Tile Tunes — Final Prototype

Melissa Avila
Cornell University
Ithaca, USA
ma796@cornell.edu

Alan Lee
Cornell University
Ithaca, USA
aml326@cornell.edu

Meredith Young-Ng
Cornell University
Ithaca, USA
mjy35@cornell.edu

DESIGN GOALS

We live in a world where there are hundreds of instruments—ways one can physically play and create music. However, there are also hundreds of ways music can exist as ideas; one can arrange and compose this music, through concepts written on paper to files generated on computer software. Composition is a critical component of musical expression. Music tracks often begin this way, and only later are recorded with live instruments. This process of arranging music, however, is noticeably more removed from the physical realm of actually playing the music on an instrument. In addition, music visualization is disconnected from ways to tangibly interface with music. Our overarching goal is to bridge this gap between musical ideas and the tangible world using physical computing, while also doing so in a fun, visual manner that allows for freedom of artistic expression.

To do so, we created a device that can allow for music to be arranged *physically*. Rather than rely on a typical computer interface or hand drawing notes, such a device involves modular musical tiles that represent various notes. These physical tiles can be easily arranged and moved around along a user drawn track (two black lines). The freedom of defining the track facilitates artistic expression; the other features work in tandem to allow the user to rapidly and physically interface with musical concepts. A song can be implemented with the tiles, and then played back by a robot reading them as it drives along the track. Changes can then be quickly made by adding, removing, or changing existing tiles.

The goal for the implementation of these tiles requires the association of color with musical notes as well as a track to place the tiles on. The physical tiles could have a number of different design implementations ranging from thin colored blocks to magnets on a whiteboard. The track would be a line or set of black lines defined by the user that the robot would track with IR sensors, independent of the colors on the track. A robot driving over the tiles along a track can use a color sensor to identify these colors and reference a predetermined mapping of that color to a certain note on the musical scale. This mapping would seek to map lower frequency notes to lower frequency colors and higher frequency notes to higher frequency colors. The robot would then play this note on a speaker. It is a key goal of our design for this audio output to be complex with MIDI or MP3 output rather than a tonal buzzing sound. To further aid in the visualization of this music and to add some complexity, our last design goal is to implement an OLED screen on top of the robot with a live display of the notes being detected by the robot. This may help with the overall understanding of our project and is an active way of showing what the robot is seeing and doing. It is our hope that these features combined will create a modular, tangible, and unique music arrangement experience.

RELATED WORK

Our work is primarily inspired by Yuri Suzuki's Colour Chaser project and Specdrums, a wearable ring device that converts colors into different musical notes. Our work focuses on the intersection of the broader research areas of physical music composition and intelligent physical systems.

COLOUR CHASER

Able to detect colored lines, Suzuki's small rectangular Colour Chaser vehicle follows a black line track circuit, simultaneously translating the other non-black RGB-colored lines drawn on top of the circuit into different musical pitches [1]. This design enables users to output pitches of different lengths with an adjustable track, but can be further improved by implementing chord functionality as well as an undo function to allow for pitch and pitch length modifications. Similarly to the Colour Chaser, our design seeks to convert color into sound through a moving robot vehicle that detects colored note tiles and outputs musical notes, with an adjustable track length and design. Our design improves upon the Colour Chaser through enhanced and more complex audio output, the addition of musical tiles as a replacement for colored lines, and the visual display of what is being interpreted. The user is able to modify their composition by simply adding or removing these musical note tiles.

SPECDRUMS

Specdrums incorporate the concept of translating color to sound through the use of wearable ring devices connected to a phone app [2]. While still in the developmental stage (not yet sold to the public), Specdrums offers a wide variety of features, including sound sampling (note that sounds do not have to necessarily be musical notes), re-mapping of colors to sounds, and chord modification. In addition, Specdrums can be connected to external music software (i.e. GarageBand, Ableton Live) via Bluetooth MIDI; however, our work seeks to remove these external dependencies through the usage of our colored musical tiles. Our focus is also merely tangentially related, as we seek to provide a more hardware-oriented alternative with our vehicle, track, and musical tiles for color to sound conversion. Our hardware alternative also allows for a user to physically edit a track after it's been made and play the entire thing back again with the new modifications.

EYEMUSIC & CHROMESTHESIA: MUSIC FROM COLOR

Possible applications of generating music from color include EyeMusic, a sensory-substitution device that converts visual shape and color information into different associated instruments (particularly music) for the blind [3]. Our design builds upon this concept by associating an MIDI ocarina with our tile color spectrum. The idea of incorporating color and sound also relates to the research behind synesthesia, which includes the phenomenon where people "hear" colors based on certain music or notes [4]. This particular phenomenon is more commonly known as chromesthesia, as different tempos and moods of a piece affect many listeners' visual perception of the piece through color [5].

MUSIC SIGHT READER

There is a music sight reading device that allows the user to slowly scan across a page of sheet music and have the device identify the note on the page and play it, effectively sight reading the sheet music for the user. Such a device requires formal sheet music, however, which requires a basic understanding of music theory to make. Our robot would hopefully circumvent such a need through intuitive colored tiles [6].

SoundTRACK

Designed as a wooden children's toy train that generates music based on the arrangement of the track pins, SoundTRACK uses the track as an reconfigurable music composing tool [7]. Our robot seeks to build upon this concept of a reconfigurable track with additional complexity through incorporating color.

NOVATION MUSIC LAUNCHPAD

Created as a physical extension of the Ableton Live software, the Launchpad has the capability to play a variety of notes, sounds, and videos through colored buttons arranged in a square pad [8]. We seek to associate colors with certain musical actions (more specifically, notes) in our design, without any external software dependencies.

NINTENDO LABO PIANO

The Nintendo Labo Piano is a small piano that you can construct out of cardboard that interfaces with the Nintendo Switch to play music [9]. Though it is not meant for track arrangement, the piano serves as a bridge between the physical world and music; the simplicity of constructing an instrument out of cardboard goes hand in hand with the simplicity of our device being able to generate music from mere plastic tiles.

DROPMIX

The Dropmix music game [10] shares the concept of generating tracks by combining and arranging predefined physical objects. In the case of Dropmix, different cards represent different music samples. Up to 5 cards can be combined to make a mix. This allows for less freedom for creativity and expression, however, since you can only generate tracks preloaded by the game.

MOGEES

Through usage of a Mogees sensor, users can transform everyday objects into new music and sounds [11]. Users can personalize their experience through modifying their sensor detection input to a specific default music/sound recording. Similarly to Mogees, our work seeks to detect different colored objects (musical tiles) and transform them into generated music.

FL STUDIO

Ultimately, one must acknowledge influences of track arranging software to our physical track arranger. Programs like FL Studio have become increasingly popular [12] as ways to express music with digital notes and symbols as well as track layering. Though you can do much more in FL Studio to edit and create sounds, our goal is to recreate part of the arrangement experience in software such as FL Studio without requiring a computer interface.

DESIGN: FEATURES, ITERATIONS, AND CHALLENGES

Note: Schematics and Videos of each iteration of our robot are available in the appendix.

Our original idea for this project, shown in Figure 1, was to have a robot that would read colored tiles along a fixed-width expandable railed track. Each colored tile would correspond to a different musical note which would be interpreted and processed in real-time and then be displayed on an OLED display and played by a speaker connected to an audio breakout board.

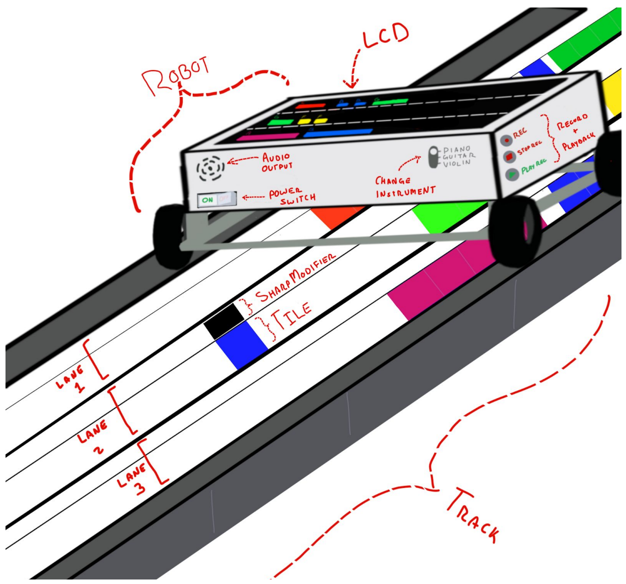

Figure 1. Initial design of the robot

However, each feature of our design has evolved tremendously over the course of the last few weeks, with design descriptions and iterations described in detail in the following sections. Our final design is ultimately a line-tracking tracking robot which can interpret one colored sticker (representing notes) at a time which is then displayed on an OLED screen and outputted as a MIDI note through an on-board speaker. The robot is shown in Figures 2 and 3.

The functional design of our robot can be separated into four primary groups: movement, interpretation, output, and frame, as well as six sub components: line tracking, colored tiles, motor control, color sensing, OLED output, and audio output.

Our general robot design consists of an acrylic three-tiered box frame. The main box-frame (the bottom tier) houses two motors (with wheels) on each side at the front, two ball casters at each side of the back of the box (parallel to the motors), two 5V portable USB battery packs, and a color sensor wedged between two IR transmitter/receivers (all facing toward the ground) the front pane. Cutouts and brackets allow for the secure mounting of the IR and color sensors in the front of the robot as well as the ball casters in the back. On top of the first-tier plane is the RedBear Duo processor responsible for motor control, line tracking, and motor speed. It should be noted that there is also a 9v battery on this tier as well as two small breadboards here separated by an isolator for the 9v components driving the motors. Raised by 4 acrylic pillars glued onto the main frame of the robot is another level/layer of acrylic which holds an Adafruit Feather M4 microcontroller, responsible for the control of the color sensor, OLED display, and audio breakout board. Also on this layer is the mini speaker, connected to an AUX output for the audio breakout board. The last tier (the topmost plane) is by far the smallest and holds the OLED screen and a potentiometer which adjusts the motor speed.

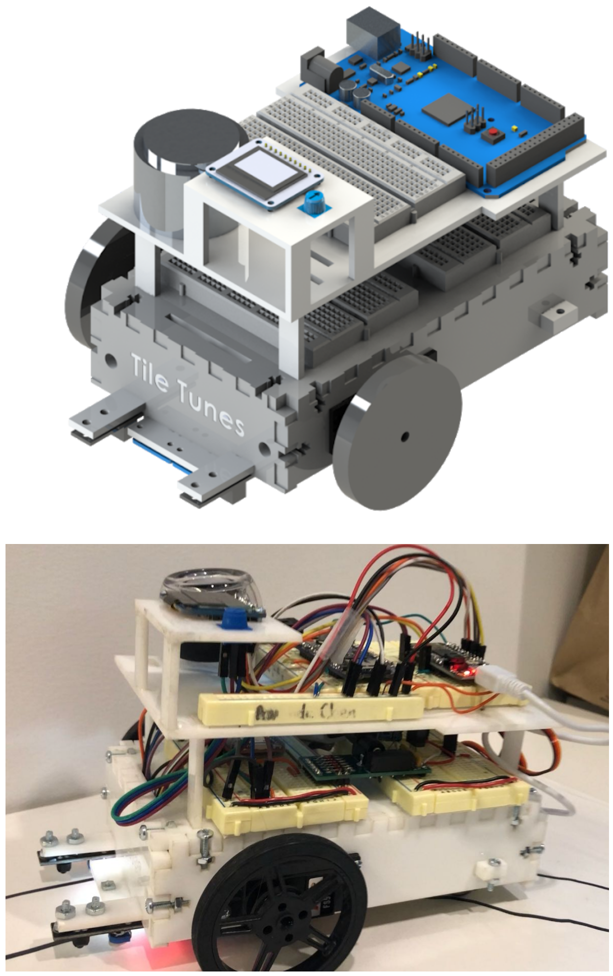

Figure 2. Final design of our robot in CAD (above) and reality (below)

We decided to use two microcontrollers, the Adafruit Feather in addition to the RedBear, to solve latency issues with the OLED display and the audio output, the difficulty of porting the audio breakout code to the RedBear, and a lack of pins. When all components were on a single device, the responsivity to color was very low and the robot would often miss several notes. While during the design of our final iteration we did attempt to use the Arduino Mega rather than the Adafruit Feather, we were ultimately given no choice but to use the Feather because the Mega was almost three times as slower and would cause large delays in displays and audio output.

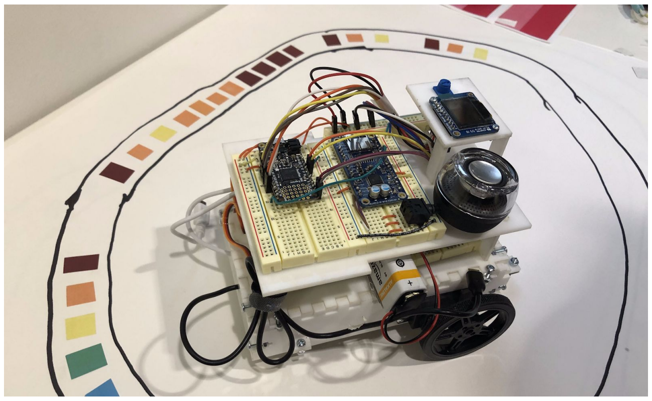

Figure 3. Final design of our robot, three-quarter view

<u>MOVEMENT</u>

Line Tracking: Originally, our robot was not intended to perform any line tracking, but rather, move along a fixed track. However, for added complexity and to provide users greater freedom in terms of track design and movement, we moved to a line tracking system for our second and final iteration.

Between the second and third iteration, we changed from single-line tracking to double line tracking. We created a custom 3-D printed part (Figure 4) that would hold two Sharpies and allow users to easily create a double-lined track using the printed part. The decision to change to a double-lined track came from a desire to minimize erratic movements and the slow edge-following that came from using the two IR sensors, moderately spaced apart, (sandwiching the color sensors) moving along a single line.

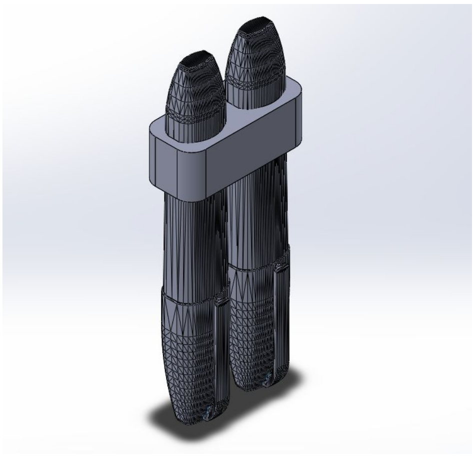

Figure 4. The sharpies inside the holding device.

Currently, line tracking works by following three basic conditions: if the left sensor detects a line (indicating that the robot is drifting too far to the right), then the robot should move left; similarly, if the right sensor detects the line, the robot should move right. Finally, if a line is detected by both or neither sensor, then the robot should continue to move forward. To account for issues regarding the line being obscured by tiles on tight turns (causing the robot to lose track of the black line and thus veer off course), in our second iteration, we decided to add black lines to the center of the colored tiles. This way, the robot can continue to track the line while also identifying the colored tiles. This problem was eventually solved in our third iteration by moving to the double-lined track with the colored tiles between the lines of the track.

The last significant design change to line tracking was the programming addition of interrupts to increase responsivity and movement. There was a noticeable decrease in how often the robot accidentally moved off course after the incorporation of interrupts, as well as the reaction time of our line tracking as our program did not have to complete all of its processes before interpreting information from the IR sensors (polling).

Motor Control: Our movement is controlled by two continuous rotation servos placed at the front of our robot. Our biggest challenges with motor control have stemmed from inconsistencies between motor speeds due to hardware differences and changes in each servo's zero-point (the servo value at which the servo is at a full-stop). There seemed to be a hardware issue in one of our servos because we constantly had to manually adjust it's zero-point in order for it to match the speed of the other servo. Ultimately, we designed our code such that speed and zero-point adjustments could be handled more easily, without the need for manual hardware changes. In the event that a manual hardware change is needed, holes were implemented in the final iteration to allow a screwdriver to easily be placed through the front of the robot into the motor to physically adjust the zero-point.

We had originally planned to use an H bridge motor controller to facilitate motor movement, however, given that it never arrived, we implemented motor controls individually for each wheel, and had to invert and manually adjust our values when necessary (because the motor control and wiring are reflected on the opposite sides).

From the initial iteration to our last iteration, the only changes our motors and motor controls underwent were slight adjustments to their speed and controls.

<u>INTERPRETATION</u>

Colored Tiles: The color sensing components are likely the aspects of our design that have evolved the most between each iteration. Our original design featured a single color sensor interpreting colored tiles along a fixed track. After our decision to implement a freely-drawn line tracking robot, we intended for our color sensors to detect square-cut colored magnetic tiles while moving on top of a magnetic whiteboard. However, this design proved challenging in that in order to improve the accuracy of our line tracking and color sensing, our IR and color sensors had to be extremely close to the ground, causing them to scrape against the raised magnetic strips. Moreover, our ball caster would get stuck on the raised tiles and derail our line tracking.

Thus, our second iteration prototype featured taped strips of paper on cardboard, rather than magnetic strips. We drew a single-lined tracking using sharpie and added black lines to our colored tiles to prevent the IR sensors' misinterpretation of the tiles as black lines (which would cause the robot to move off the track). However, having paper taped down on the floor was problematic in terms of reusability (the colored paper would rip upon removal) and restricted a user's ability to make quick changes to their track.

Consequently, for our final design we used label paper as it was easy to print on, stuck flat to the ground, was sticky enough to be reused, and did not rip as easily on removal. The label paper provided us with the flexibility of the magnets (our original idea) and the reliability/accuracy of the taped paper. Since the final iteration features a two lined track, color tiles could simple be placed in between the lines without need to make any additional modifications.

Color Sensing: As our robot moves along the track, the TCS34725 color sensor is continuously attempting to interpret colors, with the Adafruit Feather M4 Express microcontroller checking if the colors satisfy one of the predetermined HSL (Hue Saturation Lightness) ranges (based on our colored tiles), mapped to musical notes. From our first iteration to our last, there were not many significant changes in our method of color sensing, with the exception of the issues we encountered regarding the colored tiles. The small changes we did make for our final prototype mostly involved adjusting the color sensors to the correct height from the ground to maximize accuracy, and refining the HSL ranges (shown in Table 1) in which different detected colors fall into.

For our last iteration, we attempted to implement a second color sensor, which would have allowed us to interpret two notes, placed side by side, simultaneously. Unfortunately, hardware wise, integrating the second I^2C port proved problematic as the Feather only had one I^2C port available and there was no compatible software online available to convert any of the digital pins to I^2C. While we did implement our own library to convert two GPIO pins to I^2C, this library caused SERCOM conflicts which crashed our audio breakout, making note interpretation impossible. This library can be modified with a different SERCOM to allow for full integration; however, due to time constraints and hardware limitations, we only mounted one color sensor as we wanted to focus on adjusting this color sensor for maximum accuracy of colored tile reading.

When colors are interpreted by the color sensor, the values are converted to HSL. If an HSL value falls within a certain saturation and lightness range, then the code determines color color the hue value can best be interpreted as. Each color value (and hue range) is associated with a specific note as follows:

Table 1. Colors mapped to hue values and musical notes

Color	**Hue (H) Range**	**Musical Note**
Red	0 <= H < 21	C4
Orange	21 <= H < 45	D4
Yellow	45 <= H < 76	E4
Green	76 <= H < 159	F4
Cyan	159 <= H < 200	G4
Blue	200 <= H < 230	A4
Purple	230 <= H < 293	B4
Pink	293 <= H < 361	C5

As we purchased a relatively cheap color sensor, we were limited in the reliability of the interpreted colors. Consequently, our output would sometimes be inaccurate and inconsistent between notes of the same color due to slight differences in lighting on different sections of our tracks. We frequently had issues with the interpretation of red, orange, and pink and tried many different variations of each color until we found a physical variation of each that worked the most reliably with the color sensing.

OUTPUT

OLED Output: Visual output of interpreted notes was achieved using an Adafruit 16-bit OLED breakout board. After colors were processed into musical notes, we sent the respective note letter information to the display and the OLED library performed the work of turning the string object into pixels on the display. Furthermore, for visual consistency between tiles and notes on the screen, we colored the the letter on the OLED to the same color as the tile it interpreted.

The OLED component is one of the components that changed the least between iterations. While we prepared the code to work with two distinct interpreted notes on the

OLED screen, after the removal of the second color sensor due to aforementioned issues between the second color sensor and the audio breakout, we reverted back to our original OLED design.

Audio Output: We originally designed audio to be processed by an audio breakout board (the VS1053 Codec), and then streamed to an external bluetooth speaker. However, for our initial and second iterations we encountered complications integrating our audio breakout board with the RedBear due to SPI space as well as software issues: (1) there was not enough room for SPI control on our RedBear for integration of both the Codec and OLED display, and (2) there was little to no software or technical support online for the codec. Porting the Arduino code for compatibility with the RedBear is extremely difficult; we deemed it not a cost-effective use of our time.

For our final iteration, we edited the existing the Arduino audio breakout code for compatibility with the Adafruit Feather to manually drive the reset pin. While we had initially planned to play MP3 files, we switched to MIDI to achieve a crisper sound, have the option of playing multiple notes simultaneously, and because MIDI output did not require SPI, and, as a result, saved us the difficulty of implementing multiple slaves on the SPI bus.

FRAME

Our initial CAD (Figure 5) featured a T-slotted wooden box design, with mounts for the motors at the front and the ball casters at the back. We planned to have all of our boards mounted on the top of our box design.

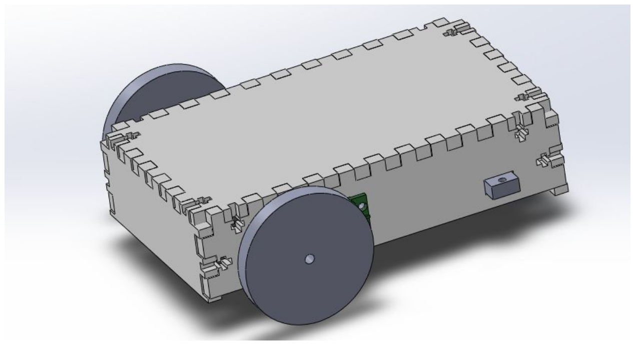

Figure 5. Initial CAD of the robot, three-quarter view

However, after realizing that not all components would fit appropriately on the single plane, we designed a stacked frame for our robot, with the OLED and audio components/breadboard at the top of the stack (near the front of the robot), and all the other components beneath it. We maintained the design of the wheels and ball casters from our original CAD. A significant change we made was the addition of mounts from which the IR sensors and color sensor were screwed into. These mounts offered more stability and security than simply using tape, and prevented wires from accidentally becoming unplugged. The following (Figure 6) is an image of the stacked frame design from the second iteration of our prototype:

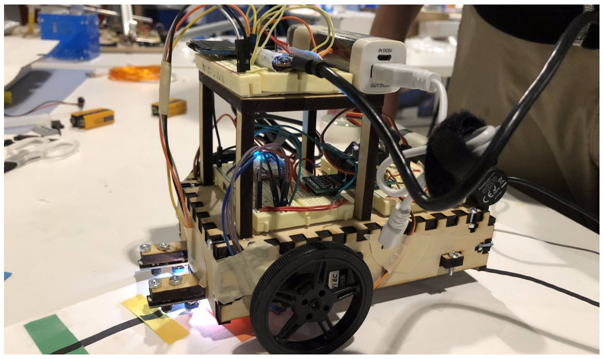

Figure 6. Second iteration of the robot, three-quarter view

For our final iteration, we focused on adding an additional room to the second layer of our frame to house the Adafruit Feather processor and audio breakout board. We also added a small third layer which was a third of the size of our other two layers and solely held the OLED display and speed potentiometer. To minimize the size of our overall robot, we reduced the vertical distance between layers as much as we could, leaving only enough room for the wires to reach different boards and components comfortably. The larger 3.3V batteries which powered our processors were mounted at the bottom of the device (Figure 7) next to our servo motors. We also incorporated slots throughout the robot through which we ran our wires in order to maintain a cleaner and more user-friendly aesthetic. Lastly, for the sake of aesthetic design and joint stability, we changed our frame material from wood to acrylic.

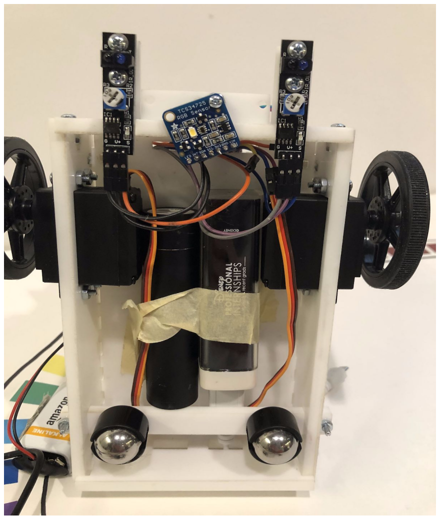

Figure 7. Final design of our robot, underside view

FUTURE PLANS AND REVISIONS

Given time constraints and hardware limitations, we were unable to implement all of our robot design ideas. In terms of revisions we would like to make, we believe that it is possible for a second color sensor to be integrated, but it would require additional modifications to our current custom I^2C bus library. Another option would be to use a different color sensor that allows for a reconfigurable or different slave address. Moreover, the addition of a second color sensor may require further adjustment in the width of our track (and consequently distance between our IR sensors).

In hindsight, and for future reference, we may have been able to create a more robust and accurate musical scale with a higher-quality color sensor. Our current TCS34725 color sensor would sometimes be inconsistent in its readings (displaying the wrong note or failing to read a tile at all), and those readings were highly affected by the speed of our robot. If we had access to a better color sensor, we would be able to incorporate more colors as notes for sharps and flats, and also allow the robot to move and interpret notes at a faster rate.

Our current line tracking code, while sufficient for our design and project, is slightly rudimentary. Following tight turns and multiple paths is a major limitation of our current design and, in the future, our goal would be to resolve both of these issues, with the possibility of adding complex line tracking commands such indicating stops/rests through intersecting lines and including edge detection to prevent the robot from falling off of tables or moving off the paper.

During a demonstration of our design, a couple of audience members commented that our design could possibly be a children's toy. If we had more time and resources, we could potentially turn our device into a child's/music enthusiasts toy, one to rival the Colour Chaser (which currently has many limitations in its controls and its sound). Developing the device into a toy would require a closed-frame design (so that it wouldn't be dangerous to the user, particularly children) and more permanent wire soldering (to prevent unwanted disconnections and breaking).

Lastly, we would have liked to incorporate greater user flexibility in audio output and note interpretation. One cool feature of a competing product, the SpecDrums, is the ability to set custom sounds and change color configuration for notes through a mobile application. We believe that our design would improve immensely from a similar implementation where we could allow users to set the instrument and octave played by the robot, upload custom sounds, and configure what musical note each color corresponds to.

REFERENCES

1. Yuri Suzuki. 2010. Colour Chaser. Retrieved September 16, 2018 from http://yurisuzuki.com/archive/works/colour-chaser/
2. Specdrums. 2018. Specdrums. Retrieved September 16, 2018 from https://www.specdrums.com/
3. Sami Abbouda, Shlomi Hanassya, Shelly Levy-Tzedeka, Shachar Maidenbauma, and Amir Amedi. 2014. EyeMusic: Introducing a "visual" colorful experience for the blind using auditory sensory substitution. *Restorative Neurology and Neuroscience* 32, 247-267.
4. Siri Carpenter. 2001. Everyday fantasia: The world of synesthesia. *American Psychological Association* 32, 3 (March 2001).
5. Stephen E. Palmer. 2015. What Color Is This Song? Retrieved September 16, 2018 from http://nautil.us/issue/26/color/what-color-is-this-song
6. Dan Nasowitz. 2012. I Will Destroy This Robot Sheet-Music Sight-Reader, I Swear I Will. Retrieved September 16, 2018 from https://www.popsci.com/technology/article/2012-11/i-will-destroy-robot-sheet-music-sight-reader-i-swear-i-will
7. Elizabeth Stinson. 2013. A Brilliant Toy Train That Plays Music But Won't Drive Parents Crazy. Retrieved September 16, 2018 from https://www.wired.com/2013/07/a-clever-toy-train-that-play-music/
8. Novation. 2018. Launchpad. Retrieved September 16, 2018 from https://global.novationmusic.com/launch/launchpad
9. Garrett Martin. 2018. The Nintendo Labo Piano Is a Surprisingly Useful Little Synthesizer. Retrieved September 17, 2018 from https://www.pastemagazine.com/articles/2018/05/what-ive-learned-about-the-nintendo-labo-piano.html
10. Hasbro. 2018. Dropmix Music Gaming System. Retrieved September 17, 2018 from https://www.hasbrotoyshop.com/en/htsusa/dropmix--1/-c3410as00?catName=
11. Victoria Turk. 2014. This Gadget Turns Any Object into Electronic Music. Retrieved September 17, 2018 from

https://motherboard.vice.com/en_us/article/vvbe7j/this-gadget-turns-any-object-into-electronic-music

12. Dan Wells. 2016. The Unlikely Rise of FL Studio, The Internet’s Favorite Production Software. Retrieved September 17, 2018 from https://noisey.vice.com/en_us/article/d33xzk/fl-studio-soulja-boy-porter-robinson-madeon-feature

APPENDIX

Code

Link to GitHub: https://github.com/MelissaAvila1/TileTunes

Videos

First Iteration: https://youtu.be/q3bHx5FvDkM

Second Iteration: https://youtu.be/9Ad1SqMqIWo

Final Iteration: https://youtu.be/n2oRfrANPMU

Schematics

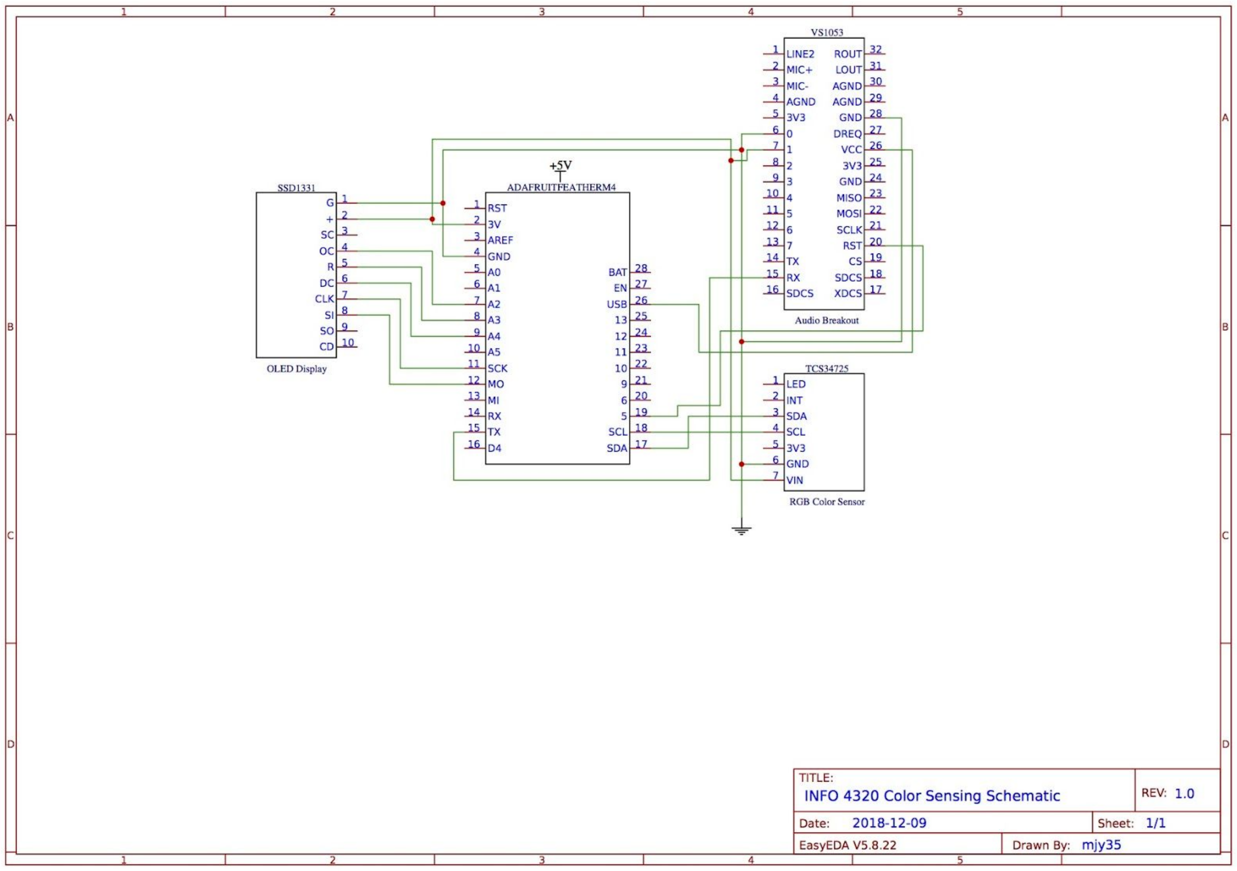

Figure A1. Schematics for Adafruit Feather controlling the color sensor, OLED display, and VS1053 audio breakout

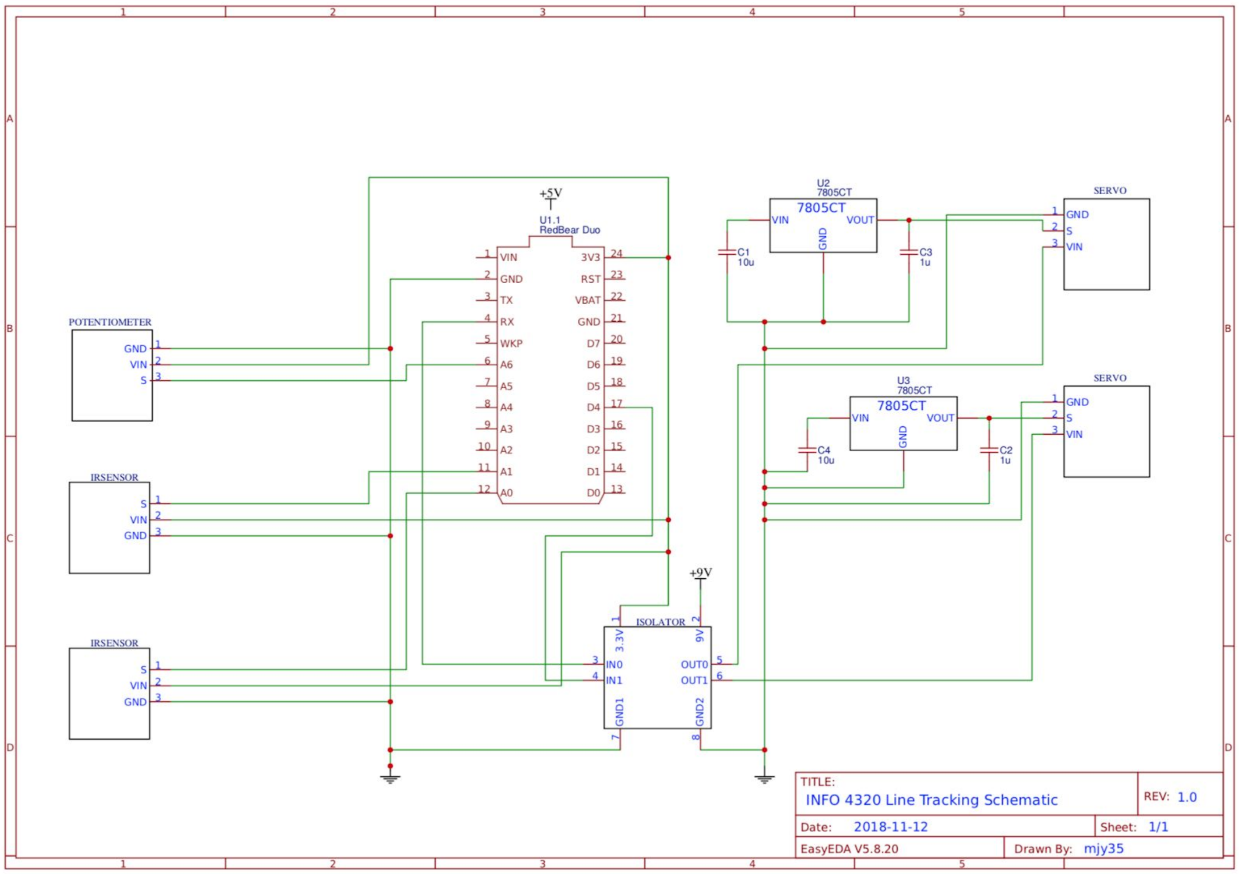

Figure A2. Schematics for Redbear Duo controlling the motors, IR sensors, and and potentiometer