Authors: Brendan Zelikman and Tan Vu
Professor: Felix Heide
Advised by Chloe Qiu

The Color of Music

Abstract

Using the codebase for a virtual piano from Borja Morales and inspired by the VSKeys project, The Color of Music is a 3D music-to-color visualizer featuring a functional piano keyboard that maps musical notes to color. As the user plays notes on the keyboard, the corresponding colors are dynamically blended and painted as background gradients. The visualization features incredibly diverse customization and can be highly relaxing, exciting, and stimulating. With a robust GUI, users are able to modify various effects, including the style and sound of the keyboard as well as color and spotlight visualizations. One can additionally choose to display other objects within the scene, including a 3D sphere and wave which react to the music and get mutated by sound effects, such as reverb and phaser.

1. Introduction

The goal of this project was to visualize chromesthesia, a type of synesthesia in which sounds can manifest as visuals of colors, shapes, and movements. Most existing chromesthesia visualizers (and music visualizers) demonstrate the visuals as psychedelic and geometric. Such visualizers rely on the usually harmonic nature of music to create an organized and calming visual aspect that adds to a more enlightening listening experience. However, music is not always harmonic. For example, smashing the piano keyboard rarely gives a pleasant song. It would not make total sense if the visualizer for such a sequence was calming. The same thing can be said for the psychedelic portion of the visualization, as not all psychedelic visuals are peaceful. We wanted to create a visualizer that breaks from the framework of harmonic relationships and focuses on capturing aesthetically pleasing results tied to pitch.

While some aspects of the project, e.g. background gradients, are systematic, other aspects, e.g. spiky spheres, are noisier and more unpredictable. Each visualization is intentionally random and fast paced, creating an overall more exciting and stimulating display. Our project is a playful experiment to create a different, more chaotic model of music visualization. It shows the potential disorder of the usually harmonic music and creates intense but unique psychedelic visuals that are not often described or depicted.

We draw inspiration from multiple existing projects from both the computer graphics and media domain. Most notably, we were inspired by VSKeys—a 3D piano harmony visualizer by Diego Zamalloa-Chion and Aaron Skepasts, created as their COS426 Final Project in Spring

2021. VSKeys employed a spherical implementation of the Tonnetz grid along with geometric spirographs based on the keys pressed. We followed VSKeys to use an open 3D space to create an immersive experience and forked Borja Morales's piano code as boilerplate for our project.

2. Approach

Given the idea and goal of the projects, our approach was straightforward. For the MVP, we sought to fork the Borja Morales keyboard and implement a primitive color visualizer. We use ThreeJS to handle the graphical elements of our project and ToneJS to handle the audio. After updating the keyboard to work with a more recent version of ThreeJS and ToneJS, we add custom event handlers to detect key presses and flesh out the functionality of the GUI. After the MVP, we introduced a 3D sphere and 3D wave to complement the keyboard as alternative models, and worked on adding more settings and effects.

3. Implementation

3.1 Keyboard Model and Sound Effects

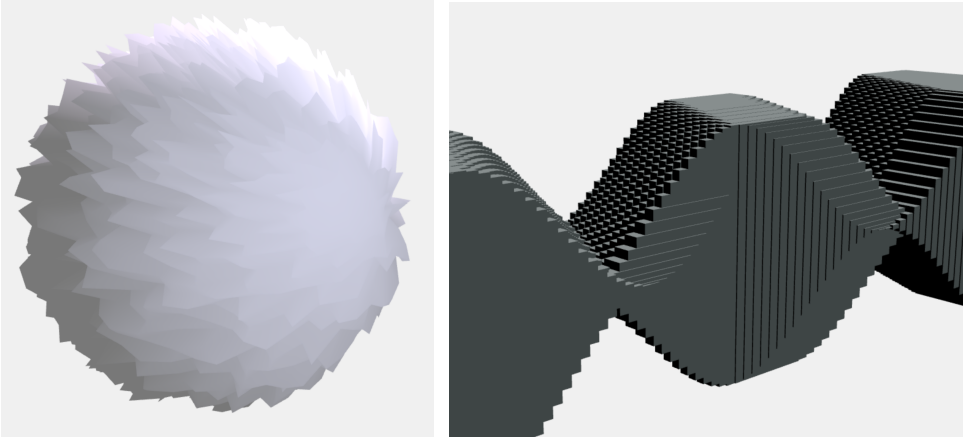
To create the keyboard model, we adapted a 3D piano player from Borja Morales and used key labels from VSKeys (Figure 1). We wanted to let the user play with a wide range of notes using their keyboard, while avoiding cumbersome utility keys like Tab and Delete that are often used for other kinds of event handling. We reworked the key layout to start with “Z” for C3 and end with “]” at C5, encompassing 36 notes or three octaves total. The octave can be changed by the user, transposing the piano and moving the key labels across the keyboard. One can additionally choose to show/hide the key labels, which are laid across the piano keys.

Figure 1: Piano keyboard with key labels

Using ToneJS, we included the ability to change the instrument to a sampled grand piano, an AMSynth, an FMSynth, or a DuoSynth. Each instrument has a distinct timbre and selecting one applies a different color palette to the keyboard. The generated sound is chained through five audio effects—reverb with a decay control, chorus with a frequency control, phaser with a frequency control, vibrato with a frequency control, and volume with a level control—that can be controlled on a scale from 1–100. One can additionally choose to mute the keyboard or sustain the sound after the keys are released.

3.2 Sphere and Wave Models

To incorporate further customization of the scene beyond our sources of inspiration, we added a 3D sphere model and a 3D wave model that can be toggled to render instead of the piano. Contrary to the static keyboard, these two models are dynamic and react to the music: the sphere pulsates in the center of the screen with a noise function that randomly offsets the vertices (Figure 2) and the wave oscillates across the screen with a trigonometric function that manipulates the size of moving cubes (Figure 3). The material of the models can be set to a Basic, Phong, Toon, or Wireframe material, and the reactions can be set to occur automatically or upon key press. The models are additionally mutated by the current sound effects applied to the keyboard—reverb adds size, chorus adds stutter, phaser adds jitter, and vibrato adds noise.



Figures 2 and 3: The sphere and wave models

3.3 Color and Spotlight Visualization

We created five note-color maps by generating a pattern of colors across the circle of fifths, a circular arrangement of all 12 notes of a musical octave. Rainbow consists of a red-to-violet spectrum starting from C going clockwise. Reverse Rainbow consists of a red-to-violet spectrum starting from C going counterclockwise. Scriabin and Messiaen correspond to the chromesthetic associations of 20th century composers Alexander Scriabin and Olivier Messiaen, respectively. Mars corresponds to the associations of Warren Mars, a contemporary musician with a published website covering chromesthesia[1].

Individual notes are mapped to colors on the screen, and simultaneous notes are blended together into a radial, conic, or animated linear gradient. The color changing effect is accomplished by changing the scene's background color. We experimented with numerous ways to display color, including changing the ThreeJS scene background, and using materials and textures. Ultimately, we decided to render it by changing the HTML document background color, as this allowed us to easily make use of CSS gradients and transformations. Changing document

[1] https://warrenmars.com/visual_art/theory/colour_wheel/music_colours/music_colours.htm

background color is also the most inexpensive way to do this, increasing the project's performance. By nulling the background of the Borja Morales scene, we are able to display a 3D keyboard over a 2D HTML page.

We also used ThreeJS's SpotlightHelper to create cone-shaped line drawings which pop up across the scene as the user plays and disappear when the keys are released. SpotlightHelper displays a cone shaped object that traces a Spotlight object's lighting, which coincidentally looks interesting when clustered. The spotlight helpers are randomly generated around the scene and fired in a circular, top-down, bottom-up, or random direction in bursts, as shown in Figure 4.

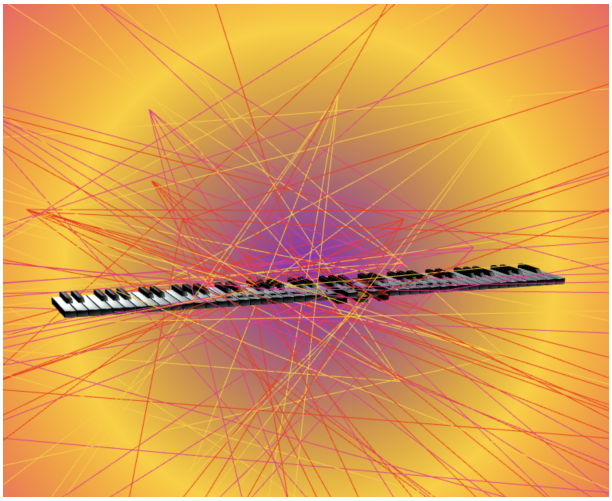


Figure 4: Keyboard in front of a radial gradient with spotlight helpers

For conic direction spotlight helpers, we used a CircleGeometry with 16 vertices as the base where the spotlights originate. Each spotlight originates from one of the vertices and points at a different target coordinate. For spherical direction spotlight helpers, each spotlight originates from a location on screen where the x, y, z coordinates are randomly chosen within a range and points to the same target coordinate. For random direction spotlight helpers, each spotlight originates from a random location and has ten possible target positions. We find spotlights with different origins pointing to the same target to be aesthetically interesting.

In addition, we created a vintage color mode by generating a giant pulsating sphere that encapsulates the scene and dims the lighting, making the colors more muted and wavy.

4. Results

Though the concept of a music visualizer is broad, we were successfully able to hone a specific aesthetic using background gradients and create a robust application. The visualizer ended up very immersive and stimulating, due to the naturally pleasing gradients and the expansive amount of features present. The noisy sphere and randomized spotlight added another intriguing aspect to the visualizer but did not make it seem completely random. We added a few

custom presets into the GUI featuring some of our favorite settings and effects for each synth, showcasing the breadth of the project.

5. Discussion

Though the project turned out successful, there are still plenty of avenues for further work. We were able to observe the connection between harmony and color anecdotally, but we did not add any explicit functionality, as VSKeys made the main focus of their project. During development, we experimented with rendering MIDI files but scrapped the feature after finding that the MIDI player was not capable of properly rendering the audio effects applied to the instrument. We also experimented with rendering 3D OBJ files, but found them overly difficult to manipulate and instead focused on native ThreeJS meshes and geometries. Some potential steps might be to revisit these two ideas and integrate their functionality. Additionally, the instruments, models, color maps, and effects were all chosen out of personal preference, but it would certainly be possible to incorporate more.

6. Conclusion

We are overall satisfied with the result of our project as we accomplished our goal of creating a music visualizer that breaks from the traditional harmonic visual model. The project incorporated the harmony and the noise of music into a coherent and immersive visualization.
Our approach to implementing our ideas, however, was quite naive. While the 3D sphere, waves and spotlights all successfully expanded previous music visualizer models, our implementation was simple. I.e. changing sphere by random noise, creating spotlight at random location and using simple trig functions to create waves. There are also many features we wanted to implement but were not able to, leaving a lot of potential for future work.

7. Contributions

Brendan adapted the piano keyboard from Borja Morales and used ToneJS to implement the instruments and effects.
Tan implemented the color gradients, spotlight helpers, and the 3D models.
We both contributed to the GUI and added effects to the visuals after the MVP.

References

Morales, Borja, "3D Piano Player", https://github.com/reality3d/3d-piano-player
Zamalloa-Chion, Diego, and Skepasts, Aaron, “VSKeys”, https://deegzc.github.io/VSKeys/
"Three.js - JavaScript 3D Library," *Three.js*, https://threejs.org/docs/
“Tone.js Docs”, *Tone.js*, https://tonejs.github.io/docs/14.7.77/index.html
"Chromesthesia," *Wikipedia*, https://en.wikipedia.org/wiki/Chromesthesia