EG1311 B02 Team 04
Project Report

Table of Contents

1. Introduction

This report provides a detailed account of our learning experiences and outlines the step-by-step process of building the final robot. It covers various stages of the development process, including initial ideation, designing, programming and prototype testing. It offers a comprehensive overview of how our project progressed from a concept to a fully functional robot. Furthermore, we will discuss various challenges we encountered during our prototype testing and some modifications we made to address these challenges. Finally, this report summarises key lessons learnt throughout the building process.

1.1 Objective

The primary goal of this project is to build a self-powered robot that navigates an obstacle course consisting of a starting area, a bump, a ramp, and a wall and then delivers a ping-pong ball over the wall (See Fig.1). In addition to the primary objectives, extra points are awarded based on the robot's weight and its capability to reverse back to the starting point.

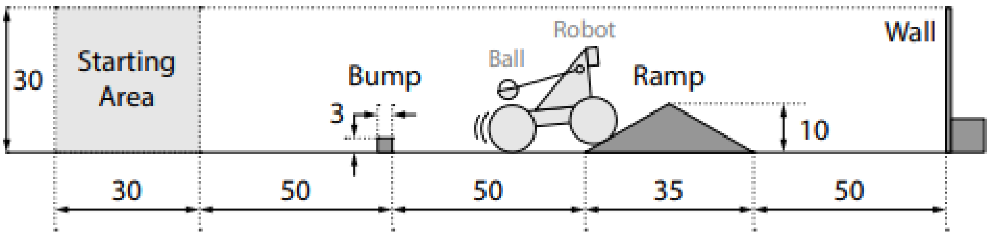


Fig.1. Layout of the Obstacle Course

1.2 Specifications

There are a few specifications imposed on the robot as follows:

I. Dimension Constraint: The robot must fit within 30 x 30 x 30 cubic centimetres.
II. Power Constraint: The robot must carry one 9V and four AA batteries.
III. Limited Materials: The robot must be made only using the materials provided.
IV. Time: The robot has up to 30 seconds to complete the course.

2. Ideation and Prototyping

This section of the report details our initial brainstorming process for designing each hardware component of our robot and the flow of the software program. Our team decided to adopt a four-motor design which offers several advantages that enhance the robot's performance and reliability during the obstacle course. The four-wheel design provides multiple contact points between the wheels and the ground which distributes weight more evenly and ensures greater stability. Additionally, it produces greater overall mechanical power output which is important for tasks such as pushing the vehicle over bumps or climbing inclines, where increased torque and consistent power are necessary. The following subsections further evaluate each component's design.

2.1 Wheel Design

We chose acrylic material for our car wheels due to its better structural integrity than cardboard. Furthermore, the acrylic wheels were cut out using laser cutting machines which ensures the diameter of the wheels is consistent throughout, as compared to hand cutting a cardboard wheel. A more rounded wheel ensures that the car will move in a straight line and prevent deviation from the obstacle course. The radius of the car wheels must at least be greater than the height of the bump in order to pass over it. Therefore, a diameter of 90mm is used for our wheels to ensure the car has sufficient height to pass over the bump and ramp. (See Appendix B)

2.2 Car Body

For the car body, we chose the corrugated plastic board instead of cardboard because of its stronger structural integrity and lightweight characteristics. The length of the car was an important aspect we considered during our initial ideation. In order for the car to pass over the ramp successfully and not get stuck at the maximum point, we applied the trigonometry formula to first compute the angle of incline of the ramp, as shown in Figure 2. Based on the calculation, the length of the car, including wheels, should maximally be about 250mm. This would guarantee that the vehicle would not get stuck at the top of the ramp.

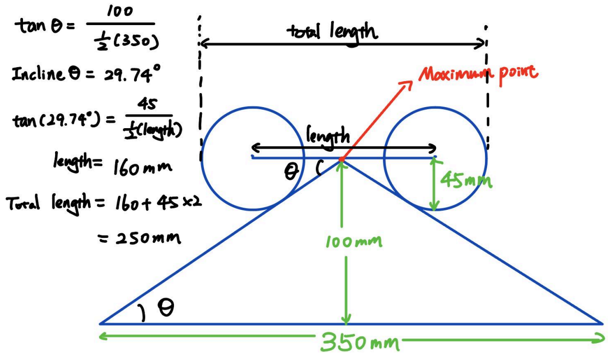


Fig.2. Maximum Length of the Car

2.3 Motors

Since we use 4 motors to power the car, the motors must output sufficient mechanical power to push the car over the bump and climb the ramp. The load on the motors is at its highest point when the car is climbing up the ramp because the motors need to generate sufficient torque to overcome both the weight of the car as well as the friction between the wheels and the ground as shown in Figure 3. When the torque required exceeds the maximum capacity of the combined motors, the motors will be in stall condition where the angular velocity drops to zero and the car will come to a halt as shown in Figure 4. Therefore, to ensure sufficient power, we used 4 AA batteries instead of 1 9V battery since 4 AA batteries can produce higher current and power.

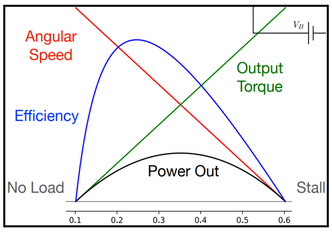


Fig.4. Motor Physics

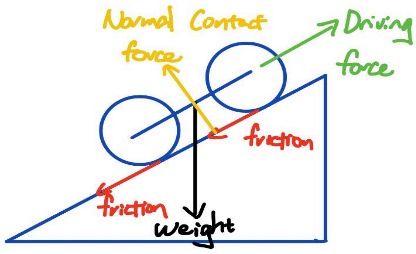


Fig.3. Forces Acting on the Car

2.4 Catapult Launching System

For our catapult launching mechanism, we initially thought of using the tension of a rubber band to slingshot the ping-pong ball over the wall. However, this approach has a few challenges. Firstly, the tension of the rubber band is significant which might cause damage to the structure of the car. Secondly, we need to use the servo provided to release the rubber band. This increases the complexity of the launching system and does not guarantee a 100% success rate. Therefore, to simplify its engineering principle, we directly attached the catapult launcher to the servo arm and used the servo's rotational speed to throw the ping-pong ball over the wall. Instead of using elastic potential energy stored in the rubber band, we directly convert the electrical energy of the servo into the kinetic energy of the ball as shown in Figure 5.

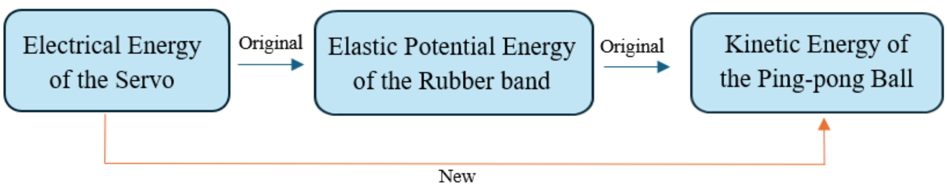


Fig.5. Energy Conversion of the Catapult Launching System

2.5 The Code

Based on the project requirement, we can use the following pseudocode and flowchart to represent the flow of our program for our robot as shown in Figure 6. In summary, the ultrasonic sensor measures the distance between the robot and the wall at a regular time interval. When the distance between the robot and the wall is at a certain length (in this case 10 cm), the robot will stop, throw the ball and reverse for a certain number of seconds. Otherwise, the robot will move forward. The complete code for our robot is shown in Appendix D.

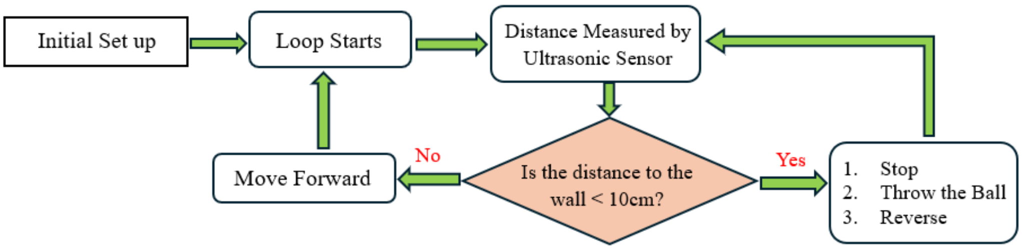


Fig.6. Flowchart of the Program

3. Prototyping and Challenges

After the initial ideation of each robot component, we began implementing these ideas and constructing the robot. In this section, we will address some of the challenges we encountered during the prototyping phase of our robot in preparation for the final run.

3.1 Lack of Friction of the Wheels

After testing the initial template project in Tutorial 1, we observed that the car got stuck in front of the bump while the wheels were spinning freely. This was because the wheels did not have sufficient grip of the ground, preventing the car from climbing up the bump and the ramp. To overcome this problem, we need to increase the frictional force between the wheels and the ground by changing the wheel design. We will discuss the improvement further in Chapter 4.

3.2 Understanding of the L293D Motor Driver

For our second prototype, we managed to use 1 H-Bridge Motor Driver and 4 AA batteries to power 4 motors by connecting the positive terminal of each motor to one output terminal on the motor driver and grounded all the negative terminals with the H Bridge Motor Driver as shown in Figure 7. When a high digital signal from Arduino is sent to the input terminals of the motor driver, the motors spin. When the digital signal is low, the motors will not spin. However, the car could only move forward and stop at the wall. It was unable to reverse back. After understanding the principle of this L293D motor driver. We decided to use 2 motor drivers for our final robot, each powering 2 motors by connecting one of the motor terminals to output 1 and the other to output 2. The same connection was done for the other motor with output 3 and output 4. The motor driver was then grounded. When Input 1 is high and Input 2 is low, the motors spin in a certain direction. When Input 1 is low and Input 2 is high, the motors spin in the opposite direction. When both inputs are low, the motors will not spin as shown in Figure 8.

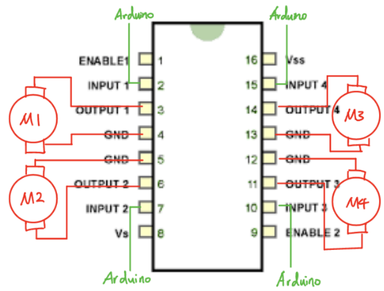


Fig.7. Original H Bridge Motor Driver Connection

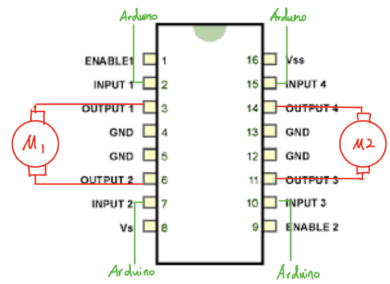


Fig.8. New H Bridge Motor Driver Connection

3.3 Height of the Ultrasonic Sensor

For our third prototype, the robot managed to pass over the bump but started reversing before reaching the ramp. After analysing the problem, we realised that the ultrasonic sensor was placed too low on the car body which misled the Arduino to think that the ramp was an obstacle. Therefore, the reverse code got triggered too early. To solve this problem, we decided to elevate the height of the ultrasonic sensor. We will discuss this further in Chapter 4.

4. Improvement

In the previous chapter, we outlined some difficulties encountered during the prototype testing phase. This chapter will discuss some of the modifications we made to our previous prototypes along the way to come to our final product.

4.1 The Wheels

Our final wheels were designed using AutoCAD Fusion and were cut out from acrylic material using the laser cutting machine. Each wheel has three triangular holes being cut out as shown in Appendix E 2D CAD drawings. This minimises the weight of the wheels while ensuring their structural integrity. To increase friction between the wheels and the ground, we wrapped the wheels with the anti-slip mat provided, to ensure the car has a better grip of the ground.

4.2 Height of the Ultrasonic Sensor

In this project, we are using the HC-SR04 ultrasonic sensor. After searching for its datasheet, we realised that this ultrasonic sensor has a range between 2cm and 400cm, and an angle of effect of 15 degrees. Therefore, to prevent the ultrasonic sensor from false triggering due to the ramp, we have to place the ultrasonic sensor at least 14.7cm from the ground. This can be proven using the trigonometry formula as shown in Figure 9.

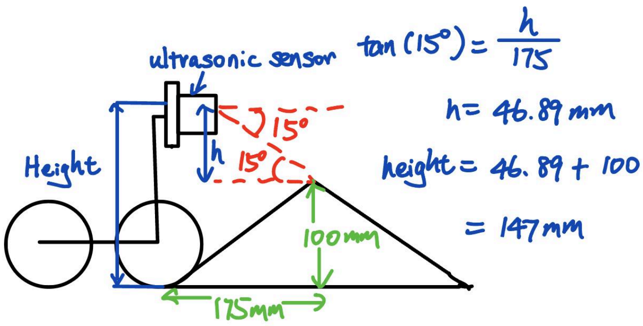


Fig.9. Height of Ultrasonic Sensor

4.3 Length of the Catapult Arm

We wanted to use minimum energy to get the ball past the wall, hence the ball had to be released at the optimum angle. Since the robot had to stop about 5 cm before the wall, we found that the ideal release point of the ball needed to be very close to the highest point of trajectory because the time to gain height (after throwing the ball) was minimal. As such, we had to reach the maximum trajectory height using the length of the catapult. The length of the catapult has to be smaller than 30 cm as the dimensions of the robot have to be less than 30 * 30 *30. Owing to these constraints and after many trials, we decided that the length of the catapult would be 20.25 cm

5. Lesson Learnt

In this section, our group will explore the key insights and valuable lessons gained throughout this project.

5.1 Understanding limitations, practicality and concepts

Given the fixed materials that we can use for our robot and its dimensions, we have to decide on which material to use for each part of the robot and take into account the weight of the robot. The material used must also be sturdy enough for the robot to successfully cross the obstacle course. We thus used the corrugated board for the body of the robot as it is more sturdy than cardboard, we even used ice cream sticks to make it more stable, so that it would not fold when all the electronic components were placed on top of it.

5.2 Learning from Mistakes

During the prototyping stage, our goal was to make sure that the next prototype was better than the current one and not to commit the same mistake that we had earlier. When something fails, it is very important to avoid making hasty conclusions, instead, we should systematically approach the problem to find the root cause. For example, a couple of our motors would just stop functioning in between trial runs, we hastily concluded that the motor was faulty and tried changing new ones, but it turned out that the breadboard was faulty all along.

5.3 Open-mindedness

When we were coming up with ideas for the robot, specifically the body, the wheels and the catapult of the robot, everyone had a different idea. We embraced each of our ideas and decided to create different bodies, wheels and catapults to test which was the best. A prime example is the creation of the wheels for our robot, we ideated on the shape, size and number of holes in the wheel. We thus laser printed wheels with different numbers of holes to test on the robot, in attempts to minimise its weight but ensure stability. By keeping an open mind to the various ideas, we were able to broaden our pool of options to work with.

5.4 Iterative Prototyping

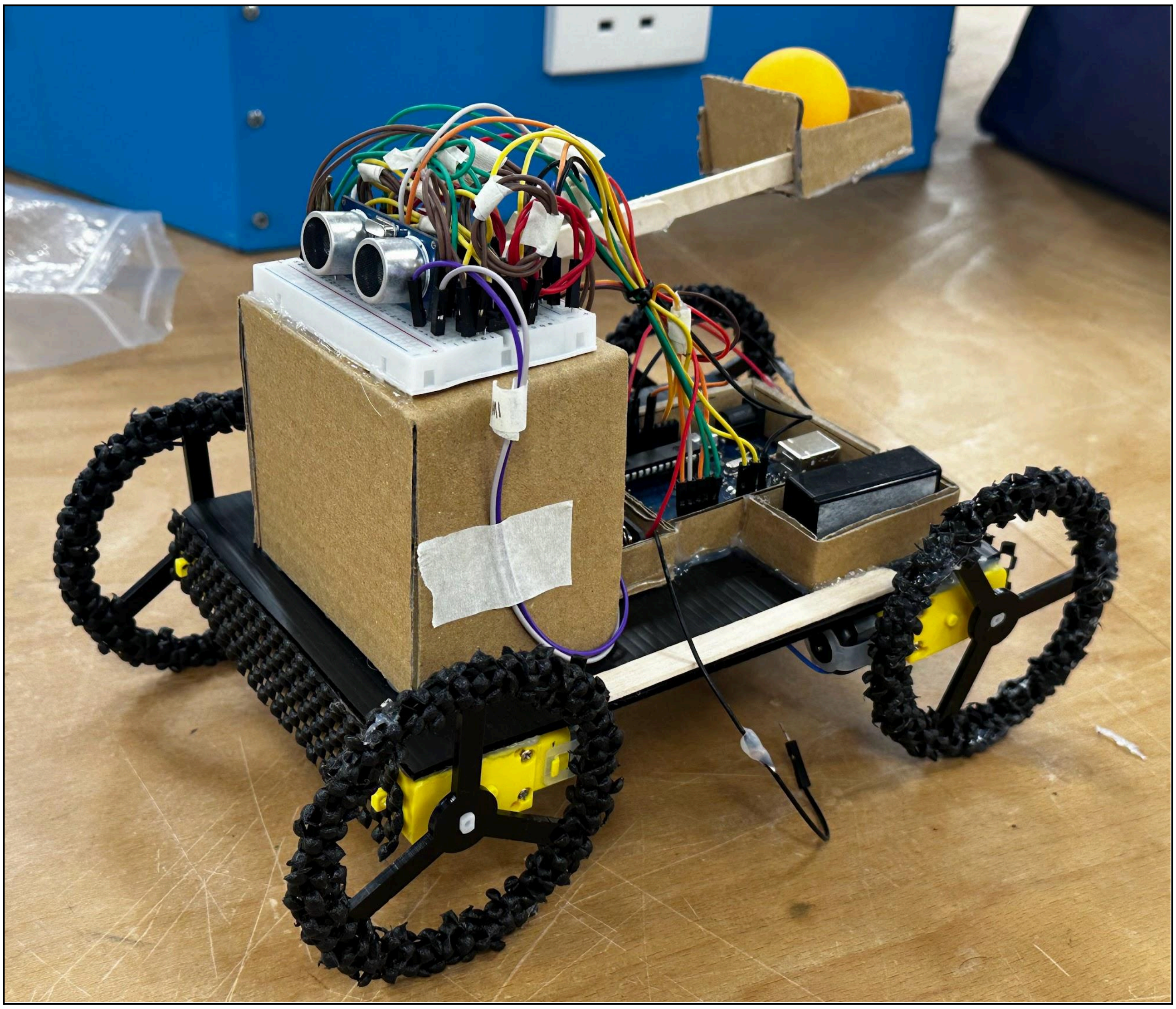
This method was highly suggested by our Professor. We cannot build the best robot in our first attempt, we have to make one model, take it for a trial run, find out where it failed, and analyse and rectify it in the next prototype. By doing various prototypes we were able to recognise what was necessary for the robot to succeed. This approach also helped us improve the robot after every prototype.

6. Conclusion

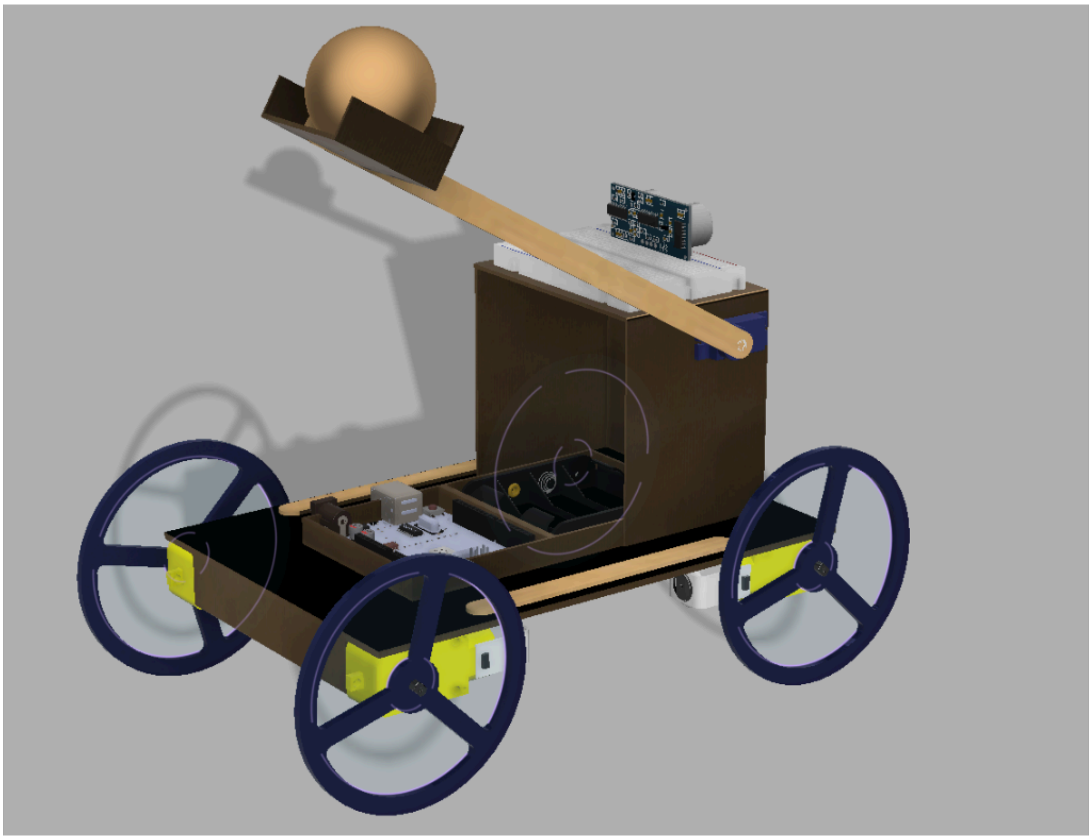
The process of creating a self-powered robot capable of traversing through an obstacle course and delivering a ping-pong ball over a wall, given no building instructions and challenges in operations, gave us opportunities to apply the design thinking model to brainstorm creative, innovative and practical solutions. This creation process has also taught us meaningful lessons such as proper prototyping, adaptability, and teamwork, to solve complex engineering problems.

Appendix

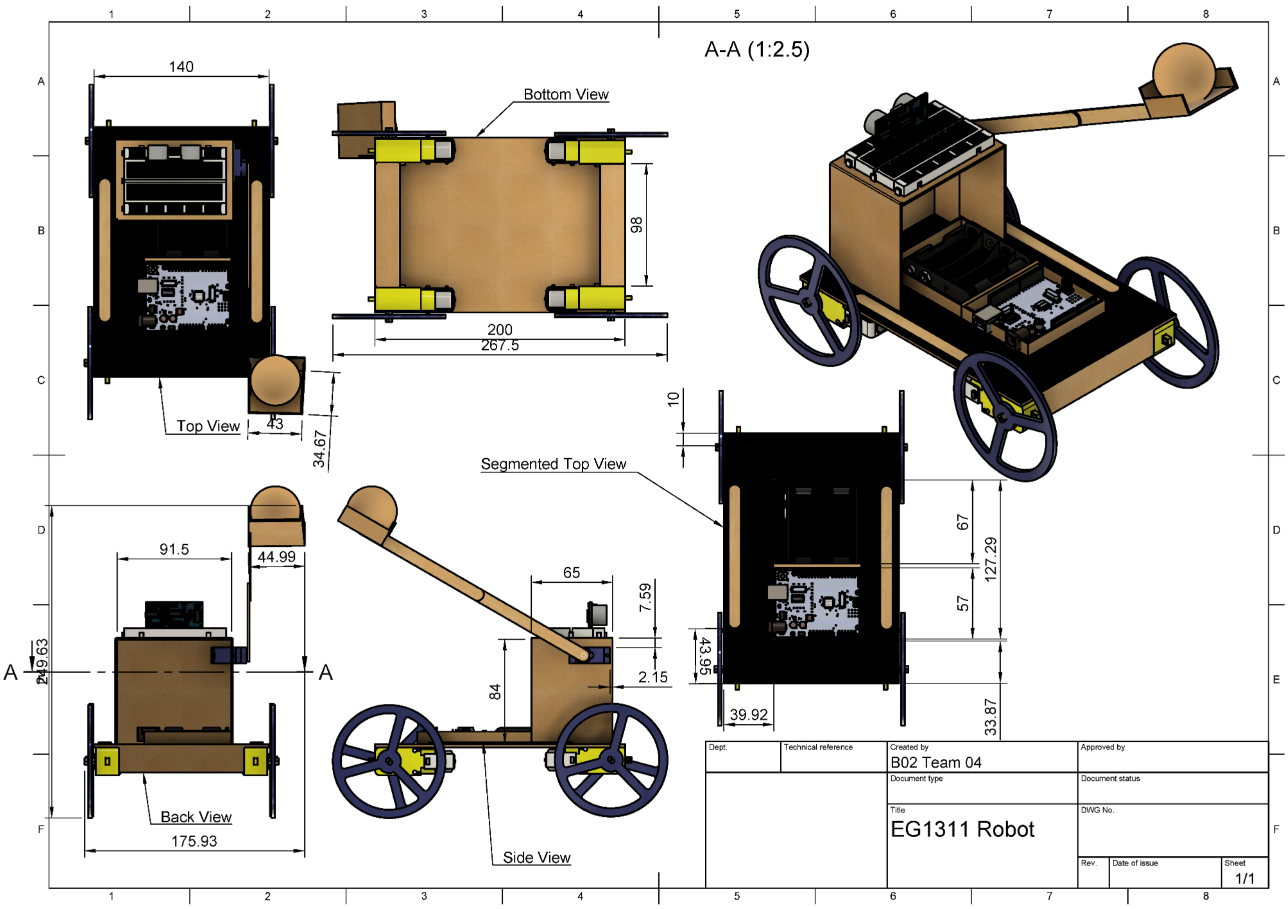
Appendix A (Photograph of our Final Robot)



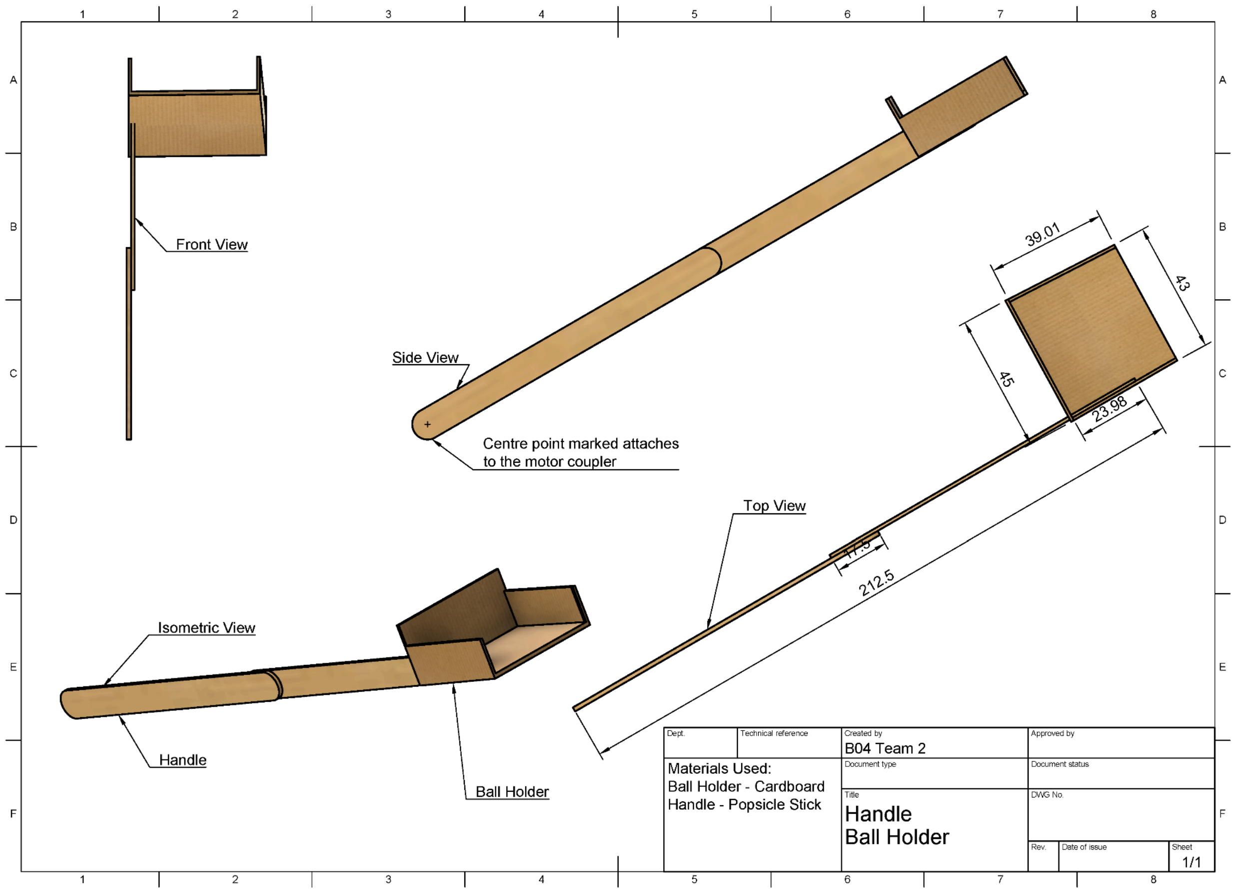
Appendix B (CAD Rendering and 2D Drawings of Our Final Robot)



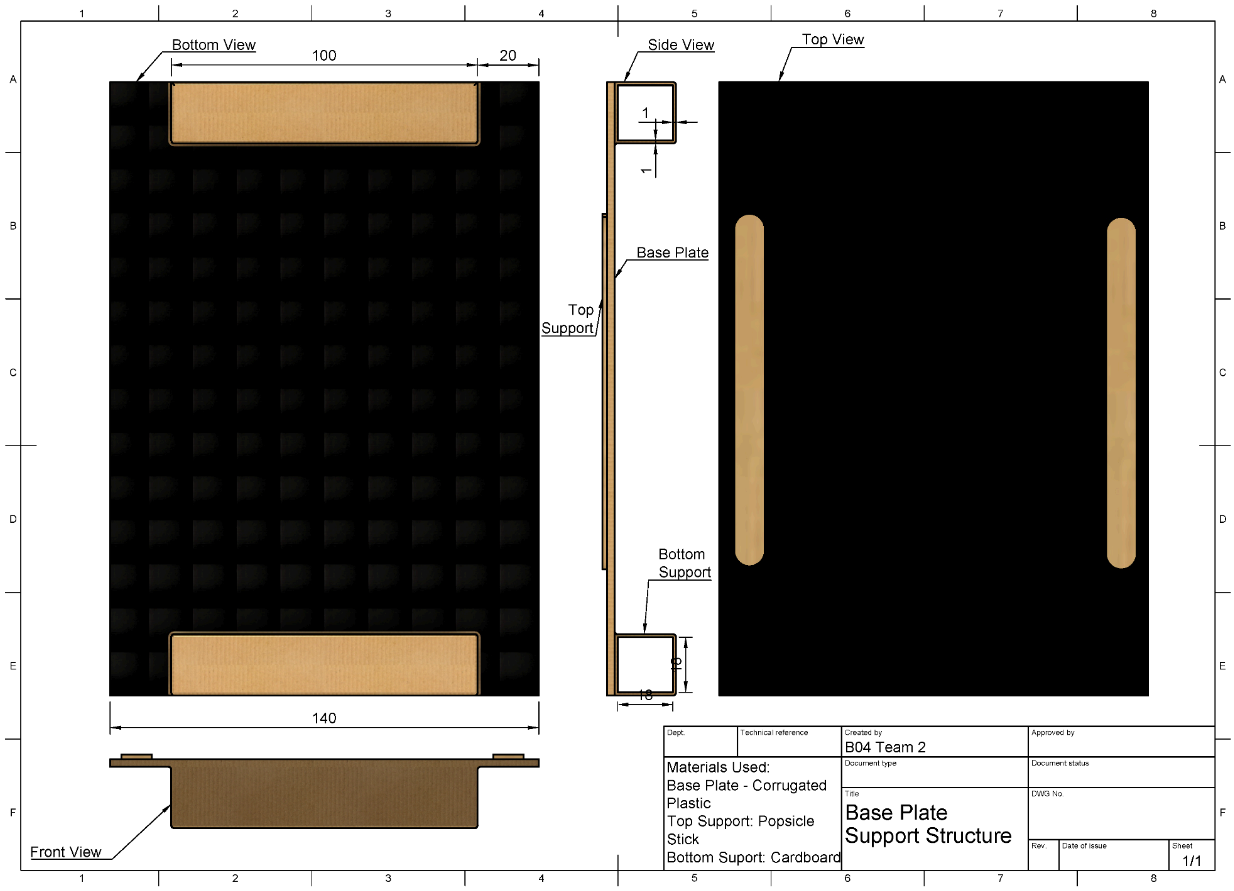
Drawing of Final Robot



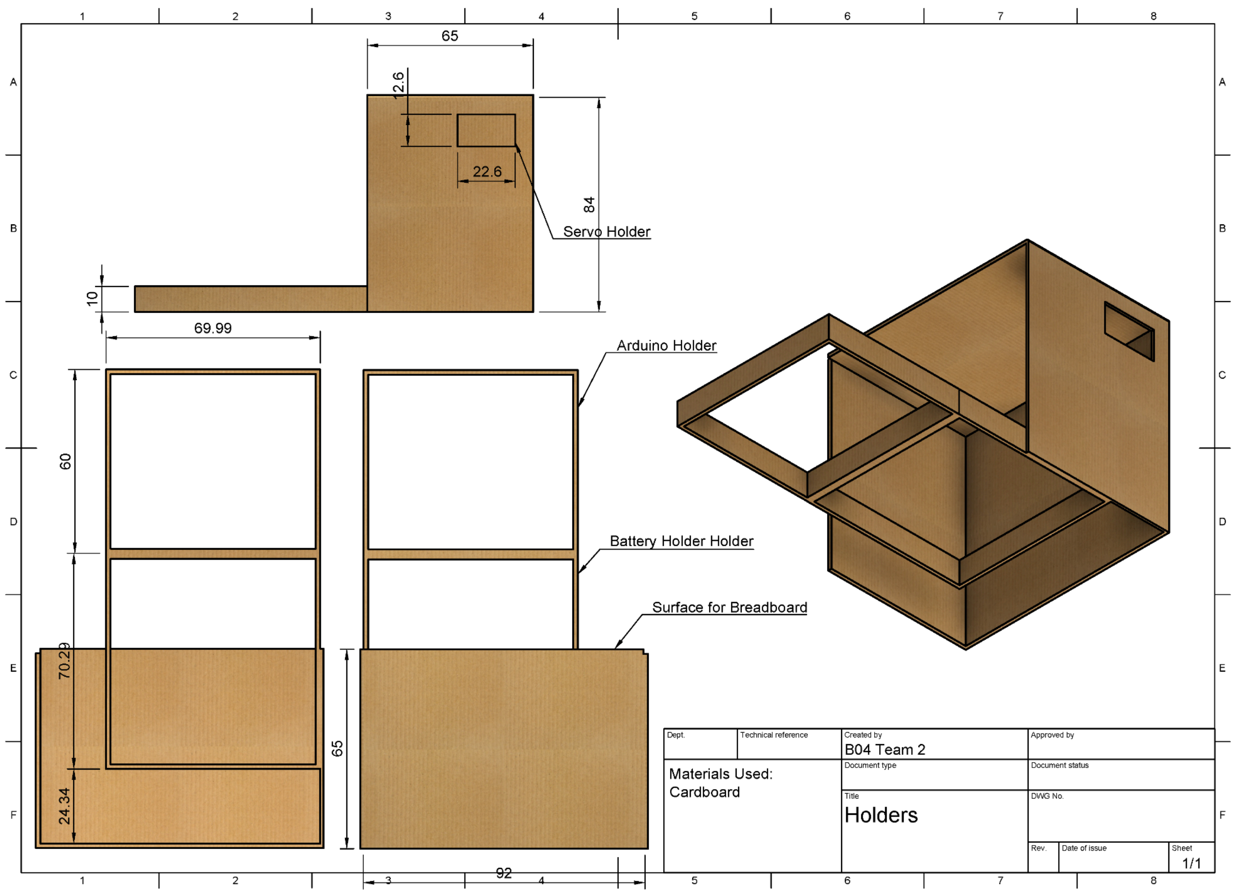
Drawing of Handle and Ball Holder



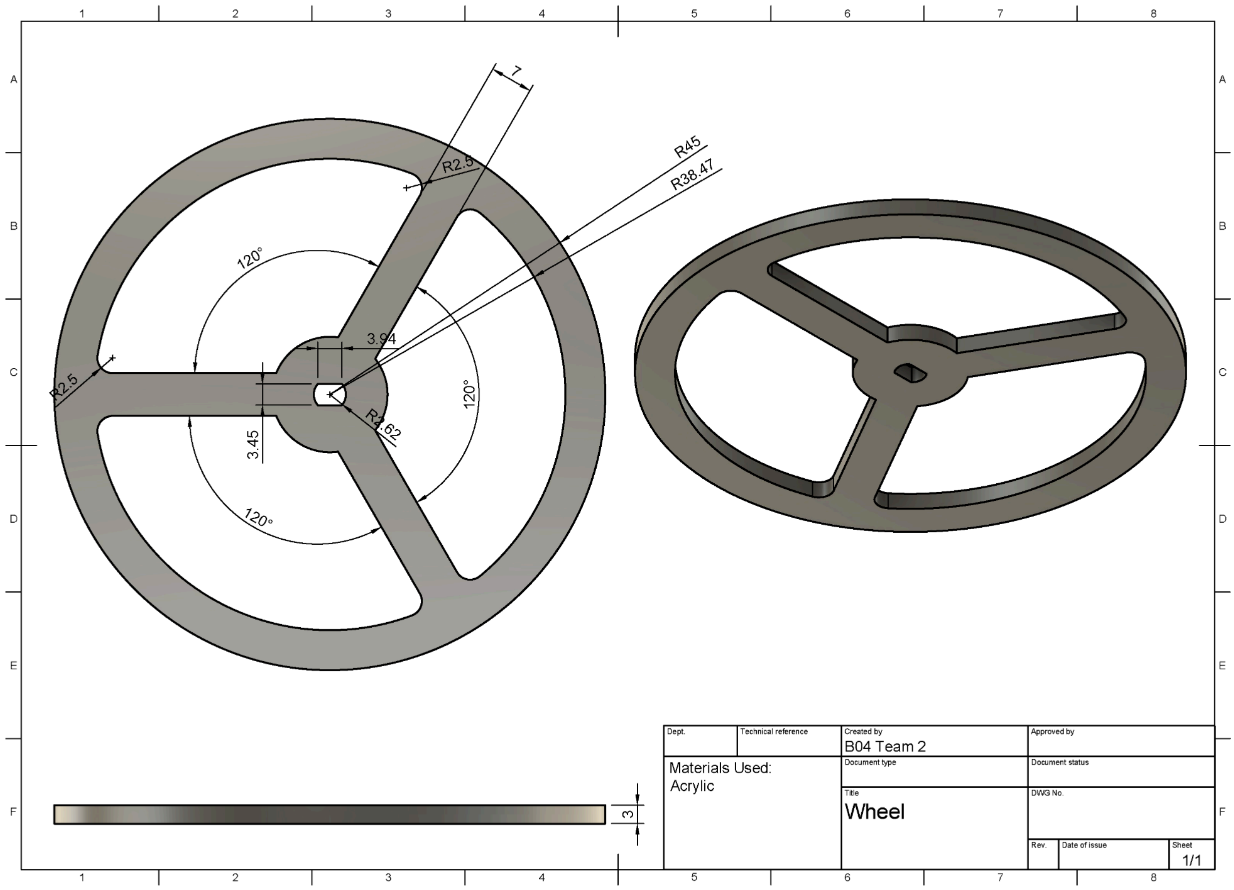
Drawing of Base Plate and Support Structure



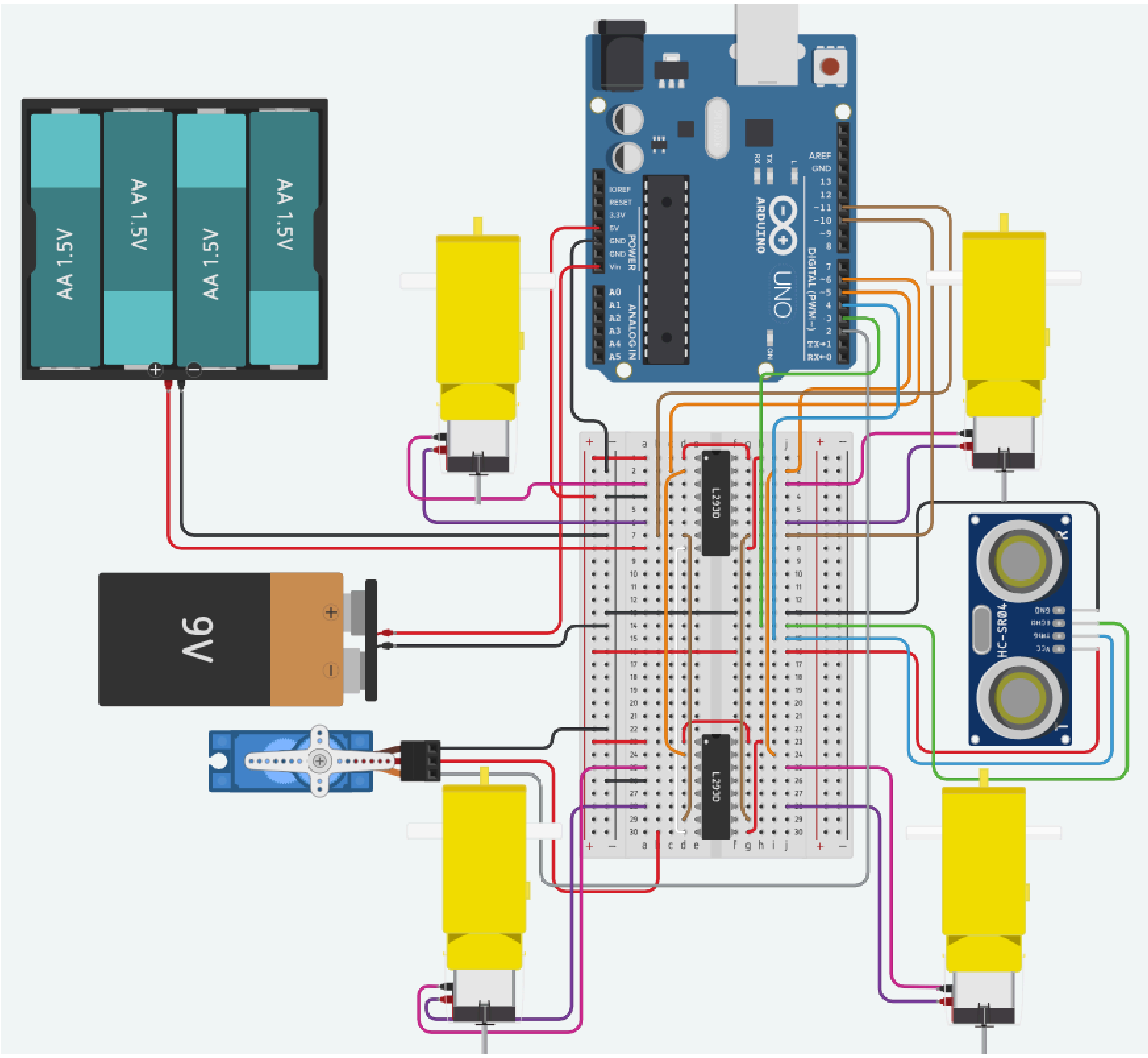
Drawing of Holders



Drawing of Wheels



<u>**Appendix C (Electrical Circuits – TinkerCAD Diagram)**</u>



Appendix D (Arduino Code)

```
#include <Servo.h>
//declaring all the digital output pin numbers used for the arduino
int TRIG_PIN = 4;
int ECHO_PIN = 3;
int MOTOR_PIN1 = 10;
int MOTOR_PIN2 = 11;
int MOTOR_PIN3 = 6;
int MOTOR_PIN4 = 5;
int servo_pin=2;
//defining the speed of sound for the ultrasonic sensor
float SPEED_OF_SOUND = 0.0345;
//declaring the name of the servo
Servo myservo;

void setup() {
  pinMode (MOTOR_PIN1, OUTPUT) ; //set all 4 motor pins as output
  pinMode (MOTOR_PIN2, OUTPUT) ;
  pinMode (MOTOR_PIN3, OUTPUT) ;
  pinMode (MOTOR_PIN4, OUTPUT) ;
  pinMode (TRIG_PIN, OUTPUT) ; //set trigger pin of ultrasonic sensor as output
  digitalWrite(TRIG_PIN, LOW); //initally, trigger pin is set to low
  pinMode (ECHO_PIN, INPUT) ; //set echo pin of ultrasonic sensor as input for reading distance
  Serial.begin (9600); //establish communication between computer and arduino
  myservo.attach(servo_pin);
  myservo.write(0); //set inital servo position
}


void loop() {
  digitalWrite(TRIG_PIN, HIGH); //ultrasonic sensor sends out a pulse wave
  delayMicroseconds (10);
  digitalWrite (TRIG_PIN, LOW);
  int microsecs = pulseIn (ECHO_PIN, HIGH); //ultrasonic sensors time the duration for the pulse to reflect back
  float cms = microsecs * SPEED_OF_SOUND/2; //use the time to calculate the distance to the wall
  Serial.println(cms); // print the distance on serial monitor for debugging purposes
  if(cms < 15) { //if distance to the wall is less than 15cm
    digitalWrite (MOTOR_PIN1, LOW); //motor will stop spinning
    digitalWrite (MOTOR_PIN2, LOW);
    digitalWrite (MOTOR_PIN3, LOW);
    digitalWrite (MOTOR_PIN4, LOW);
    delay(1000); //delay for 1 second
    myservo.write(90); //servo to change from position 0 to position 90, catapult arm will swing up
    delay(3000); //catapult arm will stay up for 3 seconds
    myservo.write(0); //catapult returns back to inital position
    delay(2000); //delay for 2 seconds
    digitalWrite (MOTOR_PIN1,LOW);  //motor will spin in reverse direction
    digitalWrite (MOTOR_PIN2, LOW);
    digitalWrite (MOTOR_PIN3, HIGH);
    digitalWrite (MOTOR_PIN4, HIGH);
    delay(11000);
    } else { //if distance is more than and equal to 15 cm
    digitalWrite (MOTOR_PIN1, HIGH); //motor will spin in forard direction
    digitalWrite (MOTOR_PIN2, HIGH);
    digitalWrite (MOTOR_PIN3, LOW);
    digitalWrite (MOTOR_PIN4, LOW);
   }
   delay (10);
}
```