Translation between Image and Music

Rongzhong Li

Physics

Music is the art of time while painting is the art of space. Both of them provide people with beauty and joy. Sometimes the feelings can even overlap. There has been software that visualizes music to colorful animations. The procedure is analyzing the digitalized music data, mapping the frequency and amplitude components to different colors and shapes, and linking them using smooth animations. People can feel the emotion of the music by looking the visualization so it's an extra way to understand and appreciate the music.

But we can notice that this way of visualization takes the same time as the music itself; Most of the beauty comes from the transform function, not the music by itself. What if we develop an algorithm to project the music onto a still 2D or 3D image? People can look at its structure and colors to expect what they will hear.(music note is a too accurate visualization to understand and still not intuitive at a glance.) The projection can be standardized so the pattern can even act as a preview of the music. With the boom of music production, this will provide people a faster way to pick what may interest them.

The algorithm can also be applied reversely. Notice a picture taken by a fine camera is about 4M, which is roughly the same size as an mp3 song. From the view of information storage, the transform can be lossless. It would be fantastic if we can play a music by simply scanning its pattern. If the algorithm is good enough and the pattern it generates looks natural, we can even apply the image-to-sound process to any graphics to stimulate composition, or just for fun.

An early stage of this project has been developed as below:

Input:

Any music file has to be transformed to wav file using audacity for processing. Then use Matlab read in the file as changing amplitudes over time.

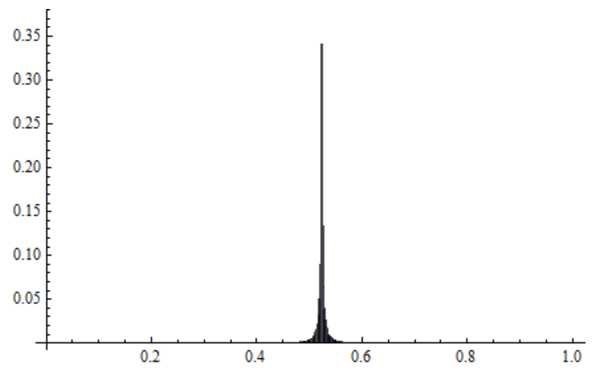
DCT:

We need to divide the length to shorter blocks. A natural choice is to apply DCT on each bar to get its frequency component. To reduce data size and catch the main feature of the music, we can limit the frequency up to some reasonable high level, and record their normalized (0~1) frequency amplitude. So we get a Time×Frequency Matrix and it will be the raw data for our visualization.

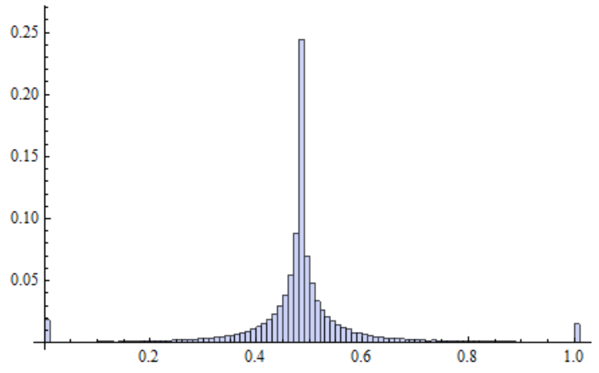
Raw Data Analysis:

I'm using Mathematica to generate the graphs.

Make a histogram of the frequency amplitude to find the most popular range. Rescale over that range to spread the amplitude. To compare different music, the rescaling operation should be standardized. The rescaling will also affect the background (the average data value) of the image. (See attached images.)

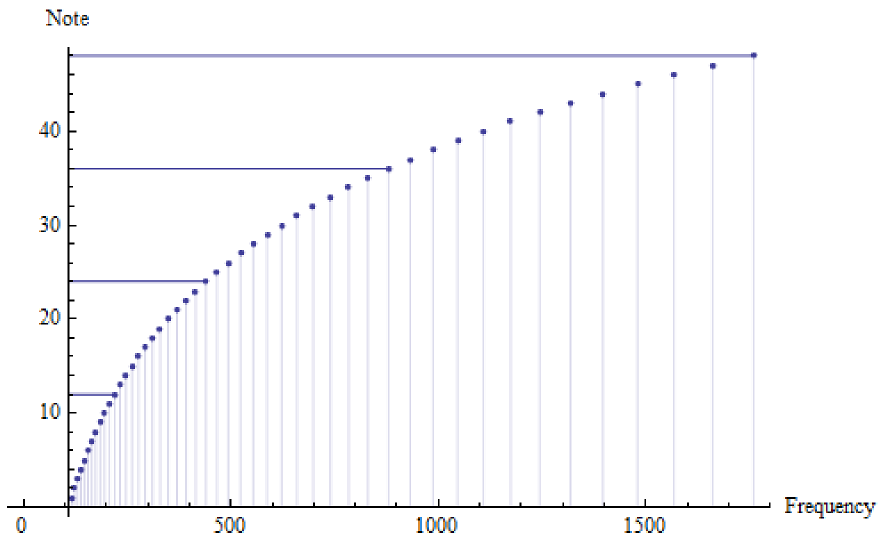


F1. Original Histogram



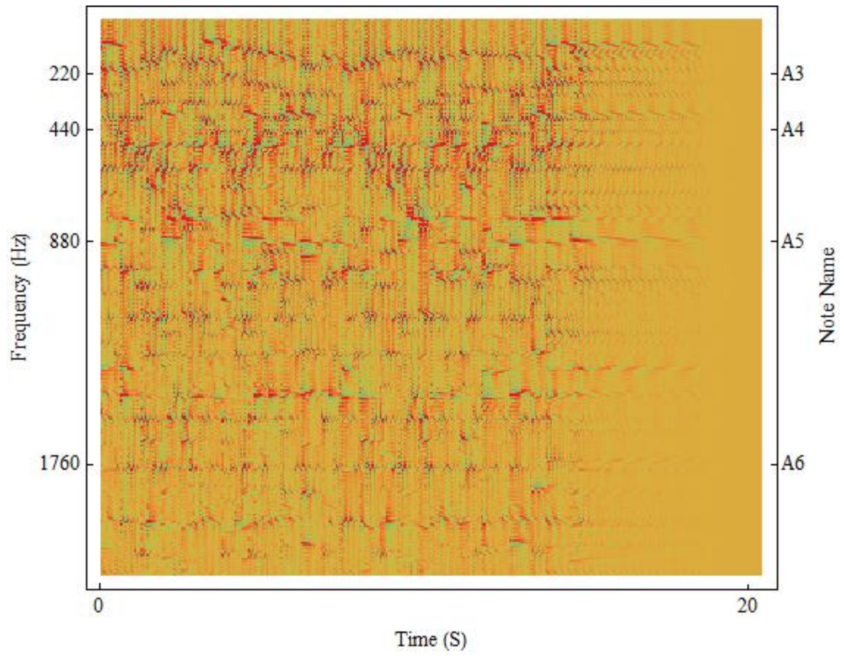
F2. Rescaled Histogram

We need to understand the relationship between pitches. It's F'=F0*2^(n/12).



F3. Pitch Curve

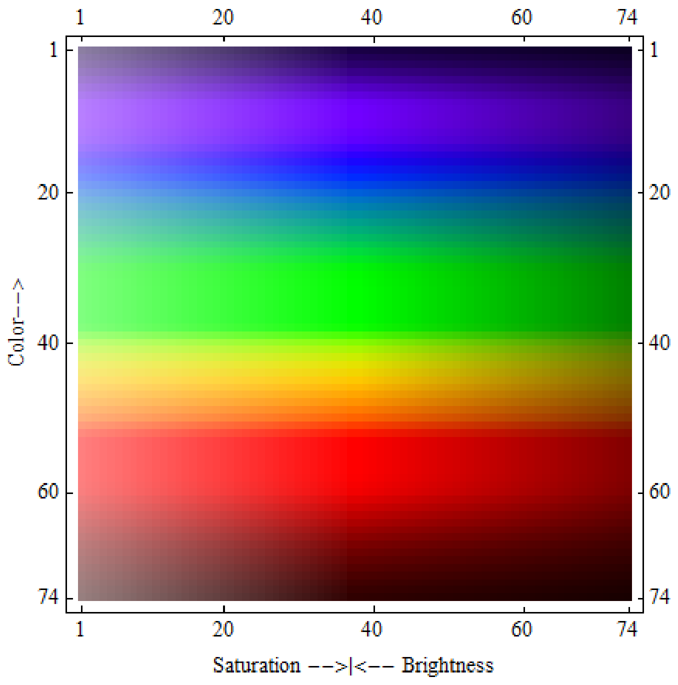
Now we can plot the raw matrix. The labeling is based on the conversion between coordinates and frequency.



F4. A Preview of the Raw Data Structure

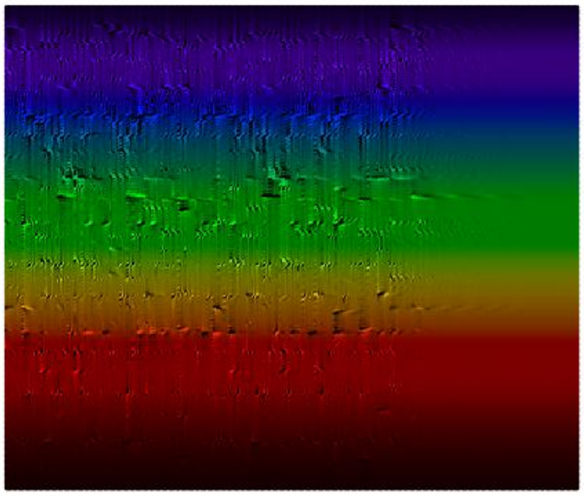
Mapping Frequency to Image:

To map audio to image, we need to use indexed color. Considering the similarity between light and wave, I chose the colormap of visible spectrum.

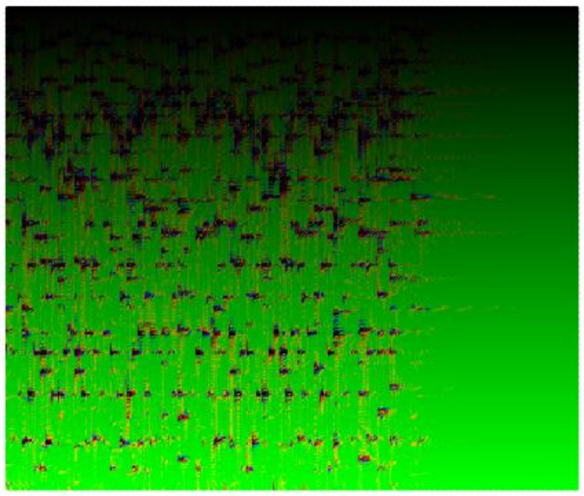


F5. Visible Spectrum

I tried two methods to tune the colormap. One is projecting the frequency to color, and amplitude to color brightness. The other is projecting the amplitude to color, and frequency to color brightness. Their effects are shown in the graphs below. Though the first projection is more intuitive, the second represents frequency components better.

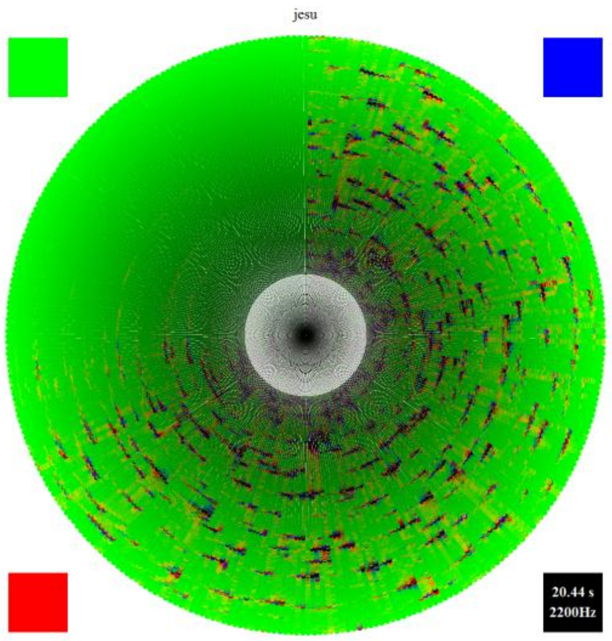

F6. Frequency to Color,

Amplitude to Brightness.


F7. Amplitude to Color,

Frequency to Brightness.

To make the graph more symmetric and artistic, I plot the same content in a polar coordinate. The image to the right is for a MIDI piano music. The angle is time going clockwise and the radius is frequency raising outward. Every element is a colored disk with a radius proportional to frequency. This transform sacrifices the resolution in low frequency end and causes some moire pattern (The "central ring"), but the more expressive high frequencies (middle region notes) are magnified along radius. It can be solved using larger image size, or other rendering method (discussed later).



When looking at this image, we can see the frequency components at each moment and their fading afterwards. Because it's played by MIDI piano, the frequencies are clear and discrete (compared with other instruments). Similar pattern indicates repeating paragraphs. The middle tone is green for this piece of music. But if we use a standardized range to rescale other music, it can be different.

In principle, this image contains most of the audio information. The RGB rectangular on the corners are for inverse transform (IDCT) purpose. Using their position and color, we can calibrate the image shape and color during scanning. The text in the black box tells the duration of audio, and the highest frequency on the radius.

Discussion:

1. **Efficiency:**

The DCT part is fast, but rendering the image is very slow and often causes the Mathematica kernel die due to memory exhaustion. This can be solved using another rendering method. Currently I'm defining the colored disks' position and size, then let the program make a vector graph. It's good for resolution, but consumes lots of memory. Instead, I can generate a colormap using indexed color and define the color of each pixel, then write the color matrix to a BMP file**.**

2. **Parameter:**

Because currently the time is arranged over angle, the length of DCT blocks is depended on the total length of the music. Longer music only allows a more coarse grained bar division. The range of frequency is taken from 0 to 2200Hz, which is the main range of a piano. However lower frequencies are compressed in the central region and the higher frequencies are cut off.

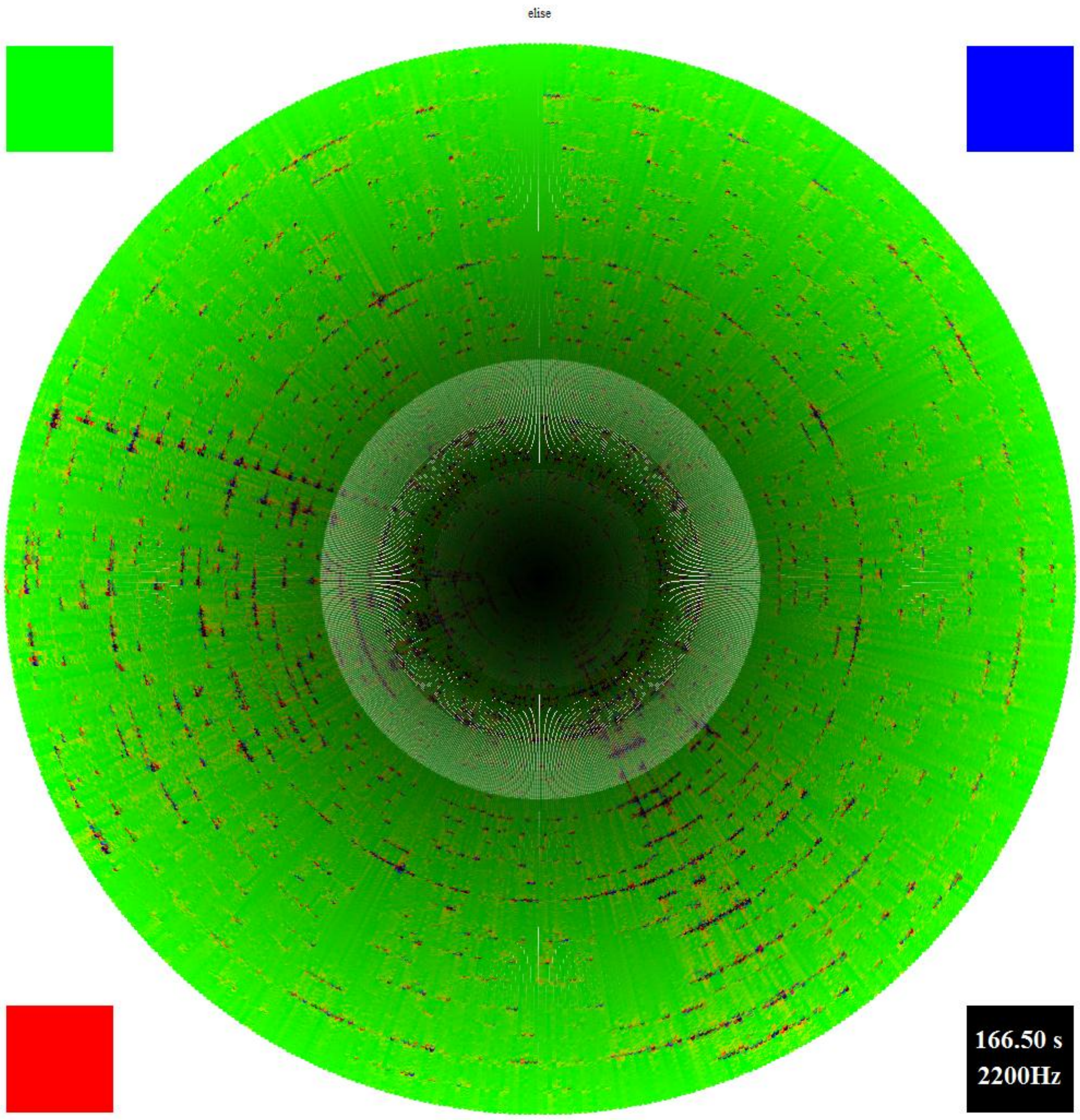
3. **MIDI Implementation**

The transform described is from a mixed audio to separate frequency components. With a MIDI file, we can actually know all the knowledge. So we can try to mark the major notes beyond the mixed background later.

Appendix:

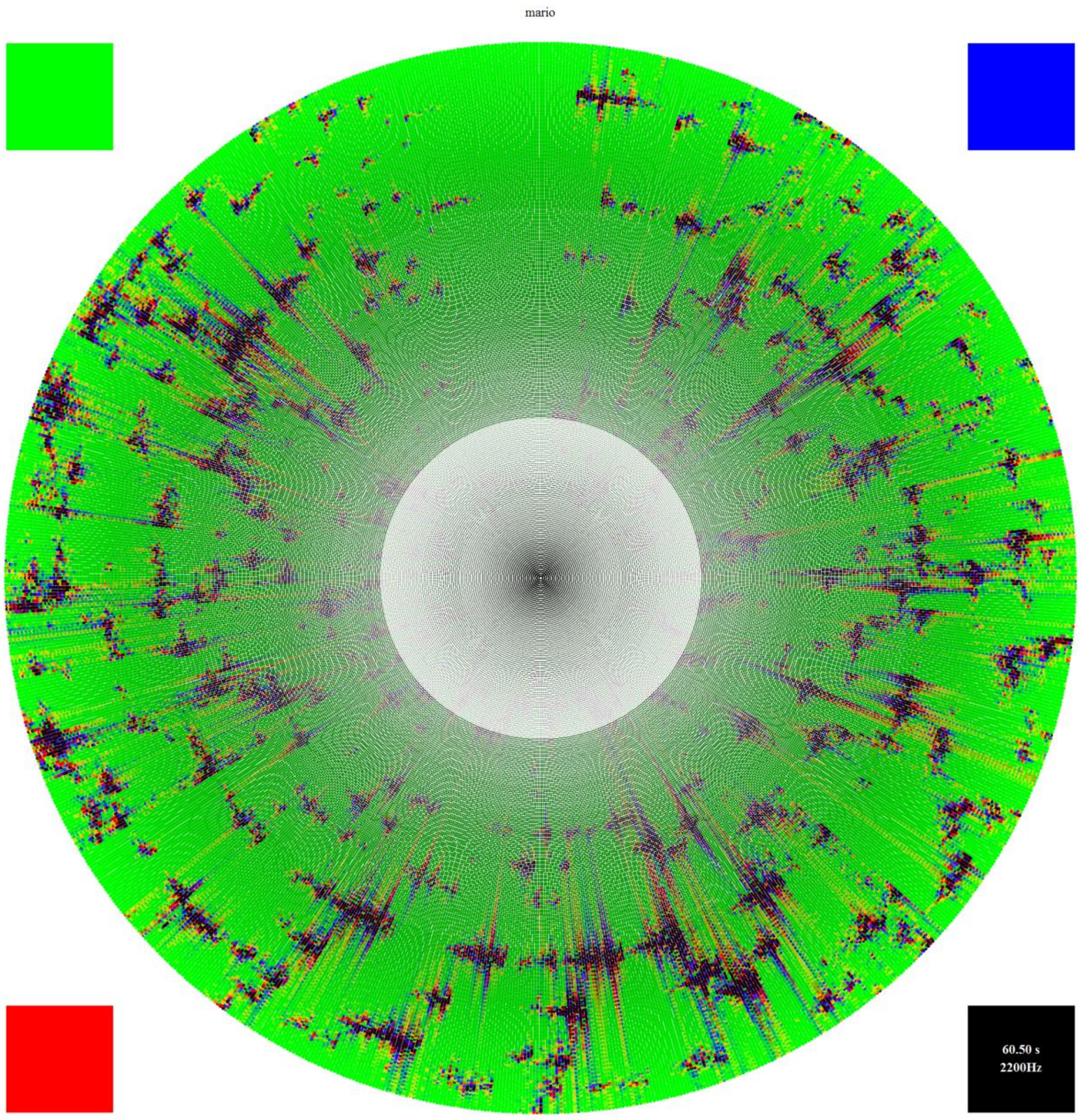
Below are the images of some other sample music.

Fur Elise (piano)



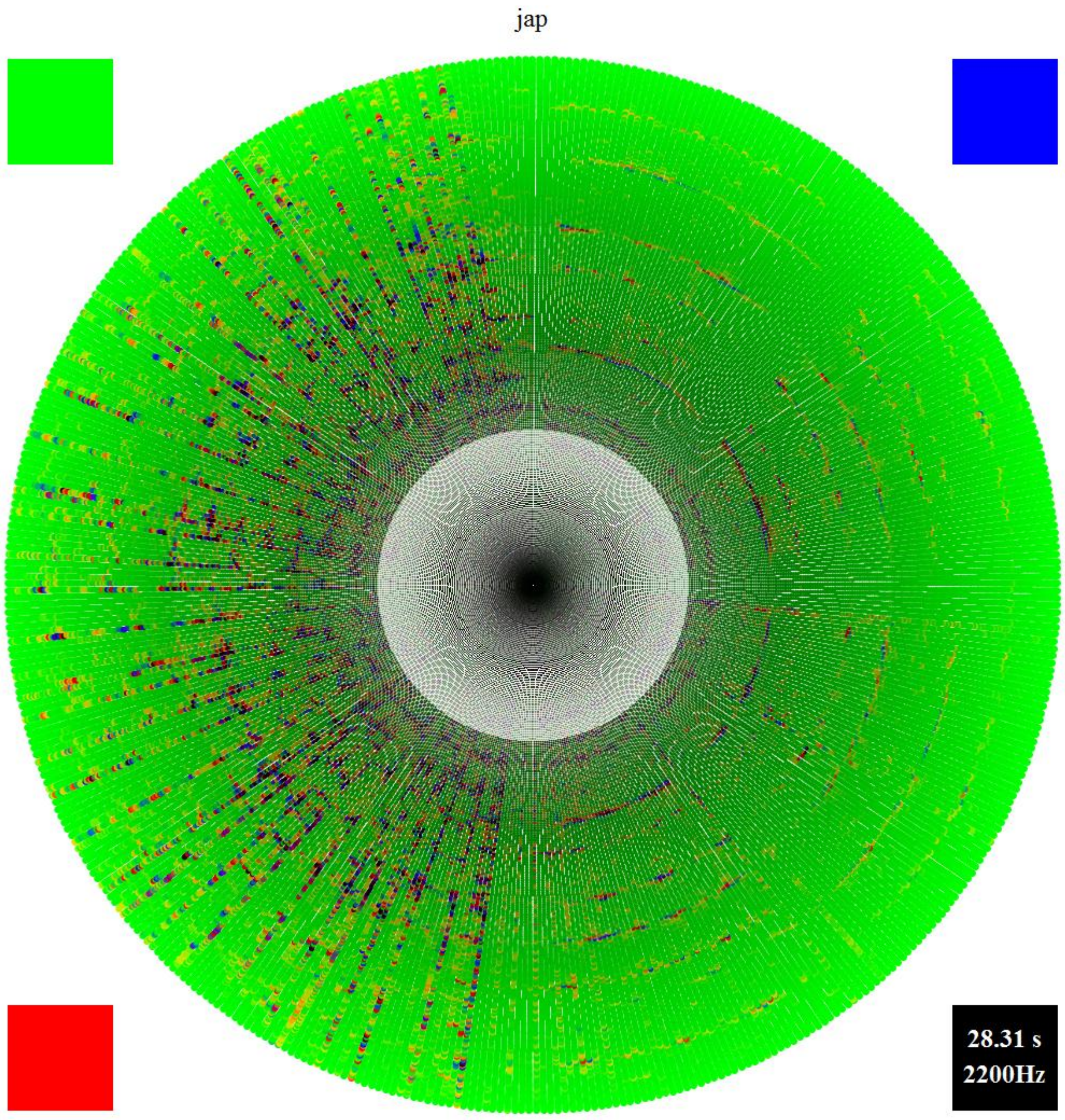
Uniformed structure as it’s a classical music. Pure tones of piano.

Super Mario (harmonica)



A vivid mood as it’s a game background music. The frequencies distributes continuously due to the bending skill when blowing the harmonica.

Chromatic Wind (multi instruments)



Shows the existence of different instruments. The beginning is pure and light while the later part becomes busy and strong.