New York University

Department of Mechanical and Aerospace Engineering

Advanced Mechatronics Project

Term Project

Submitted to

Professor Vikram Kapila

Raghunath Kishan Reddy- **KRR9721**
Arun Kumar Srivathsa- **SA6360**
Indupuru Sriharsha Reddy- **SI2151**

Table of Contents

Contents

Introduction 3

Project Overview: 3

Selection of Components 4

Raspberry Pi 3: 4

Propeller Activity Board: 4

PI Camera: 4

QTR-8RC Sensor: 5

Ultrasonic Sensor: 5

Continuous Servo Motor: 6

Micro Servo Motor: 6

Piezo Buzzer: 6

LED: 7

Robot Structure: 7

Working Principle: 9

Circuit Diagram: 12

Bill of Materials: 13

Annexure 1: 14

Raspberry Pi Code: 14

Simple IDE Code: 16

Introduction

We are given the task of designing an autonomous robot that drives around the Manhattan streets while detecting enemy intruders basically enemies among us who troubled civilians in the city. The robot must look out for the enemies and eliminate them by scouring through the streets while following all the traffic rules and should avoid obstacles if any.

Project Overview:

The bot starts from the safe houses either H1 or H2 to traverse through the streets of manhattan depicted through the black tape with 3 lines A-line, Centre(i) line, and B line. It is also given that an obstacle will be placed randomly at i2 or i3 or i5 locations. The total arena consists of 8 objects 4 friends(indicated in blue) and 4 enemies(indicated in red) as shown below. Also, we were given that few are one-way streets as directed by the green arrows in the below image. one more condition that was given to us is that the bot shouldn't stop at any place unless it detects a friend or enemy which is placed at a height of 20cms and has aruco tags to them. Once the robot detects the enemies, it knocks down the enemies.

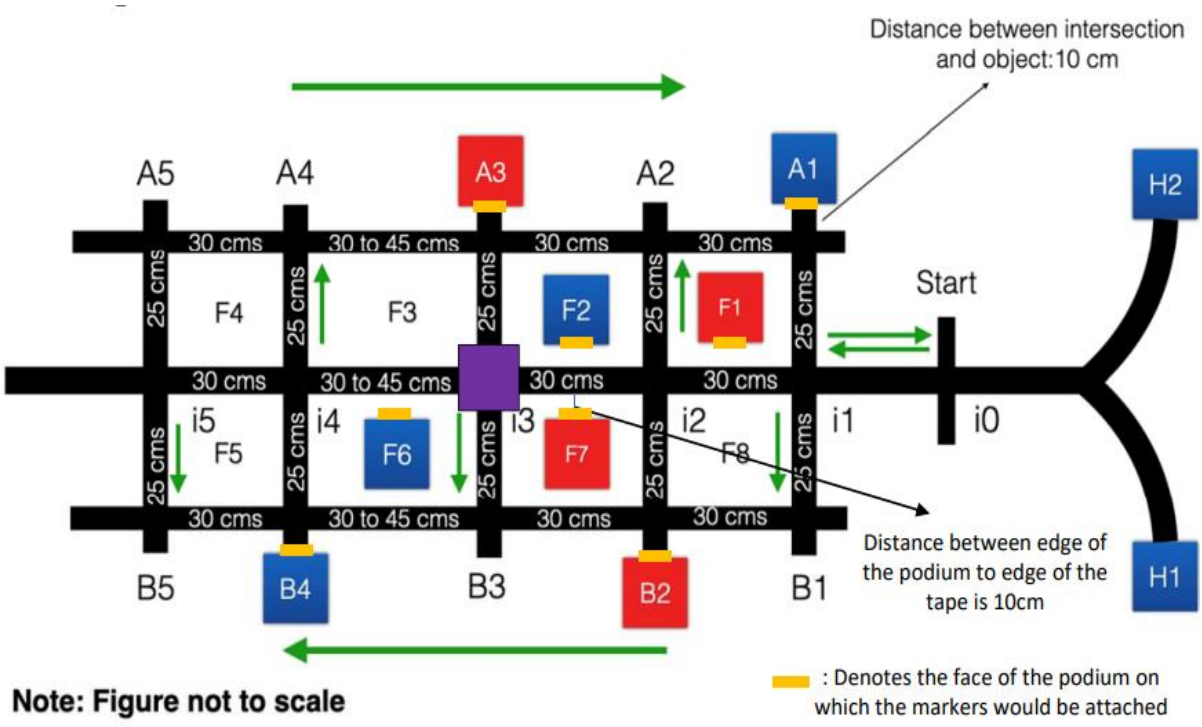

Figure 1: Scenario

Selection of Components

Raspberry Pi 3:

For the term project, we decided to use raspberry pi and integrate the raspberry pi with the propeller activity board. RPI microcontroller was used to get the tag ids with the help of a pi camera. The RPI also controlled the actuation mechanism. Once the pi camera detects the tag ids and if the ids are found to be an enemy. the data is shared with the motor to knock off the enemy block.

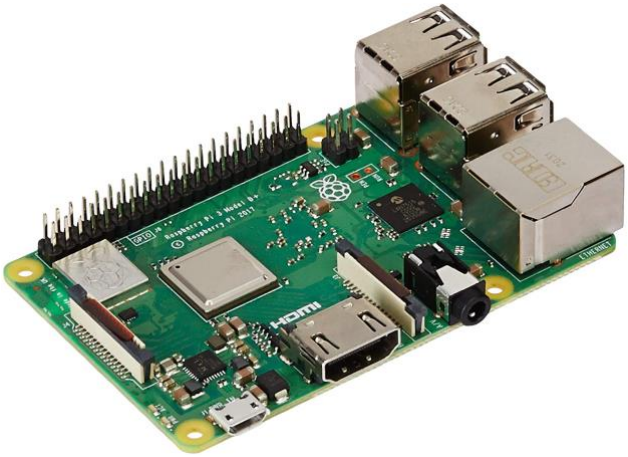

Figure 2: Raspberry Pi 3

Propeller Activity Board:

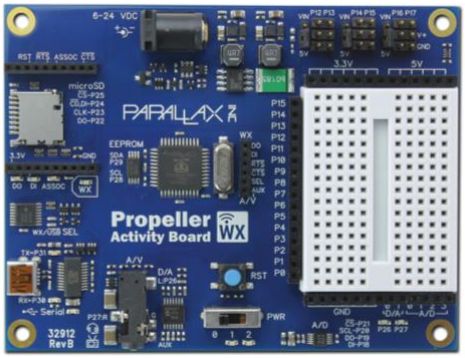

Figure 3: Propeller Activity Board

PI Camera:

To find the enemy or friendly block in the scenario, we used a pi camera where each block has different aruco tag ids. A pi camera detects all the tag ids and identifies each individual block as friendly or enemy.

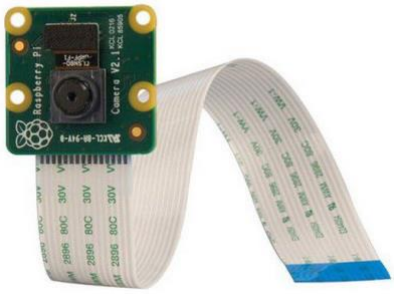

Figure 4: Pi Camera

QTR-8RC Sensor:

QTR-8RC reflectance sensor array is a line sensor, which can also be used as a proximity or reflection sensor. This sensor consists of 8 emitter and receiver pairs, this sensor sends the data to the microcontroller when it reflects the surface either 0 or 2500. This data can be used to make the robot follow a path and complete the task accordingly.

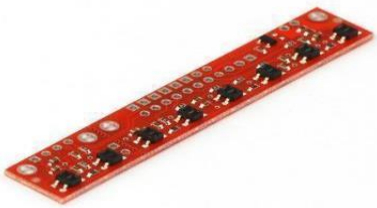

Figure 5: QTR Sensor

Ultrasonic Sensor:

The ultrasonic sensor is a distance sensor, which detects the object at a certain distance and sends back the data to the microcontroller. The sensor consists of an ultrasonic transmitter, a receiver, and a control circuit. This sensor will help detect the object and send feedback to the robot so that the robot moves toward the object.

Figure 6: Ultrasonic Sensor

Continuous Servo Motor:

This servo motor rotates continuously in a clock or anti-clockwise direction. Servo is attached to the wheels of the robot for movement. Sensor feedback is sent to the servos to stop at a particular junction, to move forward or backward, and to turn right or left. According to the sensor data, the servo works and helps the robot to follow the path.

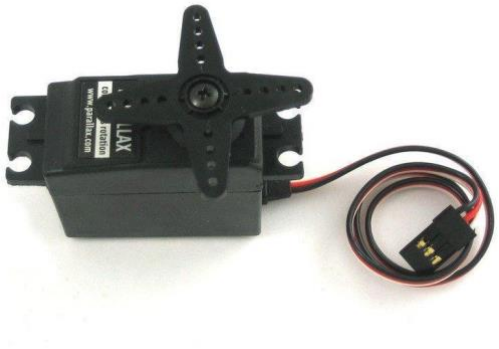

Figure 7: Continuous Servo Motor

Micro Servo Motor:

We have used micro servo motor SG90 for the actuation mechanism. The mechanism is attached to the left side of the robot, where it knocks down the enemy block once the pi camera detects the enemy. To make this actuation proper we choose this motor and also of its compact size, it has helpful for to place the motor in the right position.

Figure 8: Micro Servo Motor

Piezo Buzzer:

One piezo buzzer was used for the indication of intersections. Different frequencies were used in the buzzer for the different intersections So that it will be easy for the person to identify the intersections.

Figure 9: Piezo Buzzer

LED:

LEDs were used to indicate whether the object is found on the left side of the streets or the object on the main street of Manhattan. LED also blinks when it detects the block is a friend or an enemy.

Figure 10: LED

Robot Structure:

We have used Boe Bot for the structure and rebuilt the structure according to our needs. Added a few extra components to the boe bot to fix the sensors and the bottom and top of the bot.

Figure 11: Boe Bot

Actual Bot Structure:

The image below shows the rebuilt and redesigned bot as per the requirement for the

project. This includes some 3D printed parts for holding the QTR sensor, UltraSonic sensor, servo motor for a mechanism to destroy the enemies, and pi camera.

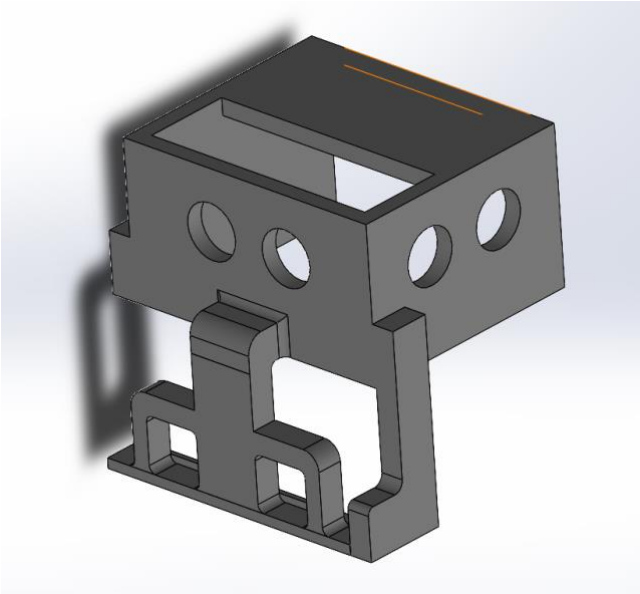

Figure 12: Front Part

This 3D printed part was designed to place QTR, ultrasonic, and pi camera. The pi camera was mounted on the part using an L clamp so that we have a correct coverage of the view to identify the id tags.

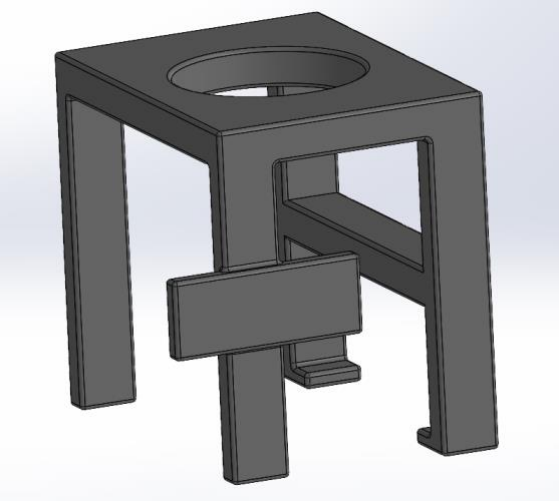

Figure 13: Main Part

The image shown above was designed to hold the major parts of the project. Propeller activity board was placed on top of the model and raspberry pi was placed at the bottom of the model. On the side, micro servo motor is fixed to destroy the enemy blocks as shown below.

Final Bot picture

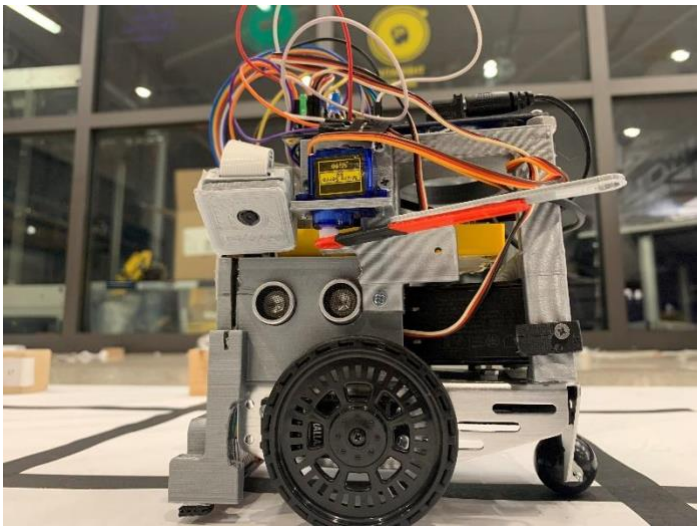

Figure 14: Side View of Robot

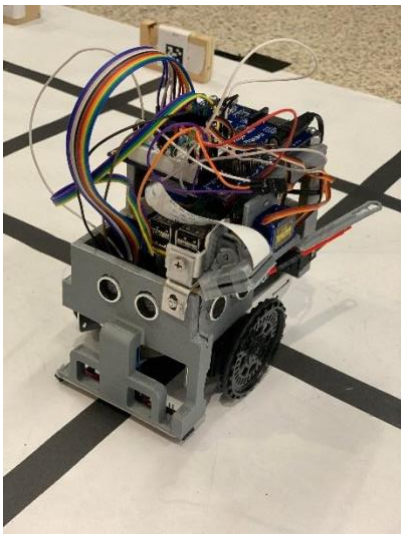

Figure 15: Front View of Robot

Working Principle:

Our Entire program is divided into following key steps which include Linesensing and line following, Object Sensing and Obstacle sensing, and Main program which includes our logic for traversal through the given streets with given directions to follow and communication from propellor to raspi when there is an object placed at the left side so that Raspi sends signal to start camera and check for ID tag. Once ID is detected internally it scans if its enemy or friend. If its friend then it blinks Green led and if its enemy then it sends signal for the microservo which is connected to raspi itself to actuate and knock the enemy out.

Step1: We programmed the bot in such a way that we use multiple cogs to run different parts of programs simultaneously.
We used Cog1 for Line following and Cog2 for Linesesnsing so that all the time the bot tries to follow the given black line while sensing the reading using polulu sensor.

```
void linefollow()
{  while(1)
   {
   //linesense();

    if(sensor1<=a && sensor3>=a && sensor4>=a && sensor8<=a)            //IR on black line move forward
    {
    //Move both the Motors
    servo_speed(right_servo, right_forward);
    servo_speed(left_servo, left_forward);
    }

    else if((sensor2<=a || sensor3<=a) && (sensor4>=a||sensor5>=a))
    {
    // stopping the left wheel and move left wheel for moving left SIDE
    servo_speed(right_servo, right_forward);
    servo_speed(left_servo, 0);
    }
                                                          //IR not fully on black line: perform correction action move Left
    else if((sensor2>=a || sensor3>=a) && (sensor4<=a || sensor5<=a))
    {
    //stopping the right wheel and move left wheel for moving right SIDE
    servo_speed(right_servo, 0);
    servo_speed(left_servo, left_forward);
    }

    else
    {
    //Move both the Motors
    servo_speed(right_servo, right_forward);
    servo_speed(left_servo, left_forward);
    }
    }
}
```

```
//Line sensing function
void linesense()
{
    set_directions(7, 0, 0b11111111); // Set P0-P7 as output pins.
    set_outputs(7, 0, 0b11111111); // Set P0-P7 as high pins
    pause(1);
    sensor1 = rc_time(0, 1);
    sensor2 = rc_time(1, 1);
    sensor3 = rc_time(2, 1);
    sensor4 = rc_time(3, 1);
    sensor5 = rc_time(4, 1);
    sensor6 = rc_time(5, 1);
    sensor7 = rc_time(6, 1);
    sensor8 = rc_time(7, 1);
    //printf(" %d ,%d ,%d ,%d ,%d ,%d , %d \n", sensor1, sensor2, sensor3, sensor4, sensor5, sensor6, sensor7, sensor8);
    //pause(10);
}
```

Step2:

We are using two Ultrasonic sensors one at the front and one towards the left of the bot so that we can detect the objects and delivery locations and I, B & A lines respectively. Our program is programmed in such a way that only at junctions or intersections our Ultrasonic will sense object lights up blue led once an objection or location is detected.

```
void object_sensing()
{
  ultrasense_left();
  print("my distance %d cm\n", distance_left);
  print("Sensing object left");

  if (distance_left<=10)
   {
    high(15);
    pause(100);
    low(15);
    print("object detected at %d cm\n", distance_left);
    //pause(200);
    count_object++;

    if (count_object==2 && obstacle_found==1){
       high(15);
      pause(100);
      low(15);
      high(15);
      pause(100);
      low(15);
      servo_speed(right_servo, 0);
      servo_speed(left_servo, 0);
      high(15);
      print("Completed run\n");
      pause(15000);
      stop = 1;
      l_line1=105;
      ultrasense();
     }
    }
}
```

```
int ultrasense()
{
 low(trigPin_front);
 pulse_out(trigPin_front, 10);
 long tEcho = pulse_in(echoPin_front, 1);
 distance_front = tEcho / 58;
 //print("%d cm\n", distance_front);
 //pause(200);
}


int ultrasense_left()
{
 low(trigPin_left);
 pulse_out(trigPin_left, 10);
 long tEcho = pulse_in(echoPin_left, 1);
 distance_left = tEcho / 58;
 //print("%d cm\n", distance_left);
 //pause(200);
}
```

Step3: This is our raspi code where we tell the pi camera to actuate and sense for friends and enemies ID tags and tells the microservo to knock the enemy down when pi camera detects enemy ID tag while green led is glowing and for friends it blinks the green led twice.

```
dictionary = cv2.aruco.Dictionary_get(cv2.aruco.DICT_6X6_250)

# Initialize the detector parameters using default values
parameters = cv2.aruco.DetectorParameters_create()
camera = cv2.VideoCapture(-1)

pwm=GPIO.PWM(ServoPin, 50)
pwm.start(0)

while True:

    _, img = camera.read()
    sleep(0.1)
    inputValue=GPIO.input(TrigPin)

    if(inputValue == True):
        # creates an "img" var that takes in a camera frame
        _, img = camera.read()
        sleep(0.1)

        # Convert to grayscale
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

        # detect aruco tags within the frame
        markerCorners, markerIds, rejectedCandidates = cv2.aruco.detectMarkers(gray, dictionary, parameters=parameters)

        # draw box around aruco marker within camera frame
        img = cv2.aruco.drawDetectedMarkers(img, markerCorners, markerIds)
        print(markerIds)

        if (int (markerIds or 50) >= 10 and (markerIds or 50)<=20):

            GPIO.output(LEDPin, GPIO.HIGH)
            sleep(1)
            GPIO.output(LEDPin, GPIO.LOW)
            sleep(1)
            pwm.ChangeDutyCycle(6.5) # move to 90 deg forward to hit object
            sleep(0.5)
            pwm.ChangeDutyCycle(1.5) # move back to original position
            sleep(0.5)
            pwm.ChangeDutyCycle(0) # to make jitters go
            sleep(0.5)
            PwmDuty=0

        else:

            GPIO.output(LEDPin, GPIO.HIGH)
            sleep(0.1)
            GPIO.output(LEDPin, GPIO.LOW)
            sleep(0.1)
            GPIO.output(LEDPin, GPIO.HIGH)
            sleep(0.1)
            GPIO.output(LEDPin, GPIO.LOW)
            sleep(1)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

# When everything done, release the capture
camera.release()
cv2.destroyAllWindows()l
```

Step4: This is our main logic which tells the bot to traverse through the given map. So the Logic we have given is to scout for Object first in the external box and then enter the midline I for obstacle detection and object sensing, so once the 8 objects which includes 4 enemies and 4 friends are found and obstacle is detected the bot stops in the center (Mid) I line. if the

bot doesnt detect 8 objects and detects less objects it means the object is possible at B5 so the after to find the obstacle at the centre line the bot searches again then enters B line and goes to B5 to detect the objects and stops there.

Circuit Diagram:

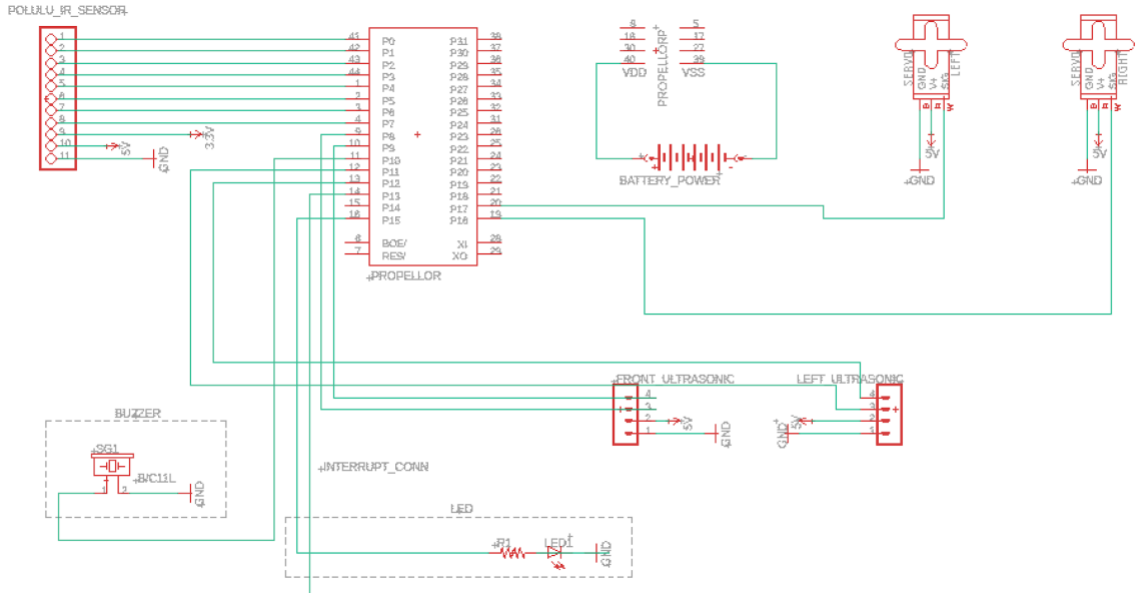

Figure 16: Circuit Diagram 1

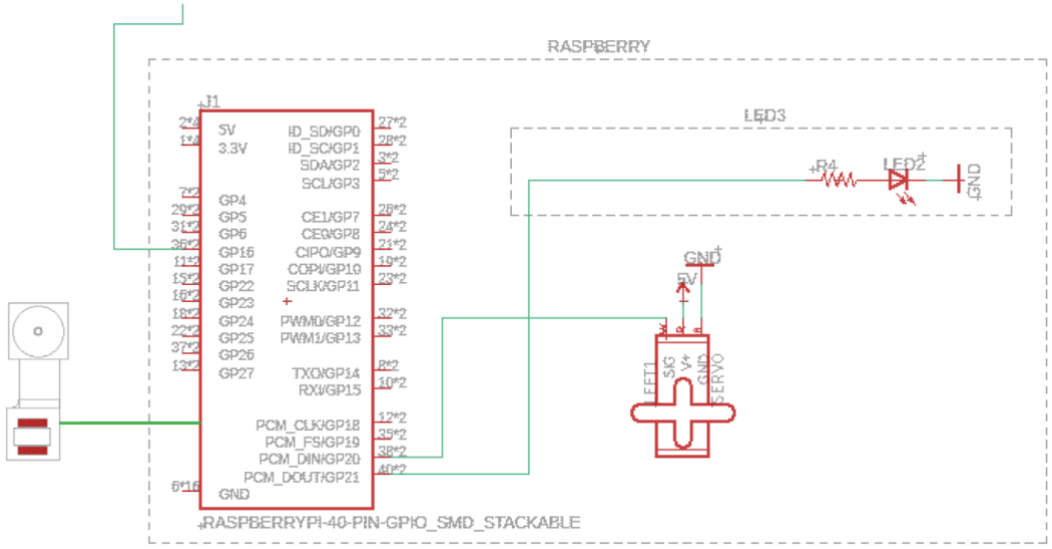

Figure 17: Circuit Diagram 2

Bill of Materials:

Part Name	Quantity	Amount
Raspberry Pi 3	1	$ 140

Propeller Board	1	$ 72
Pi Camera	1	$ 25
QTR Sensor	1	$ 10
Ultra Sonic	2 ($4 per piece)	$ 8
Continous Servo Motor	2 ($20 per piece)	$ 40
Micro Servo Motor	1	$ 6
Piezo Buzzer	1	$ 2
Batteries	1 pack	$ 4
Miscellaneous		$ 30
Total		$ 337

Results:

We approached differently for this project. When the robot reaches i1 intersection it takes the left and scouts the left lane of Manhattan after that it turns right and scouts the right lane of Manhattan for the object. The robot scouts for friends and enemies on both the left and right sides of the lane with the help of the Pi camera. If the camera detects the id of an enemy block our robot knocks off the block and moves ahead. If a friend is detected only the led blinks and robot moves forward. After scouting the left and right lanes, our robot starts to scout the centre lane, where it detects the enemy and friend on both sides of the center lane and stop when it detects the traffic block. This is the scene if there is no block at B5. If there is a block at B5 then our robot takes a u-turn and scouts for the B5 in the left lane and stops at B5. We designed a simple knocking system to destroy the enemy from the block.