Northeastern University
College of Engineering

Cornerstone of Engineering II

Agent Lux

SUBMITTED BY:

Annabelle Lilly, Dylan Mouri, Dean Patel

Section 6

Report Submitted: **April 17, 2019**

Annabelle Lilly, lilly.a@husky.neu.edu
Dylan Mouri, mouri.d@husky.neu.edu
Dean Patel, patel.dea@husky.neu.edu
291 St. Botolph Street
Boston, MA 02115

April 17, 2019
Dr. Constantine Mukasa
368 Snell Engineering
Boston, MA 02130
c.mukasa@neu.edu

Dear Professor Mukasa,

Our team of agents was tasked by the commander of the National Espionage Unit (NEU) with engineering an autonomous robot capable of seeking out a small light source in a dark room. Our team has accomplished this task, and this report will explain the details of our successful robot, Agent Lux.

Since the introduction of this project, our team has been working tirelessly to ensure the successful creation of the robot, in order for the NEU to be able to find chemical threats more effectively. Over the weeks our team has brainstormed different solutions, created an effective code, and tested the prototype a multitude of times to ensure the success of the project. With the fully functioning robot, the NEU will effectively complete missions without putting agents in dangerous situations.

Our team is very excited about our robot, and we are eager to share the details about how we created Agent Lux. We believe our robot is not only functioning but also efficient, and will be reproduced by the NEU to be used in many future missions.

Sincerely,
Annabelle Lilly

Dylan Mouri

Dean Patel

Table of Contents

List of Tables

List of Figures

Abstract

Our team of agents was given the task of creating an autonomous light seeking robot, capable of finding a lamp from across the room and celebrating when the light source was found, in addition to having an emergency stop button. Our solution was a robot with two distance sensors to avoid walls and objects, as well as four photoresistors for sensing light. There was one on each side of the robot, as well as one on top. The robot would compare the light values read by the photoresistors and move in the direction of greatest light. When the top photoresistor read a value 1.05 times greater than the other four, the robot would celebrate finding the light source by playing the Pirates of the Caribbean theme song. On demonstration day, our solution did not work as intended, as our robot celebrated too early. However, our team worked very hard, and learned valuable lessons from this project. We know exactly what went wrong and how to improve in the future. This report contains all of the information about the construction of our robot.

1. Failure Analysis

1.1 Problems Discovered

While testing the robot, the team observed positive results. The robot was tested in a small, dark room with a cell phone flashlight. The robot was able to successfully traverse the room and find the light within 90 seconds.

Upon demonstrating the robot, the team did not achieve the desired results. The room was brighter than expected, and the robot celebrated as if it had found the light source immediately, as the top photoresistor was reading a value 1.05 times higher than the rest (as dictated by the code). In an effort to salvage the situation quickly, the team decided to cover up the top sensor with tissue paper in order to attempt to fix this problem. The robot was then able to move to the other end of the room, in the general direction of the light. Unfortunately, the robot then ran into the back wall, and was not able to get away from this wall. It would move in order to get away from the wall, but would not move far enough, and repeatedly ran into the wall. The robot met the functions and objectives of giving off a signal under light, a stop button, and having aesthetic portions. Unfortunately, the robot failed in the objective of finding the light in the allotted time and not becoming immobilized.

1.2 Solutions Identified

Upon demonstrating the robot, the team learned a few key lessons. To start, the light difference between the top photoresistor and other photoresistors required to trigger the "found the light" celebration was too low. In other words, the robot was tested in too dark of a room to start. A larger difference between the top photoresistor and the other photoresistors should have been coded. Additionally, the robot was tested in a much smaller room than the demonstration room. The testing for the robot should have been conducted in a larger space in order to simulate the actual conditions of the demonstration. Also, it is suspected that the turn time of the robot when sensing an object was too short, as the robot repeatedly turned into the same wall. This should have been increased in order to ensure the robot would get away from the wall. Furthermore, it is overwhelmingly clear that the team should have put more consideration into the decision to purchase new parts. The parts provided by the Sparkfun kit were not reliable, which was known information. The work in meetings often ended up being about coding around issues with the hardware, rather than replacing the materials and coding accordingly. If more consistent and reliable photoresistors and distance sensors had been integrated, the team is confident the robot would have been successful.

Overall, this project was not as simple as the team thought it would initially be. At the beginning, it seemed rather easy, as the majority of the project had been completed during the previous semester. However, upon working on the project, many issues arose. The initial code for the autonomous movement, the initial code for the light sensing, and the initial aesthetic components all failed. Fortunately, the team remained level-headed and worked through all of these issues until a functioning robot was developed in testing. Even when the demonstration did not go as planned immediately, covering the top photoresistor with tissue paper to stop the premature celebration was at least a clever manipulation that demonstrated understanding of how the robot worked.

2. Introduction

The issues found with the distance sensors and the photoresistors during the first trial of this project were immediately addressed upon the restarting of this project, in order to create a successful light seeking robot.

2.1 Background

Initial research for this iteration of the project was concerned with finding an appropriate foundation that would provide the robot functionality. The circuitry and general coding had been finalized in the first iteration, with the help of online resources such as the Sparkfun Light-Seeking Robot [1] and the Science Buddies Light-Tracking robot [2]. These sources were used for inspiration; however, the vast majority of the project was conducted without the use of these references. This was because many of the examples found through research could not entirely be translated for this mission's set of objectives and constraints. For example, most robots looked at during initial research did not have any form of a distance sensor to avoid obstacles or walls. Theoretically while the device ought to find the light regardless of a distance sensor, it was important to the group that extra security was added to ensure the robot could safely and consistently find light. Thus, the background research allowed the group to set up the foundation for an Arduino-based light sensing capability in terms of the circuitry, but there remained work to be done by integrating new components and new code.

Another area of research, primarily during the second iteration of the project, was concerned with finding appropriate hardware components that were compatible with the foundation. This required a thorough understanding of how the photoresistors interacted with the Arduino board, since the group wanted to be able to replace them seamlessly with a new part and upload essentially the same code. Research of compatible light sensors was conducted and several parts sold on Adafruit.com were compared (exact part to be mentioned in methodology section). Overall, this area of research was more group-based than the area mentioned above because it required the group understand the robot's circuitry and software compatibility as well as component tolerances.

2.2 Problem Statement

The underlying problem for the National Espionage Unit (NEU) is that the agents do not currently have a system which is capable of safely and consistently finding chemical threats. While previous solutions exist, such as humans and dogs trained to seek and handle these harmful chemicals, the National Espionage Unit has decided that they do not want to expose living beings to this danger, and seeks a solution that would not risk their safety. Instead, the National Espionage Unit aims to accomplish this through the utilization of autonomous robots. Previously, our group was tasked with developing the robot's ability to autonomously move around an area without getting immobilized by anything. In this project, we are building upon our previous model to develop a system capable of detecting a light source in an enclosed, dimly-lit environment. Per the agency's requirements, we cannot spend more than fifty dollars, the robot must have an obvious red stop button, and it must detect the light source within 5 minutes.

The functions of the robot are that it needs to be able to move autonomously; specifically, it must navigate a room independently. This way, NEU agents are not required to complete the mission, but rather, the robot will do so. Additionally, this robot should detect light detect from by detecting where the chemical radiation is coming from, for NEU missions of this type. Furthermore, this robot must include a red button for emergency stopping purposes. Finally, it is necessary for this robot to give a signal when it has found a light source, with either an audio or visual signal, so the NEU is aware that the robot has completed its mission.

The objectives for this robot include that it cannot collide with other other objects or walls. This way it is not impaired and can complete its mission for the NEU. Also, the robot must be able to find a small source of light in a dimly lit room from 15 to 20 feet away, without prior knowledge of the specifics of the situation. The robot needs to accomplish the task as quickly as possible, as this is a competition with the other NEU agents, as to who can create the most effective robot. The robot must also include a noticeable signal once the light source is found, in order to alert the NEU of its success. Additionally, the robot must be aesthetically pleasing. A creative and unique element is also necessary to make the robot marketable. The robot needs to stand out from the competition. Our team has decided to include a countdown in the robot, so that it does not need to be timed, but rather times itself.

The constraints on the production of this robot include that it must cost the agents less than $50, after the base is provided. This way it is affordable and reproducible for the NEU. Additionally, due to increasing national security concerns, there is a deadline for the robot to be finished. In addition, the robot must locate the light source within five minutes. This ensures that in the field the robot will perform efficiently, as needed by the NEU for high risk tasks.

2.3 Unique Idea

As one of the members of the team was a timekeeper, transforming this process into an autonomous one (keeping with the theme of autonomy) was one idea generated to add to the functionality in a creative way.

The implementation of this feature is relatively simple in terms of both hardware and software. The circuitry is given in the Sparkfun kit and a Fritzing diagram has been taken from Sparkfun online shown below. Considering the number of pin connections the LCD requires, the team will decide between purchasing another microcontroller or purchasing a mega-board. This mainly depends on the final design and how much space can be afforded.

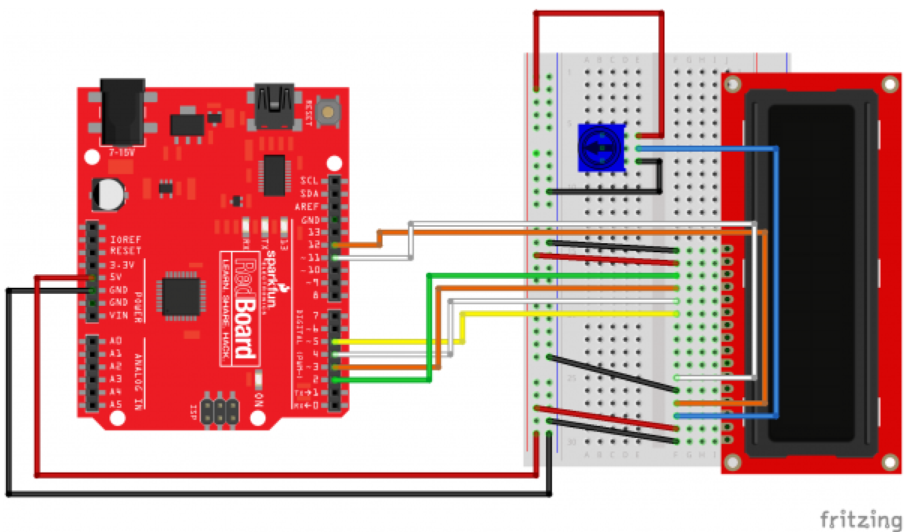


Figure 1: Fritzing Diagram for LCD Circuitry [3]

The LCD will be coded to display a countdown from 5 minutes and increment down to 0 seconds by steps of 1 second. This is a simple program that calls the lcd.print function to print the time in min:ss format (e.g. 4:31 means 4 minutes and 31 seconds remain). From background research online, it was also seen that the program must clear the display each time in quick succession (since there is a quick 1 second increment). To implement the timer feature then, the program must also use the lcd.setCursor to clear the information displayed immediately and display the next piece of information (the time left) for one second [3].

Figure 2: LCD code

```
lcd.print("5:00");

//loop to increment down by 1 second

lcd.setCursor(0,1);   // Set the cursor to the position
  // lcd.print("       "); // Erase the largest possible number
  // lcd.setCursor(0,1);   // Reset the cursor to the original position
  // lcd.print(millis()/1000); // Print our value
```

2.4 Stakeholders

This robot will aid the National Espionage unit in detecting chemical threats in a timely manner. This device was created in order to assist the NEU in their need for a way to find a chemical threat without putting lives in danger. The robot may be placed in a dark room by an agent, and within five minutes, the threat will be found by the robot. The robot will notify the agent that the source has been found, and will remain at the location of the threat. This way, any agent using the device will not have to be in harm's way, and will know the location of the threat. Thus, this device is far more ethical to use, as opposed to putting agents in the field. When the commander of the NEU gives an agent a dangerous mission, there is the ethical dilemma of the agent possibly getting harmed. With this device, the agent is far safer. Additionally, if the device is damaged during a mission, it is easy for the NEU to replace, as it is small, inexpensive and relatively simple to make.

3. Methodology

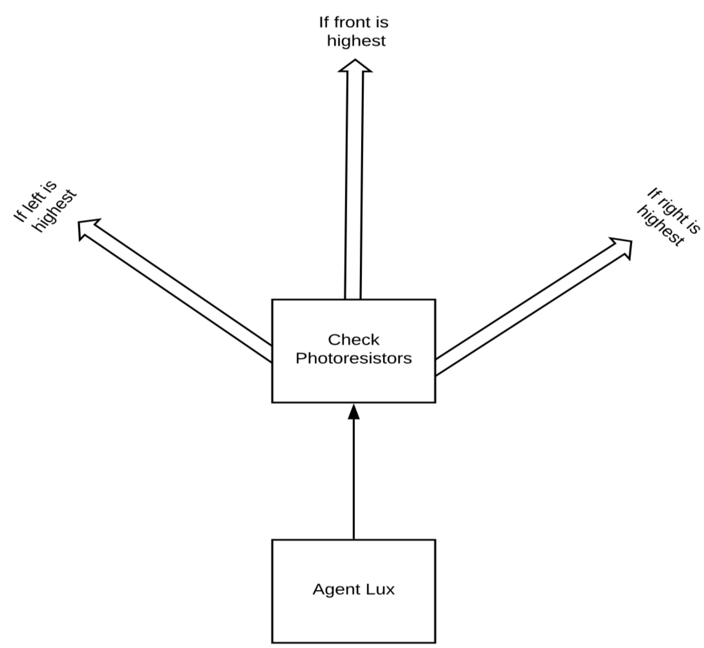
3.1 Possible Solutions

1. Solution 1: The first solution our team came up with was called the "Specific Path" solution, or the lawnmower motion. In this solution, the robot would be programmed to run through a specific back-and-forth path across a room, similar to how one may mow a lawn. The robot would follow this path until a photoresistor read a high enough value, and thus the light source was located. The robot would then give an audio signal, so it would be obvious to the NEU that the source had been found. The robot would also be equipped with an emergency stop button.

Figure 3: Lawnmower path

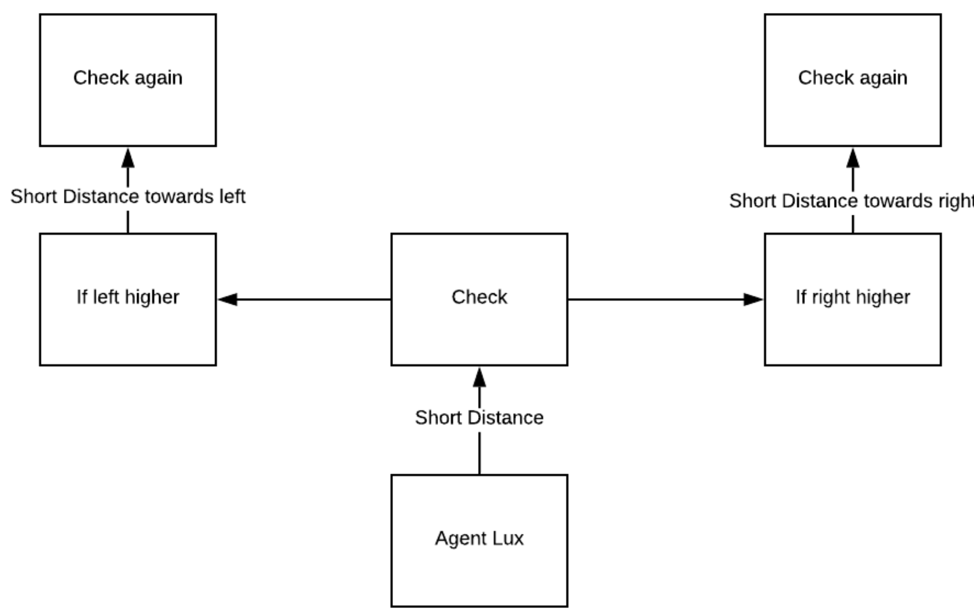
2. Solution 2: The second idea our team came up with was called the "Move Forward" motion. The robot would move forward while checking the three photoresistors on the sides of it. When one of the photoresistors read a value higher than the others by a certain percentage, the robot would turn in that direction and continue to move forward. Additionally, if the distance sensor read a low value, the robot would turn away from the obstruction. This solution would also include a noticeable audio signal that the source of light had been found when the fourth resistor on top read a high enough value. The robot would also include an emergency stop button.

Figure 4: "Move Forward" motion diagram



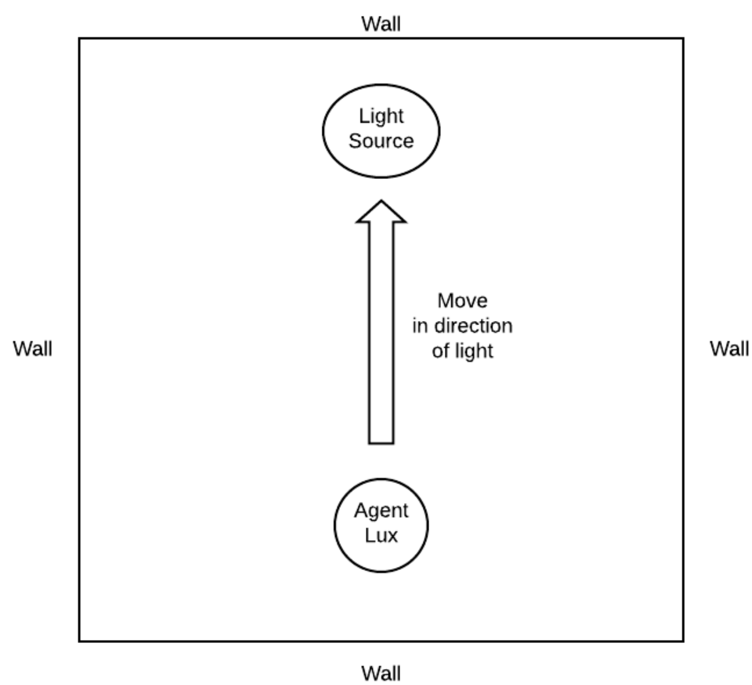
3. <u>Solution 3</u>: The final solution our team created was called the “Waddle” motion. The robot would move forward for very short distances. After each small distance, the robot would check the two photoresistors on the sides, and then move in the direction of the highest value for a short period of time. This action would continue until the robot discovered the light source. When the value read by one of the photoresistors was appropriately large, the robot has found the light source. Upon finding the light source, the robot would stop moving and relay an audio signal, informing the NEU that the threat was found. The robot would also include three distance sensors, so it would not collide with walls or objects, in addition to an emergency stop button.

Figure 5: Diagram of the “waddle” motion.



4. Solution 4: The fourth idea generated was the idea to guide the robot motion solely with the light sensor, called the "Light Sensor" motion. This meant at all times the robot would take inputs from its three light sensors and proceed in the direction of greatest light value. This was proposed because even in the event of an obstacle or wall, since it is not the light source, the robot could potentially be relied upon to maneuver out of the way. It was also considered because of its simplicity and ease of replication.

   Figure 6: Diagram of "Light Sensor" motion



3.2 Chosen Solution

Of all of the proposed solutions above, we decided to proceed forward with Solution 3, the waddle motion. Using a rank order diagram, shown in Table 1, the team determined exactly which elements were of utmost importance, and how they compared to others. With these factors considered, a rank order diagram was developed using Kepner Tregoe Analysis to determine which solutions fulfilled our required functions. All of the solutions passed, so we developed a Kepner Tregoe Decision Matrix, weighing the objectives and their potential to be achieved between each solution. From this, it was determined that Solution 3 was clearly the best option, outscoring the others. As such, we determined to proceed forward with Solution 3.

Table 1: Rank Order Diagram of Objectives

	Seeks Light	**Autonomous**	**Unique**	**Signal**	**Aesthetic**	**Sum**
Seeks Light	-	1	1	1	1	4
Autonomous	0	-	1	1	1	3
Unique	0	0	-	1	1	2
Signal	0	0	0	-	1	1
Aesthetic	0	0	0	0	-	0

Table 2: Go-No Go Diagram for must objectives

Must Objectives	**Seeks Light**	**Autonomous**
Alternative Solutions	**Go or No-Go**	
Solution 1	Go	Go
Solution 2	Go	Go
Solution 3	Go	Go
Solution 4	Go	Go

Table 3: Kepner Tregoe Decision Matrix for Wants

Want Objectives	Unique	Signal/Stop Button	Aesthetic	Total
Weights	3	1	2	
Alternative Designs	Rate Value / (Rate Value * Weight)			
Solution 1	4/12	1/1	2/6	20/12
Solution 2	8/12	2/1	4/6	40/12
Solution 3	11/12	6/1	6/6	95/12
Solution 4	10/12	5/1	5/6	80/12

3.3 Evolution of Project

The team learned many important lessons about coding from the previous attempt at this project. The most important lesson was that coding around hardware issues is not a safe idea and the project will be more successful if the financial commitment to new parts is made. With the new and improved sensors, the team wanted to implement a new code. Not all of the code would be different; the structure would remain, but a few key changes would be made. Firstly, the robot motion would be more controlled in the sense that it would move for smaller increments of time. Instead of moving forward for about a second, the code was designed to have the robot only turn in the direction of greatest light reading (left or right) and stop for a larger period of time. The team believed this provided for a more “patient” code that would give more time for the robot to react accordingly. Additionally, the code results in a “waddle” type of motion as opposed to the erratic motion observed in the previous attempt. While this compromises to a small extent the speed at which the robot can find the light, it is a safer and more reliable way of achieving the objective (finding light), which was considered as the more important priority in this attempt.

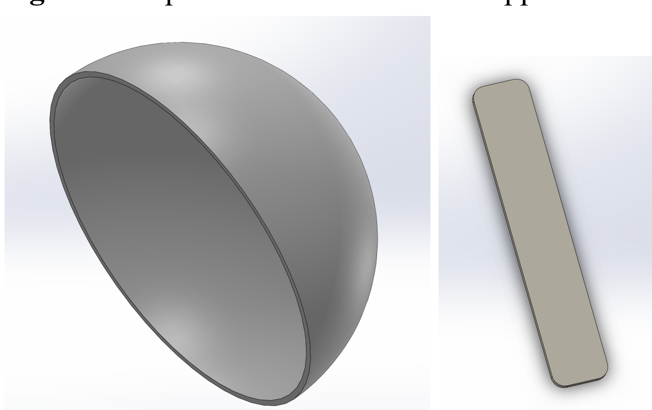
Another important lesson learned from the duration of the project was that the aesthetics were as integrated in the design choice as the hardware or software. The aesthetics had to work with the overall design and functionality, not simply be a task saved for the final day and then applied. The following two sections show the physical evolution of the aesthetic design developed using SolidWorks, in addition to other technical issues.

3.3.1 First Iteration

An initial prototype was created based on our chosen solutions, in order to test the robot and solve any problems found. The initial prototype consisted of the parts described in solution three: two photoresistors, three distance sensors, a stop button and a "found light source" audio signal. As for aesthetics, a paper bowl with cardboard supports was used.

Upon testing, it was found that this prototype had a few issues that needed to be corrected. First, the robot was continually moving in a curved path. The right motor was lagging behind the left. Additionally, the aesthetic pieces did not look as nice as desired.

Figure 7: Paper bowl and cardboard supports.

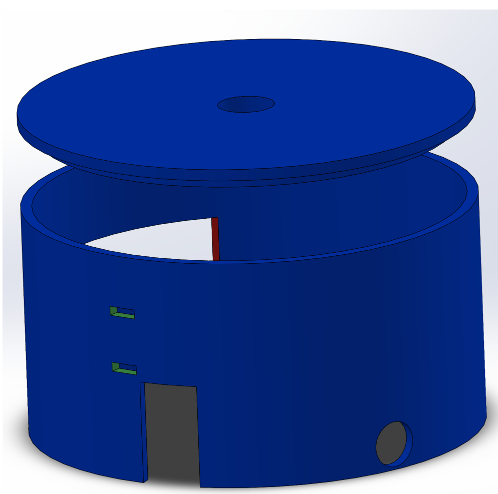


3.3.2 Second Iteration

To deal with our motor issue, our team began by adjusting the motor speeds. When this did not work, the batteries were replaced. This temporarily solved the issue, in order for other issues to be addressed.

As for the aesthetics, the next design was a big jump in terms of complexity and sophistication from the bowl and cardboard support. It included a 3D printed case and lid that would encircle the robot from its top plate upwards. There would be windows for the necessary visible components such as the LCD display, light sensors, switch, and Arduino port. There was also a hole in the lid for the big red button. The design was put through the Cura software, an application which estimates the time a 3D print will take. It was observed that the design took longer to print than desirable (<5 hours), as the design needed to be easily replicated by the NEU.

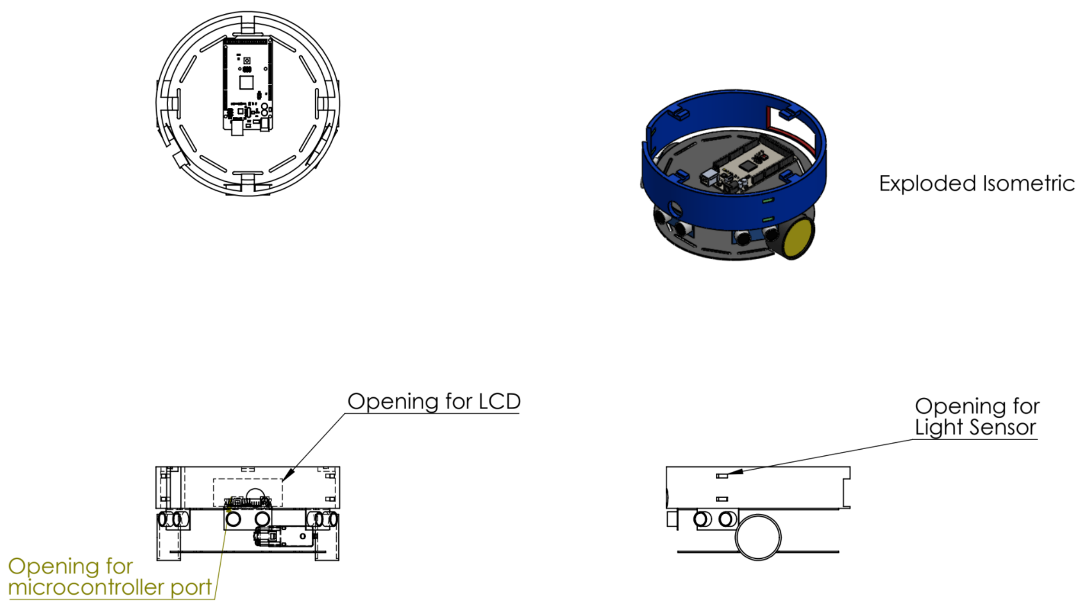
Figure 8: Solidworks model for case and lid



3.3.3 Third Iteration

After several runs through the simulator, the design was reduced to the tallest it could be while still retaining its complexity in terms of openings in the curvature. This would eventually serve as the casing used in the final design. A bowl was purchased and used as the lid instead of printing another piece. Additionally, the motor issues were finally fixed with the “waddle” motion. The specifics of the final design will be discussed in the following sections.

Figure 9: Final 3D print ring design



4. Final Design

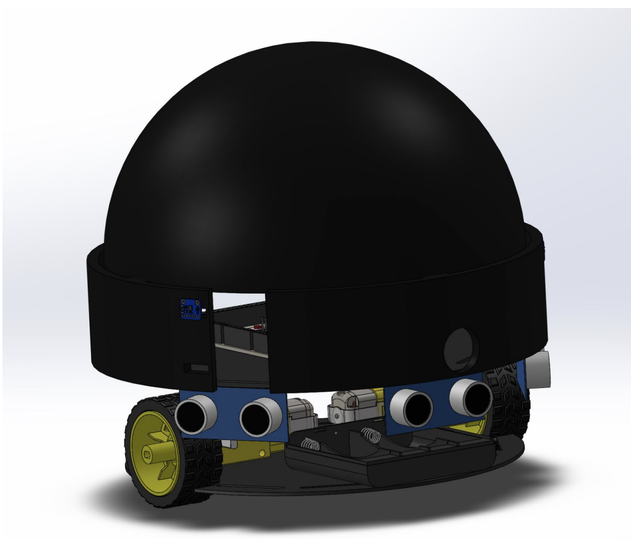
4.1 Description

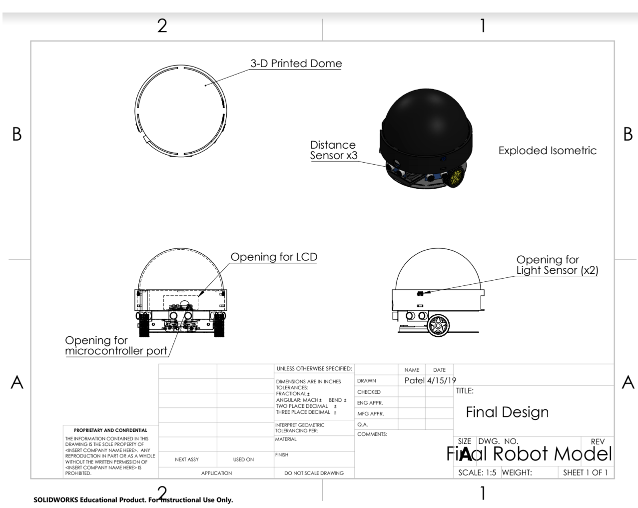
Agent Lux included two photoresistors, three distance sensors, a red stop button, a timer, a buzzer, a mega redboard, a 3D printed ring, and a bowl. Only two photoresistors were needed on each side of the robot, since the movement was short, so the robot would "waddle" back and forth. One of the photoresistors reading a high enough value would cause the robot to celebrate. An audio signal, emitted by the buzzer, would go off. This signal was the Mission Impossible Theme song, as this goes along with the theme and would be obvious to the NEU that the source had been found. Furthermore, a large emergency stop button used. It was placed on the top of the robot, in order for it to be easily accessible. The distance sensors and photoresistors used were both purchased from Adafruit.com, which provided far more accurate readings, as well as longer distance readings, than the hardware provided by the sparkfun kit. Since the redboard in the sparkfun kit did not provide enough pins for all of the pieces of Agent Lux, a mega redboard was also purchased in order to accommodate all portions of the project. As for the unique portion, an LCD was used to display the status of the mission and the countdown. Thus, the robot was completely autonomous. A 3D printed ring was used for the aesthetics of the project, as well as a bowl. The 3D ring projected upward from the top disk of the robot base, to protect the breadboard and redboard. The ring was equipped with slots that the bowl would fit into to cover the top of the robot, as well as holes for the photoresistors to be placed in. The red button was placed through a hole in the top of the bowl. The bowl could still be lifted in order to make adjustments, but protected the hardware of Agent Lux. The bowl was painted in order to resemble a tuxedo, as the robot is essentially a miniature, non-living NEU agent going on missions.

4.1.1 Solidworks Model

In this model, the final design of our robot can be seen, with all the proper components. The tuxedo painting on the bowl is not visible, but can be seen on the front page of the report.

Figure 10: Solidworks model of Agent Lux

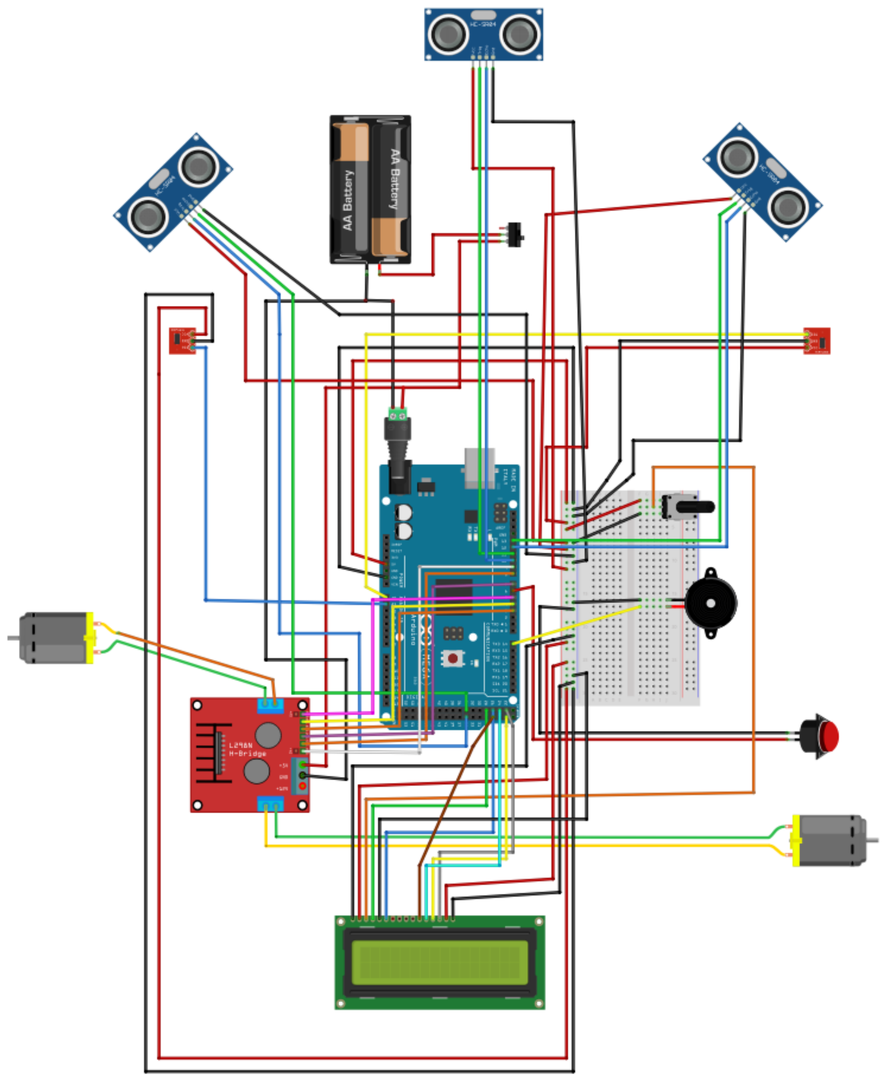




4.1.2 Fritzing Diagram

Figure 11 shows a wiring diagram for all of the components of the robot. This includes the distance sensors, light sensors, arduino mega, breadboard, battery pack, switch, motor controller, motors, potentiometer, speaker, and LCD display.

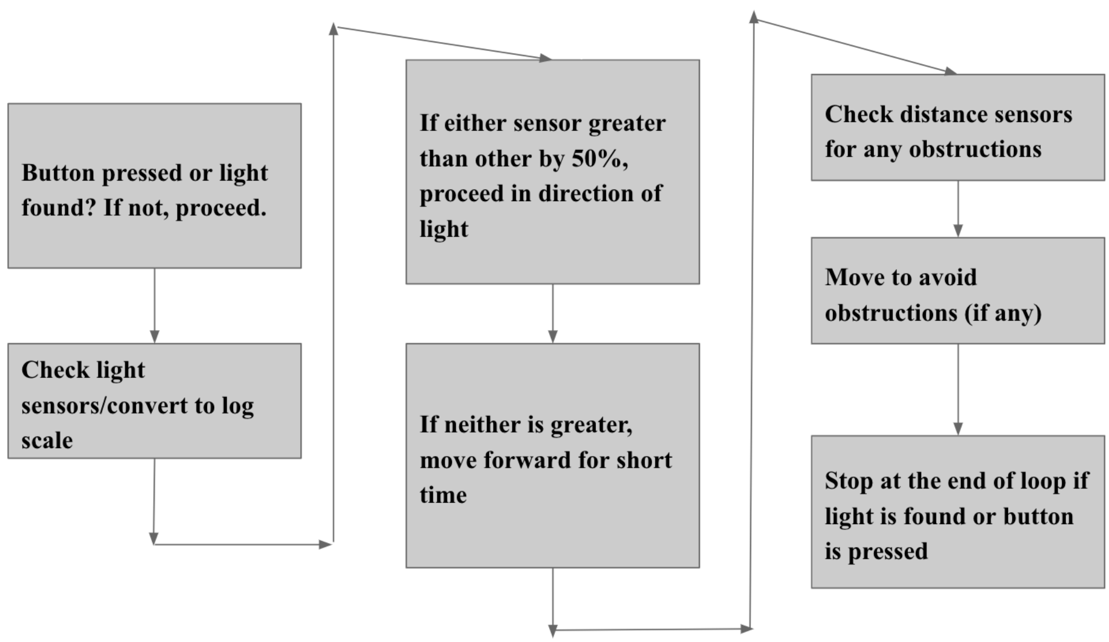
Figure 11: Fritzing diagram of the wiring of the robot.



4.1.3 Pseudocode

In order for the robot to successfully navigate the test environment, it must follow a relatively simple program. A very simplified flowchart of such code is shown in Figure 12. In the beginning of the program, pins are assigned to each component and variables and functions are defined and initialized. Within the loop, the robot checks to determine whether or not the button was pressed or the light was found. If either is true, the robot stops in the loop, stopping for the emergency stop button or celebrating if the light is found. If neither of the above conditions are true, the robot checks the light sensors and converts the values of the readings into a logarithmic scale. It then determines if either of the readings is greater than the other by 50% and turns in the direction of the greater light source while also moving forward. If neither light source is greater, it proceeds forward for a short period of time. Once the light readings have been read and accounted for, the ultrasonic sensors read the distance between the robot and any obstructions in the path. If anything is detected in the path, the robot is programmed to avoid it, moving in the direction of the least obstructed sensor (if all are obstructed, the robot performs a 180 degree turn). Finally, the loop repeats itself continuously, waiting for the stop button to be pressed, the light to be found, or the time to run out.

Figure 12: An explanation of the code that Agent Lux uses to operate



4.2 Cost Analysis

Table 4: Budget for the project.

Item	Quantity needed	Total Cost to Purchase and Install ($)
Adafruit Log-scale Analog Light Sensor	2	7.90
Adafruit Ultrasonic Sonar Distance Sensor	3	11.20
Mega Redboard	1	13.99
Bowl	1	.79
Total		**$33.88**

5. Conclusion

5.1 Evaluation

On demonstration day, our robot functioned effectively. The robot was autonomous, as it did not collide with any walls. The robot was light seeking, as it found the light source in well under five minutes (approximately a minute and a half). Additionally, upon finding the light source, the robot appropriately celebrated, informing the NEU of its successful mission. The emergency stop button functioned when pressed, and the unique element, the LCD, also was a success. Updates about the mission were displayed on the screen throughout the mission, as well as the countdown. Finally, the robot was very aesthetically pleasing, with the printed 3D ring and bowl, in addition to the tuxedo. The robot cost under $35 to create, and could therefore be replicated by the NEU inexpensively.

As seen by the success on demonstration day, Agent Lux is an effective and sleek solution to accomplishing NEU missions and protecting agent lives in the field.

5.2 Future Work

In the future, our team recommends rebuilding the base of the robot completely. While the structure was effective, there were repeated issues with the motors that were frustrating to the team. Thus, different and better robots could be purchased in order to ensure the robot could move forward in a straight line.

Furthermore, our team would like to increase the distance at which light could be found. The robot found the light from approximately 15-20 feet; however, the NEU may send Agent Lux on missions in rooms much bigger. Thus, the robot should have larger range. Additionally, the demonstration room was empty. In the future, a robot that could effectively navigate a room with obstacles, such as chairs and tables, would be effective.

Our robot did not have a photoresistor on the back of it. This means that, if placed in a room with the light source behind it, the robot would take significantly longer to find it. If a back photoresistor was utilized, the robot would find a light source behind it much faster.

Finally, the last improvement our team would like to make is making the emergency stop button both a start and stop button. The on/off switch is not well located on the robot, while the red button is right on top and easily accessible.

While we are very proud of the robot we created, it is important to not get complacent, and to always look for improvement in one’s work.

6. Rubric

Table 5: Rubric for Report

Task	Explanation	Excellent	Good	Poor
Introduction to report	Report includes a title page, a persuasive cover letter, an abstract and background information.	5	3	1
Failure Analysis	Describes the problems with the first trial of the project clearly, with plans on how to fix these issues.	10	7	4
Problem Description	The problem is clearly explained, in addition to the objectives, functions, constraints and the stakeholders in the project.	15	11	6
Unique Idea	The unique idea is explained clearly, with diagrams or pictures to supplement the explanation.	10	7	4
Solutions Considered	At least three solutions that were considered are explained with the use of pictures or diagrams, as well as the justification for the solution chosen, with decision analysis diagrams included.	10	7	4
Evolution of Project	The design of the robot is explained for multiple iterations of the project, with the problems found and solutions for these problems clearly stated. Pictures and diagrams are included.	15	11	6
Final Design	The final design is explained completely, including the pseudocode, fritzing diagram, solidworks model and budget.	15	11	6
Conclusion	A conclusion in the report wraps up the project, discusses the success of the project, and recommends future work.	10	7	4
Formatting	The entire report is neat and contains easily understood headings with a table of contents, as well as all tables and figures listed, plus an appendix for any pertinent information not found in the report (code, Gantt Chart, etc). References are cited properly.	5	3	1
Grammer	The report has few grammar mistakes, flows well, and is nicely written.	5	3	1
Total		100		

7. References

[1] S. Hymel, “Light-Seeking Robot,” *Spark Fun*. [Online]. Available: https://learn.sparkfun.com/tutorials/light-seeking-robot/all. [Accessed: 16-Apr-2019].

[2] “Build a Speedy Light-Tracking Robot (BlueBot Project #2) | Science Project,” *Science Buddies*, 21-Sep-2018. [Online]. Available: https://www.sciencebuddies.org/science-fair-projects/project-ideas/Robotics_p022/robotics/light-following-robot. [Accessed: 16-Apr-2019].

[3] “Experiment 15: Using an LCD,” *SIK Experiment Guide for Arduino - V3.2*. [Online]. Available: https://learn.sparkfun.com/tutorials/sik-experiment-guide-for-arduino---v32/experiment-15-using-an-lcd. [Accessed: 16-Apr-2019].

8. Appendix

Appendix A – Gantt chart

Table 6: Gantt Chart describing the work of the group over the timeline of the project.

	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
Planning							
Brainstorming Solutions							
Autonomous Robot							
Add Initial Photoresistor							
Functional Prototype							
Editing/Improving Code							
Final Design							
Wiring Diagram							
SolidWorks Model							
Matlab Simulation							
Presentation							
Final Report							

Appendix B – Code

Figure 13: Final Code for Agent Lux

```
//Project 4 Code\\
#include <LiquidCrystal.h>
LiquidCrystal lcd(28,26,25,24,23,22);   // Initialize the library with the pins we're using.
const int trigPinFRONT = 11;          //connects to the echo pin on the FRONT distance
sensor
const int echoPinFRONT = 10;          //connects to the trigger pin on the FRONT
distance sensor
const int trigPinLEFT = 35;          //connects to the echo pin on the LEFT distance sensor
const int echoPinLEFT = 34;          //connects to the trigger pin on the LEFT distance
sensor
const int trigPinRIGHT = 13;          //connects to the echo pin on the RIGHT distance
sensor
const int echoPinRIGHT = 12;          //connects to the trigger pin on the RIGHT distance
sensor
int light_sensor_left = A1;
int light_sensor_right = A0;
float rawRange = 1024;
float logRange = 5.0;
int threshold = 100;
int celebration = 0;

// motor one
int in1 = 8;  // Motor A (Right Motor)
int in2 = 7;
int enA = 9; //enable motor A, PWM pin

// motor two
int in3 = 3;  //Motor B  (Left Motor)
int in4 = 4;
int enB = 5;

const int buttonPin = 6;
int buttonpressed = 0;

void start()
```

```
{
 // set all the motor control pins to outputs
 pinMode(enA, OUTPUT);
 pinMode(enB, OUTPUT);
 pinMode(in1, OUTPUT);
 pinMode(in2, OUTPUT);
 pinMode(in3, OUTPUT);
 pinMode(in4, OUTPUT);
}

float distanceFRONT = 0;              //stores the distance measured by the FRONT distance
sensor
float distanceLEFT = 0;              //stores the distance measured by the LEFT distance
sensor
float distanceRIGHT = 0;              //stores the distance measured by the RIGHT distance
sensor

void play(char, int);

const int buzzer = 14; //Define pin 14
const int songspeed = 1; //Change to 2 for a slower version of the song, the bigger the
number the slower the song
//****************************************
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
```

```
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
//****************************************
int notes[] = {      //Note of the song, 0 is a rest/pulse
   NOTE_D5, 0, NOTE_D5, 0, NOTE_F5, NOTE_G5,
   NOTE_D5, 0, NOTE_D5, 0, NOTE_C5, NOTE_CS5,
   NOTE_D5, 0, NOTE_D5, 0, NOTE_F5, NOTE_G5,
   NOTE_D5, 0, NOTE_D5, 0, NOTE_C5, NOTE_CS5,
   NOTE_F5, NOTE_D5, NOTE_A4,
   NOTE_F5, NOTE_D5, NOTE_GS4,
   NOTE_F5, NOTE_D5, NOTE_G4,
   NOTE_F4, NOTE_G4, 0
};
//*******************************************************************\\
int durations[] = {        //duration of each note (in ms) Quarter Note is set to 250 ms
  375, 187.5, 187.5, 375, 375, 375,
  375, 187.5, 187.5, 375, 375, 375,
  375, 187.5, 187.5, 375, 375, 375,
  375, 187.5, 187.5, 375, 375, 375,
  187.5, 187.5, 1500,
  187.5, 187.5, 1500,
  187.5, 187.5, 1500,
  187.5, 187.5, 1500,
};
/********************************************************************\\

void setup()
{
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin (9600);       //set up a serial connection with the computer
  pinMode(trigPinFRONT, OUTPUT);   //the trigger pin will output pulses of electricity
  pinMode(echoPinFRONT, INPUT);    //the echo pin will measure the duration of pulses
coming back from the distance sensor
  pinMode(trigPinLEFT, OUTPUT);   //the trigger pin will output pulses of electricity
```

```
  pinMode(echoPinLEFT, INPUT);    //the echo pin will measure the duration of pulses
coming back from the distance sensor
  pinMode(trigPinRIGHT, OUTPUT);   //the trigger pin will output pulses of electricity
  pinMode(echoPinRIGHT, INPUT);    //the echo pin will measure the duration of pulses
coming back from the distance sensor
  lcd.begin(16, 2);              //initializes screen with 2 rows
  lcd.clear();                  //clears the screen of anything previously saved
  lcd.print("Time Starts Now");   //says the timer is starting
}

void loop()
{
    if(digitalRead(buttonPin) == LOW) //functions based off of button pulling input pin
LOW
    {
      buttonpressed = buttonpressed + 1;
    }
    Serial.println(buttonpressed);

    if(buttonpressed > 0)
    {
      Move(0, 0, 10);
      while (buttonpressed > 0)
      {
        lcd.clear();            //clears the screen of anything previously saved
        lcd.print("Emergency!");   //says emergency stop
        lcd.setCursor(0,1);   // Set the cursor to the position
        lcd.print("Mission Stopped"); // print mission stopped
      }
    }

    if(celebration > 0)
    {
      Serial.println("FOUND LIGHT");
      Move(0, 0, 10);
      lcd.clear();            //clears the screen of anything previously saved
      lcd.print("Light Found");   //says the light was found
      lcd.setCursor(0,1);   // Set the cursor to the position
      lcd.clear();
      lcd.print("Mission Complete"); // print success
```

```
    delay(500);
    for (int i=0;i<36;i++){             //36 is the total number of music notes in the song
    int wait = durations[i] * songspeed;
    tone(buzzer,notes[i],wait);         //tone(pin,frequency,duration)
    delay(wait);}                       //delay is used so it doesn't go to the next loop before
tone is finished playing
   }

   while(300-millis()/1000 < 1 || 300-millis()/1000 > 300)
   {
    Move(0, 0, 10);
    lcd.clear();         //clears the screen of anything previously saved
    lcd.print("Time Exceeded");   //says the timer is exceeded
    lcd.setCursor(0,1);   // Set the cursor to the position
    lcd.print("Mission Failed"); // print failure
   }

   if(buttonpressed == 0 && celebration == 0 && 300 - millis()/1000 > 0)
   {
    lcd.setCursor(0,1);   // Set the cursor to the position
    lcd.print("      "); // Erase the largest possible number
    lcd.setCursor(0,1);   // Reset the cursor to the original position
    lcd.print(300 - millis()/1000); // Print our value

    int left = analogRead(light_sensor_left);
    int right = analogRead(light_sensor_right);
    float luxleft = RawToLux(left);
    float luxright = RawToLux(right);

    Serial.print("Left Light: ");
    Serial.println(luxleft);
    Serial.print("Right Light: ");
    Serial.println(luxright);

    if ((luxleft > threshold && luxright > threshold) || luxleft > 135 || luxright > 135)
    {
     celebration++;
    }

    else if (luxleft > luxright * 1.5)
```

```
{
  SpinLeft();
  delay(10);
  Move(1, 100, 50);
}
else if (luxright > luxleft * 1.5)
{
  SpinRight();
  delay(10);
  Move(1, 100, 50);
}
else
{
  Move(1, 100, 50);
}

distanceFRONT = getDistanceFRONT();   //variable to store the distance measured
by the FRONT sensor
distanceLEFT = getDistanceLEFT();   //variable to store the distance measured by the
LEFT sensor
distanceRIGHT = getDistanceRIGHT();   //variable to store the distance measured by
the RIGHT sensor

Serial.println("FRONT Distance: ");      //print direction
Serial.print(distanceFRONT);     //print the distance that was measured
Serial.println(" in   ");      //print units after the distance
Serial.println("LEFT Distance: ");      //print direction
Serial.print(distanceLEFT);     //print the distance that was measured
Serial.println(" in   ");      //print units after the distance
Serial.println("RIGHT Distance: ");      //print direction
Serial.print(distanceRIGHT);     //print the distance that was measured
Serial.println(" in   ");      //print units after the distance

if (distanceFRONT > 12 && distanceLEFT > 12 && distanceRIGHT > 12) //if the
area all around is clear, move forward
{
  Move(1 , 100 , 100);
}
if (distanceFRONT <= 12)
{
```

```
        delay(50);
        distanceFRONT = getDistanceFRONT();   //variable to store the distance measured
by the FRONT sensor
        Serial.println("Something detected in FRONT: ");      //print direction
        Serial.print(distanceFRONT);     //print the distance that was measured
        Serial.println(" in. away  ");      //print units after the distance
        if (distanceFRONT <= 12)
        {
         if (distanceRIGHT > distanceLEFT)
         {
          if (distanceRIGHT <= 12) //if the distance all around is less than 12", reverse
            {
           Move(0 , 100 , 100);
           delay(10);
           Spin(1 , 100);//Spin Right
           delay(10);
           Spin(1 , 100);//Spin Right
            }
          if (distanceLEFT > 12) //if the right is clear, go right
          {
           Spin(1 , 100);//Spin Right
           //stop(); //Stop Spinning
           Move(1, 100, 100);//Move Forward
          }
         }
         if (distanceLEFT > distanceRIGHT)
         {
          if (distanceLEFT < 12) //if the distance all around is less than 12", reverse
          {
           Move(0 , 100 , 100);
           delay(10);
           Spin(1 , 100);//Spin Right
           delay(10);
           Spin(1 , 100);//Spin Right
          }
          if (distanceRIGHT > 12) //if the left is clear, go left
          {
           Spin(0, 100);//Spin Left
           //stop();
           Move(1, 100, 100);//Move Forward
```

```
          }
        }
      }
    }

    if (distanceRIGHT < 3)
    {
     delay(50);
     distanceRIGHT = getDistanceRIGHT();   //variable to store the distance measured
by the RIGHT sensor
     Serial.println("Something detected to RIGHT: ");      //print direction
     Serial.print(distanceRIGHT);     //print the distance that was measured
     Serial.println(" in. away  ");      //print units after the distance
     if (distanceRIGHT < 3)
       {
        Move(0 , 100 , 100);
        delay(10);
        Spin(0 , 100);//Spin Left
       }
    }

    if (distanceLEFT < 3)
    {
     delay(50);
     distanceLEFT = getDistanceLEFT();   //variable to store the distance measured by
the LEFT sensor
     Serial.println("Something detected to LEFT: ");      //print direction
     Serial.print(distanceLEFT);     //print the distance that was measured
     Serial.println(" in. away  ");      //print units after the distance
     if (distanceLEFT < 3)
       {
        Move(0 , 100 , 100);
        delay(10);
        Spin(1 , 100);//Spin Right
       }
    }

    delay(100);      //delay 100ms between each reading
   }
}
```

```
//------------------FUNCTIONS-------------------------------

//RETURNS THE DISTANCE MEASURED BY THE FRONT HC-SR04 DISTANCE
SENSOR
float getDistanceFRONT()
{
  float echoTimeFRONT;                   //variable to store the time it takes for a ping to
bounce off an object
  float calculatedDistanceFRONT;         //variable to store the distance calculated from the
echo time

  //send out an ultrasonic pulse that's 10ms long
  digitalWrite(trigPinFRONT, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPinFRONT, LOW);

  echoTimeFRONT = pulseIn(echoPinFRONT, HIGH);      //use the pulsein command to
see how long it takes for the
                                                    //pulse to bounce back to the sensor

  calculatedDistanceFRONT = echoTimeFRONT / 148.0;  //calculate the distance of the
object that reflected the pulse (half the bounce time multiplied by the speed of sound)

  return calculatedDistanceFRONT;              //send back the distance that was calculated
}

//RETURNS THE DISTANCE MEASURED BY THE LEFT HC-SR04 DISTANCE
SENSOR
float getDistanceLEFT()
{
  float echoTimeLEFT;                   //variable to store the time it takes for a ping to bounce
off an object
  float calculatedDistanceLEFT;         //variable to store the distance calculated from the
echo time

  //send out an ultrasonic pulse that's 10ms long
  digitalWrite(trigPinLEFT, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPinLEFT, LOW);
```

```
  echoTimeLEFT = pulseIn(echoPinLEFT, HIGH);      //use the pulsein command to see
how long it takes for the
                                        //pulse to bounce back to the sensor

  calculatedDistanceLEFT = echoTimeLEFT / 148.0;  //calculate the distance of the
object that reflected the pulse (half the bounce time multiplied by the speed of sound)

  return calculatedDistanceLEFT;              //send back the distance that was calculated
}

//RETURNS THE DISTANCE MEASURED BY THE RIGHT HC-SR04 DISTANCE
SENSOR
float getDistanceRIGHT()
{
  float echoTimeRIGHT;                  //variable to store the time it takes for a ping to
bounce off an object
  float calculatedDistanceRIGHT;        //variable to store the distance calculated from the
echo time

  //send out an ultrasonic pulse that's 10ms long
  digitalWrite(trigPinRIGHT, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPinRIGHT, LOW);

  echoTimeRIGHT = pulseIn(echoPinRIGHT, HIGH);      //use the pulsein command to
see how long it takes for the
                                        //pulse to bounce back to the sensor

  calculatedDistanceRIGHT = echoTimeRIGHT / 148.0;  //calculate the distance of the
object that reflected the pulse (half the bounce time multiplied by the speed of sound)

  return calculatedDistanceRIGHT;             //send back the distance that was calculated
}

//CONVERT TO LOG SCALE
float RawToLux(int raw)
{
  float logLux = raw * logRange / rawRange;
  return pow(10, logLux);
```

```
}


//THIS FUNCTION MOVES THE ROBOT EITHER FORWARD OR BACKWARD
void Move(int dir, int speed, int duration) {
  if (dir == 1) {
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    analogWrite(enA, speed - speed/5); // Put left motor at full speed
    analogWrite(enB, speed); // Put right motor at full speed
    delay(duration);
  }
  else if( dir == 0){
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
    analogWrite(enA, speed - speed/5); // Put left motor at full speed
    analogWrite(enB, speed); // Put right motor at full speed
    delay(duration);
  }
}

//THIS FUNCTION SPINS THE ROBOT
void Spin(int dir, int speed) {
  if (dir == 1) { //spin right
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    analogWrite(enA, speed); // Put left motor at full speed
    analogWrite(enB, speed); // Put right motor at full speed
    delay(500);
  }
  else if( dir == 0){
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
```

```
    digitalWrite(in4, LOW);
    analogWrite(enA, speed); // Put left motor at full speed
    analogWrite(enB, speed); // Put right motor at full speed
    delay(500);
  }
}

void SpinRight()
{
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    analogWrite(enA, 150); // Put left motor at high speed
    analogWrite(enB, 25); // Put right motor at low speed
    delay(100);
}

void SpinLeft()
{
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
    analogWrite(enA, 25); // Put left motor at low speed
    analogWrite(enB, 150); // Put right motor at high speed
    delay(100);
}

void stop() {
  analogWrite(enA, 0); // Put brakes on left motor
  analogWrite(enB, 0); // Put brakes on right motor
}
```

Appendix C – Time Table

Table 7: Time Table

Group Member	Activities	Date	Time
Annabelle	Made rubric, started shared report, unique idea	3/11/19	1 hr 20 min
Dylan	Made rubric, started shared report, unique idea	3/11/19	1 hr 20 min
Dean	Made rubric, started shared report, unique idea	3/11/19	1 hr 20 min
Annabelle	Discuss ordering new parts	3/13/19	30 min
Dylan	Discuss ordering new parts	3/13/19	30 min
Dean	Discuss ordering new parts	3/13/19	30 min
Annabelle	researched and ordered new distance and light sensors	3/19/19	30 min
Dylan	researched and ordered new distance and light sensors	3/19/19	30 min
Dean	researched and ordered new distance and light sensors	3/19/19	30 min
Annabelle	pseudocode, planning for upcoming weeks (scheduling and different solution options, i.e. aesthetics)	3/20/19	1 hr
Dylan	pseudocode, planning for upcoming weeks (scheduling and different solution options, i.e. aesthetics)	3/20/19	1 hr
Dean	pseudocode, planning for upcoming weeks (scheduling and different solution options, i.e. aesthetics)	3/20/19	1 hr
Annabelle	Worked on final report	3/23/19	1 hr
Dylan	Worked on robot motion	3/24/19	1 hr
Annabelle	Finalized motion, began light sensing	3/25/19	1 hr
Dylan	Finalized motion, began light sensing	3/25/19	1 hr
Dean	Finalized motion, began light sensing	3/25/19	1 hr
Annabelle	Worked on final report	3/31/19	1 hr
Dylan	Worked on motion sensing	3/31/19	1 hr
Dean	Worked on final report	3/31/19	1 hr
Annabelle	Worked in timer (unique element)	4/1/19	1 hr
Dylan	Worked in timer (unique element)	4/1/19	1 hr
Dean	Worked in timer (unique element)	4/1/19	1 hr

Annabelle	effectively made robot find light/presentation	4/5/19	3 hrs
Dylan	effectively made robot find light/presentation	4/5/19	3 hrs
Dean	effectively made robot find light/presentation	4/5/19	3 hrs
Annabelle	worked in unique element/3D print/presentation	4/6/19	3 hrs
Dylan	worked in unique element/3D print/presentation	4/6/19	3 hrs
Dean	worked in unique element/3D print/presentation	4/6/19	3 hrs
Annabelle	presentation/testing/3D print	4/7/19	3 hrs
Dylan	presentation/testing/3D print	4/7/19	3 hrs
Dean	presentation/testing/3D print	4/7/19	3 hrs
Annabelle	aesthetics	4/9/19	1hr
Dean	aesthetics	4/9/19	1hr
Annabelle	final report	4/12/19	2 hrs
Dylan	final report	4/12/19	2 hrs
Dean	final report	4/12/19	2 hrs
Annabelle	final report	4/15/19	1 hr
Dylan	final report	4/15/19	1 hr
Dean	final report	4/15/19	1 hr
Annabelle	final report	4/16/19	3 hrs
Dylan	final report	4/16/19	3 hrs
Dean	final report	4/16/19	3 hrs

APPENDIX D – DEPLOYMENT TABLE

Table 8: Deployment Chart

	Annabelle	Dylan	Dean	Complete
Planning				yes
Brainstorming Solutions				yes
First Report				yes
Autonomous Robot				yes
Second Report				yes
Add Initial Photoresistor				yes
Functional Prototype				yes
Third Report				yes
Editing/Improving Code				yes
Final Design				yes
Wiring Diagram				yes
SolidWorks Model				yes
Matlab Simulation				yes
Presentation				yes
Final Report				yes