# Sound Stories

Sketching the Music of Film

---

Katherine Chui

M.S Data Visualization, Parsons School of Design

*Submitted in partial fulfillment of the requirements for the degree of Master Science in Data Visualization at Parsons School o fDesign*

# Table of Contents

# Introduction

## Sound and Music in Film

Sound is an important aspect of modern day film and storytelling. It comes in three main forms according to Jessica Green: dialogue (sounds made by characters in the film), noise, and music [1]. All of these aspects help create the atmosphere of a movie: setting the tone and signifying important plot cues to the audience.

In films, music does everything from setting tones and conveying intensity to foreshadowing and reminding its audience of past scenes. [2] Stam, Burgoyne, and Flitterman-Lewis break music into categories including: Redundant music which matches the main emotion of a scene, contrapuntal music which runs counter to that emotion, Empathetic music which conveys the characters emotions, didactic contrapuntal music which works to distance the audience from a scene, and  a-empathetic music which is neutral.

Music can work to develop motifs, signify themes, and build empathy between characters and the audience. Consider the Pixar Film, UP; in the beginning they introduce a musical motif that represents the connection between the main character and his wife. This musical theme starts cheerful as they meet, fall in love, and get married. As they grow old and his wife passes, the theme morphs–its tempo slows, the pitches shift, and notes are played with a softer, more reflective tone. This scene feels emotional although the character on screen isn't speaking or crying. As he reorganizes his house, keeping his wifes chair next to his, the musical theme chimes in, reminding us of how they fell in love as kids and all the life they lived together. It makes us understand what the main character is feeling without words: sad, reflective, nostalgic, and desperate to hold onto every memory of her. "UP" develops this musical theme at the start and uses it throughout the

1

2

film to remind us of the main character's motivations and memories. This just one example shows how music can be incredibly powerful in storytelling.

### Objectives

In this project, my goal is to explore how sound and music can be conveyed in a movie using only visuals. To do this, I will focus primarily on the music and capturing the emotions, intensity, and patterns it conveys. I also hope to explore how captions can further support the music and dialogue of a story, which I'll do by building a dynamic typographic library to display captions that match the intensity of the movie sounds.

In order to understand music in film from both a statistical and artistic point of view, this project is broken down into two levels of scale. The first level focuses on the imagery and captions that can be displayed in synchronization with a video and its audio; this level breaks a single video file into small time increments. The second level works to give a more general view of many videos in a movie showcasing different auditory trends across a full film. Through delving into both levels of generality, we can both experience music and the art that it creates as an audience member while also understanding sound in its larger context as it works to progress a story.

---

## Background

### History of Music in Film

In our modern era of entertainment, movies from Star Wars, Toy Story, Spirited Away, and Interstellar use sound effects, music, and dialogue to emphasize emotion, build intensity and action, and create music scores that people associate with each film after watching. This wasn't always the case; films in the 1890s to the 1920s were silent, either using live

music in theaters like plays or relying on the expressions of actors to convey the film's narrative. *The Jazz Singer* in 1927 by the Warner Bros was the first movie with synchronized sound. In 1931, the sound-on-film system inscribed sound waves as lightwaves onto the photographs, allowing soundtracks and videos to be stored in the same strip.[3] During the war, comedies, action movies, and musicals increased in popularity. This increased the production of orchestra music composed to match the pacing and emotions of the films. As the media industry grew through the late 1900s, music in films started to merge with jazz, rock and roll, and pop artists. Furthermore, synthetic music techniques started being added to film in the 1980s with movies like *Tron* and *Star Wars.* [4]

## Captions and their Significance

According to over 100 studies "... captioning a video improves comprehension of, attention to, and memory for the video" for all audiences. This includes people of all ages, those who are deaf or hard of hearing, and those watching movies in their non-native language.[5] Historically, captions first were introduced as "intertitles" which showed written dialogue in between scenes in silent films[6]. In the 1920, films with synchronized sound took over silent films, eliminating accessibility for the D/deaf community, and in the 1950s, the federal government authorized captioned films for cold war education. Captioning didn't begin appearing on television until the 1970s-80s, and by the 1990s, captions on TV shows became mandatory by law in the US. [7]

---

[3] Museum of Modern Art. "Experimentation with Sound." *MoMA*, accessed April 1, 2025. https://www.moma.org/collection/terms/film/experimentation-with-sound.

[4] Calgary Philharmonic Orchestra. "A Brief History of Film Music." *Calgary Philharmonic Orchestra*, accessed April 1, 2025. https://calgaryphil.com/blog-a-brief-history-of-film-music/.

[5]

[6]

[7] Gernsbacher, Morton Ann. "Video Captions Benefit Everyone." *The Permanente Journal* 21 (2017): 16-230. https://pmc.ncbi.nlm.nih.gov/articles/PMC5214590/.

### Limitations of Captions

Captions lead to greater comprehension and can enhance film experiences for all people. However, one existing limitation of movie captions is the written cues of a film's music. For example, many films denote sound changes through bracketed descriptions like: "[Music intensifies]" or "[Emotional music plays]". I hope to address this limitation by creating visuals through typography, shapes, and colors to better translate the music of films into the visual world. In this project, I explore a range of audio data driven visual representations of music in film working to reflect the emotional tone, intensity, and thematic motifs that the original scores communicate.

---

## Methods: Data Collection and Processing

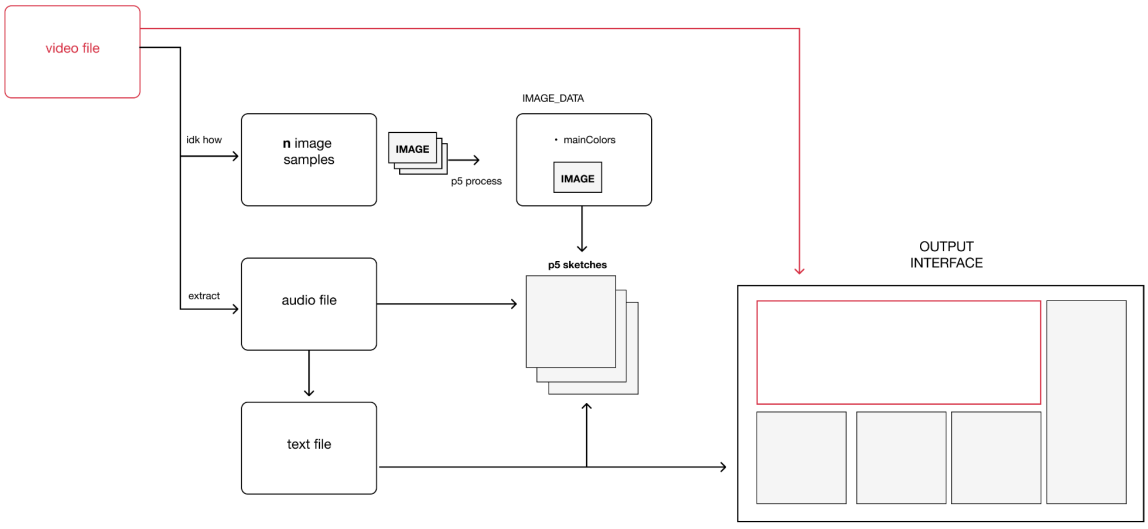
The majority data in this project is from existing films and video clips which I then process into various datasets summarizing the audio and image elements of the videos. In the audio processing phase of this project, I used two machine learning models which are trained on external datasets which are described in greater detail below.

By analyzing the music on both a specific level and a general level, it provides a picture of the experience of the music while also summarizing iit as an overall progression of the movie. On the most zoomed in scale, I extract various audio features from a sound clip which are specified as time series data. This includes features like amplitude, energy, tempo, the notes being played, the potential emotions represented in the sound, and the distribution of instruments being played.

## Video Processing

Taking as input either a youtube link or a video file, I created a function which does the following actions. If the input is a youtube link, it downloads the video. This step uses yt-dlp and saves it in a local folder It then breaks the inputted video (or downloaded youtube video) into n_samples sections by using ffmpeg to split the video into n_samples of equal duration clips which it then saves to a “videos” subfolder. Using ffmpeg, it then extracts audio + images from each of the n_samples video clips into “audios” and “images” folders respectively.

These folders ( “videos”, “audios”, “images”) contain the raw data for our movie. This function then delegates to two other functions, getAudioData and getImageData, to process the raw audio files and raw image files into json data.




The audio processing is the bulk of the data collection in this project, but the image processing provides necessary color information to create visualizations that are cohesive with each scene.

## Image Processing

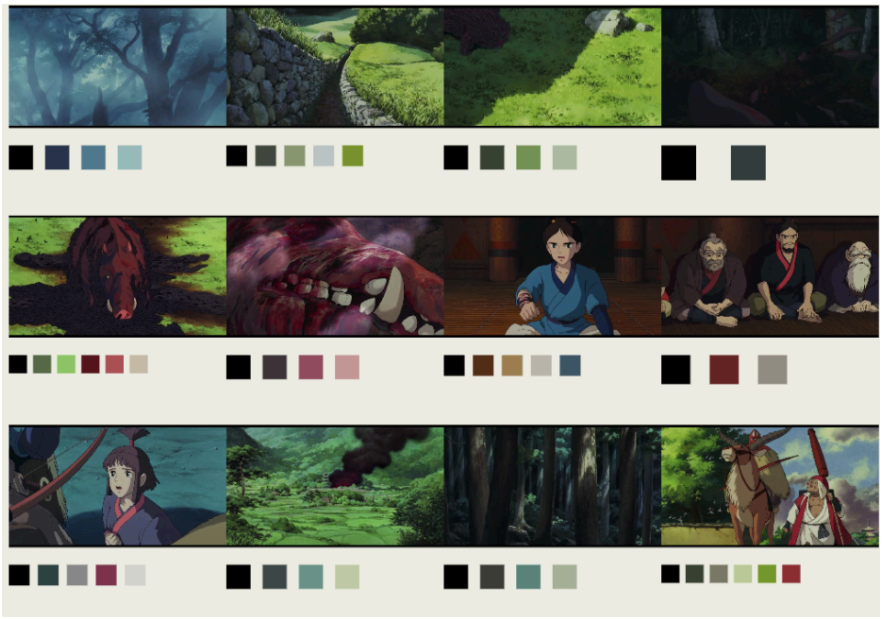
From each image, I extracted the “main colors”. To start, I read in the pixel data and converted it into a dataframe where each row represented the color of a pixel broken down into its rgb values. I then explored two methods for extracting the main colors.

For the first method, I used a Gaussian Clustering model to cluster the pixels into 8 clusters. I then used the cluster center coordinates to get the average color for each cluster. Since our data has three dimensions (R,G,B), we can easily get the color of each cluster by taking the first coordinate of the cluster center to be the red component, the second coordinate to be the green component, and so on. I then output the top 5 most saturated colors out of the 8 clusters.

The second method is conceptually very similar to a clustering algorithm, but works primarily to separate colors by a color_distance_threshold to create a list of unique colors. To do this, I iterate through each pixel in the image and add that color to a list of unique colors if it is “unique”. To determine if it’s “unique”, I compute the euclidean distance (distance of the line between two points in R3) between that color and every color that is already in the list of unique colors; if the pairwise distances are all greater than the color_distance_threshold , then it considered “unique” and it is added to the unique colors list. This process ensures that all colors in the final unique color list have pairwise distances greater than color_distance_threshold. I then return the top 5 most saturated colors from this list. In practice, I chose a color_distance_threshold of 100.


*Color extraction of a scene from the film, "Past Lives" 2023. This shows both methods with the selected colors as well as the pixels in $R^3$ space. We see that the Color distance threshold yields a brighter range of colors.*

After previewing the colors extracted using both methods, I found that the second method produced a greater variety of colors which better represented each image. This is likely due to the fact that the second method primarily isolates the pairwise distances between each color while Gaussian Mixture Models also factor in the size of each cluster causing these models to create more even sized clusters. As a result, if an image has a lot of dark colored pixels, it'll end up returning mostly dark colors instead of picking out the few "unique" colored pixels.



*Example or colors selected from a series of scenes from Studio Ghibli's "Princess Mononoke"*

## Audio Processing

The first step in this is extracting useful features from the audio file. For images, our goal was to understand the colors so breaking down each pixel into its rgb values made a lot of sense. With the audio files, it's less straightforward which features will tell us what. For example, one feature of audio is its volume, but we could have two very different audio clips that both have similar volumes. Since we are exploring different visualizations, emotional tones, and intensities of music, we need to capture a variety of features from our audio.

The audio library, pyAudioAnalysis provides two functions, shortTermFeatureExtraction and midtermFeatureExtraction to get features from a sound file. Both of these functions take a signal and sample rate from an audio file, normalize the signals and extract ~26 features including. I also used the librosa to extract other features like tempo.

### Initial Feature Extraction

All of the sound files start as .wav files which store the audio signal. My first goal is to process this file into a dataset of audio features like beat, tempo, energy, amplitude, and frequency. We start by looking at the fine grained sound features and then average these out for more general understandings of each sound file.

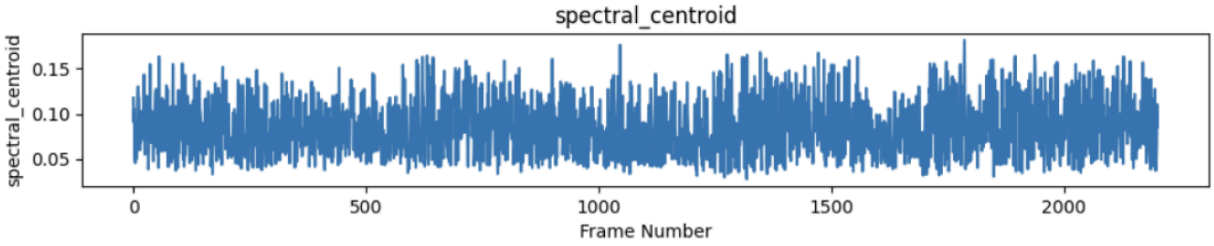
### What is Sound and How Do We Represent It?

Sound vibrates through a medium (like air or water) and travels in waves. When we listen to sound in real life, we are processing many sound waves added onto each other. That seems similar to a Fourier series which is essentially a sum of (sine) waves that can approximate functions like complicated sound signals. Therefore, it makes sense to

deconstruct these sums of waves using Fourier Transforms which gives us the coefficients of different frequency waves to match our sound file. These coefficients tell us how big or small the amplitude of the sound wave for that frequency should be which can help us understand how loud the sound is at different parts of the frequency spectrum.

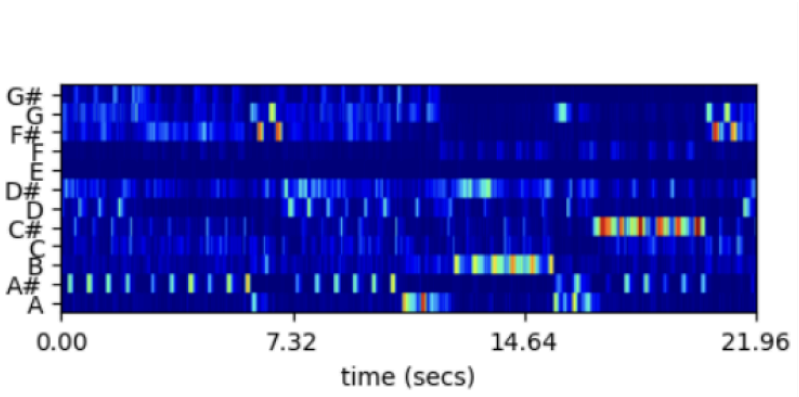
Another thing to consider is how large the amplitude of a frequency wave is doesn't necessarily correspond to how loud we perceive it to be; for example, humans process lower frequencies as louder than higher frequencies at the same amplitude. In order to account for this, I use the Mel scale, which rebalances the coefficients to better represent the perceived loudness of these frequencies. In my dataset, I use Fourier transforms combined with the Mel scale to get coefficients for each frequency.

I also analyzed features like energy, spectral centroids, and chroma at each timestamp of the given audio file.




Spectral centroids represent the center of mass of the spectrum (if we think of sound as lying on a spectrum and giving weight to different parts of this spectrum based on the loudness of different frequencies, the center of mass would be our spectral centroid). This is very useful in getting an overall picture of how frequencies are changing throughout the audio file. Chromagrams help us understand the specific music notes of an audio file; for this feature, I select the active notes at each time step by detecting which chroma values are above a certain threshold. [8]

8




*Example of a Chromagram*

### Data Aggregations

Using the fine grained audio features extracted for each scene, I computed aggregate values for each feature. For energy and spectral centroids I computed the average, and for amplitude I saved the maximum amplitude. For the notes determined by the chromatograms, I saved a list of all notes played during the audio file. I also used the librosa library to compute audio features over the entire audio file instead of the small timestamps in the previous features. This includes features like tempo and beat.

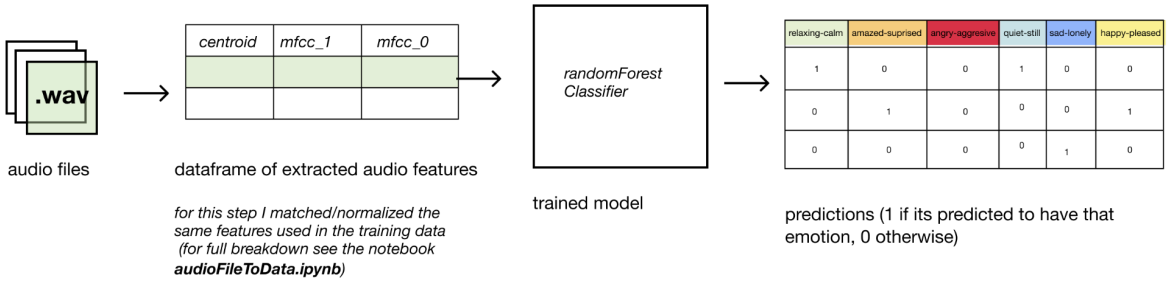
Finally, I used machine learning to make estimates at the emotions in the audio file. This was motivated by my curiosity as to how emotions can be conveyed in music and how they might relate or contrast to the film plots.

## Emotion Detection

The dataset I worked with was from the research study "Multi-Label Classification of Music by Emotion" which had a group of people label the emotions that they felt after listening to

each music piece. [9] The data consists of the audio features (Mfcc constants, spectral centroids, chromagram features, and more) for a sound file and the corresponding emotion label. There were around 300 entries in the entire dataset which I split 80:20 for my train:test split.

I used a random forest classification model which yielded a low classification error on both training and test sets. However, this dataset is relatively small and I wanted to see how this model performed on some of my actual data–do these emotion labels match my perceived emotions for different film scores?



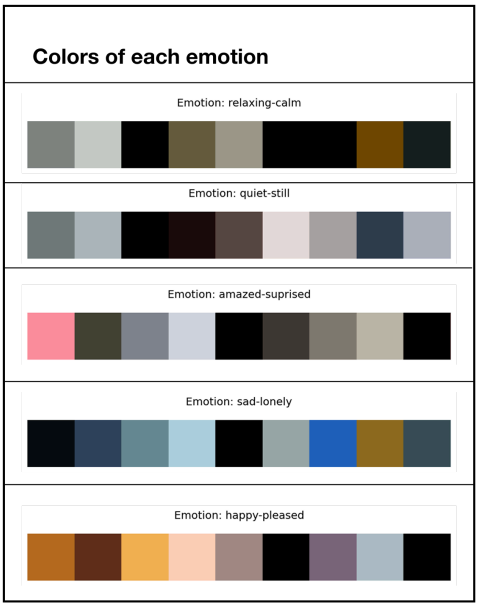

audio files

dataframe of extracted audio features

*for this step I matched/normalized the same features used in the training data (for full breakdown see the notebook **audioFileToData.ipynb**)*

trained model

predictions (1 if its predicted to have that emotion, 0 otherwise)




In order to apply this classifier to new data, I created an audioFeatureExtraction class and used the *librosa* python library to compute the same audio features from my .wav files as the training dataset used. I then loaded my audio files, extracted the audio features, loaded my pretrained model, ran predictions on my audio files, and saved the resulting labels in a csv file.

To understand these predictions, I used Chat-GPT to label the emotions of these films, and compared

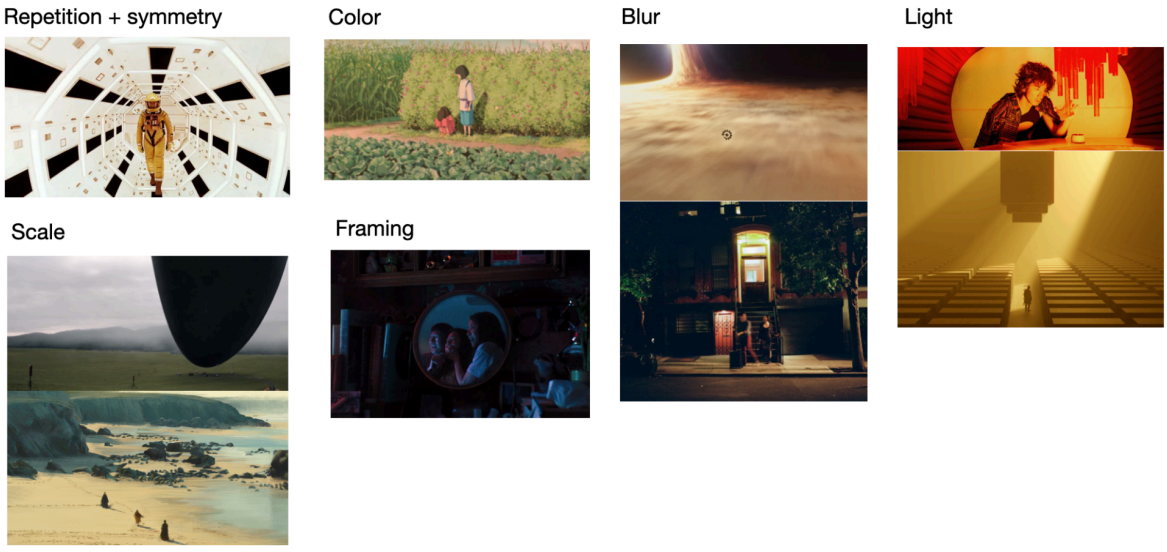
[9] Trohidis, Konstantinos, Grigorios Tsoumakas, George Kalliris, and Ioannis Vlahavas. "Multi-Label Classification of Music by Emotion." *EURASIP Journal on Audio, Speech, and Music Processing* 2011, no. 4 (2011). http://asmp.eurasipjournals.com/content/2011/1/4.

my results to these outputs. These results ended up corresponding. Finally, I decided to aggregate the same colors I had extracted from these films with the emotion labels to see if there were correlations between certains emotions and colors. The results revealed that more warm tones corresponded to the ‘happy-pleased’ label while more cool tones corresponded to the ‘sad-lonely’ label.

---

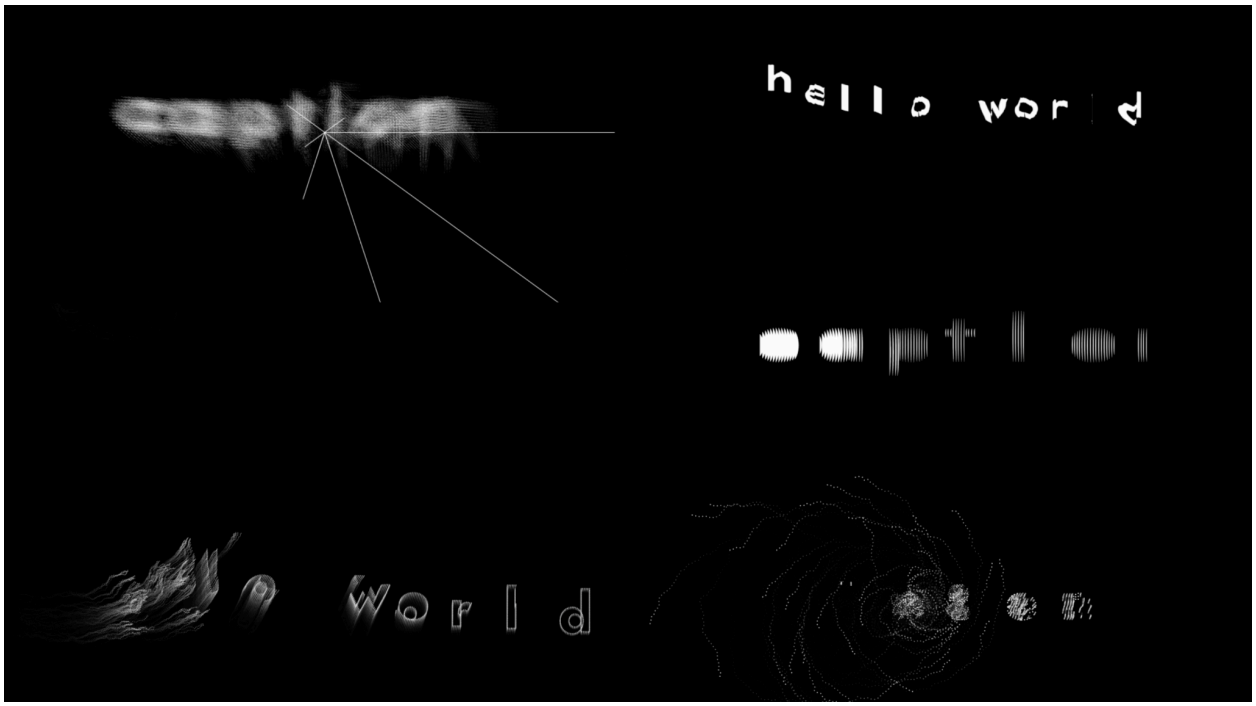
## Sound Sketches

I explored sound through a series of p5.js sketches which synchronize with the audio being played, translating the audio data into different visuals. These sketches are data driven as the colors are pulled from our imageSceneData and our audioSceneData dictates the motion and shapes of our visuals in real time. I was inspired by cinematography and how elements like repetition, symmetry, color, motion blur, light, framing, and scale can be used to set tones and convey plot. In my sketches I wanted to explore how these ideas can extend into sound.




To start, I was curious about how captions and text can reflect the sentiments of dialogue and music. My first dynamic type sketch works by transforming the text into lists of points

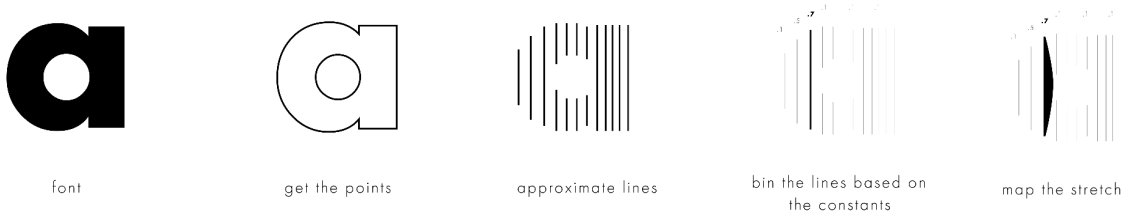
indicating the outer shape of each letter. Working with points allowed me to manipulate the text and distort its shape. One concept that I use throughout my sketches is mapping the amplitudes of frequencies determined by the FFTs (which I then rescale and normalize using the Mel scale as described previously) to different aspects of a sketch. In this exploration of type and sound, I divide the frequency spectrum into bins evenly for each letter, and then map the average amplitude of that letter bin to a feature. In this example, the first line of text maps amplitude to vertical position. This means the louder that range of the spectrum is, the higher up the letter will move. The second line of text maps amplitude to noise movement speed which means that a louder sounds will make the letters warp at a faster rate. I created this effect by adding noise to each of the points on the letter and animating it by using p5's frameCount variable. For the final version of this sketch, I used a combination of these amplitude mappings on the text so that it moves, warps, and stretches based on the sound.




This second sketch can be used on both text and image inputs. It works by first drawing the text or image onto the blank p5 canvas, then reading each pixel of the canvas, and

saving its coordinates into a data structure if the color of that pixel is *close enough*[10] to some inputted color. I then bin 360 degrees into n bins, and stretch lines stemming from each pixel towards the angle of each bin based on the frequency amplitudes. I used this similar idea for another font, “strings”, which emulates the plucking of string instruments by mapping the audio values to the curve pull of each string. Instead of processing an image, I get the lines for each font by simplifying the text points by their x coordinates (two points are the same if they are within some x_threshold distance of each other). This idea is illustrated below.

**Breaking down a font: Strings**



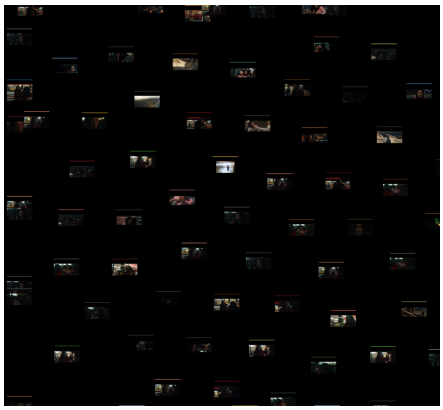

My final font sketches worked with flow fields. I created a *Particle* class for each point on the text, initializing it with a velocity (speed + direction) and position. Using a similar frequency binning process, I assign each particle to an audio value, and as the audio value increases or decreases, so does the speed of the particle and the magnitude of the noise applied to its position. The direction of the particle is modified at each frame of the animation to point towards the top of the canvas using basic trigonometric functions; once the particle reaches the top of the canvas, it redirects the particle to its starting position.

---

[10] *Close enough* - a color is close enough to another color if the euclidean distance between them is less than some constant threshold. The euclidean distance reverse to the distance between the colors in R3 space where the R,G,B components of each color determines its coordinates

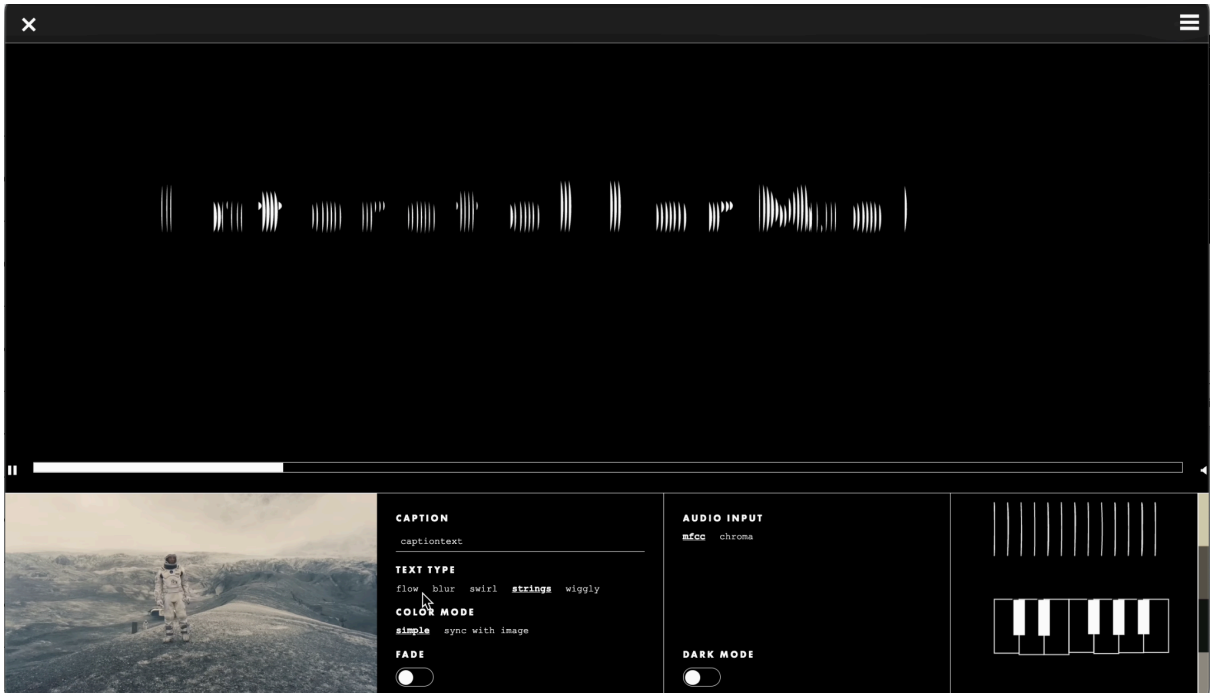
## Design

In order to keep the focus of this project on the art of film, sound, and image, I decided to keep the external interface minimal. I chose primary colors of black, and white for all of the functional elements, while using the colors that I extracted from the scenes to illustrate all of the film specific data visualizations in order to make the experience more immersive. To balance making the interface easy to navigate while also encouraging exploration, I created a generous interface of all the scenes of a film which users can click on to see more. This is in the opening page which gives users a preview of many of the features in the page including the dynamic fonts, color extractions, and range of movie scenes represented. As the user hovers over different scene previews, the audio plays which then drives the text to respond.

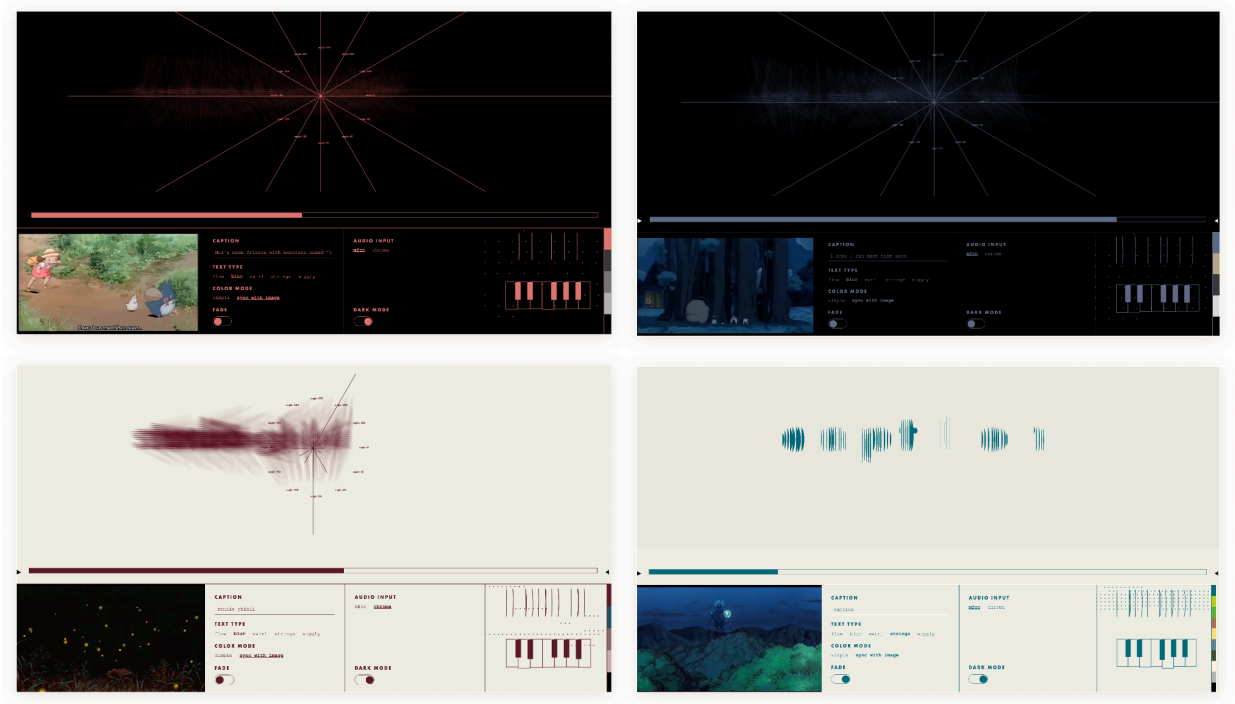



### Video Player Interface

This interface serves three main functions. It allows users to watch and control the video, experience and modify the typography, and see the live instruments and colors extracted from the audio. If the scene has caption data, the typography should also sync with the live captions of the video.




I wanted this interface to feel multi sensory; this meant capturing the sound, colors, and tones of each film. In order to do that, I wanted to make the CSS dependent on the image data that I extracted from each video. To start, I created a dark/light mode option which toggles the background and interface colors from light to dark; this is intended to match the darkness level of the different films. Secondly, I created a color mode called “sync with image” which sets the colors of the interface to change based on the colors of the image. Instead of changing the color at every second, this function is only called when a “beat” is detected in the audio; this is another audio feature which I extracted using a tempogram.



I also wanted to expose as many parameters to my fonts as possible to encourage exploration as each film might have a different font, audio range, or color mode that works best. The bottom bar contains the caption value which will sync with caption data if available, but is also a modifiable input. The “text type” input allows users to select between 5 dynamic font options from my library including ***flow***, ***blur***, ***swirl***, ***strings***, *and* ***wiggly.*** Finally, the users can select the input to the fonts: ***mfcc*** or ***chroma***. These constants were described in the audio processing section; essentially the ***chroma*** highlights the different notes in the signal while the ***mfcc*** represents the loudness of each frequency that we *perceive.* Changing this input will change which features will drive the font’s movement.

## Movie Dashboard

This view allows users to navigate the different scenes of the larger film while also viewing sound trends of the film through time. The line plot on the top syncs its x-axis with the timestamps of the scenes laid out on the bottom and depicts the spectral centroid. This

feature is sampled hundreds of times per scene, so in order to simplify the trends, I plotted the average of the spectral centroid within some chunk interval. On the selected scene, it plots every sample, on the previous and next scene, it plots the average for every 10 samples, and for the remaining scenes, it averages over every 50 samples. With this dynamic plot, I was interested in playing with scales and decided that changing the precision based on the selected scene offered both a detailed and general picture of the

data. This is because without the aggregations, the line plot looks very chaotic and it's difficult to notice trends, but with the aggregations, we lose our visibility of outliers.

The second plot depicts the chromagram data. Since a chromagram helps pick out notes on a piano (regardless of the octave), I decided to use a piano to depict this. Each set of piano keys represents the notes found in the corresponding scene, and the notes are filled in if they are played. I was interested to see if this helps us notice dominant keys or notable key changes in the film. Furthermore, I added a feature that plays the selected keys in a

scene when you click it in order to bridge the visual understanding of the piano with our intuitive understanding of notes.

## Conclusion

### Summary

In this thesis, I created an audio driven interface that allows users to watch a film with real time, audio reactive captions, instrument illustrations, and beat sync-ed colors.
This immersive interface ultimately works towards my goal of visualizing the sound of movies through type, color, and illustration. The dynamic font library works to capture the intensity, emotion, and pacing of film dialogue that we miss in traditional captioning. Furthermore, the interface colors are driven by the films themselves, creating an interface that feels like an extension of the film itself. Through this project, I also created my own data generation pipeline pipeline that takes a video, sections it into scenes, extracts the main colors at each second, and computes the audio data using spectral analysis and emotion classification. Therefore, this project can be extended to more films and used as a tool for generating typography from music.

### Limitations and Future Directions

In this project, I faced several challenges. To start, the readability of different fonts can vary depending on the audio inputs and selected fonts. I partially addressed this by listing the caption in plane text while also exposing the different font parameters and font types to allow users to tweak the fonts to improve legibility. Secondly, larger videos like entire films can reduce the performance and increase the loading times of the site. Consequently, I mainly used single scenes, but am therefore lacking some insights on overall audio trends across a film in its entirety. I also struggled with getting the captions for different films in a consistent file format which I hope to remedy by extending the compatible caption formats. I am also interested in using LLMs to generate my own caption files from the audio files.

One of my main goals going forward, is to add functionality for users to upload their own videos which the backend will then process and load onto the site. I also hope to integrate the emotion labeling and visualize more trends of the music across the entire plot. In general I want to keep exploring more of the larger scheme trends in the audio data over films, composers, and genres. For example, in my emotion detection, It was interesting to see how the colors correlated to different emotions. I could extend my current loading screen to also show the scenes clustered by emotions, color, and genre.

### Reflection

Film is a multisensory and interdisciplinary art form that can be enhanced through data visualization to create a more accessible and immersive experience. I hope this project can be a space for exploring and enjoying the visual beauty of sound, the intersection of music, narrative, and design.

## Bibliography


Calgary Philharmonic Orchestra. “A Brief History of Film Music.” *Calgary Philharmonic Orchestra*, accessed April 1, 2025. https://calgaryphil.com/blog-a-brief-history-of-film-music/.

Blaszke, Maciej, and Bożena Kostek. "Musical Instrument Identification Using Deep Learning Approach." *Sensors*(Gdańsk University of Technology). Accessed March 31, 2025.

Gernsbacher, Morton Ann. “Video Captions Benefit Everyone.” *The Permanente Journal* 21 (2017): 16-230. https://pmc.ncbi.nlm.nih.gov/articles/PMC5214590/.

Green, Jessica. "Understanding the Score: Film Music Communicating to and Influencing the Audience." *The Journal of Aesthetic Education* 44, no. 4 (December 2010): 81–94. https://doi.org/10.1353/jae.2010.0009.

Lu, Yun-Jhen, I-Chun Kuo, and Ming-Chou Ho. "The Effects of Emotional Films and Subtitle Types on Eye Movement Patterns." *Acta Psychologica* 230 (2022).

Ikhsan, Sandi Putra. "Analysis of the Structural Complexity of 'Time' in Hans Zimmer’s Score for *Inception*." *Interlude* 3, no. 1 (November 2023). https://doi.org/10.17509/interlude.v3i1.70083.

Trohidis, Konstantinos, Grigorios Tsoumakas, George Kalliris, and Ioannis Vlahavas. "Multi-Label Classification of Music by Emotion." *EURASIP Journal on Audio, Speech, and Music Processing* 2011, no. 4 (2011). http://asmp.eurasipjournals.com/content/2011/1/4.

Sable, Anuj. "Introduction to Audio Analysis and Processing." *Paperspace Blog*. Machine Learning, 4 years ago. Accessed March 31, 2025. https://blog.paperspace.com/introduction-to-audio-analysis-and-synthesis/.

Sparsh, I.M. "MusicNet Dataset." Kaggle. Accessed March 31, 2025. https://www.kaggle.com/datasets/imsparsh/musicnet-dataset/data.