

Western University of Canada

Faculty of Engineering

MSE 2202B - Introduction to Mechatronic Design

A.M.A.N.D.A.

Final Design Report

Zach De Haan, Danielle Myers, Brendan Ng, Lac Truong

Michael Naish

MSE 2202B - Section 002, Team 10

Date Submitted: April 7th, 2017

**Executive Summary**

The following report outlines the problem given by Tesseract Power Co., and provides a solution to solve it. AMANDA, A Mobile Auto-Nomous Driving Arduino, is the suggested solution to accomplish the task given. The report describes how AMANDA operates and all its features. The report also provides multiple Figures found the in appendix to aid in the understanding of each component designed and provides insight as to why they were chosen. The report also explains how each component work and how they work together to accomplish the task in a step by step manner. At the end of the report is a performance review of how well AMANDA faired during the design competition. In addition to the performance review, multiple recommendations are made as to how AMANDA can be improved upon. With this report, it will guide you through the ins and outs of how AMANDA operates to resolves Tesseract Power Co.’s presented problem.

**Table of Contents**

Introduction

Problem Definition………………………………………………………………………...4

Background………………………………………………………………………………..5

Project Specifications……………………………………………………………………...6

Discussion

Evaluation…………………………………………………………………………………8

Prototype Description/Operation………………………………………………………...10

Iterations, Analysis, and Design Development…………………………………………..12

Final Design……………………………………………………………………………...14

Performance……………………………………………………………………………...15

Conclusion……………………………………………………………………………………….19

Recommendations

Hardware Recommendations…………………………………………………………….20

Software Recommendations……………………………………………………………..22

References………………………………………………………………………………………..24

**Appendix**

Figure 1: Isometric View of the final design…………………………………………………….25

Figure 2: Top view of the final design…………………………………………………………..26

Figure 3: Front view of the final design…………………………………………………………27

Figure 4: Side view of the final design…………………………………………………………..27

Figure 5: A Decision Matrix of top secondary objectives……………………………………….28

Figure 6: Isometric view of the final prototype………………………………………………….28

Figure 7: Top view of the final prototype………………………………………………………..29

Figure 8: Front view of the final prototype………………………………………………………30

Figure 9: Side view of the final prototype……………………………………………………….31

Figure 10: AMANDA searching for the tesseract……………………………………………….32

Figure 11: AMANDA picking up the pyramid…………………………………………………..33

Figure 12: AMANDA lifted the power pyramid onto the ramp…………………………………34

Websites where every component of the final report may be purchased………………………..35

Entire final code used to run AMANDA…………………………………………………………………36

**Introduction**

A mechatronics engineer is one who can incorporate robotics into their solutions for everyday issues. Some issues maybe very general however, many issues are very specific and solutions must be catered to them. In this report, there is a specific problem that must be tackled.

Problem Definition:

The problem within this report was provided by Tesseract Power Co. (TPC), a company which uses a new source of power that is generated from cube-like-tesseracts. Within TPC’s power plant there exists multiple tesseracts that must be retrieved autonomously. These tesseracts are located along the rightmost wall of the power plant which is an approximate square ranging from the size of 6 - 10 feet. The walls that the tesseract rests on are 3.5 inches high.

Once the tesseract is obtained, it must be transferred to a power pyramid which is located within the power plant. The power pyramid constantly emits a frequency of 38 kHz which is modulated at either 170 Hz or 340 Hz. Then the tesseract must be placed underneath the power pyramid which has a square base of 4 inches and a height of 3 inches. Although the base of the pyramid is 4 x 4 inches, it has a square base opening of 1.5 x 1.5 inches centered under the pyramidion.

Finally, when the tesseract is placed underneath the power pyramid the task is complete. During the entire process, the design must be able to traverse over power conduit lines which are located randomly on the power plant’s floor. The overall design will be assessed based on performance, design, and potential failures.

Along with the problem provided by Tesseract Power Co., they also provided a few constraints for the solution. The solution must be able to fit within a 12 x 17 x 17-inch locker. The solution must also be able to traverse in any conditions without any markings on the ground or any other form of installed guide.

Background:

Given the problem, multiple sensors must be implemented to detect the power pyramid, and the tesseract. All the sensors implemented were: an infrared receiver, an infrared proximity sensor, an ultrasonic proximity sensor, a hall effect sensor, and a limit switch.

1. Infrared Receiver

Infrared Receiver receives infrared signals transmitted from an infrared transmitter. The receiver sends a signal once it receives an infrared signal. (2015)

1. Infrared Proximity Sensor

Infrared Proximity Sensor combines an infrared transmitter and receiver. The transmitter sends out an infrared signal which will bounce off an object and reflect towards the sensor. The reflected signal will be read by the receiver and it will display a value which is the time difference between sending and receiving. The time difference can then be converted to a distance. (2015)

1. Ultrasonic Proximity Sensor

The Ultrasonic Proximity Sensor operates the same way as the infrared proximity sensor however, it utilizes ultrasonic waves instead of infrared waves.

1. Hall Effect Sensor

The Hall Effect Sensor utilizes an external magnetic field to trigger it. Within the hall effect sensor there is a semi conductive material. The material is split up into two sides, one side with electrons (p-side), and the other without (n-side). Once the sensor is within a magnetic field, the electrons will begin to flow from the p-side to the n-side, creating a voltage which results in a current. The current and voltage are proportional to the magnetic field’s strength and it will display its value. (Woodford, 2017)

1. Limit Switch

A Limit Switch can be simplified to a button that displays on or off values. It contains a button with a beam mounted across it and if the beam were to depress, so would the button. (Bayne)

Once all the sensors are mounted onto the robot, it will be able to determine its surroundings and perform the task with efficiently.

Project Specifications:

A Mobile Auto-Nomous Driving Arduino (AMANDA) is a mobile robot which will accomplish all the given tasks. Figure 1 through 4, are solidworks drawing of final design. Within the final design there are four major components:

1. **Driving System**

Driving was accomplished by utilizing four wheels, of which only the rear two would spin. The wheels are large enough that the traversal of bumps becomes no issue occurred once coded properly. Proximity sensors are mounted such that driving in a straight line alongside the walls are possible.

1. **Tesseract Retrieval System**

A permanent magnet, along with two servo motors were used to retrieve the tesseract from the wall. The magnet could extend, retract, and sweep because of the two servo motors.

1. **Power Pyramid Retrieval System**

A ramp, an arm, and an infrared sensor were used to locate and retrieve the power pyramid. When the sensor detects the power pyramid, AMANDA will approach it. Once AMANDA has contacted the pyramid, it backed up and lowered the ramp and the arm. The arm will push the power pyramid, tilting it onto the ramp high enough such that the tesseract may slide underneath.

1. **Drop Off System**

The Driving System, Tesseract Retrieval System, and the Power Pyramid Retrieval System were all in effect to drop the tesseract underneath the power pyramid. The permanent magnet which has the tesseract on it was swung to the centre of AMANDA. The tesseract was knocked off and dropped below AMANDA. AMANDA reversed the required amount until the tesseract was under the pyramid. Finally, the arm released the pyramid off the ramp, sliding it onto the ground with the tesseract underneath it.

Further details regarding the selection and operations of each component will be discussed in detail in later sections. Within this report, it will outline the entire project from beginning to end, providing a timeline on the overall progress. Along with the timeline, a discussion of why each method was selected will be provided. Multiple figures and tables were found in the appendix to provide further insight.

**Discussion**

In the following sections the report will go into details on the final design. It will provide details, and insight into the process of finalizing the design. Within the product development file, there is a detailed timeline as to how the project progressed.

Evaluation:

Prior to the production phase, a list of objectives was constructed to guide the direction of the final design. Besides the objective to resolve the problem given, secondary objectives for the project were created. The objectives are:

1. Create an energy efficient product.
2. Create an aesthetically pleasing product.
3. Create a simple product that is easily remanufacturable.
4. Create a cost-efficient product.

With these objectives in mind, they were used as criteria to decide which concept provides the best solution to the problem. A decision matrix (Figure 5) was created which outlined the rankings of each objective.

Given the concept generation provided by the product development files, multiple go-no-go screenings were created to finalize which concept would be implemented into the final design. The screenings weighed each concept against each other with multiple criteria. The go-no-go screenings are also located in the product development files however, simple explanations of the decisions follow:

1. Wheels

Rear wheel drive provides the simplest and lightest concept compared to the rest. Compared to tank treads, wheels are small and easier to mount.

1. Locate the tesseract

Hall Effect sensor provides the most efficient concept in finding the tesseract. Traversing the entire wall is not a viable option as there is a large amount of time consumed.

1. Obtain the tesseract

Permanent magnet provides a simple, light, and energy efficient concept as opposed to the others. Secondary claws require a lot of space and energy. An electromagnet also requires a large power source to power it.

1. Transport the tesseract

Physical stop paired with the funnel is the simplest method to transport the tesseract to the desired location. Based on the permanent magnet, a platform to move the tesseract is not needed.

1. Locate the power pyramid

Infrared sensor is the most efficient method in finding the power pyramid. Traversing the entire power plant floor is not a viable option as there is a large amount of time consumed.

1. Pick up the power pyramid

Ramp and arm method is the simplest and most diverse method in retrieving the power pyramid. Any overhang claw method results in a large space requirement, it may exceed the size constraint.

1. Place power pyramid on tesseract

Ramp and arm method is the simplest method with the fewest potential errors. Any claw method is too large and may exceed the size constraint

1. Collision Avoidance

Proximity sensors pairs with limit switches most efficient and diverse method. Generating an internal map requires a lot of coding which may or may not be accomplished within the given time to complete the project.

Prototype Description/Operation:

The prototype was a close representation of how the final design will look and operate however, different electrical components and parts were used to model it. Diagrams of the prototype can be found in the appendix and was labelled as Figure 6 to Figure 9.

To begin, the body of the prototype closely resembles the final shape of the product, it is Figure 1. The shape closely resembled a rectangular car with an area in the centre left open for a funnel system to operate. Like a car, the prototype had four wheels which operate on a rear wheel drive basis.

Secondly, the prototype had an ultrasonic proximity sensor and an infrared proximity sensor which guided the driving system of AMANDA. The ultrasonic sensor was strategically mounted above the wall and the infrared sensor is mounted to the outer left side of the frame. These two sensors allowed for a straight traversal along the wall while the robot was searching for the tesseract. The ultrasonic sensor detected the distance from the top of the wall and notified AMANDA that we were in the right position. If the ultrasonic sensor detected a different value AMANDA knew that it had strayed too far to the left or right. The infrared sensor provided side distance values from the wall which will allowed AMANDA to determine which way it must reposition itself.

Next, at the back left of the prototype is the magnet retrieval system. The system consists of two small servo motors and a hall effect sensor which were connected to a beam and modelled in solidworks found in the product development file. One servo (servo 1) was mounted below the beam and the other (servo 2) was mounted on top of the beam. The hall effect sensor was then mounted at the end of the beam on top of the long-angled platform. A long magnet was then connected to servo 2 with an arm and a pin joint that connected the magnet’s arm and rested on the long-angled platform. Servo 1 rotated the entire system which allowed the system to sweep for corners and rotated it towards the funnel system. Servo 2 extended and retracted the magnet on the long-angled platform which allowed for the retrieval of the tesseract. Finally, the hall effect sensor notified AMANDA when to extend and retract the magnet.

Power pyramid retrieval system was the next major component of the prototype. The system was located at the front of AMANDA and consisted of an arm, ramp, infrared sensor, and a limit switch which can be seen in Figures 6 to 9. Once the retrieval of the tesseract was complete, AMANDA began scanning the power plant floor to locate the power pyramid with the infrared sensor. Once found, the robot began to move forward until the limit switch was activated against the side of the pyramid. Next AMANDA reversed, lowered the ramp, moved forward and push the pyramid onto the ramp with the arm which completed the retrieval system of the power pyramid

Finally, the drop off system combined the two retrieval systems along with the funnel system which consumed most the space and can be seen in Figure 7 and Figure 9. Once the retrieval of the pyramid was complete AMANDA began the final phase of the operation, placing the tesseract under the power pyramid. The magnetic arm swung and hit the tesseract off a physical stop and dropped it into a funnel system. The funnel system precisely dropped the tesseract under the center of AMANDA. The robot then reversed until the tesseract is under the power pyramid. With multiple combinations of lifting the arm and reversing the power pyramid would be over the tesseract.

Overall, AMANDA required a storage space no greater than 12 x 17 x 17 inches.

Iterations, Analysis, and Design Development:

Provided in the product development file are all the details regarding the entire process of the project however, the major iteration and analyses was outlined in this section.

The largest iteration done was the removal of a claw system and the implementation of a ramp and arm system for the retrieval of the power pyramid. Through testing and analysis, it was discovered that the claw was too large and was not capable of clearing walls. Along with an oversized claw, due to inaccurate measurements the claw did not assemble correctly which resulted in an unreliable method of obtaining the power pyramid. Thus, the claw system was opted out and replaced by the arm and ramp system. The new system required a smaller frame compared to the previous concept, and could be autotomized through servos instead of motors which simplified the coding process. Through the testing phase, the new system proved to be more reliable and efficient than the previous system.

Testing was conducted for each of the stages that AMANDA went through. A few tests were conducted to determine whether the driving system could traverse over bumps as well as the tesseract because of the retrieval system. Wheel modifications were implemented and full details can be seen in the product development file. Testing was conducted by, driving AMANDA over the speed bump provided in many different configurations and angles. In the end the results showed that AMANDA could traverse over any conduit lines that may be found within the power plant.

Testing the hall effect sensor was done in a systematic manner. Based on how AMANDA retrieved the tesseract, the hall effect values were measured. The table regarding the measured hall effect values can be found in the product development file.

Similar testing methods were used to determine how well the infrared sensor operated with the power pyramid. Different angles and sides of the power pyramid were applied to determine the field of view of the infrared sensor which can also be found in the product development file.

The remaining tests were conducted in a less structured manner. Code was written based on what AMANDA was required to do. Based on the performance of the task the code was modified and re-uploaded. Therefore, the remaining tests to determine how well AMANDA operated were conducted on a trial and error basis.

Final Design:

The selection process which determined the best components to implement in the final design are found in the product development file. In the product development file, all the potential components that were compared to can be found. However, all the final components which were implemented in the final design are as follows:

1. Frame Material: Aluminium 6061 ($12.67)
2. Wheels: Actobotics 4-inch heavy duty wheels ($9.20 each)
3. Motors: 165 RPM Planetary Gearmotor ($52.63)
4. Servos: HS-645MG Motor ($41.99)
5. Hall Effect Sensor: US5881LUA hall ($2.00)
6. Infrared Proximity Sensor: GP2Y0A21YK0F ($14.95)
7. Infrared Sensor: TSOP38238 ($1.95)
8. Linear Actuator: LIN-ACT1-02 ($51.99)
9. Electromagnet: EM075 round electromagnet ($34.09)
10. Battery: 1286 12V Li-Polymer Battery ($24.99)
11. Arduino Board: MSEduino ($75)

Costs listed beside each part can be found in their respective URL’s located in the appendix of this report. The total cost for a single AMANDA unit was an estimated cost of $321.46, without tax. This price also excluded the manufacturing cost that may be incurred if a mass production AMANDA is required.

Performance

A performance competition was conducted on April 6th, 2017 between all the groups competing for the best potential solution to Tesseract Power Co.’s problem. During the competition, each group was required to demonstrate a simulation of how the final design will operate within the modeled power plant floor. Each group was faced with a time restriction of three minutes to dictate the efficiency of their solution.

AMANDA’s performance during the competition was not the greatest in terms of solving the entire problem. However, in comparison to every group that competed, AMANDA’s performance was average. During the competition, each group was presented with a simulated power plant floor which included a tesseract, power pyramid, and 2 large conduits. Each group was required to traverse alongside the rightmost wall starting on top of a conduit and pick up the tesseract. From there the power pyramid was located on the other side of the course emitting a frequency which would display ‘I’ and ‘O’. In total, each group could demonstrate their solution three times, with a time limit of three minutes per trial.  During the competition, nearly all the groups were not able to complete the course, and no one group was able to complete it consistently.

Trial 1

AMANDA’s first trial was not successful. To begin, AMANDA was placed at the starting position, on top of a conduit and at the correct distance from the wall. Once the battery pack provided was connected, AMANDA adjusted into its starting position. The code was designed such that AMANDA would traverse alongside the wall until the hall effect sensor detected the tesseract however, it was noticed that AMANDA was unable of driving over the conduit. With a few resets of the battery, AMANDA was still incapable of traversing over the conduit.

To resolve the issue, the set motor speeds were adjusted such that AMANDA would have enough momentum to carry itself over the initial conduit. With that in mind, AMANDA was ready for Trial 2 since no additional information on any other issues were gathered during Trial 1.

Trial 2

For each group’s second trial, they could place their robot past the first conduit such that the new first conduit their robot would see was roughly 5 feet away. The new initial position however, conflicted with the changes made after Trial 1 as a new issue arose.

During AMANDA’s second trial, it was started after the first conduit which resulted in a clear path between the robot and the tesseract. With the new starting position, and increased motor speeds after Trial 1, AMANDA drove too fast for the hall effect sensor to detect any magnetic field before it drove past it. The result was that AMANDA drove past the tesseract, stopped, then extended the magnetic arm and began panning for the power pyramid. On occasion, AMANDA drove too fast and strayed away from the straight wall, which resulted in the hall effect sensor never activating.

During the panning phase of the code, AMANDA was not capable of panning over bumps as again the motor speeds were too slow. Also, there were occasions while panning where the rear end of the robot would hit the wall. To resolve these issues, the panning sequence was adjusted such that the motor speeds were increased; the time between panning directions were increased; and finally, during the panning sequence the robot would inch forward every few seconds to avoid collisions with the wall it.

During the last attempt in Trial 2, AMANDA was positioned at the same starting position as Trial 1, and it was capable of driving over the initial conduit. Therefore, the adjustment of the motor speeds was enough such that AMANDA can start at the original initial position.

However, to make the course easier, adjustments were made such that AMANDA could run the course after the first conduit.

Trial 3

For Trial 3, additional course adjustments were made such that it would be easier to perform the overall task. Since up to this point no group completed the course, it was modified such that the pyramid now rested closer to the starting point. The power pyramid was placed between the starting position and the first conduit. The intention was that since the conduits were such an obstacle, the adjustment eliminated them completely.

During AMANDA’s Trial 3 attempt it showed signs of success. The hall effect sensor did not trigger fast enough again however, AMANDA was left alone to continue the course. During the continuation of the course the tesseract was placed on the magnetic arm to simulate what would happen if the robot had retrieved the cube. AMANDA panned for the power pyramid and tracked it. However, due to reflected infrared rays, AMANDA lost the pyramid occasionally and was incapable of finding it again based on the way the panning sequence was coded.

For the last attempt during Trial 3, similar results occurred however, this time the pyramid was nudged into the limit switch placed at the front of the robot. Once the limit switch was activated the rest of the run was a success. AMANDA reversed, lowered the ramp and lowered the arm over the power pyramid. The pyramid was then raised onto the ramp in the correct position. The tesseract dropped off the magnetic arm and into the correct position underneath the robot. AMANDA then began to reverse the pyramid over the tesseract however, a wall behind the robot blocked it from reversing far enough such that the tesseract would catch on the pyramids edge. The pyramid was released correctly and the code ended. Figure 10 to Figure 12, depicted the last attempt on Trial 3.

Based on the performance of the robot, many modifications are required for the entire system to work perfectly. However, the latter end of the course showed that AMANDA could pick up the pyramid and placed the tesseract underneath it given that there is no wall near the robot. With that said, the initial part of the course could be improved upon such that every component comes together and AMANDA runs the course smoothly.

**Conclusion**

The final design is an efficient solution to the problem presented by Tesseract Power Co. Each stage of the process has been met with a method to complete them. Various sensors and innovative ideas were used to accomplish each task. Based on the performance of the simulated run, additional modifications are required however, AMANDA proved that it has all the mechanical components required to solve the problem.

Using multiple sensors and motors, AMANDA can retrieve the tesseract and place it under a power pyramid. Using a magnetic arm which was activated by a hall effect sensor, retrieval of the tesseract is a possibility. Using multiple proximity sensors travelling in a straight line alongside of the power plant walls is also another possibility. Using four large wheels and a rear wheel drive system, traversing over conduits has been resolved. In addition, using an arm with a ramp, has proved to resolve the issue of lifting the power pyramid. Using a sequence of wheel and arm movements, AMANDA has proven that it can drop the tesseract underneath the power pyramid and drop the pyramid, with a slight modification of the positioning of the robot. With all the mechanical components put together, the only issue that remains is programming AMANDA to perform the task effectively and consistently.

Therefore, with more time, AMANDA is capable of being an efficient solution to solving the problem presented by Tesseract Power Co.

**Recommendations**

There are multiple recommendations that would improve the overall performance of AMANDA which comprise of both mechanical and programmable suggestions.

Hardware Recommendations

Based on the performance of both the infrared and hall effect sensor, it is recommended that they are to be replaced with a more accurate model. During the testing phase of the project, the range at which the hall effect sensor detects a change in magnetic field is very small. The difference between the sensor detecting a magnet and not detecting a magnet was very minimal (all values can be found within the product development file). Also, the range at which the sensor noticed a magnetic field was also very limited. With these two limitations, it was very difficult to detect the tesseract with accuracy. Therefore, it is recommended that the hall effect sensor model that was used during the prototyping phase should be replaced with a more effective model that can provide better values.

On the other hand, the infrared sensor provided was too sensitive. The infrared sensor detected infrared values with an approximate 360-degree field due to infrared reflections against multiple surfaces. The solution the resolve this issue was to produce a cover where the sensor was mounted to that minimized the field of view at which the sensor can detect. A recommendation to further improve upon this issue is to use a sensor that comes with a cover which works 100 % of the time. Or use one that is designed to detect infrared in a straight line. These sensors would provide consistent values and would minimize the amount of errors that may arise.

In addition, it is recommended that a proximity sensor should be mounted at the rear end of the robot. Mounting a proximity sensor would resolve the issue of backing into a wall during the drop off phase of the code. During the performance, the robot reversed into the wall while it tried to reverse and place the pyramid onto the cube. To resolve this issue, it is recommended that an additional proximity sensor should be mounted on the rear end of the robot and programmed such that before the drop off phase, check to see if there is a wall behind it. If there is then the robot should move forward enough such that while it is completing the drop off phase of the code, that it will not run into the wall. Therefore, it is recommended that an additional proximity sensor should be mounted at the rear end of the robot.

Finally, the last recommendation is to develop a suspension system for the robot. Since there was an issue with motor speeds at the beginning of the competition, they were increased enough such that AMANDA would be able to go over the bumps. The increased motor speeds however conflicted with the hall effect sensor as AMANDA would drive too fast for the sensor to detect the tesseract. An addition of a suspensions system would improve the amount of friction between the wheels and the ground as there would be constant contact between them. The suspension system would then assist in the traversal over the conduits at a lower motor speed and would potentially be slow enough such that the hall effect sensor would be able to detect the tesseract. Also, two battery packs may be used to separately power each motor, which would result in a better performance overall as they would not lose power.

Changing the model of the infrared sensor; the hall effect sensor; adding an additional proximity sensor at the rear end of the robot; and finally adding a suspension system are four hardware recommendations that would greatly improve upon the quality of the final design.

Software Recommendation

Software remains to be the main issue associated with the poor performance of the overall design. With the hardware recommendations taken, programming the robot would be a lot simpler. The following recommendations will assume that the hardware recommendations were taken and implemented.

The initial issue faced was that AMANDA drove too fast for the hall effect sensor to detect the tesseract. Reducing the motor speeds was not a viable solution, as lowering them resulted in the robot getting stuck on the conduits. Therefore, it is recommended that a more accurate hall effect sensor would be used to provide more accurate results when testing for whether the sensor detects the tesseract. The major issue was that during the testing phase, the values the hall effect sensor read were constantly changing when it detected the tesseract and when it did not. Along with altering values, the difference between the values that differentiated whether the hall effect sensor detected the tesseract were very small. Therefore, it was very difficult to determine the correct range at which the hall effect sensor would consistently detect the tesseract. Additionally, an amplifying circuit may be applied to increase the distinction between on and off values, resulting in better performance. Therefore, along with changing the hall effect sensor, it is recommended that additional tests should be conducted to find a concrete range at which the hall effect sensor will consistently detect the tesseract. In addition an amplifying circuit may be applied to assist the issue as well.

Further tests for the proximity sensors are also recommended. Due to the modified motor speeds during the competition, the proximity sensors may not have been able to keep up with the changing distances as quickly. Additionally, the ranges which determine whether AMANDA drives straight must be retested as they were not as effective. Therefore, further tests must be conducted to improve on the effectiveness of the proximity sensors while driving straight along the wall.

Additionally, with the changed infrared sensor, it is recommended that it should be tested for the new values at which it detects the power pyramid. Further tests should also be conducted to improve upon the accuracy of detecting the pyramid. During the performance, it was observed that the robot would detect infrared readings while it was not facing the pyramid, and occasionally the robot detected values when it should not have. It was previously recommended that the sensor should be replaced, or the cover should be improved upon however, further tests should be conducted to determine whether those solutions solve the issue.

Finally, encoder counts and motor speeds should be tested to improve on the driving system of the robot. During the competition, it was observed that while AMANDA was panning for the power pyramid, it was not sweeping left and sweeping right at the same rate. This may be cause by the encoders not starting at zero, or simply an error in the code. Additionally, motor speeds should be adjusted to effectively scan for both the tesseract and the power pyramid. This modification would lower the speed/response time at which the sensors would need to detect the magnetic field or infrared sensor. Lowering the response time required then results in optimized sensors and potentially eliminate the need to replace them.

With both hardware and software recommendations applied, AMANDA would be a fully functioning solution in resolving Tesseract Power Co.’s problem.

References

Bayne, G. (n.d.). How Does a Limit Switch Work? Retrieved April 4, 2017, from http://sciencing.com/limit-switch-work-4924572.html

IR (Infrared) Obstacle Detection Sensor Circuit. (2015, February 02). Retrieved April 04, 2017, from http://www.electronicshub.org/ir-sensor/

What is an Ultrasonic Sensor? (n.d.). Retrieved April 04, 2017, from http://education.rec.ri.cmu.edu/content/electronics/boe/ultrasonic\_sensor/1.html

Woodform, C. (2017, March 30). How Hall effect sensors and probes work. Retrieved April 05, 2017, from http://www.explainthatstuff.com/hall-effect-sensors.html

**Appendices**

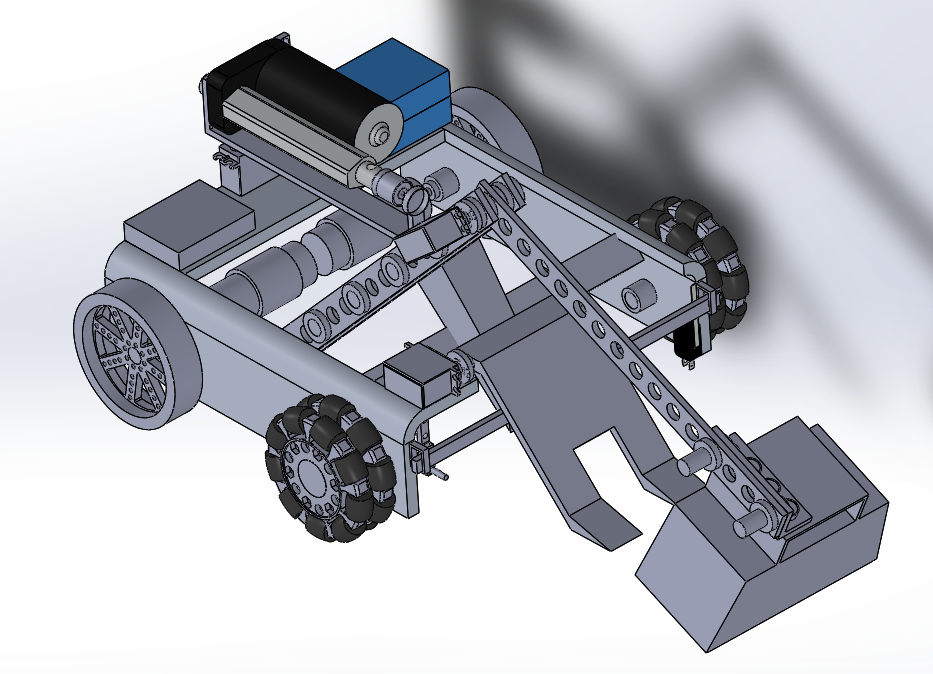
****

Figure 1: Isometric View of the final design

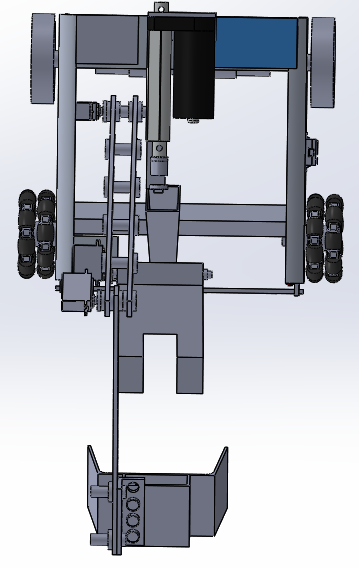


Figure 2: Top view of the final design

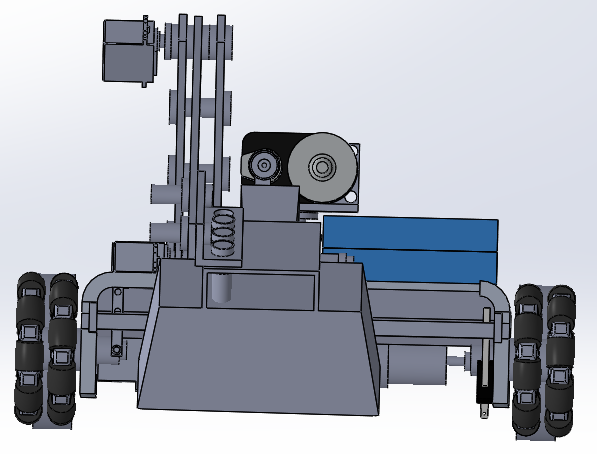


Figure 3: Front view of the final design

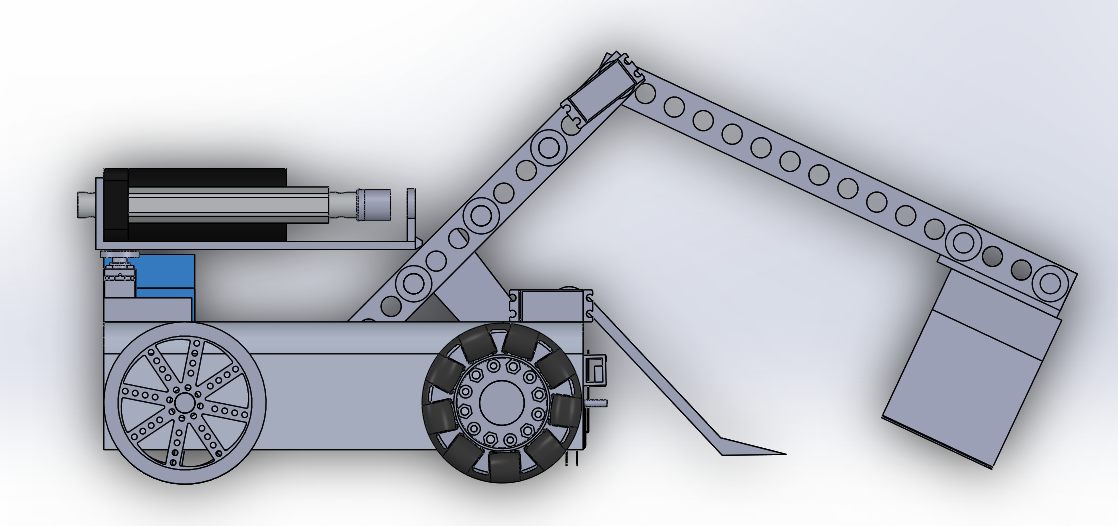


Figure 4: Side view of the final design

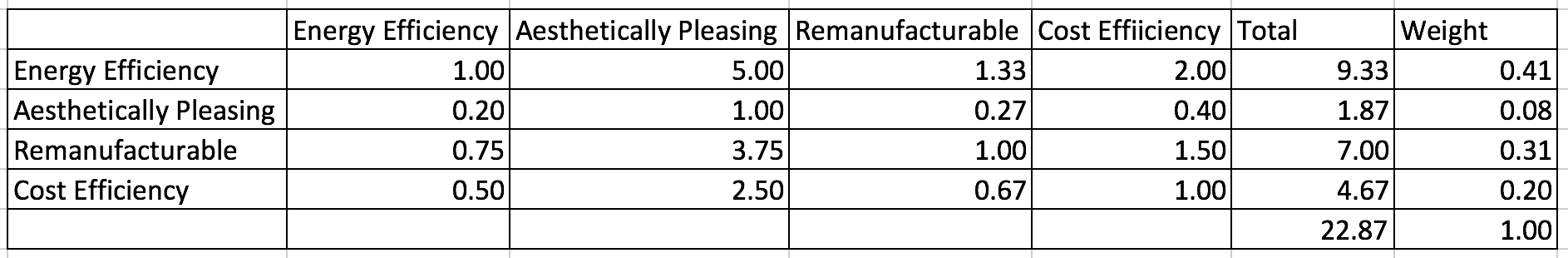


Figure 5: A Decision Matrix of top secondary objectives

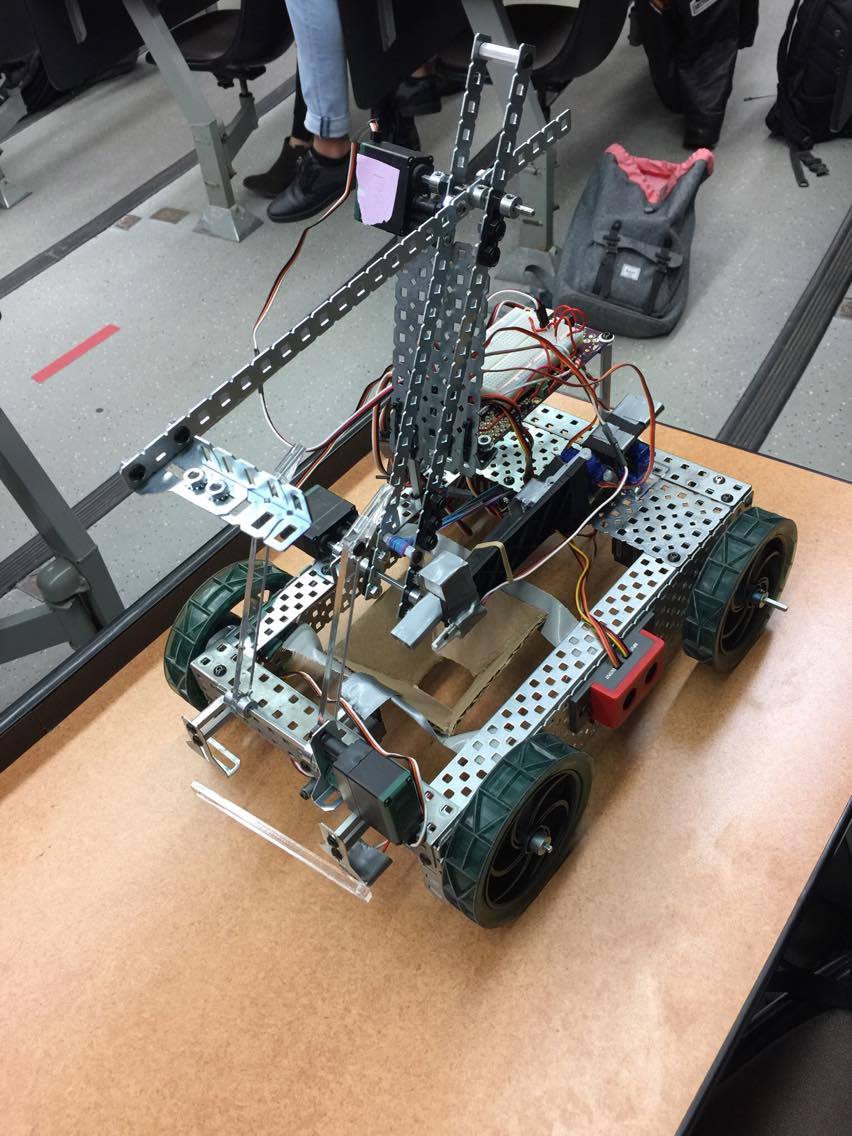


Figure 6: Isometric view of the final prototype.

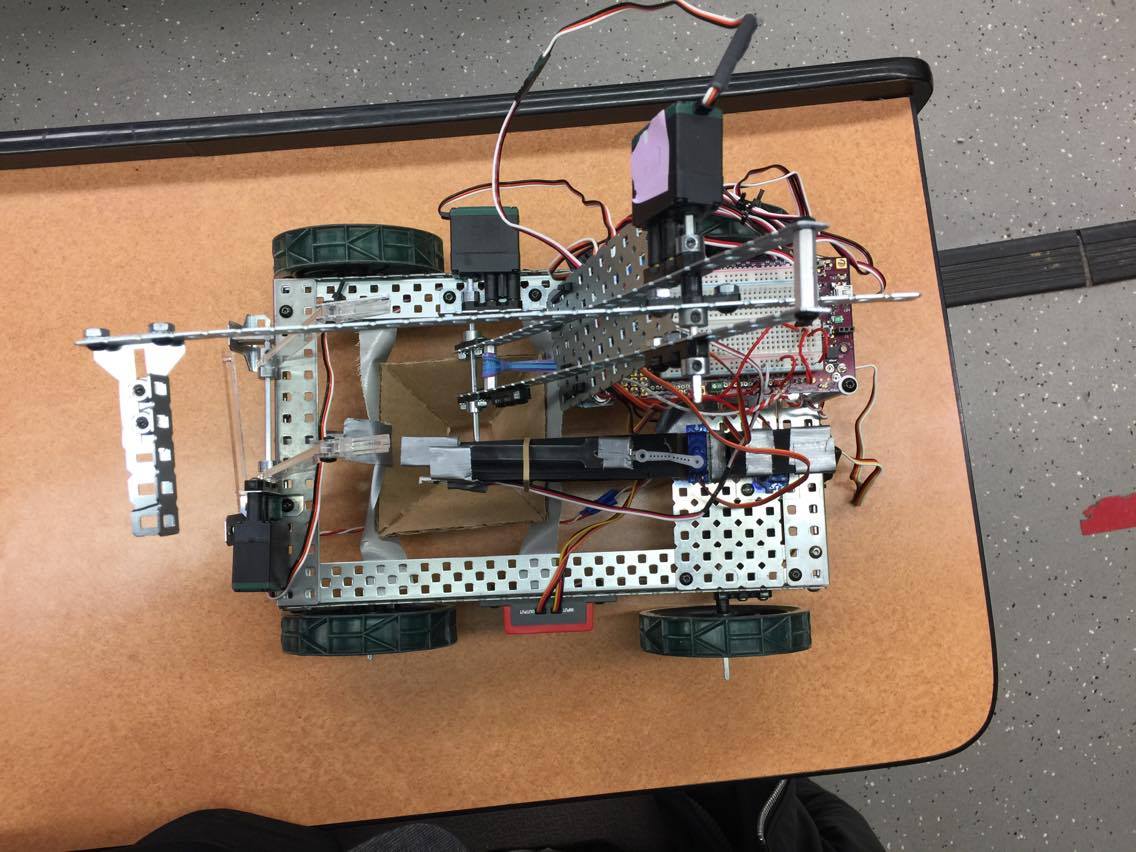


Figure 7: Top view of the final prototype.

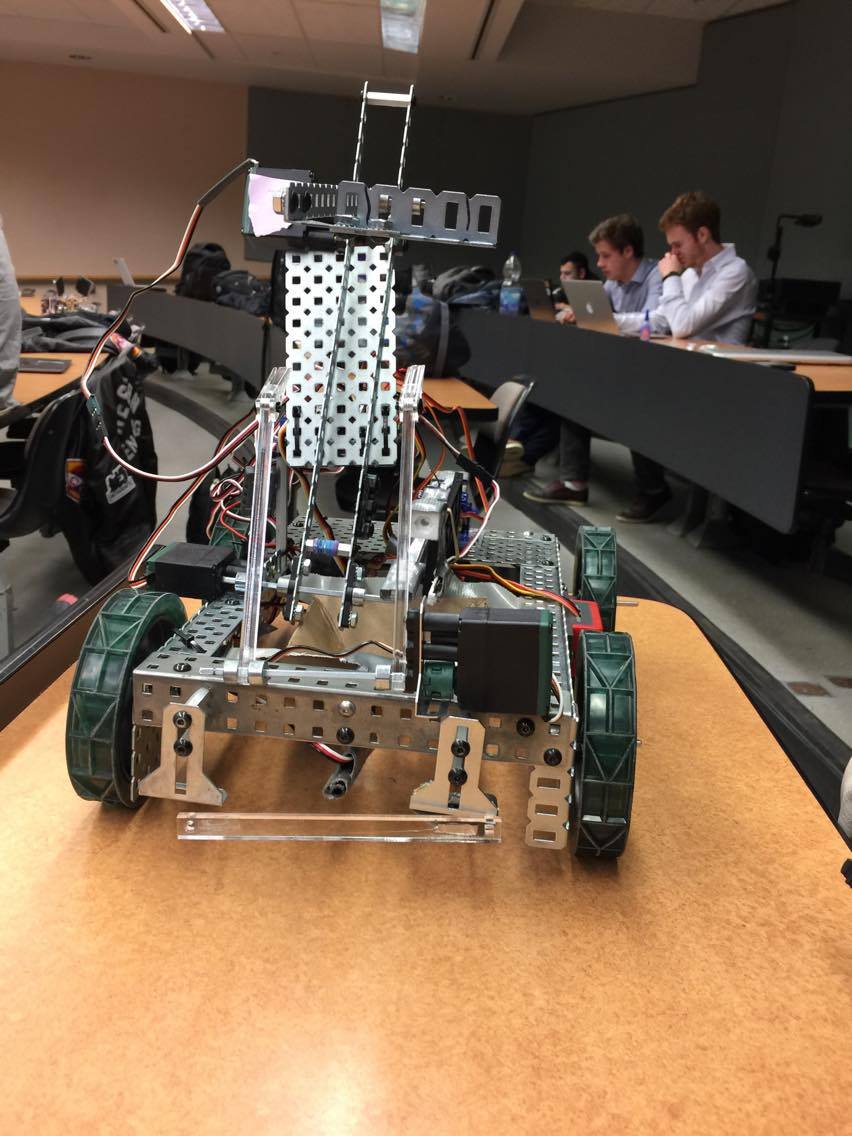


Figure 8: Front view of the final prototype.

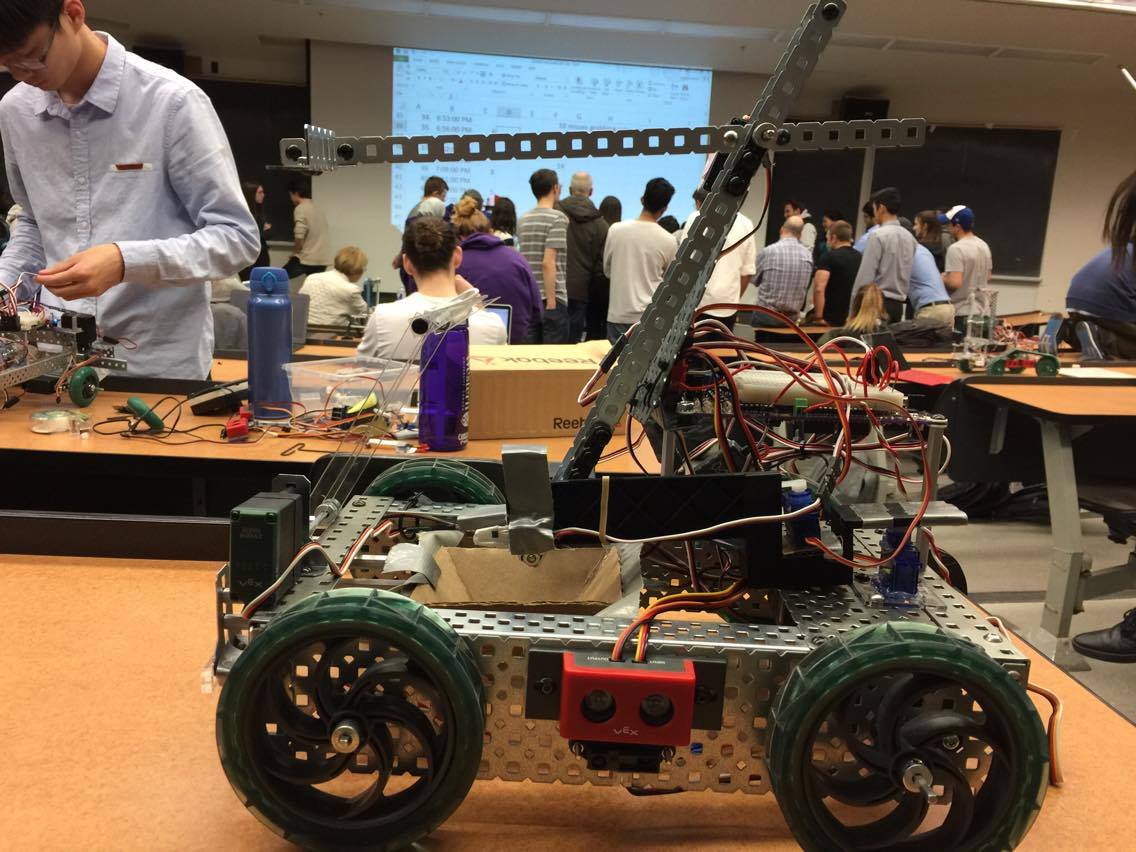


Figure 9: Side view of the final prototype.



Figure 10: AMANDA was traversing along the wall looking for the tesseract with the hall effect sensor.



Figure 11: AMANDA has found the power pyramid and has begun picking it up.

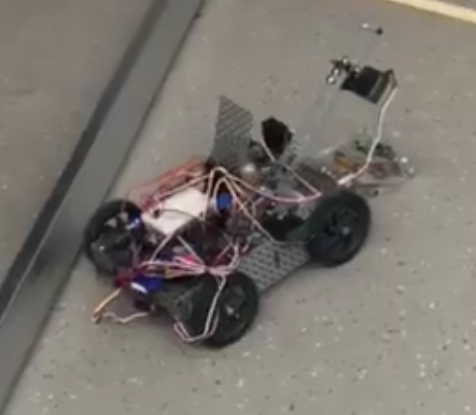


Figure 12: AMANDA has successfully lifted the power pyramid onto the ramp.

The following links are where each finalize component can be purchased:

Wheels: <http://www.robotshop.com/ca/en/actobotics-4-heavy-duty-wheel.html>

Motors: <http://www.robotshop.com/ca/en/12v-165rpm-6805oz-in-precision-planetary-gearmotor.html>

Servos: <http://www.robotshop.com/ca/en/hitec-hs645mg-servo-motor.html>

Infrared Proximity Sensor: <https://www.adafruit.com/products/164>

Infrared Sensor: <https://www.adafruit.com/products/157>

Hall Effect Sensor: <https://www.adafruit.com/product/158>

Linear Actuator: <https://www.amazon.com/Stroke-Linear-Actuator-Pounds-Maximum/dp/B00OGQLN14>

Electromagnet: <https://apwelectromagnets.com/em075-12-112.html>

Battery:<http://www.dx.com/p/1268-12v-6800mah-rechargeable-li-polymer-battery-blue-black-12v-227733?tc=CAD&gclid=CLnZs5HPjtMCFQEJaQod86IL2Q#.WOWRtjvyvD4>

Final Design Code:

// MSE 2202B

// Group 10

// Final Prototype Code

// Including Files

#include <Servo.h>

#include <EEPROM.h>

#include <I2CEncoder.h>

#include <Wire.h>

#include <uSTimer2.h>

// Pyramid Tracking Variables

int AbsoluteMotorSpeed = 0;

int firstpan = 0;

int FoundPyramidCounter = 0;

int EncoderCountToFindCube = 0;

int ui\_Left\_Motor\_Speed = 1500;

int ui\_Right\_Motor\_Speed = 1500;

int temp = 0;

int prevS = 0;

int currentS = 0;

int interval = 100;

int JumpInterval = 2000;

int PyramidCharacter1 = 73; //I 65

int PyramidCharacter2 = 79 ; //O 69

int PanRight = 1;

unsigned long JumpStart = 0;

// Ultrasonic Variables

const int ci\_Ultrasonic\_Ping = 6; //input plug

const int ci\_Ultrasonic\_Data = 5; //output plug

unsigned long ul\_Echo\_Time;

//Hall Reference

unsigned long startT = 0;

// Motor Variables

I2CEncoder encoder\_RightMotor;

I2CEncoder encoder\_LeftMotor;

int l\_Left\_Motor\_Position;

int l\_Right\_Motor\_Position;

Servo servo\_RightMotor;

Servo servo\_LeftMotor;

const int ci\_Right\_Motor = 8; // Right

const int ci\_Left\_Motor = 9; // Left

int ui\_motor\_speed = 1800;

int ui\_left\_motor\_speed;

int ui\_right\_motor\_speed;

int ui\_left\_motor\_offset = 75;

int ui\_right\_motor\_offset = 0;

// Limit Switch Variables

const int LmtSw = 7;

// Magnet Arm Variables

Servo servo\_SpinServo;

Servo servo\_PushServo;

const int ci\_Spin\_Servo = 2;

const int ci\_Push\_Servo = 3;

// Pyramid Arm Variables

Servo servo\_ArmServo;

Servo servo\_GripServo;

const int ci\_Arm\_Servo = 11;

const int ci\_Grip\_Servo = 12;

// Ramp Variables

Servo servo\_RampServo;

const int ci\_Ramp\_Servo = 10;

// Hall Effect Variables

int hallState = 0;

const int hallPin = 5;

// IR Sensor Variables

const int IRsensor = 0;

// Counters

int stage = 1;

int count = 0;

void setup() {

// Pyramid Tracking set up

Wire.begin();

delay(4000);

Serial.begin(2400);

pinMode(ci\_Right\_Motor, OUTPUT);

pinMode(ci\_Left\_Motor, OUTPUT);

AttachMotors();

encoder\_RightMotor.init(1.0 / 3.0 \* MOTOR\_393\_SPEED\_ROTATIONS, MOTOR\_393\_TIME\_DELTA);

encoder\_RightMotor.setReversed(false); // adjust for positive count when moving forward

encoder\_LeftMotor.init(1.0 / 3.0 \* MOTOR\_393\_SPEED\_ROTATIONS, MOTOR\_393\_TIME\_DELTA);

encoder\_LeftMotor.setReversed(true); // adjust for positive count when moving forward

startT = millis();

encoder\_LeftMotor.zero();

encoder\_RightMotor.zero();

pinMode(hallPin, INPUT);

pinMode(ci\_Spin\_Servo, OUTPUT);

pinMode(ci\_Push\_Servo, OUTPUT);

pinMode(ci\_Arm\_Servo, OUTPUT);

pinMode(ci\_Grip\_Servo, OUTPUT);

pinMode(ci\_Ramp\_Servo, OUTPUT);

pinMode(ci\_Ultrasonic\_Ping, OUTPUT);

pinMode(ci\_Ultrasonic\_Data, INPUT);

// Attaching Motors, Spin/Push servos, Arm/Grip servos

// IR Sensor

pinMode(IRsensor, INPUT);

// Limit Switch

pinMode(LmtSw, OUTPUT);

digitalWrite(LmtSw, 1);

AttachArmGrip();

AttachSpinPush();

delay(1000);

// Writing servos to intial positions

servo\_PushServo.write(0);

servo\_SpinServo.write(95);

servo\_ArmServo.write(90);

servo\_GripServo.write(90);

delay(2000);

DetachArmGrip();

// Writing ramp to initial position

servo\_RampServo.attach(ci\_Ramp\_Servo);

servo\_RampServo.write(180);

delay(1000);

servo\_RampServo.detach();

}

void loop()

{

if (stage == 1) // Moving until Hall Effect finds the cube

{

Serial.println("Stage 1");

delay(10);

int HallVal = analogRead(5);

// Prints the read from the Hall Effect

Serial.println(HallVal);

if (HallVal < 450 || HallVal > 490) // If the Hall Effect sees the cube

{

ui\_motor\_speed = 1500; // Motors stops

servo\_LeftMotor.writeMicroseconds(ui\_motor\_speed); // Sets motor speeds

servo\_RightMotor.writeMicroseconds(ui\_motor\_speed);

stage++; // Increments stage

}

else

{

StraightAgainstWall(); // Keeps motors at speed of 1700, to keep it moving across the wall

delay(150);

}

}

else if (stage == 2) // Picking up the Cube

{

Serial.println("Stage 2");

AttachSpinPush(); // Attachs the Spin and Push Servo

delay(1000);

servo\_PushServo.write(180); // Attempts to grab cube

delay(1000);

servo\_SpinServo.write(20); // Spins cube towards the middle

delay(1000);

DetachSpinPush(); // Detachs the Spin and Push Servos

stage++; // Increments stage

encoder\_LeftMotor.zero(); // Zeroes the encoders

encoder\_RightMotor.zero();

}

else if (stage == 3) // Panning until we touch/find the Pyramid

{

Serial.println("Stage 3");

unsigned long currentT = millis();

if (currentT - JumpStart >= JumpInterval)

{

JumpStart = currentT;

ui\_Left\_Motor\_Speed = 1800;

ui\_Right\_Motor\_Speed = 1800;

delay(500);

}

else if (currentT - startT >= interval) //checking IR on set interval to allow sensor to react

{

startT = currentT;

if (Serial.available() > 0) // Checking if serial is available

{

currentS = Serial.read(); // Reading from IR sensor

}

if (currentS <= 0) // If the sensor doesn't read anything -> pan

{

ui\_Left\_Motor\_Speed = PanSpeed (); //Pan Function to be built

AbsoluteMotorSpeed = ui\_Left\_Motor\_Speed - 1500;

ui\_Right\_Motor\_Speed = 1500 - AbsoluteMotorSpeed;

Serial.println("STUCK IN IF < 0 PAN");

}

else if (currentS == PyramidCharacter1 || currentS == PyramidCharacter2) // If the sensor reads the right value

{

ui\_Left\_Motor\_Speed = 1800; // Moves forward

ui\_Right\_Motor\_Speed = 1800;

if (digitalRead(LmtSw) == 0) // We hit the pyramid

{

ui\_Left\_Motor\_Speed = 1500; // stops

ui\_Right\_Motor\_Speed = 1500;

stage++;

}

}

else

{

ui\_Left\_Motor\_Speed = PanSpeed (); //Pass whatever to Encoder count

AbsoluteMotorSpeed = ui\_Left\_Motor\_Speed - 1500;

ui\_Right\_Motor\_Speed = 1500 - AbsoluteMotorSpeed;

Serial.println("STUCK IN ELSE PAN");

if (digitalRead(LmtSw) == 0) //We hit the wall

{

ui\_Left\_Motor\_Speed = 1400; // Will back up and turn

ui\_Right\_Motor\_Speed = 1200;

}

}

currentS = 0;

servo\_LeftMotor.writeMicroseconds(ui\_Left\_Motor\_Speed); // Sets motor speeds

servo\_RightMotor.writeMicroseconds(ui\_Right\_Motor\_Speed);

}

}

else if (stage == 4) // Picking up the Pyramid and putting the cube underneath

{

Serial.println("GOT IT");

// Attachs the Ramp Servo, and writes the initial position, then detachs it

servo\_RampServo.attach(ci\_Ramp\_Servo);

servo\_RampServo.write(90);

delay(5000);

servo\_LeftMotor.writeMicroseconds(1200); // Backs up slightly so the pyramid is at the right distance

servo\_RightMotor.writeMicroseconds(1200);

delay(300);

servo\_LeftMotor.writeMicroseconds(1500);

servo\_RightMotor.writeMicroseconds(1500);

servo\_RampServo.write(25); // Moves ramp down, then detaches ramp

delay(5000);

servo\_RampServo.detach();

AttachArmGrip(); // Attach Arm/Grip

DetachMotors();

servo\_GripServo.write(70); // Puts the arm and pushing arm in the right position

servo\_ArmServo.write(120);

delay(5000);

servo\_ArmServo.write(150); // Brings the Arm down and pushes the pyramid into the robot

servo\_GripServo.write(90);

delay(5000);

servo\_GripServo.write(120); // Further pushing the pyramid into the robot

delay(5000);

AttachSpinPush(); // Attach Push/Spin

servo\_PushServo.write(180); // Brings the servo with the arm outwards, then spins it inside to knock off the cube

servo\_SpinServo.write(150);

delay(5000);

servo\_SpinServo.write(0);

delay(5000);

DetachSpinPush(); // Detaching Push/Spin

AttachMotors(); // Attaching Motors

servo\_LeftMotor.writeMicroseconds(1200); // Backs up slighlty, then stops

servo\_RightMotor.writeMicroseconds(1200);

delay(3000);

servo\_LeftMotor.writeMicroseconds(1500);

servo\_RightMotor.writeMicroseconds(1500);

delay(5000);

servo\_GripServo.write(115);

delay(5000);

servo\_LeftMotor.writeMicroseconds(1200); // Moves the arm down then backs up and stops

servo\_RightMotor.writeMicroseconds(1200);

delay(300);

servo\_LeftMotor.writeMicroseconds(1500);

servo\_RightMotor.writeMicroseconds(1500);

delay(5000);

servo\_GripServo.write(112); // Moves the arm down then backs up and stops

delay(5000);

servo\_LeftMotor.writeMicroseconds(1200);

servo\_RightMotor.writeMicroseconds(1200);

delay(300);

servo\_LeftMotor.writeMicroseconds(1500);

servo\_RightMotor.writeMicroseconds(1500);

delay(1000);

servo\_GripServo.write(110); // Moves the arm down then backs up and stops

delay(5000);

servo\_LeftMotor.writeMicroseconds(1200);

servo\_RightMotor.writeMicroseconds(1200);

delay(300);

servo\_LeftMotor.writeMicroseconds(1500);

servo\_RightMotor.writeMicroseconds(1500);

delay(5000);

servo\_GripServo.write(70); // Puts the Arm back up

servo\_ArmServo.write(90);

stage++; // Increments the stage

}

if (stage == 5)

{

servo\_LeftMotor.writeMicroseconds(1500); // Stops the motors from moving

servo\_RightMotor.writeMicroseconds(1500);

}

}

// Function to attach the arm and grip servos

void AttachArmGrip()

{

servo\_ArmServo.attach(ci\_Arm\_Servo);

servo\_GripServo.attach(ci\_Grip\_Servo);

}

// Function to detach the arm and grip servos

void DetachArmGrip()

{

servo\_ArmServo.detach();

servo\_GripServo.detach();

}

// Function to attach the spin and push servos

void AttachSpinPush()

{

servo\_SpinServo.attach(ci\_Spin\_Servo);

servo\_PushServo.attach(ci\_Push\_Servo);

}

// Function to detach the spin and push servos

void DetachSpinPush()

{

servo\_SpinServo.detach();

servo\_PushServo.detach();

}

// Function to attach the motors

void AttachMotors()

{

servo\_LeftMotor.attach(ci\_Left\_Motor);

servo\_RightMotor.attach(ci\_Right\_Motor);

}

// Function to detach the motors

void DetachMotors()

{

servo\_LeftMotor.detach();

servo\_RightMotor.detach();

}

// Function that makes the robot move straight along a wall

void StraightAgainstWall()

{

Ping(); // Calling Ping Function

Serial.println(ul\_Echo\_Time / 58); // Printing how many cm robot is away from wall/object

if ((ul\_Echo\_Time / 58) < 11) // If it is less than 11 cm away it is too close

{

Serial.println("CLOSE TO WALL");

ui\_left\_motor\_speed = 1700; // Turns left motor faster to adjust

ui\_right\_motor\_speed = 1600;

}

else if ((ul\_Echo\_Time / 58) > 11) // If it is more than 11 cm away it is too far

{

Serial.println("FAR FROM WALL");

ui\_left\_motor\_speed = 1600; // Turns Right motor faster to adjust

ui\_right\_motor\_speed = 1700;

}

else // If it is exactly 11 cm away

{

ui\_left\_motor\_speed = 1700; // Both motors run at the same speed

ui\_right\_motor\_speed = 1700;

}

if (digitalRead(LmtSw) == 0) // If the robot hits the limit switch at all during this time

{

ui\_left\_motor\_speed = 1400; // Attempts to turn and back out of the corner/wall

ui\_right\_motor\_speed = 1350;

}

servo\_LeftMotor.writeMicroseconds(ui\_left\_motor\_speed);

servo\_RightMotor.writeMicroseconds(ui\_right\_motor\_speed);

}

// Pan Function

int PanSpeed ()

{

int motorSpeed = 1500;

if (firstpan == 0) // If it is the first time panning(after cube pickup)

{

encoder\_LeftMotor.zero(); // Set encoder counts to zero

encoder\_RightMotor.zero();

firstpan = 1;

}

l\_Left\_Motor\_Position = encoder\_LeftMotor.getRawPosition(); // Gets position of encoder counts

l\_Right\_Motor\_Position = encoder\_RightMotor.getRawPosition();

/\*Serial.print(encoder\_LeftMotor.getRawPosition());

Serial.print(encoder\_RightMotor.getRawPosition()); // Serial prints for encoders

Serial.print("Encoders L: ");

Serial.print(l\_Left\_Motor\_Position);

Serial.print(", R: ");

Serial.println(l\_Right\_Motor\_Position);\*/

//Serial.print("NO COUNT");

Serial.println(l\_Right\_Motor\_Position);

// These encoder positions and speeds will all be dependent upon the right motor since it is more reliable for encoder counts on our robot

if (l\_Right\_Motor\_Position < 2000 && PanRight == 1) // If the encoder count has not yet reached 2000 and it is panning right

{

motorSpeed = 1700; // Motor speed is set to forward

}

else if (l\_Right\_Motor\_Position >= 2000 && PanRight == 1) // If the encoder count has reached 2000 and it is still panning right

{

PanRight = 0; // Change which way it should pan

motorSpeed = 1200; // Make motor speed backwards

encoder\_LeftMotor.zero(); // Zero encoder counts

encoder\_RightMotor.zero();

}

else if (l\_Right\_Motor\_Position >= -2000 && PanRight == 0) // If the encoder count has not yet reached -2000 and it is panning left

{

motorSpeed = 1200; // Motor speed is set to forward

}

else if (l\_Right\_Motor\_Position < -2000 && PanRight == 0) // If the encoder count has reached -2000 and it is still panning left

{

PanRight = 1; // Change which way it should pan

motorSpeed = 1800; // Make motor speed forward

encoder\_LeftMotor.zero(); // Zero encoder counts

encoder\_RightMotor.zero();

}

else // If none of these conditions apple (which they should)

{

motorSpeed = 1500; // Set motor speed to stop

}

return motorSpeed; // Return the motor speed

}

// Ping Function

void Ping()

{

digitalWrite(ci\_Ultrasonic\_Ping, HIGH);

delayMicroseconds(10); // The 10 microsecond pause where the pulse in "high"

digitalWrite(ci\_Ultrasonic\_Ping, LOW);

ul\_Echo\_Time = pulseIn(ci\_Ultrasonic\_Data, HIGH, 10000); // Sets ul\_Echo\_Time to the time that it takes from when the pin goes HIGH until it goes LOW

}