6] J.-M. Pelletier, "Sonified Motion Flow Fields as a Means of Musical Expression," in *New Interfaces for Musical Expression*, Genova, 2008.

[7] J.-M. Pelletier, "Perceptually Motivated Sonification of Moving Images," in *International Computer Music Conference*, Montreal, 2009.

[8] A. Bregman, "Auditory Scene Analysis and the Role of Phenomenology in Experimental Psychology," *Canadian Psychology,* vol. 46, no. 1, pp. 32-40, 2005.

[9] P. Knees, T. Pohle and G. Widmer, "sound/tracks: Real-Time Synaesthetic Sonification of Train Journeys," in *International Conference on Multimedia*, Vancouver, BC, 2008.

[10] T. Pohle and P. Knees, "Real-Time Synaesthetic Sonification of Traveling Landscapes," in *International Conference on Multimedia*, Vancouver, BC, 2008.

[11] B. Sievers, L. Polansky, M. Casey and T. Wheatley, "Music and movement share a dynamic structure that supports universal expressions of emotion," *Proceedings of the National Academy of Sciences,* vol. 110, no. 1, pp. 70-75, 2012.

[12] M. Suzuki, S. Kataoka and E. Shimokawa, "Using Optical Flow in Motion Analysis for Evaluation of Active Music Therapy," in *International Federation for Medical and Biological Engineering*, 2014.

[13] B. Burger, M. R. Thompson, G. Luck, S. Saarikallio and P. Toiviainen, "Influences of rhythm- and timbre-related musical features on characteristics of music-induced movement," *Frontiers in Psychology,* vol. 4, 2013.

[14] V. E. Gonzalez-Sanchez, A. Zelechowska and A. R. Jensenius, "Correspondences Between Music and Involuntary Human Micromotion During Standstill," *Frontiers in Psychology,* vol. 9, 2018.

[15] E. Haga, "Correspondences between music and body movement," University of Oslo Department of Musicology, 2008.

[16] R. Loydell and M. Kingsley, "Thinking Inside the Box: Brian Eno, Music, Movement and Light," *Journal of Visual Arts Practice,* vol. 16, no. 2, pp. 104-118, 2016.

[17] R. Gradim and P. D. Pestana, "Overview of Generative Processes in the work of Brian Eno," in *11th Workshop on Ubiquitous Music*, Matosinhos, Portugal, 2021.

[18] B. Eno, Composer, *Generative Music 1*. [Sound Recording]. SSEYO. 1996.

[19] D. Belic, "IntoMobile," 9 Oct 2008. [Online]. Available: https://www.intomobile.com/2008/10/09/brian-enos-bloom-iphone-app-wants-to-relax-you-while-composing/. [Accessed 1 Nov 2022].

[20] "Brian Eno Back to Ambient Roots, in iPad App with Peter Chilvers, Upcoming Albumgro," CDM, 27 September 2012. [Online]. Available: https://cdm.link/2012/09/brian-eno-back-to-ambient-roots-in-ipad-app-with-peter-chilvers-upcoming-album/. [Accessed 12 10 2022].

[21] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools,* 2000.

[22] R. Kirkbride, "FoxDot: Live Coding with Python and SuperCollider," in *International Conference on Live Interfaces*, Sussex, 2016.

[23] C. J. Plack, A. J. Oxenham, R. R. Fay and A. N. Popper, Pitch: Neural Coding and Perception, Springer, 2005.

[24] A. R. Brown, T. Gifford and R. Davidson, "Techniques for Generative Melodies Inspired by Music Cognition," *Computer Music Journal,* vol. 39, no. 1, pp. 11-26, 2015.

[25] "What is Max?," [Online]. Available: https://cycling74.com/products/max. [Accessed Nov 2022].

[26] J. Kirk and L. Weisert, "Granular Wall: Approaches to sonifying fluid motion," in *International Computer Music Conference*, Utrecht.

[27] S. J. Hunt, T. Mitchell and C. Nash, "Thoughts on Interactive Generative Music Composition," in *2nd Conference on Computer Simulation of Musical Creativity*, Milton Keynes, 2017.

[28] M. Nayak, S. H. Srinivasan and M. S. Kankanhalli, "Music synthesis for home videos: An analogy based approach," in *Joint Conference of the 4th International Conference on Information, Communications and Signal Processing and 4th Pacific-Rim Conference on Multimedia*, Singapore, 2003.

[29] T. Parviainen, "How Generative Music Works," 2017. [Online]. Available: https://teropa.info/loop/#/airports. [Accessed 12 10 2022].