Analogies Between Loudness, Pitch and Color on LEDs

James Wenzel

Music 108

Abstract

Live music is often much more than just music - people often go to "see shows" rather than "listen to a concert." Live performances are often accompanied by lights and images. Often, these visuals are either rigidly scripted and controlled or performed by a lighting designer or “VJs.” In contrast, audio-reactive visuals will react differently each time, given a live stimulus. This paper explores components of audio-reactive visuals that will be explored using a modular LED screen (32x16 WS2812b LEDs) and code written in Processing. The Fast Fourier Transform (FFT), which can break down a signal into individual waveform components, can provide a computer with enough information to react “intelligently” to a signal. This information derived from FFT can be used to control parametric visuals. Because vision and hearing are both senses that respond to energy, analogies between pitch and color can be drawn, as well as brightness and loudness. Just as our eyes perceive light with different wavelengths as different colors, we perceive different frequencies of sound as different pitches. Similarly, our eyes are more sensitive to particular wavelengths of light, just as our ears are more sensitive to particular frequencies of sound. Drawing an analogy between the two can lead to the basis for more congruent sound and lighting visuals.

Analogies Between Loudness, Pitch and Color on LEDs

To explore audio-reactive visuals, I chose to work with LEDs, because they are extremely bright and relatively cheap. They make a good option for an awesome display at any event without costing a fortune. I have seen several different artists perform using some sort of LED screens. I was able to glean the necessary knowledge to build my own 32x16 matrix using a guide by Hans Lindaur, who built the wall for a band called STRFKR, which was actually my first exposure to an LED screen at a show (Lindaur, 2012). Some technical details follow. The matrix is made up of WS2812b LED strips controlled by a Teensy 3.1 microcontroller and an OctoWS2811 Adaptor. The Teensy 3.1 is running an open-source OctoWS2811 library designed to control these types of LEDs. The LEDs receive input from two outputs from the Teensy 3.1, powering 256 LEDs each. Patches written in Processing, an open-source visual-programming language based on Java, power the board. Processing will need to be installed to run the sketches included.

Pitch, as we learned in our class, is not a physical property inherent to sound. It is our perception of a particular frequency, and further, we are more sensitive to particular frequencies than others. That is to say, the loudness of a pitch is also a perception, and not perfectly related to its intensity (a physical property measured in decibels). As we noted in our class, we tend to perceive the range of 3-5kHz as louder than all other frequencies, and frequencies towards either extreme of our hearing range (20-20,000 Hz) require more intensity to be perceived at equal loudness. This is illustrated well by the Fletcher-Munson curve.

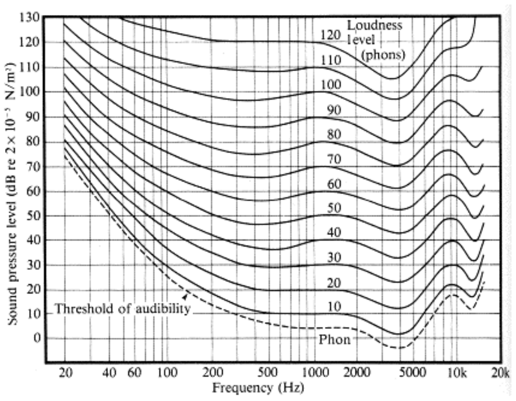

Figure 1. Fletcher-Munson Curves. From "Studio Tone vs. Live Tone and the Fletcher Munson Curve." Retrieved 15 December 2015. http://line6.com/support/page/kb/_/general-faq/studio-tone-vs-live-tone-and-the-fletcher-muns-r448

Since live music is often accompanied by visuals, we should explore the sense of sight as well. Sight, like sound, is our perception of energy, and our perceptions are not necessarily entirely indicative of the properties of the stimuli. Due to the nature of our rods and cones, our eyes are more sensitive to certain wavelengths of lights than others. There is a peak in our visual sensitivity around the 555nm range (a bright green color), and wavelengths on the extremes of visibility (red and violet) are seen as less bright. This works out very neatly for drawing an analogy between color and sound, and allows us to map a crude function between the two.

Since pitch is a logarithmic scale (every octave is doubled frequency), I first converted frequency to relative octave using the equation $f(x) = log_2(x) - log_2(20)$. This defines x as an octave above the base 20Hz, the threshold of our hearing. Using Lagrange polynomial interpolation, I was able to choose 4 points: one for our lowest visible wavelength (red), our highest (violet), the peak (555nm), and another semi-arbitrary point (around 532nm and 4kHz) so that the resulting polynomial was monotonically descending. If the function were not monotonically descending, the analogy would fail, as it would not be a 1:1 mapping of wavelength to frequency. The equation worked out to be approximately $g(x) = -.41518x^3 + 3.90566x^2 - 16.3385x + 633$, where x is octave above 20Hz, and 633 is approximately red light's wavelength in nm.

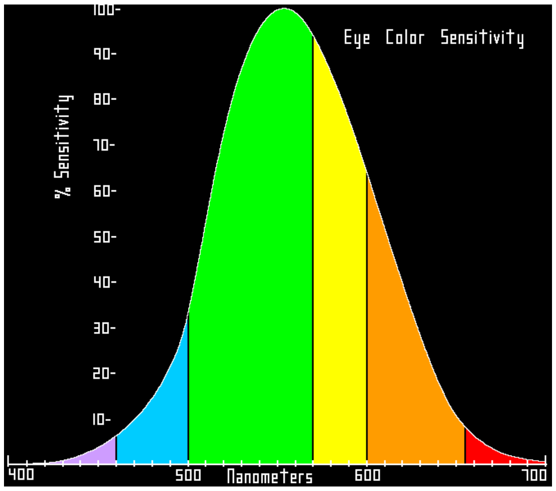

Figure 2. Eye Color Sensitivity by wavelength and color. From "Human Eye Color Sensitivity & Perception." Retrieved 15 December 2015. From http://www.astronomyassociation.org/atlas/at/ot/othcs.html

It should be noted that there are multiple ways to approach displaying color on computers – as components of red, green, and blue light (RGB - as a cube), or as hue, saturation, and brightness (HSB - as a cylinder or cone). Hue, in a few words, can be treated linearly, and thus mapped directly as a function of wavelength. At the level of simply drawing an analogy between pitch and color, we are not concerned with saturation or brightness. Whereas RGB would require three separate functions for each component, hue only requires one. I used a crude mapping of wavelength to hue, and used Java and Processing's built-in functions to convert that to data the computer screen or LEDs needed (Roberson, 2011).

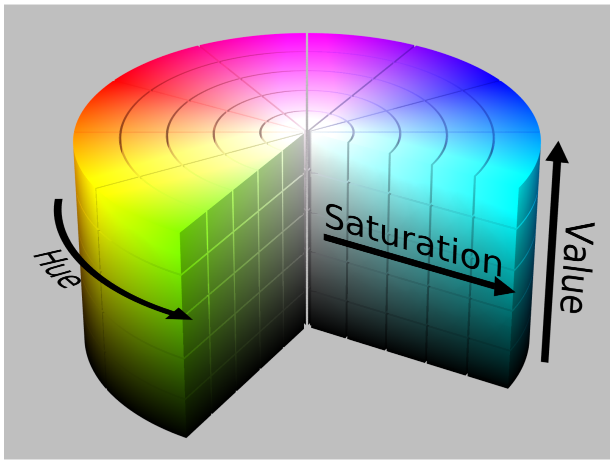

Figure 3. HSB as a cylinder. From "HSL and HSV." Retrieved 15 December 2015. From https://en.wikipedia.org/wiki/HSL_and_HSV

Before discussing loudness, it's necessary first to understand how we are able to get the frequency and pitch information into the computer in the first place. The computer program has some sort of digital input, whether it's a microphone or a pre-recorded audio file. This alone is technically enough to give us a value for loudness, since it's easy enough to read the values of the individual samples. But in terms of pitch, these values don't tell the computer anything particularly useful – so we use the Fourier Transform on a range of these values. Put simply, the Fourier Transform breaks the signal down into its periodic components (like sound waves) – which can be analyzed to see how much of a particular frequency is present. Manipulating this data, the computer can infer something similar to pitch.

One of the included sketches maps mean frequency of a logarithmic sine sweep to color, using the above methods in the folder Sin_Sweep_Color. For all sketches, a plain-Processing example is included, as well as an adapted sketch wrapped in the code necessary to power the LEDs. The LED specific sketch (those that start with “Wall_”) will not run without an attached Teensy 3.1 running the VideoDisplay sketch.

Intensity (which is perceived as loudness) can be mapped to brightness in Processing by summing the contents of the Fourier Transform, and using the ratio of that sum and some scale to determine the color’s brightness component using processing’s hue selector. The analogy between louder and brighter works in the same regard, both measure energy concentration in the signal. The included sketch that illustrates this is in the Loudness_Brightness folder.

To map color to more complex signals than just sine sweeps, we can draw from the concept of a spectral centroid – the weighted average of all of the frequency components in the signal as determined by the Fourier Transform. Most signals, especially instruments or songs, are more than a single sine wave. They are rich in layers and overtones, and the weights of these determine the spectral centroid. This relates to the timbre of a sound, which is another perceptual concept – it is the unique “color” that we hear. Timbre is part of the context that helps us know the difference between a trombone and a cello. Using the spectral centroid and mapping that to color, we can illustrate how the timbre of an instrument or song evolves over time. An example of mapping evolving timbre to color is included in the Evolving_Timbre_Color folder. To switch between the synth and trombone samples, code will need to be commented or uncommented in the setup function. A similar experiment with regards to the average frequencies in a song is include din the EQ_and_Spectral_Color folder.

Overall, these concepts and mappings provide the basis for projects that explore the relationship between sound and light in artistic ways. The analogies drawn between the senses provide a new depth for art and expression, especially in a unique live setting. The Fourier

Transform provides the interface between computer and sound – allowing our programs to "perceive" things like pitch and loudness. The project folder can be found at the following link: https://drive.google.com/folderview?id=0B34e11TWCV9wSjE0bHJkYWR6NG8&usp=sharing

References

H Lindaur. (2012, Sep 23). LED Video Wall. [Web log post]. Retrieved from http://dorkbotpdx.org/blog/armatronix/led_video_wall

W Robeson. (2011, Sep 29). Color wave length and hue. [Website comment]. Retreived from http://www.mathworks.com/matlabcentral/answers/17011-color-wave-length-and-hue