Blind-Aid Report

Cornerstone of Engineering II: GE 1501

Blue Class: MR 11:45 - 13:25

Dave Pleteau

Josie Lee

Pratap Agrawal

Daniel Tento

4.20.2018

Dave Pleteau
pleteau.d@husky.neu.edu

April 20, 2017
Professor Freeman
GE 1502 Blue Class
Northeastern University

Josie Lee
lee.josi@huky.neu.edu

April 20, 2018
Professor Freeman
GE 1502 Blue Class
Northeastern University

Pratap Agrawal
agrawal.prat@husky.neu.edu

April 20, 2018
Professor Freeman
GE 1502 Blue Class
Northeastern University

Daniel Tento
tento.d@husky.neu.edu

April 20, 2018
Professor Freeman
GE 1502 Blue Class
Northeastern University

Dear Dr. Freeman,

Our group decided to address the issue of blind people and how we could assist them in their daily activities at any age. The original problem statement was that we wanted to aid blind people with navigating through their environment. Our original design involved using a series of gyroscopes and accelerometers to create a 2D grid-based positioning system that would let out audio cues and navigate around the user's home. After realizing the difficulty and cost of implementing such a system, the design was scaled down to go on a preprogrammed path that would have to be inputted by a third party to each different house. We quickly realized the impracticality and more importantly, the dangers that obstacles such as stairs could cause. This was the point in the design process where we decided to turn the device into a child-centered robot. We then built the prototype which operated by going to set positions on a playmat created by the team. We finally decided that we should make the robot's movements random and that it should stop only when it detects a specific color based off of a button pressed. The robot would then alert the blind child to its position by letting out an audio cue repeatedly. When building the prototype we narrowed down the original problem statement to aiding blind children in navigating in their later years by fostering a sense of curiosity in them as toddlers. The prototype embodies this statement it that it was designed entirely with a child in mind. A custom playmat was built so that the robot would not have to traverse obstacles such as stairs. The prototype robot was designed to drive along a set path and stop at a certain point once the color sensor detected a certain color. The prototype represents multiple cycles of the design process. Our team realized at different points in the building of the device that some elements we wanted to include would be too expensive, too complicated, or too impractical. This led us to generating more and more idea. We would then see how the device worked and based off its performance, we would adjust different parts of it by using the engineering design process. The main takeaway our team took from this project is that while the process can be used to readjust parts of a product, it can also be used to scale down a product based on the capabilities of the team. Our team simply did not have the skills or resources to build a robot that could create its own positioning grid, so we had to find new alternatives to accomplish the problem we set out to solve. The following pages detail the steps we took to achieve our final design as well as some afterthoughts on the project.

Sincerely,
Team Blind-Aid

Table of Contents

Abstract

The objective of this project was to help the blind kids overcome their disability and not let it hinder their growth and development by training them to function more independently and rely less on their parents for basic things like locomoting. The parent presses a colour button on the robot and it looks for it in the room using its colour sensor and stops and makes a buzzing noise when it detects it. The kid follows the noise and reaches the destination. An object is placed on the destination which the kid picks up and tries to identify on the basis of touch. This helps him associate touch with an object and sound with distance and direction which will help him in the future to estimate the distances of many obstacles around him.

Section 1: Problem Definition

Children who are born blind face multiple obstacles which decrease their sense of individuality and discovery at a young age. These children have difficulties operating in their own home, which instills in them the fear of discovering the world without their parents' aid. This discouragement delays their learning process, taking them off of the general learning curve of a child. This is most apparent when they are at the toddler age as they are less inclined to explore and learn and rely heavily on their parents. Our group proposes a robot toy aimed at encouraging blind children of the age 3 to 6 to embrace individual discovery, associate sound with location, and associate braille with an object and the touch sensation of that object. The largest goal of this robot is to increase blind toddlers' desire to discover, and give them a chance to explore things individually. Of course, for safety, a parent would be with them while playing with this toy, although the child would be inputting the location, listening for the location, and finding the object on their own. They then could use most of their senses and gain a feeling of accomplishment all at the same time. A successful robot will be able to find and report location based on input, so that with further development it would be able to communicate to the user which steps to take in order to reach a location. Our definition of a successful robot changed over the duration of the project, our main goal stayed the same but by using the Duncker Diagram (Figure A) below we were able to determine the most important objectives of our robot:

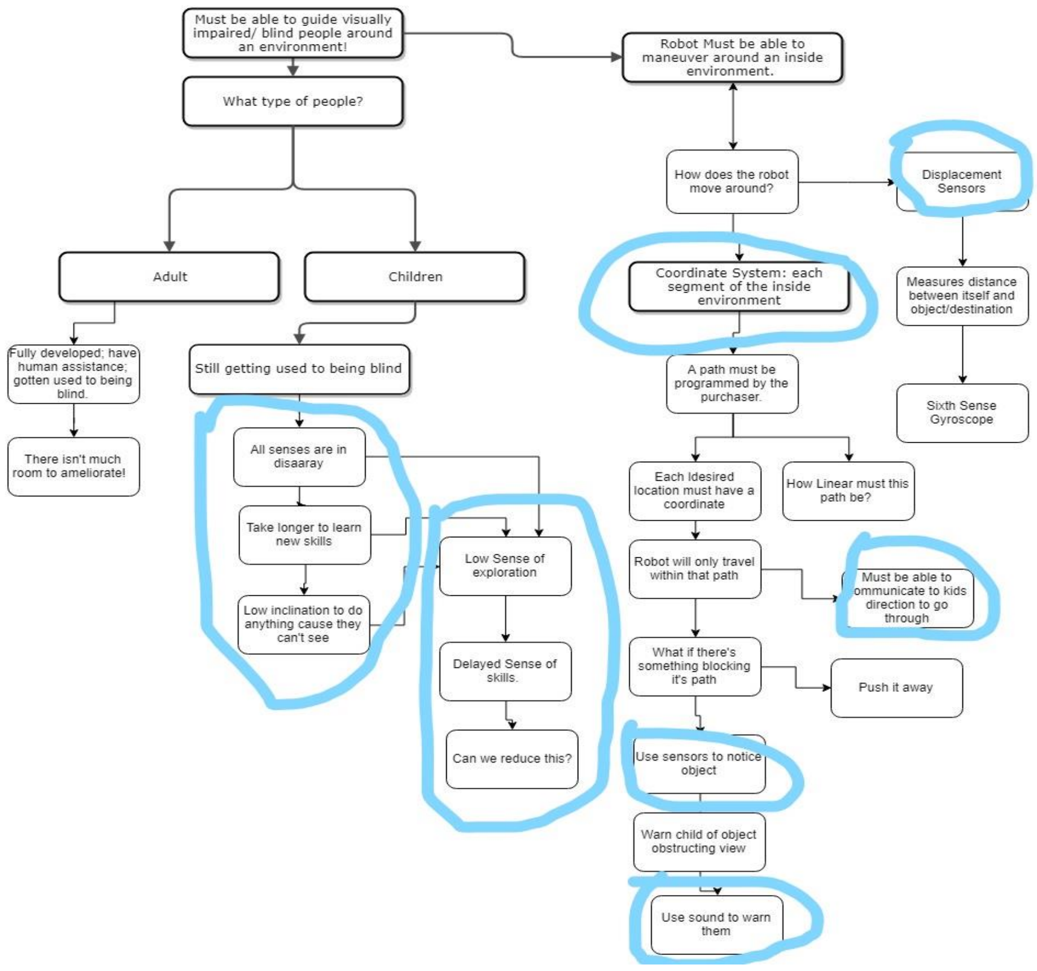

Must be able to guide visually impaired/ blind people around an environment!
What type of people?
Adult
Children
Fully developed; have human assistance; gotten used to being blind.
There isn't much room to ameliorate!
Still getting used to being blind
All senses are in disaaray
Take longer to learn new skills
Low inclination to do anything cause they can't see
Low Sense of exploration
Delayed Sense of skills.
Can we reduce this?
Robot Must be able to maneuver around an inside environment.
How does the robot move around?
Displacement Sensors
Measures distance between itself and object/destination
Sixth Sense Gyroscope
Coordinate System: each segment of the inside environment
A path must be programmed by the purchaser.
Each Idesired location must have a coordinate
How Linear must this path be?
Robot will only travel within that path
Must be able to communicate to kids direction to go through
What if there's something blocking it's path
Push it away
Use sensors to notice object
Warn child of object obstructing view
Use sound to warn them

Figure A: Duncker Diagram

The robot will find a location based off of a remote signal inputted by the user which the user will choose based off of the braille symbols relating to an object they are searching for. The robot will report its destination with sound so that the user can determine the robot's final location, at this location will be the object that the user was searching for(the object pertaining to the braille input). While the robot is moving, it will emit certain notes pertaining to the child’s movement, telling the child which direction it should move in and how many steps to take. The main functions of our robot will include ability to move forwards, backwards, and turn, as well as utilizing the color sensor. Our robot will be able to take a user input and search for a location based off of this input. Locations will be marked with specific colors, the robot will search for the color and then omit a sound upon reaching that color. In order to find the color, our robot will initiate a “search” function, which will consist of a randomized movement sequence, if the robot reaches a color that is not the inputted color, it will initiate a function allowing the robot to backup and turn in order to search in a new direction. The robot should be able to detect the location in roughly 5 seconds, depending on the size of the area it is searching. The larger the area gets, the delay of the randomizer function would dictate the length of time the robot takes. With the average skill in hearing, the user should be able to clearly detect the omitted sound from any location in an average sized room, roughly 330 square feet. The robot should be able to detect each color of the rainbow, allowing the user to start with 7 items. Cosmetically, our robot will contain an outer “shell” of wire which will allow parents to add on personalized animal-like outermost shells for the kids to play with. These outermost shells may also contain braille, and raised shapes, depending on what The parent wants to help teach their child, The constraints of our project include the area our robot will be searching, the way we set up the color course, the accuracy of the color sensor, and the length of time the robot is searching. The lighting in the room is also a constraint on our robot because it slightly alters the way the color sensor detects color, which can send it into a frenzy of recurring code where it is backing up and turning. Power is also a constraint, too much increases accuracy of the sensor and affects the color ranges, while too little does the same. Our budget as well as the size of the robot was a constraint, time was also a constraint for us. With more time we would ideally like to code a remote which would communicate user input to the robot, instead of using buttons. We also would’ve like to add in more colors, and more object for our robot to find. In terms of the coding, we had a list of goals to meet first in order to accomplish the basics of our problem statement. First, we needed to calibrate the motors and create two functions one being search and one being the recoil function where it could back up and turn around. Next, we needed to calibrate the color sensor accurately, and then associate the colors with the movement functions. We started with two colors in order to keep it simple. The next most important part is the user input based off of color, and that the robot would search for and stop on the correct color. Immediately following this piece, it was important that our robot emitted sound.

Technical research that we needed for our robot had to do with a remote and our color sensor. The TCS230 arduino color sensor uses an 8 by 8 array of 16 photodiodes to sense colors(See image 1 and 2 in appendix). There are three different filters on the diodes, clear, blue, green, and red. These diodes have two control pins since they are parallel to one another. Each of these pins can be set to low or high in order to activate certain filters on the diodes. There are two output pins which can be set to different frequencies to help make the sensor more accurate. Example code is displayed where the summary of the sensor is displayed. This arduino color sensor works the best for our purposes because we have an arduino based knowledge of coding for the sensor, and also all of the necessary equipment to connect the sensor. The sensor is only $10.00 to order online, which is effective for our purposes(Nedelkovski/). In order to use our remote, we would need both a remote and a receiver for the project. We looked into using the arduino HX1838 arduino receiver which is only $8.00 to order over amazon, well within our budget. The receiver requires only three wiring techniques, including a ground, signal, and vcc pin (See image 3 in appendix). In order to place the receiver in a visible spot on the robot we would need to adjust the receivers placement on the outer shell of the robots body. From there, there was some example code to be found online that we could use within our project (Pattabiraman).

Section 2: Generating Alternative Designs

In the beginning of the project, our problem statement was different, so most of the design we generated at first was aimed towards a different function, which consisted of moving around an entire house with the child. So for the first portion of our project that we started generating alternative designs for was the code. For instance we thought of utilizing 2 Dimensional arrays and storing positions of an array as X and Y coordinates as seen in figure A. In theory, this sounds simple and effective, since when the robot moves a certain amount in the x direction and y direction, we would know its exact position (it'll start at (0,0)). From our Duncker Diagram, we also noticed we would need a either a displacement or sonar sensor, and an IMU with 6 Degrees of Freedom to accomplish GPS-like movement around an area. From here, we decided to look at what equipment we had that we thought would be most helpful for each of the senses that we were trying to maximize on for the children.

Using the Fishbone Diagram (Figure B) below, we were able to further narrow down qualities that our robot would need to embody for the three senses:

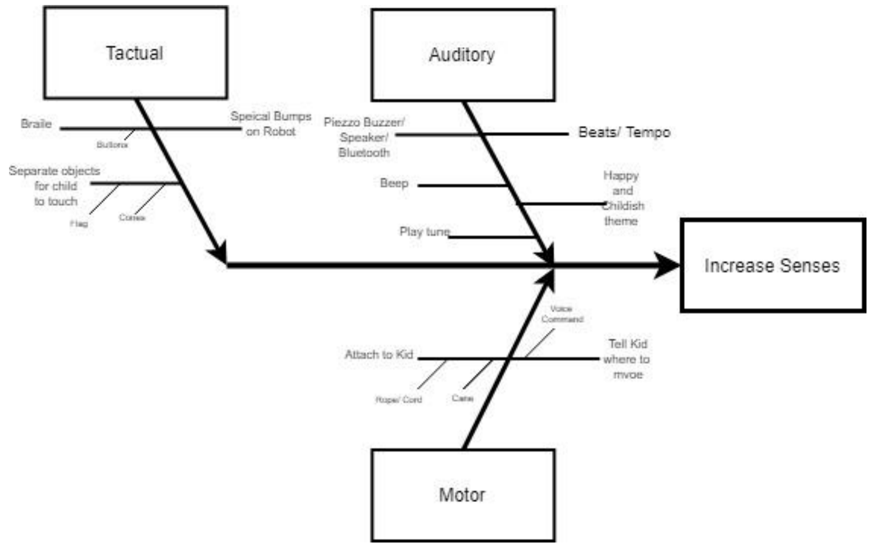


Figure B: Fishbone Diagram of Senses

For auditory, we realized our design would need to incorporate a buzzer of some sort and a segment of our code that will make noise to hear. In addition, beat, tempo, type of music and sound was also brainstormed (Figure B). For motor skills, we thought of attaching a rope or cane to the robot and to the kid to pull them in a designated direction. This also could increase the safety of the product. For tactual senses, we theorized having braille buttons or objects for the child to touch and identify. For how the body of our robot would look, we utilized C-Sketch method. Using this our team designed several body pieces of our robot, to then meet up and decide on what to accept, deny, or modify. These solidworks pieces and sketches can be seen below in figure C:

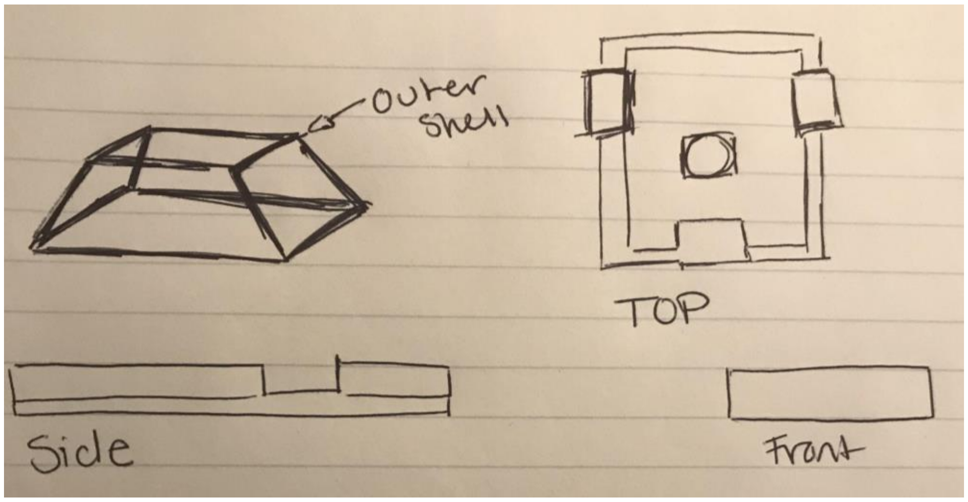


Figure C.1: Sketch of body of robot

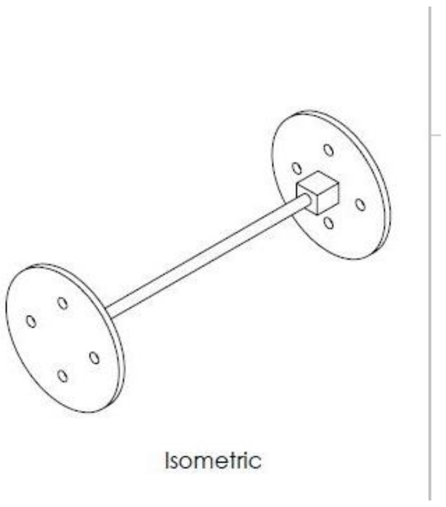


Figure C.2: Solidworks piece of wheel and axle

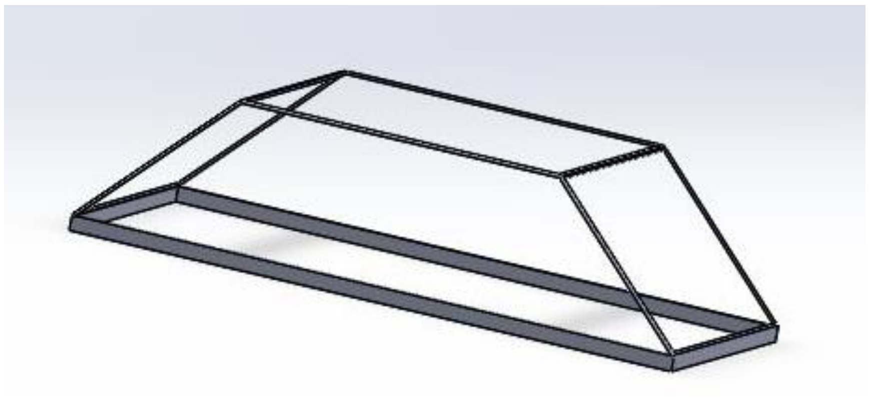


Figure C.3: Solidworks Piece for Outer Shell

After presenting our project proposal to our classmates, a series of criticism came through that had an enormous impact on our project. The first and major criticism we got was in regards to the safety of our product. They explained that there is no real way to guarantee the path of a free roaming robot won't be obstructed by stairs or random items nor are does it have any means of moving the or adjusting to the environment. They emphasized that safety of the user should be upmost. The second most impactful criticism exclaimed was to turn it into a toy. Taking these suggestions, our group decided to reduce our scale of our robot to just being a toy robot.This toy would to still need to complete the functions and goals identified in our problem statement. Therefore, we redefined our problem statement and re assessed what we wanted the maximum function of this robot to be. We developed a Kepner-Tregoe problem and decision analysis was utilized to narrow down what our design will need to embody:

Problem	Timing(H,M,L)	Trend (H,M,L)	Impact(H,M,L)
Get kid to Associate sound with direction	H	H	H
Increase Motor Skills	M	M	L
Get kid to associate touch with object	H	M	H
Keep Child Safe	H	H	H

Figure D: Kepner-Tregoe problem and decision analysis

Looking at at our Kepner-Tregoe problem analysis chart above we realized that ameliorating our users kids auditory (Score: H, H, H) and tactual skills (H, M, H) were more important than motor. This decision was made upon the fact that at a young age, sound is the first and main method of orientation. This gives it prioritization above everything else. Following sound, came the child's protection (soring: H, H, H). The reason why tactual skill won over motor skills is because for a blind child, the earlier they can start associating texture with meaning, the easier it will be for them to rely on touch reading or distinguishing things. All this must be done while warranting the child's safety as seen by its high rating in the K-P problem analysis. Using the Fishbone Diagram(Figure B) and the decision analysis (figure D), a rank ordering matrix was done for three of these problem/decision-- where we combined "increased motor skills" and "keep child safe together".

Goals	Pleasing to the ear	Tune (Iconic)	Tempo/Beat	Loud	Repetitive	Total
Pleasing to the ear		1	1/2	1	1/2	3
Tune (Iconic)	1		1/2	1/2	1	3
Single note noise	0	0		0	0	0
Loud	1	1	1		1	4
Repetitive	1/2	1	0	1/2		2

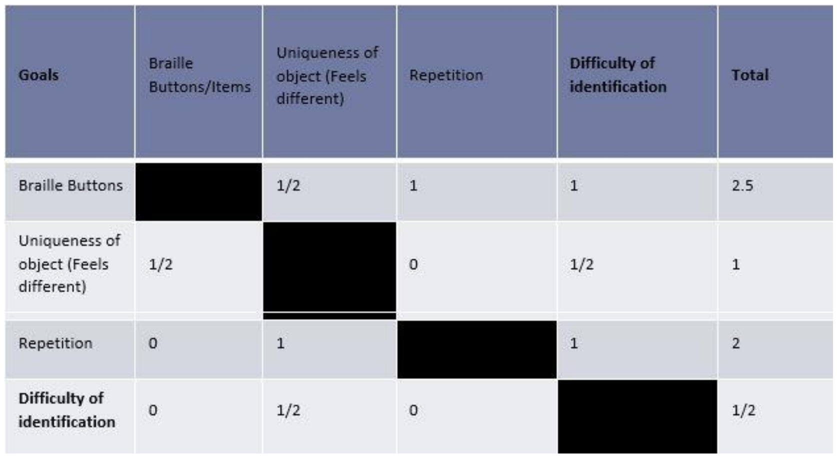
Figure E: Auditory Rank Ordering

Looking at the rankings of the solutions to increasing the auditory sense (Figure E), we can tell that the loudness, and type of tune are the most important aspect to include in our final design.

Goals	Connect robot to user	Pull In correct direction	Comfortable	Designated small Area free obstruction	Supervision Of Adult	Total
Connect robot to user		1/2	0	0	1/2	1
Pull In correct direction	1/2		0	1/2	1/2	1.5
Comfortable	0	0		0	0	0
Designated Area free obstruction	1/2	1/2	1		1/2	2.5
Supervision Of Adult	1	1	1	1/2		3.5

Figure F: Motor and Safety Solution Ranking

As seen from the figure above, it seems that our robot will need to include some sort of supervision from an adult (highest score of 3.5). In addition it looks like having a designated space for the robot toy to move around is also consequential. Looking at figure F, we decided that an attachment to the user is useless.



Goals	Braille Buttons/Items	Uniqueness of object (Feels different)	Repetition	**Difficulty of identification**	**Total**
Braille Buttons		1/2	1	1	2.5
Uniqueness of object (Feels different)	1/2		0	1/2	1
Repetition	0	1		1	2
Difficulty of identification	0	1/2	0		1/2

Figure G: Tactual Solution Rankings

Looking at the ranks, it seems that to elicit stronger identification of objects through touch, we would need to use repetition and braille buttons/items. This must mean our toy must be appealing enough to make the child play it repeatedly until he fully memorizes the object he's touching. Having problem and scale of product narrowed down, we devised a plan on how our toy would work.

Taking inspiration from last semester's Sumo Robot, we create a game that consisted of three segments: find a location based on user input, make noise when reached destination, have chil touch and identify object. Another small duncker diagram (Figure H) was used for each component of our game.

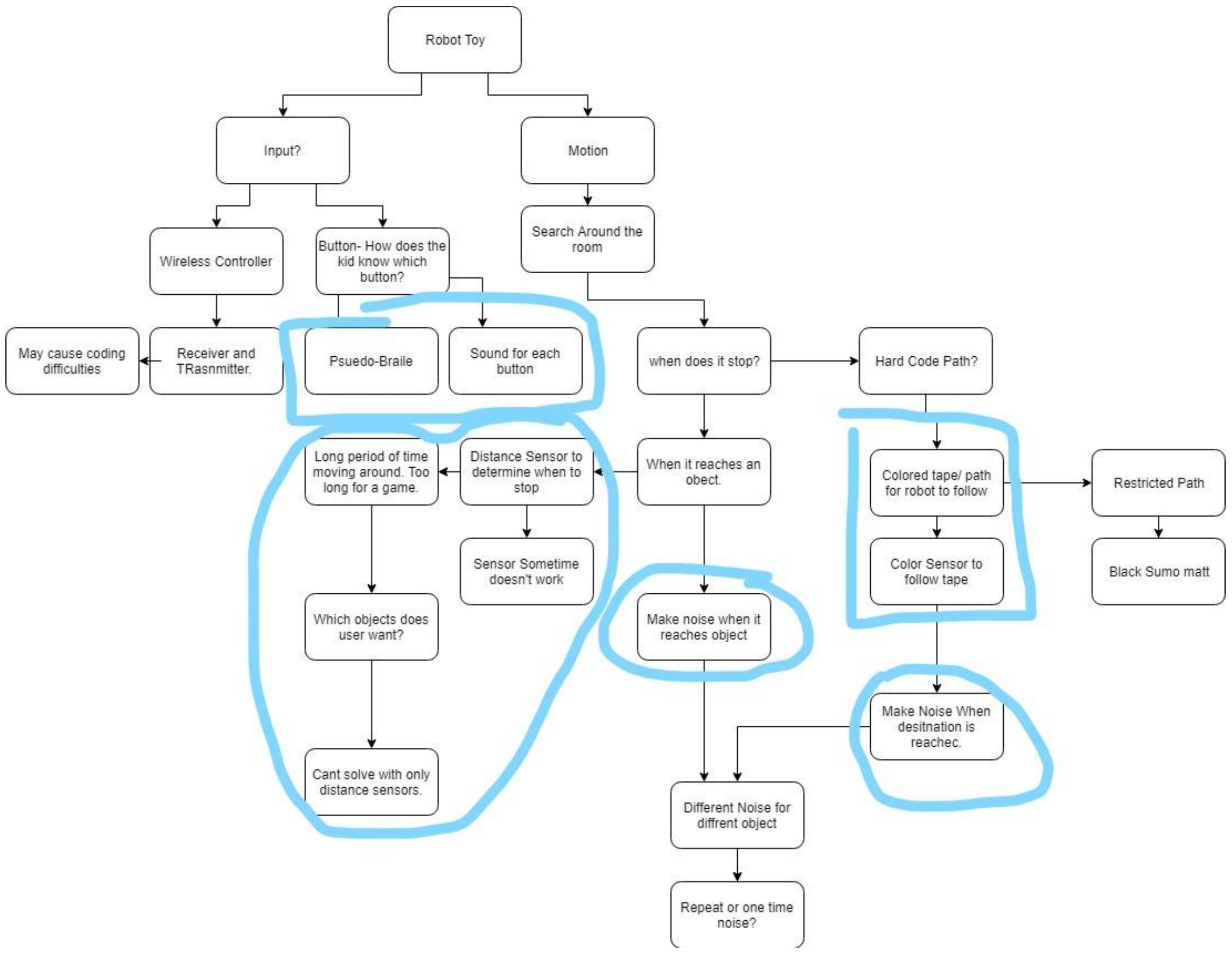

Robot Toy
Input?
Motion
Wireless Controller
Button- How does the kid know which button?
Search Around the room
May cause coding difficulties
Receiver and TRasnmitter.
Psuedo-Braile
Sound for each button
when does it stop?
Hard Code Path?
Long period of time moving around. Too long for a game.
Distance Sensor to determine when to stop
When it reaches an obect.
Colored tape/ path for robot to follow
Restricted Path
Sensor Sometime doesn't work
Color Sensor to follow tape
Black Sumo matt
Which objects does user want?
Make noise when it reaches object
Make Noise When desitnation is reachec.
Cant solve with only distance sensors.
Different Noise for diffrent object
Repeat or one time noise?

Figure H: Duncker Diagram for determining components of game

From our Duncker Diagram, there are a methods to solve each solution that are proposed and developed. For taking in user input, we decided on three buttons that will have little bumps on them represent braille. Once the button was pressed, our code was designed to have our robot search around a set area until reaching the desired location. As seen in the duncker diagram, however, there's no way for the robot to know which object the user wants based off of a distance sensor. We had further developed our idea to include a distance sensor when we realized we wanted the robot to be more of a toy, but the distance sensor would only complicate the function of our robot. We decided scratch the idea of having the robot search for a destination and an object, but instead add in a set path for it to follow using colored tape for the color sensor. Once it approaches the colored destination, the robot will stop and emit a sound, notifying the student it has reached an object. For the sound component of our design, we concluded that the toy would play different tunes for whichever object it reaches.

Section 3: Prototyping and Building to Implementation

Using the sketches in figure C,, we built the base of our prototype. The color sensor was placed in a cut out square in front of the base. The servos and wheels were placed on the far back. An omniball was attached right behind the area for the color sensor. Lastly, the distance sensor was added to the front. At this point, we had been planning on using a distance sensor to sense the object the child was looking for, therefore it was an addition to our robot. The pegboard was placed on back of the base--with two buttons attached. A picture of our prototype can be seen below:

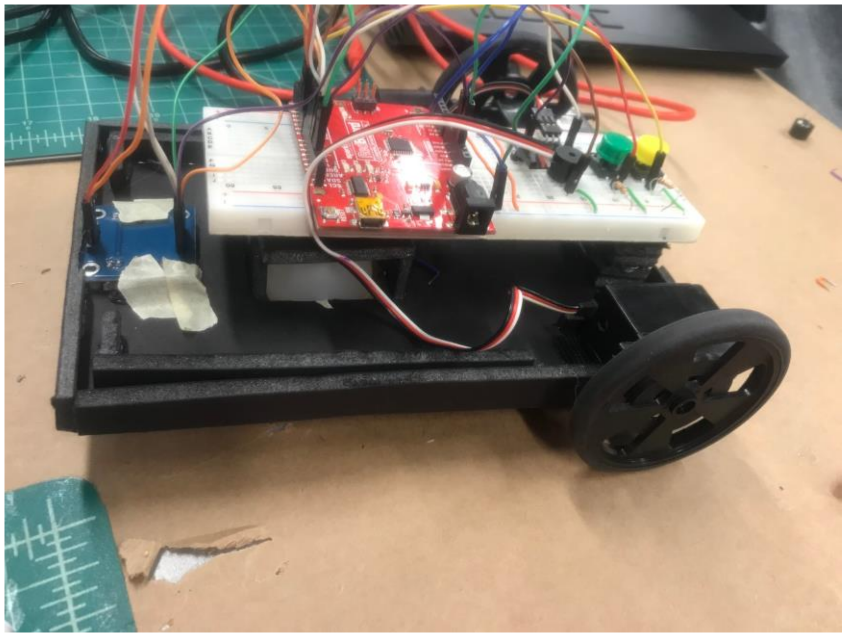
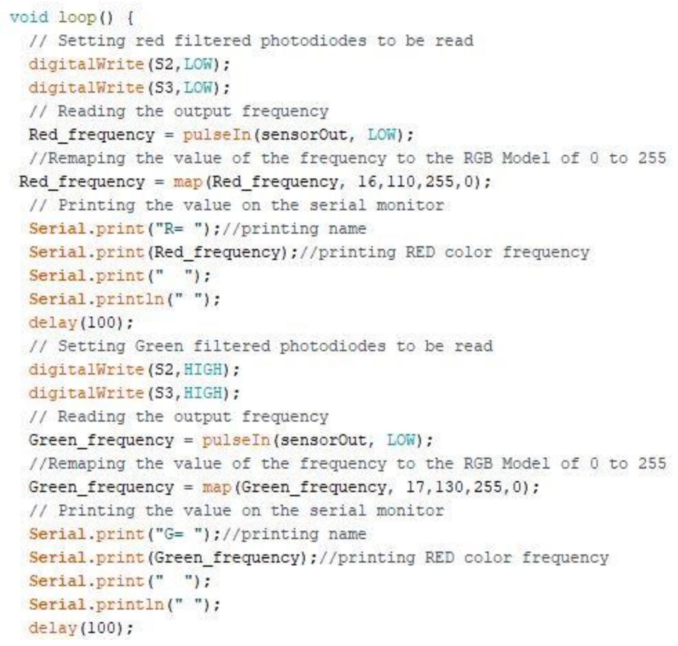


Figure I: The First Prototype

This protype helped us diagnose many of our design flaws. For starters, there was an unequal distribution in the weight. The servos, wheels, and pegboard all added more weight to the back of the robot than it was designed to handle-- resulting in our base warping. Another problem we faced was that the omniball was at a slightly greater height than the wheels. This caused the robot to be elevated in the front and drooping in the back. Our motors were also attached using velcro, which was not strong enough to support the weight of the robot, causing the motors to bend slightly and detach slightly from our robot. This issue was grew extremely problematic when the robot drove as sometimes it would sometimes move slowly due to the many disadvantages in faced in weight distribution and functionality within the design.

For the software portion, we first used a code that resembles that of the Sumo Robot from last semesters project. Inside the code, the components of the color and distance sensor were defined and initialized. Using Dejan Nedelkovski guide, we were able to outsource a skeleton code that allowed the color sensor to read different colors as seen in the figure below:



```
void loop() {
  // Setting red filtered photodiodes to be read
  digitalWrite(S2,LOW);
  digitalWrite(S3,LOW);
  // Reading the output frequency
  Red_frequency = pulseIn(sensorOut, LOW);
  //Remaping the value of the frequency to the RGB Model of 0 to 255
 Red_frequency = map(Red_frequency, 16,110,255,0);
  // Printing the value on the serial monitor
  Serial.print("R= ");//printing name
  Serial.print(Red_frequency);//printing RED color frequency
  Serial.print("  ");
  Serial.println(" ");
  delay(100);
  // Setting Green filtered photodiodes to be read
  digitalWrite(S2,HIGH);
  digitalWrite(S3,HIGH);
  // Reading the output frequency
  Green_frequency = pulseIn(sensorOut, LOW);
  //Remaping the value of the frequency to the RGB Model of 0 to 255
  Green_frequency = map(Green_frequency, 17,130,255,0);
  // Printing the value on the serial monitor
  Serial.print("G= ");//printing name
  Serial.print(Green_frequency);//printing RED color frequency
  Serial.print("  ");
  Serial.println(" ");
  delay(100);
```

Figure J: RGB Sensing Code

At this point in the project we were planning of having the robot implement a line following code, and it would start in the middle of a mat surrounded by multiple color paths. Once the user inputted which destination they were searching for, the robot would spin in a slow circle until the color was found, and then the line following code would begin until it reached the end of the colored tape. We realized after some coding that the robot wouldn't be able to stop at the end of the tape because it would be sensing black, and black was being used within the line following code. Here is where we implemented the distance sensor to sense the object the child was looking for, and to stop once the object was sensed. After calibrating our sensors to read certain colors using figure J, we started to test our robot.

Through testing our robot, we noticed two distinct problems with our code and design. Firstly, the color sensors reading would vary massively as the robot moved up and down the matt. This fluctuation in rgb values came from the fact that there was to many excess lights-- excluding the LCD-- surrounding the sensor. This made it extremely difficult to calibrate a specific rgb value for each color. In addition, some of our wiring was off which is why certain pieces of the project weren't functioning how we wanted them to. So far all we could get the robot to do was spin for a little while, sense a color, and move forward. The countless nested loops within our original code confused the robot and didn't allow it to implement any line following. We talked through the code and realized we needed to approach the problem differently. First, we drew up different design ideas for the destination search.

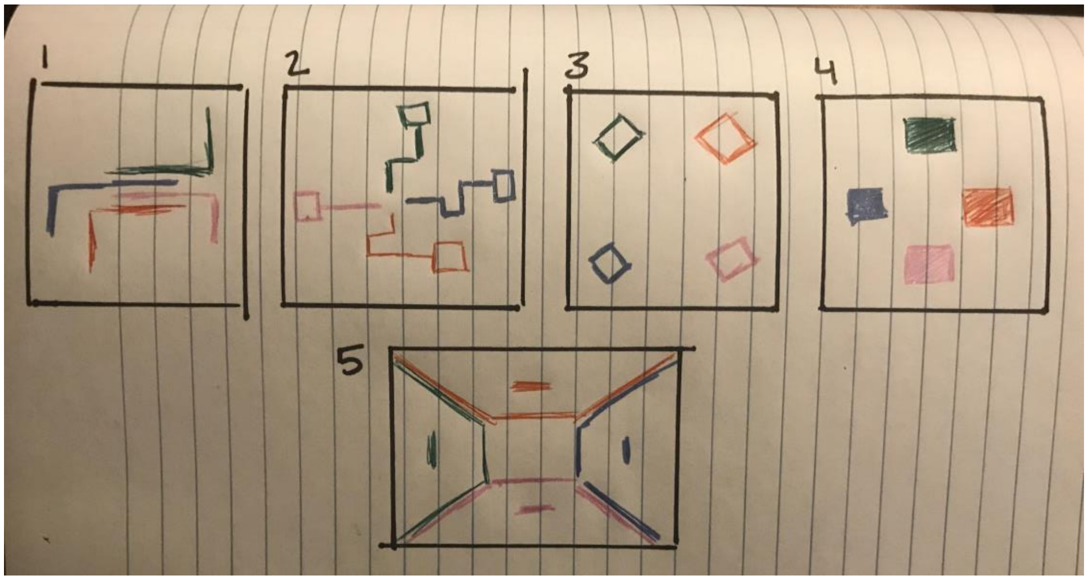


Figure K: Mat layout ideas

We had been using ideas 1 and 2, which made the coding more complex and more difficult because the robot had less of a chance to find the right color, and the distance sensor had more of a chance to notice the wrong thing. We decided to go with something more like idea 3, where the robot would be given time to search for a color and then upon finding that color it would have already reached the final destination and it would just stop. This enabled us to get rid of our distance sensor, which in turn freed up some wiring and coding. We also were able to change the idea of line following to color searching, which would require two main functions of the robot: to search for a color, and to reassess upon finding the wrong color. We started our code from scratch as we planned what we wanted the final to look like using what we had gained from this prototype.

For our final design, we knew that we wanted the base to be level. We also wanted a smoother omniball, as well as some sort of guard for the color sensor in order to obtain accurate RGB readings. First, we drew up some sketches for a modernized final design, getting an idea of where everything was to be placed.

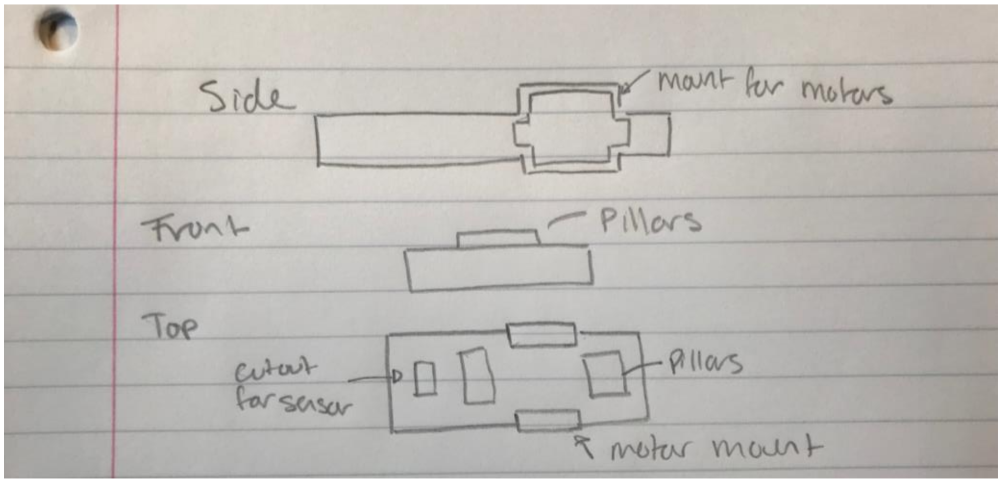


Figure L: Final Design Sketch

In order for the project to remain small in size, we would add an elevation underneath the breadboard, allowing us to place a battery pack underneath it as well as the redboard. We then tackled the issue of the motors, we decided that there needed to be some sort of socket that the motors could fit into so that the weight of the base would be evenly distributed among them. The midsection of the motors would be tangent to the side of the base, and the base would be built almost around the motor. There would be a section of support underneath it as well as above it. Next, depending on the height that the wheels would make the robot, we could measure and create a solidworks piece that acted as a socket for a marble, which would be our new omniball. This piece would be smaller and fit our purposes better for the project. If the solidworks and marble assembly was not as tall as the robot, we could simply glue more posterboard between the piece and the base until the robot was level. Below, the piece for the omniball is shown.

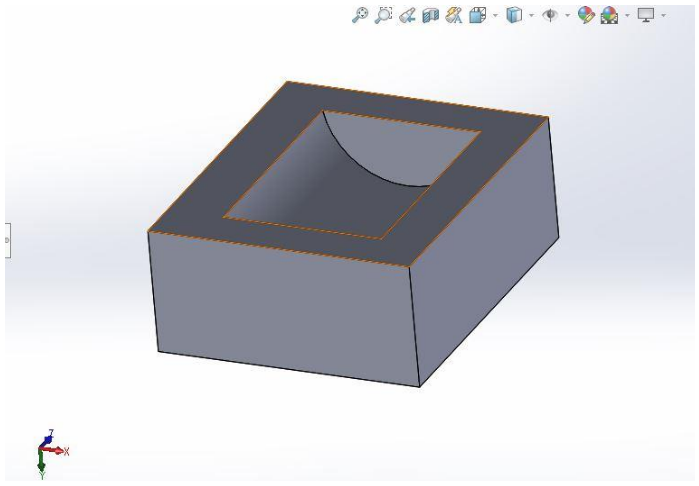


Figure M: Solidworks Socket for Marble

We also needed some sort of guard for the color sensor, and we wanted this guard to essentially almost meet the ground so that the RGB readings would be as accurate as possible. We created another solidworks piece which would act as a guard for the sensors, and almost reach the ground. We were able to determine the length of it based off of the height that the wheels would raise the base to.

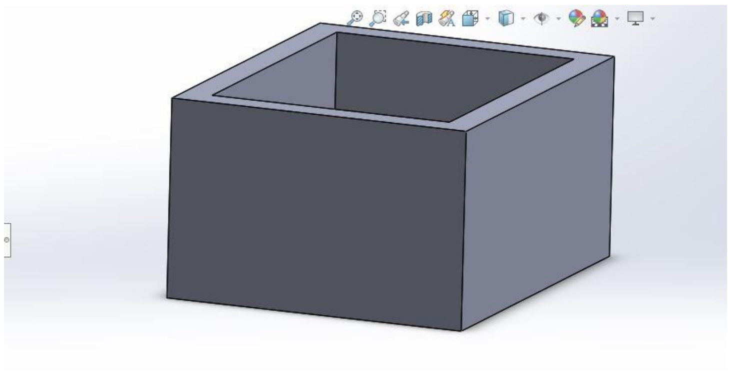


Figure N: Solidworks Sensor Guard

Once we were able to acquire the solidworks pieces and build the base to what we found to meet our goals, we tested our robot with our revamped code. This code told the robot to search for a color once the user inputted a color request. It would search for the color and upon finding it, would stop and emit a sound. If the robot ran into another color on the way, the robot would reverse and turn, and then again search for the color. Upon testing using the mat figuration 3, we found that our success was minimal. Sometimes the robot would stop, other times it would be thrown into a reversing frenzy, we figured we would start with the color sensor to determine the root of the problem. The mat was somewhat glossy and reflected the light differently in different rooms, we realized that this had a large impact on the RGB readings. Therefore, we did some more work with calibration. We then tested our code with mat figuration 4. We figured that if the destinations were filled in with color it would give the sensor more time to read and sense the color. We had slightly more success with this mat configuration, although it took an extremely long time for the robot to find any destinations. It ended up hitting the white line a lot(since we were using sumo mats) and then it would cross all the way back over t the other side of the mat, which was something we didn’t want. We re-evaluated and drew up mat figuration 5, which we felt left small room for error, and allowed the robot to reach it’s destination in a shorter amount of time. This wasn’t working exactly how we wanted it to at first, the robot wouldn’t stop right away, although it would stop eventually. We tried thinking of reasons why it wouldn’t be working right away most times, by process of elimination we decided it all stemmed from the color sensor. We did some more calibration work, and realized that the sensor would read back values that were within our range, but not consistently. We decided to make the tape thicker, which surprisingly helped a lot and the robot was then stopping on the requested color quite frequently. Although we were still having some slight problems. In order to make it as accurate as we could, we spent a lot of time tweaking the mapped values of the code. These change depending on the lighting in a room, although we were eventually able to get our robot working as accurately as we had hoped.

Section 4: Evaluate- Final Design

In our final base, the sockets we created for the motors were very reliable and provided enough support for the weight of the robot . The marble in place of the omniball was very effective and created little friction with the mat. Our robot was level and at an adequate height about the ground. The solidworks pieces that we created were hot glued to the base and were very effective(Figures M & N). To better distribute the weight of the components, we added pillars under the right and left side of the pegboard and below the base right next to the servos . There was a huge gap left for the nine volt battery.

The solidworks design for the color sensor made our RGB readings significantly more accurate, enough so that we had specific rgb values for red, green, and blue. The matt was altered to have six circular rhombi all permiterized by a colored tape.

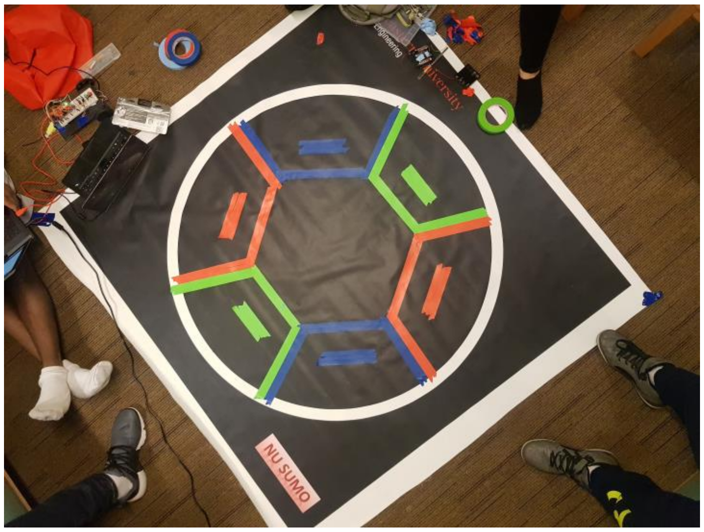


Figure O: Final Mat design/Layout

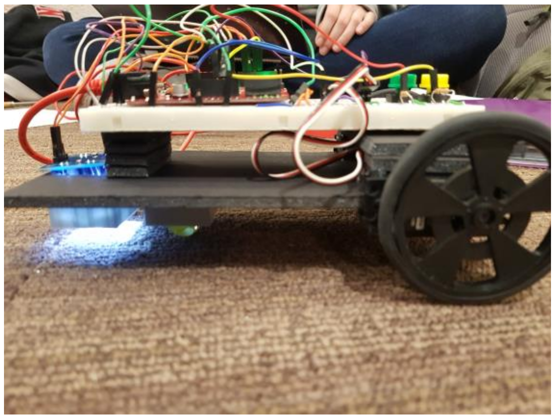


Figure P : Final Robot Design

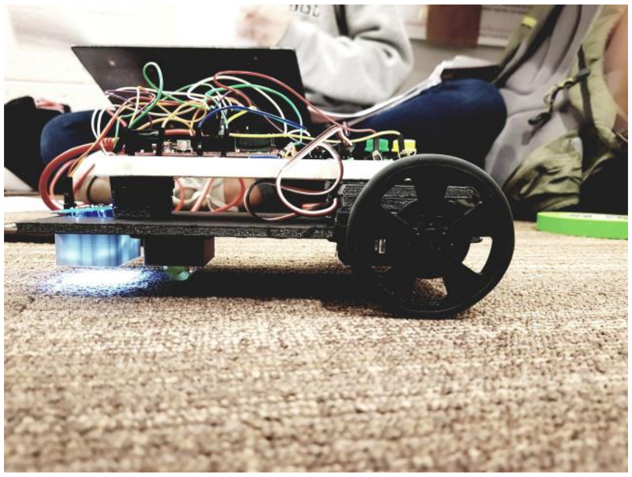


Figure P(2): Final Robot Design

Making all these adjustments for our final robot enabled the robot fulfill its objective. The robot searched for a destination based off of user input, and upon finding the destination stopped and emitted sound. To see if this would work for a blind child, we blindfolded one of our teammates and asked him to follow the buzzing sound to reach the destination. We were successful in our trial in that the teammate was able to reach his destination on the basis of nothing else but sound and was also able to identify the object placed in the destination on the basis of touch. We even demonstrated this in the final presentation and achieved success there as well. Dave had his eyes closed and reached the destination by tracking the sound and was able to identify a string of beads by feeling its texture. The main objectives of our robot were to help the user maximize on using their sound and touch senses, which our robot accomplished. With more time we would have created a more reliable outer shell for a robot following the initial goals of the outer shell. We also would have connected a remote and signal receiver to the robot, allowing the user to input form a remote. From here the robot could be further developed to function in larger and more complex obstacles, we would have created different mat layouts for the robot, and eventually expanded to an entire room filled with color destinations and paths. This technology could hopefully develop to a point where a user can input a destination, and the robot will lead them there telling them which steps to take as they go, and upon reaching the destination telling the user which object they are encountering as well.

Our final robot accomplished our main objectives, and from here we would build off of those objectives to help our robot become more advanced and personal to the user.

Section 5: Reflection

Looking back over the engineering design process for our robot, one of the major driving factors for turning our robot into a toy was draws on the deontological system of ethics. With our initial design we could not guarantee the safety of the blind person in every possible situation if the robot drove about the house freely. Our group decided it would be best to turn the robot into a toy centered around blind children and place the robot in a controlled environment, ideally the parents would be supervising the children while they play. Our group determined that as designers of a device, we had an obligation to provide a safe environment for every person using the device at all times, thus following the deontological approach for our ethical code. Our project has the potential to benefit many blind children, and prevent them from falling behind other children in terms of development. From all of our background research we understood that it was extremely important to maximize the children's use of sound and touch in order to help them become more situationally aware. Another important thing our robot helps them with is associating objects with braille and boosting their learning. Our project also has some risks, as the robot does not notify the child of any obstacles. The lack of notification could cause the child hard, but ideally the path of travel would be clear and the robot would be stopping before the object which is the only obstacle. In order to avoid the child running into the other objects in the room, we should've has the robot make a particular noise to notify the child that it had found another color, but the wrong color. The the child could associate the sound with location and start to understand that there's another object there. Our project as of right now could potentially lack in safety measures, but with more time we would make the project as safe as possible. Our team fully believes that our robot supports humanity and was created for a cause of the greater good, although some possible ethical challenges could arise from our robot. It’s possible that it could stop the user from exploring things without the robot. The robot is meant to boost curiosity and exploration, but if it accidentally supports reliance, then the child may not actually be helped by the robot. We hope this situation wouldn’t occur, but we can’t know for sure unless it was properly implemented. Another thing is the child could potentially become reliable on the sound in order to move to a location, which could be problematic because when they need to move to a certain point, sound will not always emit from that exact location. These reasons are why we recommend a certain age range for the robot, although we don’t want the child to become reliant during that time frame. Ethically, we believe the purpose of our robot and the good things that come from the technology outweigh the possible complications.

Through the design of our robot we learned a lot. We learned that in order to accomplish a big picture idea, you need to start really small and identify the most important objectives of your technology first before you try to create anything. From there we learned it’s most beneficial to perform some sort of decisional analysis to help you see how to execute a plan that matches your problem statement and maximizes on your most important objectives. We learned that prototyping is essential because it gives you the chance to see what works, reassess your problem and goals, and create another model/plan that better fits your objectives. Next time, we would improve on the group functionality and try to meet as a team more often. As a team we could get a lot done but alone it made things more difficult. With more team meetings and cohesive planning, we could produce a robot that met our objectives on a more advanced level, and we could’ve added in the other features that we’re a part of our plan. With more time, money, and skills, we would produce a toy that elaborated on all of the senses blind children need to learn to rely on and ensure the safety of the product. We would’ve been able to purchase very reliable equipment, and we could’ve created an extremely effective robot.

Reference

M. Duncan. “Engineering Concepts on Ice. Internet: www.iceengg.edu/staff.html, Oct. 25, 2000 [Nov. 29, 2003].
Frantz, Christine.”Helpful Hints for Parents of Blind Infants and Toddlers.”Internet:https://nfb.org/Images/nfb/Publications/fr/fr14/fr04se05.htm, Special Edition 2004,[Feb, 2018].

Nedelkovski, Dejan.”Arduino Color Sensing Tutorial.”Internet:https://howtomechatronics.com/tutorials/arduino/arduino-color-sensing-tutorial-tcs230-tcs3200-color-sensor/, August 2016, [March, 2018].

“Overview of Assistive Technologies for Families with a Visually Impaired Child.” Internet: http://www.familyconnect.org/info/assistive-technology/an-overview/15, [Feb. , 2018].

Pattabiraman, Krishna.”How to Setup an IR Remote and Receiver on an Arduino Board.” Internet:http://www.circuitbasics.com/arduino-ir-remote-receiver-tutorial, November 2017[March 2018].

Willings, Carmen. “Impact on Development and Learning.” Internet:https://www.teachingvisuallyimpaired.com/impact-on-development--learning.html, August 27, 2017 [Feb. , 2018].

Appendices

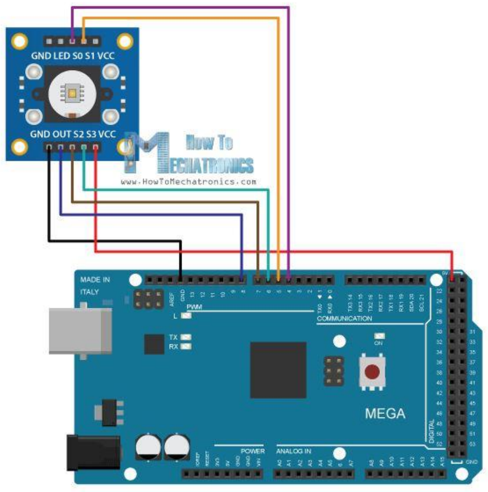
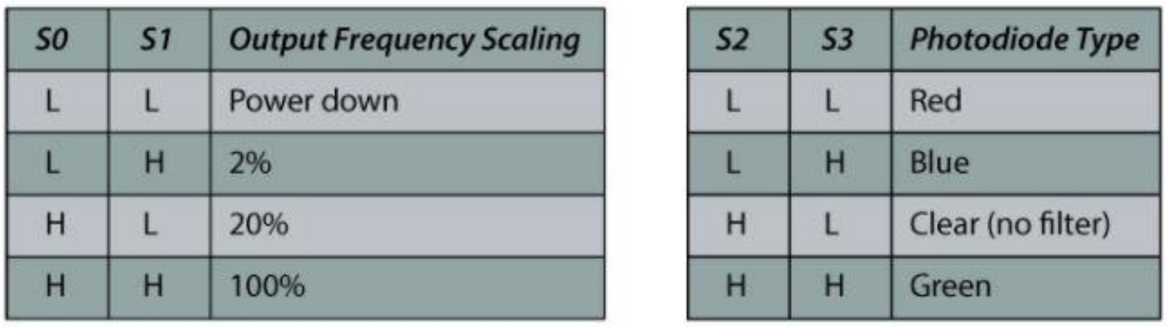


Image 1: Color Sensor Diagram



S0	S1	Output Frequency Scaling
L	L	Power down
L	H	2%
H	L	20%
H	H	100%

S2	S3	Photodiode Type
L	L	Red
L	H	Blue
H	L	Clear (no filter)
H	H	Green

Image 2: Help with Coding the Color Sensor

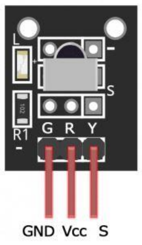


Image 3: IR Receiver

Image 4: Raised Braille Toy

Image 5:Bands with Raised Shapes

Image 6: Blind Toddler Response Screen

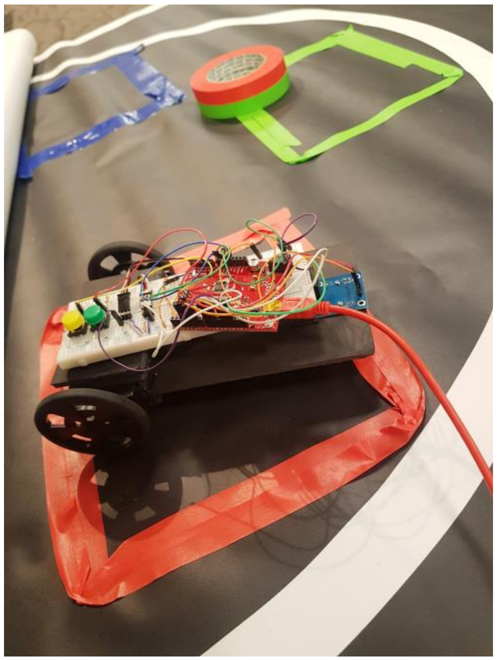


Image 7: Prototype one on mat figuration 3

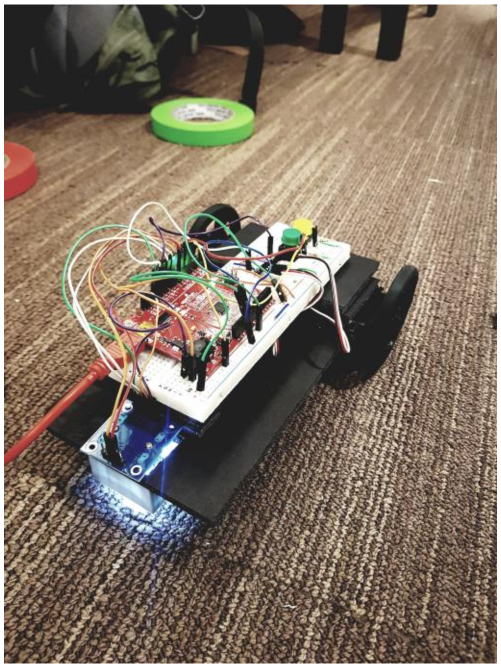


Image 8: Final Robot

Blind-Aid Code:

```
#include <Servo.h>  // servo library
#define S0 5
#define S1 6
#define S2 9
#define S3 8
#define melodyPin 7
#define sensorOut 10
#define left_forward 180
#define right_forward 0
#define right_backward 180
#define left_backward 0
#define stop_wheel 90
int Blue_frequency = 0;
int Red_frequency = 0;
int Green_frequency  = 0;
int white_frequency  = 0;
const int c = 261;
const int d = 294;
const int e = 329;
const int f = 349;
const int g = 391;
const int gS = 415;
const int a = 440;
const int aS = 455;
const int b = 466;
const int cH = 523;
const int cSH = 554;
const int dH = 587;
const int dSH = 622;
const int eH = 659;
const int fH = 698;
const int fSH = 740;
const int gH = 784;
const int gSH = 830;
const int aH = 880;
int melody[] = {
  NOTE_E7,  NOTE_E7,  0, NOTE_E7,
  0, NOTE_C7,  NOTE_E7,  0,
  NOTE_G7,  0, 0,  0,
  NOTE_G6,  0, 0, 0,

  NOTE_C7,  0, 0, NOTE_G6,
  0, 0, NOTE_E6,  0,
  0, NOTE_A6,  0, NOTE_B6,
  0, NOTE_AS6,  NOTE_A6,  0,

  NOTE_G6,  NOTE_E7,  NOTE_G7,
  NOTE_A7,  0, NOTE_F7,  NOTE_G7,
  0, NOTE_E7,  0,NOTE_C7,
  NOTE_D7,  NOTE_B6,  0, 0,

  NOTE_C7,  0, 0, NOTE_G6,
  0, 0, NOTE_E6,  0,
  0, NOTE_A6,  0, NOTE_B6,
  0, NOTE_AS6,  NOTE_A6,  0,

  NOTE_G6,  NOTE_E7,  NOTE_G7,
  NOTE_A7,  0, NOTE_F7,  NOTE_G7,
  0, NOTE_E7,  0,NOTE_C7,
  NOTE_D7,  NOTE_B6,  0, 0
};
int tempo[] = {
  12, 12, 12, 12,
  12, 12, 12, 12,
  12, 12, 12, 12,
  12, 12, 12, 12,

  12, 12, 12, 12,
  12, 12, 12, 12,
  12, 12, 12, 12,
  12, 12, 12, 12,

  9, 9, 9,
  12, 12, 12, 12,
  12, 12, 12, 12,
  12, 12, 12, 12,

  12, 12, 12, 12,
  12, 12, 12, 12,
  12, 12, 12, 12,
  12, 12, 12, 12,

  9, 9, 9,
  12, 12, 12, 12,
  12, 12, 12, 12,
  12, 12, 12, 12,
};
//Underworld melody
int underworld_melody[] = {
  NOTE_C4,  NOTE_C5,  NOTE_A3,  NOTE_A4,
  NOTE_AS3,  NOTE_AS4,  0,
  0,
  NOTE_C4,  NOTE_C5,  NOTE_A3,  NOTE_A4,
  NOTE_AS3,  NOTE_AS4,  0,
  0,
  NOTE_F3,  NOTE_F4,  NOTE_D3,  NOTE_D4,
  NOTE_DS3,  NOTE_DS4,  0,
  0,
  NOTE_F3,  NOTE_F4,  NOTE_D3,  NOTE_D4,
  NOTE_DS3,  NOTE_DS4,  0,
  0, NOTE_DS4,  NOTE_CS4,  NOTE_D4,
  NOTE_CS4,  NOTE_DS4,
  NOTE_DS4,  NOTE_GS3,
  NOTE_G3,  NOTE_CS4,
  NOTE_C4,  NOTE_FS4,NOTE_F4,  NOTE_E3,  NOTE_AS4,  NOTE_A4,
  NOTE_GS4,  NOTE_DS4,  NOTE_B3,
  NOTE_AS3,  NOTE_A3,  NOTE_GS3,
  0, 0, 0
};
//Underwolrd tempo
int underworld_tempo[] = {
  12, 12, 12, 12,
  12, 12, 6,
  3,
  12, 12, 12, 12,
  12, 12, 6,
  3,
  12, 12, 12, 12,
  12, 12, 6,
  3,
  12, 12, 12, 12,
  12, 12, 6,
  6, 18, 18, 18,
  6, 6,
  6, 6,
  6, 6,
  18, 18, 18,18, 18, 18,
  10, 10, 10,
  10, 10, 10,
  3, 3, 3
};
const int Yellow_buttonpin = 3; //declare pushbutton pins
const int Green_buttonpin = 4;
const int Blue_buttonpin = 2;
int counter = 0;

Servo servo_r;                // Declare right and left servos
Servo servo_l;
int delayRandom = 0;
void turnLeft(); void turnRight(); void forward(); void reverse();void search();
int song =0;
void setup(){
  pinMode(melodyPin, OUTPUT);//buzzer
// set up the color sens as
pinMode(S0, OUTPUT);
pinMode(S1, OUTPUT);
pinMode(S2, OUTPUT);
pinMode(S3, OUTPUT);
pinMode(sensorOut, INPUT);

//setup buttons to be inputs
pinMode(Yellow_buttonpin, INPUT);
pinMode(Green_buttonpin, INPUT);
pinMode(Blue_buttonpin, INPUT);

// Setting frequency-scaling to 20%
digitalWrite(S0,HIGH);
digitalWrite(S1,HIGH);
servo_r.attach(12);    // attach two servos to pins 12 and 13
servo_l.attach(13);
Serial.begin(9600);

}



void loop() {
  int Yellow_button_State, Green_button_State, Blue_button_state; // variables to hold buttons
```

```
servo_l.write(stop_wheel);     //left motor Clockwise for reverse
  servo_r.write(stop_wheel);

     Yellow_button_State = digitalRead(Yellow_buttonpin);
     Green_button_State = digitalRead(Green_buttonpin);
     Blue_button_state = digitalRead(Blue_buttonpin);
bool red_tape = false;
  bool green_tape = false;
  bool Blue_tape = false;
  if ((Yellow_button_State == LOW) && (Green_button_State == HIGH)&&(Blue_button_state == HIGH)){
 red_tape = true;
  }
   if ((Yellow_button_State == HIGH) && (Green_button_State == LOW) && (Blue_button_state ==
HIGH)){
      green_tape = true;
    }
     if ((Yellow_button_State == HIGH) && (Green_button_State == HIGH) && (Blue_button_state ==
LOW)){
      Blue_tape = true;
    }
while (red_tape == true)
{
  search();   // move around
      // Setting red filtered photodiodes to be read
 digitalWrite(S2,LOW);
  digitalWrite(S3,LOW);
  // Reading the output frequency
  Red_frequency = pulseIn(sensorOut, LOW);
  //Remaping the value of the frequency to the RGB Model of 0 to 255
  Red_frequency = map(Red_frequency, 16,110,255,0);
   Serial.print("R= ");//printing name
  Serial.print(Red_frequency);//printing RED color frequency
  Serial.print("  ");
  Serial.println(" ");
  delay(100);
      if (Red_frequency >= 170 && Red_frequency <250)
      {

        red_tape = false;
          //stop and make some noise
          servo_l.write(stop_wheel);     //left motor Clockwise for reverse
  servo_r.write(stop_wheel);
   Serial.println("Red Tape Reached. Stopping");
      sing(1);
      }
      else if (Red_frequency <70)
      {
        search();
        }
        else
        {
        reverse();
            int n = random(1,2);
            if (n==1){
             turnRight();

          }
          else if(n==2){
            turnLeft();
            }
        }
}

while (green_tape == true)
{
  search();
    // Setting Green filtered photodiodes to be read
    digitalWrite(S2, HIGH);
    digitalWrite(S3, HIGH);
    // Reading the output frequency
    Green_frequency = pulseIn(sensorOut, LOW);
    //Remaping the value of the frequency to the RGB Model of 0 to 255
    Green_frequency = map(Green_frequency, 17, 130, 255, 0);
    Serial.print("G= ");//printing name
    Serial.print(Green_frequency);//printing RED color frequency
    Serial.print("  ");
    Serial.println(" ");
    delay(100);
    //green tape
    if ( Green_frequency >= 170 && Green_frequency <= 230)

      {
        green_tape = false;
          //stop and make some noise
          servo_l.write(stop_wheel);    //left motor Clockwise for reverse
  servo_r.write(stop_wheel);
   Serial.println("Red Tape Reached. Stopping");
  sing(2);
 //beeep periodically

      }
      else if (Green_frequency <80)
      {
        search();
        }
        else
        {
        reverse();
```

```
            int n = random(1,2);
            if (n==1){
             turnRight();

          }
          else if(n==2){
            turnLeft();
            }
        }
      }



while (Blue_tape == true)
{
    search();
    digitalWrite(S2, LOW);
    digitalWrite(S3, HIGH);
    // Reading the output frequency
    Blue_frequency = pulseIn(sensorOut, LOW);
    //Remaping the value of the frequency to the RGB Model of 0 to 255
    Blue_frequency = map(Blue_frequency, 35, 67, 255, 0);
    // Printing the value on the serial monitor
    Serial.print("B= ");//printing name
    Serial.print(Blue_frequency);//printing RED color frequency
    Serial.print("  ");
    Serial.println(" ");
    delay(100);
    if (Blue_frequency >=190 && Blue_frequency < 255)
      {
        Blue_tape = false;
          //stop and make some noise
          servo_l.write(stop_wheel);     //left motor Clockwise for reverse
  servo_r.write(stop_wheel);
   Serial.println("Red Tape Reached. Stopping");
   sing(2);
   delay(5000);
   sing(1);
      }
      else if (Blue_frequency <70)
      {
        search();
        }
        else
        {
        reverse();
            int n = random(1,2);
            if (n==1){
             turnRight();
          }
          else if(n==2){
            turnLeft();
            }
        }
      }
}
void beep(int note, int duration)
{
  //Play tone on buzzerPin
  tone(melodyPin, note, duration);
 //Stop tone on buzzerPin
  noTone(melodyPin);
  delay(50);
}

void search()
{

forward();
    //right motor clockwise forward
            //  millisecond delay
}
void turnLeft()
{
  reverse();
  servo_l.write(left_backward);     //left motor Clockwise for reverse
  servo_r.write(right_forward);    //right motor clockwise forward
  delayRandom = random(900, 1550);  //randomized delay alters angle of turn
  delay(delayRandom);
}

void turnRight()
{
  reverse();
  servo_r.write(right_backward);      // spin right motor backwards
  servo_l.write(left_forward); // spin Left motor forwards
int  delayRandom = random(900, 1550);  //randomized delay alters angle of turn
  delay(delayRandom);
}

void reverse()
{
  servo_r.write(right_backward);  //right motor counterclockwise reverse
  servo_l.write(left_backward);   //left motor Clockwise for reverse
  delay(1500);       //  .5 second delay
}
```

```
void forward()
{

  servo_r.write(right_forward);    //right motor clockwise forward
  servo_l.write(left_forward); //left motor counterClockwise for forward
 delayRandom = random(900,  1550);
}

void sing(int s){
   // iterate over the notes of the melody:
   song = s;
   if(song==2){
     Serial.println(" 'Underworld Theme'");
     int size = sizeof(underworld_melody) / sizeof(int);
     for (int thisNote = 0; thisNote < size; thisNote++) {
       // to calculate the note duration, take one second
       // divided by the note type.
       int noteDuration = 1000/underworld_tempo[thisNote];
       buzz(melodyPin, underworld_melody[thisNote],noteDuration);
       // to distinguish the notes, set a minimum time between them.
       // the note's duration + 30% seems to work well:
       int pauseBetweenNotes = noteDuration * 1.30;
       delay(pauseBetweenNotes);

       // stop the tone playing:
       buzz(melodyPin, 0,noteDuration);

    }

   }else{
     Serial.println("'Mario Theme'");
     int size = sizeof(melody) / sizeof(int);
     for (int thisNote = 0; thisNote < size; thisNote++) {

       // to calculate the note duration, take one second
       // divided by the note type.
       //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
       int noteDuration = 1000/tempo[thisNote];

       buzz(melodyPin, melody[thisNote],noteDuration);

       // to distinguish the notes, set a minimum time between them.
       // the note's duration + 30% seems to work well:
       int pauseBetweenNotes = noteDuration * 1.30;
       delay(pauseBetweenNotes);

       // stop the tone playing:
       buzz(melodyPin, 0,noteDuration);
    }
  }
}
void buzz(int targetPin, long frequency, long length) {
  long delayValue = 1000000/frequency/2; // calculate the delay value between transitions
  //// 1 second's worth of microseconds, divided by the frequency, then split in half since
  //// there are two phases to each cycle
  long numCycles = frequency * length/ 1000;  // calculate the number of cycles for proper timing
  //// multiply frequency, which is really cycles per second, by the number of seconds to
  //// get the total number of cycles to produce
  for (long i=0; i < numCycles; i++){ // for the calculated length of time...
    digitalWrite(targetPin,HIGH);  // write the buzzer pin high to push out the diaphram
    delayMicroseconds(delayValue); // wait for the calculated delay value
    digitalWrite(targetPin,LOW);  // write the buzzer pin low to pull back the diaphram
    delayMicroseconds(delayValue); // wait again or the calculated delay value
  }
}
```