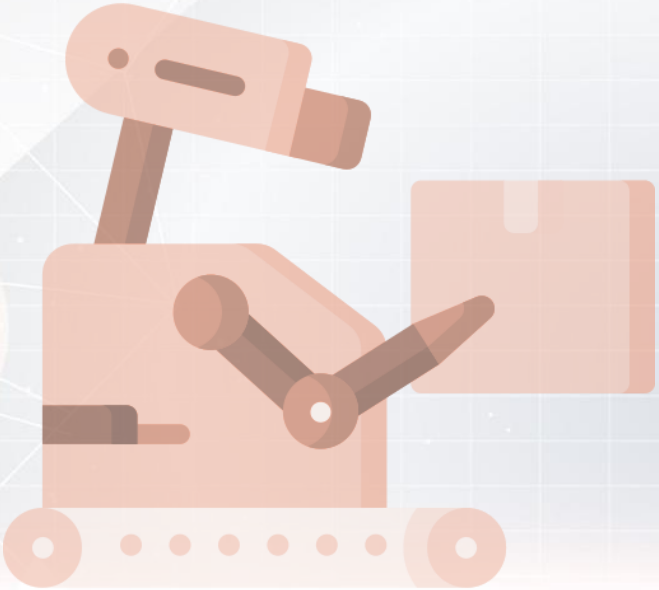


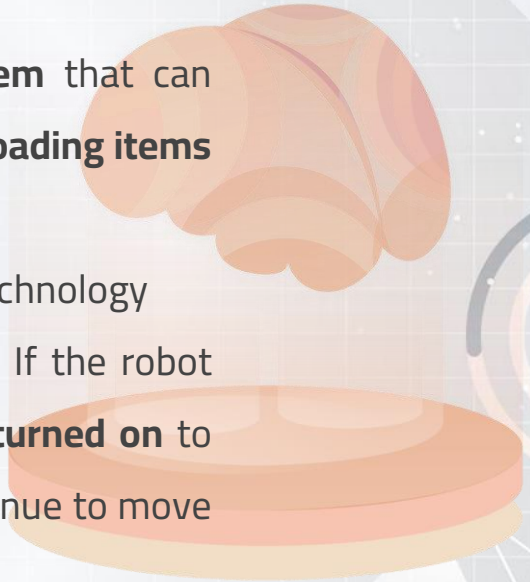
AUTONOMOUS DELIVERY SYSTEM

By Foxtrot Members:

H'NG KEAN TEONG	(206390)
SHURIIEN A/L SUGARMARAN	(206426)
LIM QI YANG	(206464)
ALIA BINTI ABU HASAN	(207544)
NUR HAZIMAH IZZATIE BINTI MISNAN	(207603)

INTRODUCTION

- This project is about creating an **automated delivery system** that can **deliver items and equipped with a sliding mechanism for unloading items from one location to another** in an automated way.
- Users can choose the unloading point through the use of IoT technology
- After unloading, the robot will **return to its original position**. If the robot encounters an **obstacle, it will stop**, and the **buzzer will be turned on** to alert people. Once the obstacle is removed, the robot will continue to move on its path.

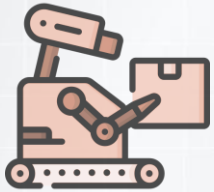


PROBLEM STATEMENT

The line-following robotic delivery device is an example of how technology can be used to improve logistics and transportation

- Through IoT, robots can be automated to perform tasks more efficiently and cost-effectively than human labor
- Can contribute to industry by scaling up the technology and replace traditional rails as a form of long-distance logistics
- It will be a more reliable, cheaper and environmentally sustainable form of transport

OBJECTIVES



01

To design an automated delivery control system

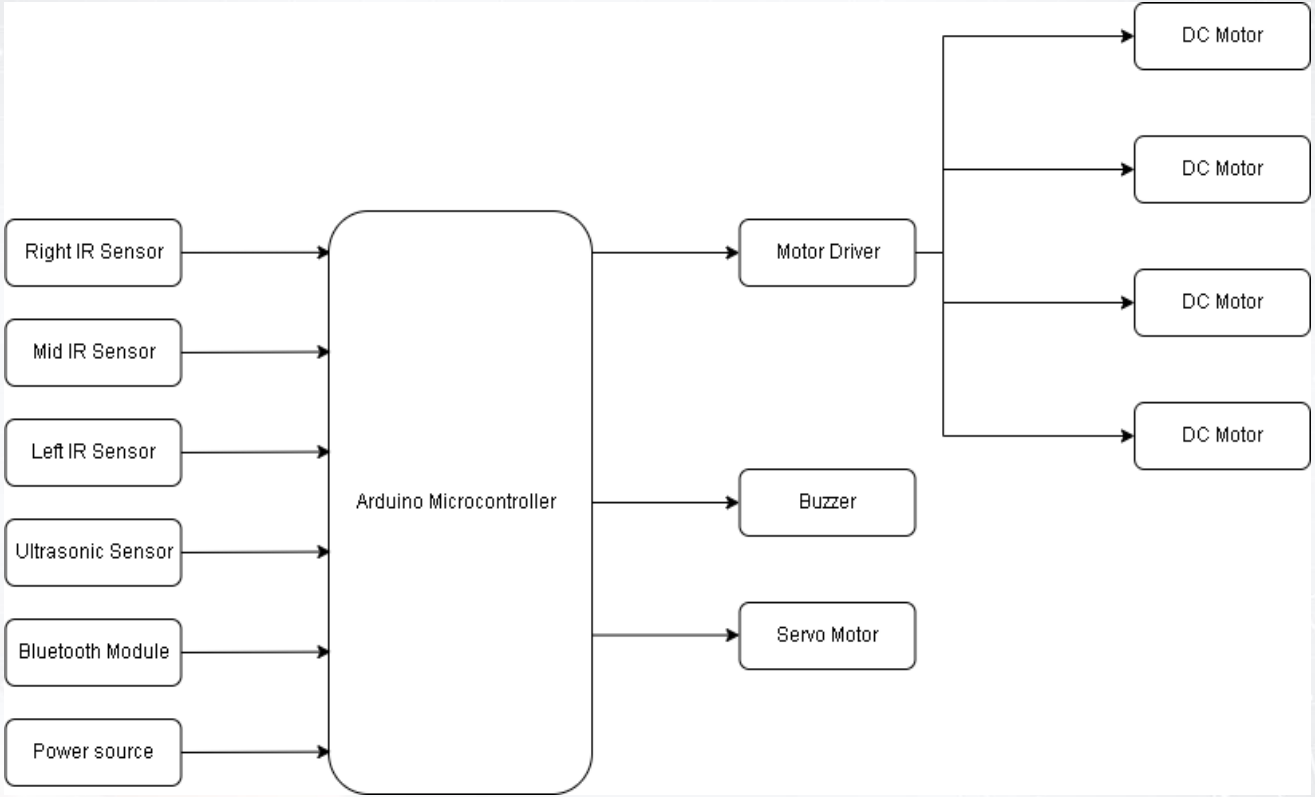
02

To transport the items from one location to the other location

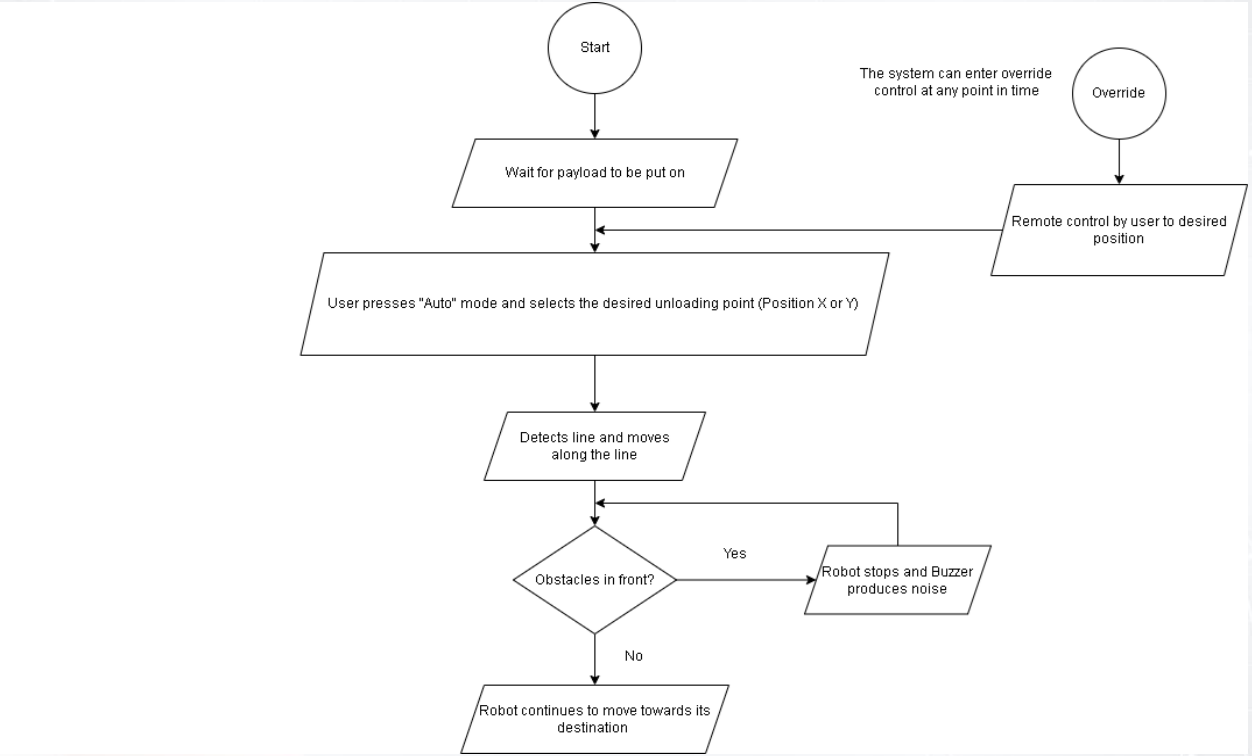
03

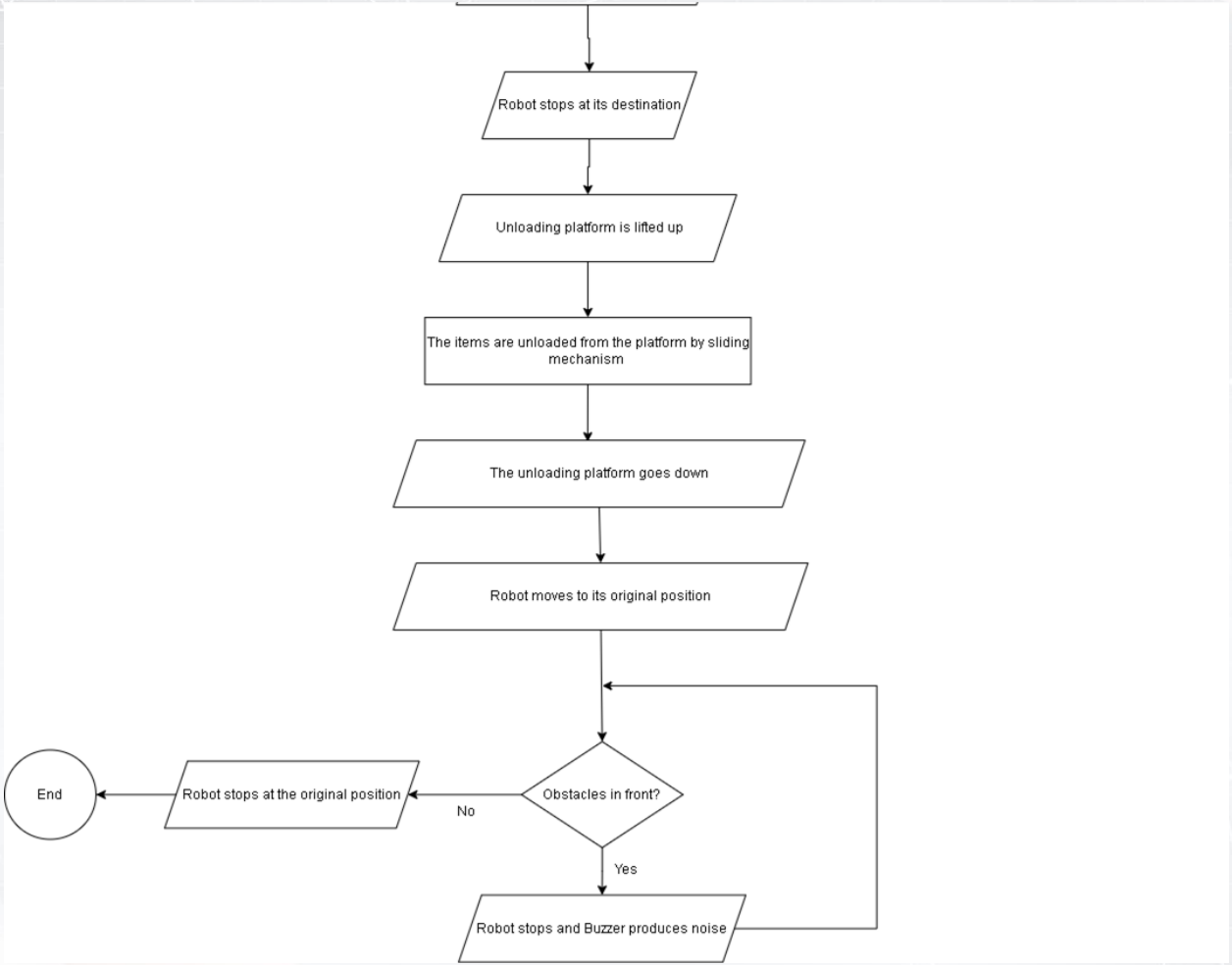
To increase the efficiency of work

BLOCK DIAGRAM



FLOW CHART


Start
Wait for payload to be put on
User presses "Auto" mode and selects the desired unloading point (Position X or Y)
Detects line and moves along the line
Obstacles in front?
Yes
Robot stops and Buzzer produces noise
No
Robot continues to move towards its destination
The system can enter override control at any point in time
Override
Remote control by user to desired position

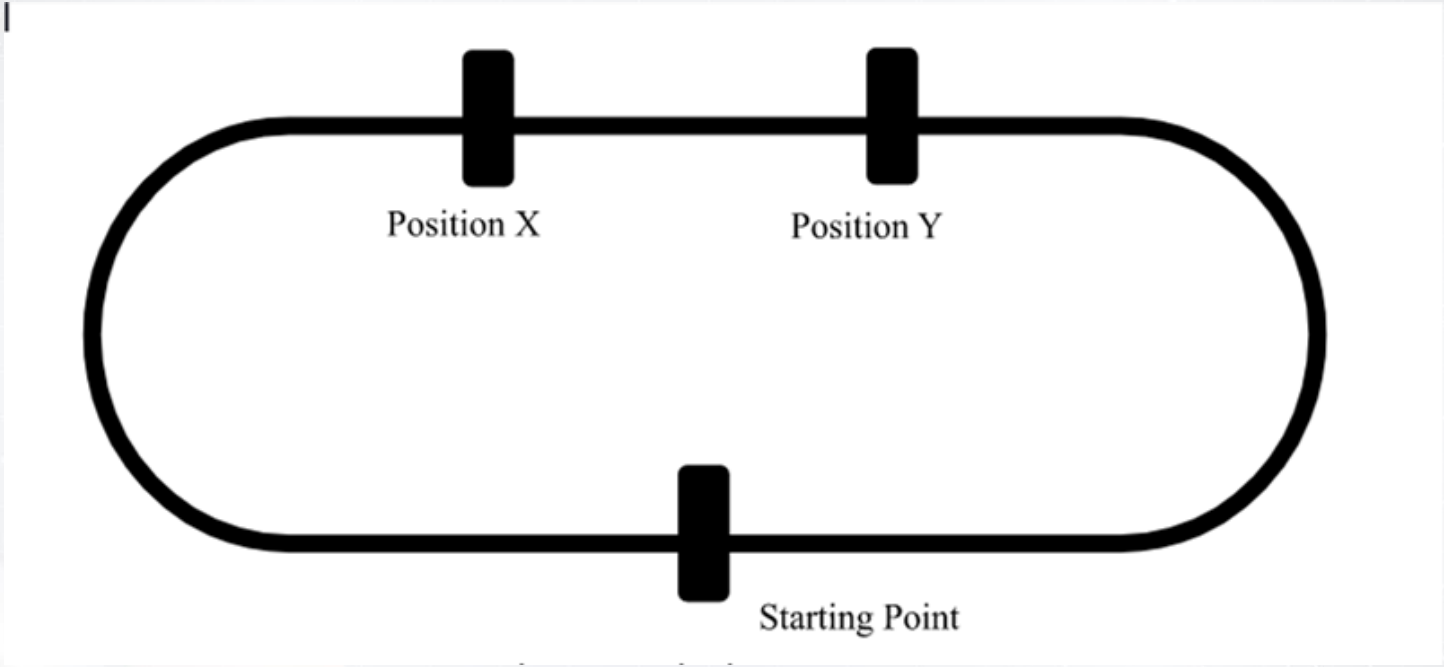

Robot stops at its destination
Unloading platform is lifted up
The items are unloaded from the platform by sliding mechanism
The unloading platform goes down
Robot moves to its original position
Obstacles in front?
No
Yes
Robot stops at the original position
End
Robot stops and Buzzer produces noise

OPERATION EXPLANATION

- After the user has put on the payload, the user is required to choose the mode for the robot car through the Bluetooth module. If the manual control mode is chosen, the robot car can be manually controlled by the user through the Bluetooth module.

- If auto mode is chosen, the user is required to choose the desired unloading point (Position X or Y). So that the item can be unloaded at the desired place. After the unloading point is selected, the robot car moves automatically in a clockwise direction to the destination on the track.

TRACK LINE



OPERATION EXPLANATION

- The delivery robot's route is created using black tape. Three IR sensor modules, namely the left sensor, middle sensor and right sensor. The robot moves forward when the middle sensor senses black. The robot turns the left side when the left sensor on a black line. If the right sensor senses a black line, the robot turns right until the middle sensor reaches a black surface. If all three sensors come on a black line, the robot stops.

- Suppose there is an obstacle in front of the delivery robot car, and the ultrasonic sensor senses it. In that case, it will immediately stop, and the buzzer produces noise to alert the people. Once the obstacle is removed, the delivery robot car continues to move on its path.

OPERATION EXPLANATION

- When the robot cat arrives at the destination, the servo motor will be activated, and the unloading platform will be lifted up. As a result, the items slide along the platform to the ground.

- After a moment, the unloading platform goes down, and the robot will return to its original position.

- Likewise, when the robot encounters obstacles in the journey of heading back, the buzzer will be activated, and the robot will be stopped immediately.

- After the robot reaches the original position or starting point, the robot is stopped and waits for the user to put on the payload.

CODING OF THE DELIVERY ROBOT

```
#include <NewPing.h> // library for ultrasonic distance sensor
#include <Servo.h> // library for servo motor

// instantiate ultrasonic distance sensors to digital pin 2, 3, and 10
NewPing ultra_sensor ( 3 , 2 , 10 );

//instantiate servo motor
Servo servo_9;

// instantiate speed (digital pin 5) and direction of rotation (digital pin 4) of left-side tyres
int spd_A = 5;
int dir_A = 4;

// instantiate speed (digital pin 6) and direction of rotation (digital pin 7) of left-side tyres
int spd_B = 6;
int dir_B = 7;

// instantiate buzzer to digital pin 8
int buzzer = 8;

// instantiate left, middle, and right IR sensors (analog pin A3, A4, and A5) and variables to store the IR sensor value
int left_IRsensor = A4;
int left_sensor_value = 0;
int right_IRsensor = A3;
int right_sensor_value = 0;
int mid_IRsensor = A5;
int mid_sensor_value = 0;

// instantiate character variable to receive incoming bluetooth data
char data ;

// instantiate mode of operation (mode 0 = autonomous, mode 1 = remote control)
int mode = 0 ;

// instantiate type of motion including forward, backward, left, right and stop
int motion = 0 ;
```

CODING OF THE DELIVERY ROBOT

```
37
38  int loc_count = 0; // counter to record number of stop points passed through
39  int loc_user = 0; // record user-desired stop point input
40
41  void setup() {
42    //setting up all pinmodes as output
43    pinMode(spd_A , OUTPUT);
44    pinMode(dir_A , OUTPUT);
45    pinMode(spd_B , OUTPUT);
46    pinMode(buzzer, OUTPUT);
47    pinMode(dir_B , OUTPUT);
48
49    // attaching servo motor to digital pin 9
50    servo_9.attach(9);
51
52    // begin serial communication of Arduino UNO board
53    Serial.begin(9600);
54  }
55
56  // function to move forward in autonomous mode
57  void forward() {
58    digitalWrite(dir_A , 1);
59    analogWrite (spd_A , 90);
60    digitalWrite (dir_B , 0);
61    analogWrite (spd_B , 90 );
62  }
63
64  // function to move backwards in autonomous mode
65  void back() {
66    digitalWrite(dir_A , 0);
67    analogWrite (spd_A , 90);
68    digitalWrite (dir_B , 1);
69    analogWrite (spd_B , 90 );
70  }
71
72  // function to turn left in autonomous mode
```

```
73  void left() {
74    digitalWrite(dir_A , 0);
75    analogWrite (spd_A , 0);
76    digitalWrite (dir_B , 0);
77    analogWrite (spd_B , 90 );
78  }
79
80  // function to turn right in autonomous mode
81  void right() {
82    digitalWrite(dir_A , 1);
83    analogWrite (spd_A , 90);
84    digitalWrite (dir_B , 0);
85    analogWrite (spd_B , 0 );
86  }
87
88  // function to move forward in remote control mode
89  void forward1() {
90    digitalWrite(dir_A , 1);
91    analogWrite (spd_A , 200);
92    digitalWrite (dir_B , 0);
93    analogWrite (spd_B , 200 );
94  }
95
96  // function to move backward in remote control mode
97  void back1() {
98    digitalWrite(dir_A , 0);
99    analogWrite (spd_A , 200);
100   digitalWrite (dir_B , 1);
101   analogWrite (spd_B , 200 );
102 }
103
104 // function to turn left in remote control mode
105 void left1() {
106   digitalWrite(dir_A , 0);
107   analogWrite (spd_A , 150);
108   digitalWrite (dir_B , 0);
```

CODING OF THE DELIVERY ROBOT

```
  analogWrite (spd_B , 150 );
}

// function to turn right in remote control mode
void right1() {
  digitalWrite(dir_A , 1);
  analogWrite (spd_A , 150);
  digitalWrite (dir_B , 1);
  analogWrite (spd_B , 150 );
}

// function to stop in remote control mode
void Stop() {
  digitalWrite(dir_A , 0);
  analogWrite (spd_A , 0);
  digitalWrite (dir_B , 0);
  analogWrite (spd_B , 0 );
}

void loop() {
  // instantiating distance variable to record distance in front of ultrasonic distance sensor
  // ultrasonic distance sensor is positioned in the front position of the car
  int distance = ultra_sensor.ping_cm();

  // reading analog values from IR sensors and storing into variables
  left_sensor_value = analogRead(left_IRsensor);
  right_sensor_value = analogRead(right_IRsensor);
  mid_sensor_value = analogRead(mid_IRsensor);

  // delay 0.2 seconds to make entire loop run slower, thus not overloading the physical components
  delay(200);

  if ( Serial.available()) {
    // read incoming data through bluetooth connection
    data = Serial.read();
```

```
  // choose between remote control (M) or autonomous (A) mode
  if (data == 'M') {
    mode = 1 ;
  }
  if (data == 'A') {
    mode = 0;
  }

  // location for load drop-off point for autonomous mode
  if (data == 'X') {
    loc_user = 1;
  }
  if (data == 'Y') {
    loc_user = 2;
  }

  // instruction input for remote control mode
  if (data == 'F') {
    motion = 1;
  }
  if (data == 'B') {
    motion = 2;
  }
  if (data == 'R') {
    motion = 3 ;
  }
  if (data == 'L') {
    motion = 4 ;
  }
  if (data == 'S') {
    motion = 0 ;
  }
}

// AUTONOMOUS CONTROL CODE BLOCK //
```

CODING OF THE DELIVERY ROBOT

```
if ( mode == 0 && loc_user != 0 ) // car will only move if autonomous mode and drop-off location is chosen
{
  if (distance == 0) { // if there is no object in front of the ultrasonic distance sensor
    digitalWrite(buzzer , 0); // the buzzer will be deactivated and motion is continued

    // IR sensor
    // value > 900 = white region
    // value < 900 = black region

    // for when IR detects all black region (stop point)
    if (right_sensor_value < 900 && mid_sensor_value < 900 && left_sensor_value < 900) {
      loc_count++; //location counter increase by 1

      // when reach desired point, car stops moving
      // object is dropped off by activating servo motor
      // then the car continues to move
      // the car will not stop if location counter does not match user desired drop-off point
      if (loc_count == loc_user) {
        delay(500);
        Stop();
        servo_9.write(0);
        delay(2000);
        servo_9.write(90);
        forward();
      }

      // when counter reach 3, it indicates that car has returned to initial position
      // counter and user location is reset to zero
      if (loc_count == 3) {
        forward();
        delay(1000);
        Stop();
        loc_count = 0;
        loc_user = 0;
      }
```

CODING OF THE DELIVERY ROBOT

```
217         // delay of 0.5s is introduced to prevent overlapping increment of location counter at stop point
218         delay(500);
219       }
220
221       // if only detect black line, go forward
222       else if (mid_sensor_value < 900) {
223         forward();
224       }
225
226       // if detect black line on the right, turn right
227       else if (right_sensor_value < 900  && left_sensor_value > 900) {
228         right();
229         delay(10);
230
231       }
232
233       // if detect black line on the left, turn left
234       else if (right_sensor_value > 900 && left_sensor_value < 900) {
235         left();
236         delay(10);
237       }
238
239       // if no black line is detected at all, meaning car is off track
240       // since the car trajectory is clockwise, the car only has to keep turning right until it detects the track again
241       else if (right_sensor_value > 900 && mid_sensor_value > 900 && left_sensor_value > 900) {
242         right();
243         delay(5);
244       }
245     }
246
247     // if there is an obstacle, the car stops
248     // the buzzer is activated
249     // the buzzer will only deactivate and car continue motion if the obstacle is removed
250     else if (distance != 0) {
251       Stop();
252       digitalWrite(buzzer , 1);
```

CODING OF THE DELIVERY ROBOT

```
    }
  }

  // REMOTE CONTROL CODE BLOCK //

  if (mode == 1) { // mode 1 is remote control mode
    // data input from bluetooth above is executed in this block
    if (motion == 0) {
      Stop();
    }
    else if (motion == 1) {
      forward1();
    }
    else if (motion == 2) {
      back1();
    }
    else if (motion == 3) {
      right1();
    }
    else if (motion == 4) {
      left1();
    }
  }
}

```

LIST OF MATERIALS

1

Arduino Uno

A microcontroller board that give command for the robot

2

DC Motor

To move the robot car

3

Motor Driver

To control the motor

4

Ultrasonic Sensor

To detect the distance between the robot and obstacle

5

IR Sensor

To keep the robot car on the specific track

6

Bluetooth Module

Connect the Arduino with phone

7

Buzzer

Produce noise to indicate obstacle

8

3.7V Battery

To supply power to the robot car

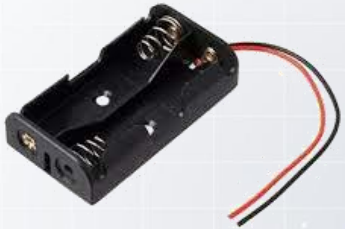
9

Servo Motor

To lift up the unloading platform

LIST OF MATERIALS

10



Battery holder

A platform for batteries to be inserted

11

Smart Robot Tires

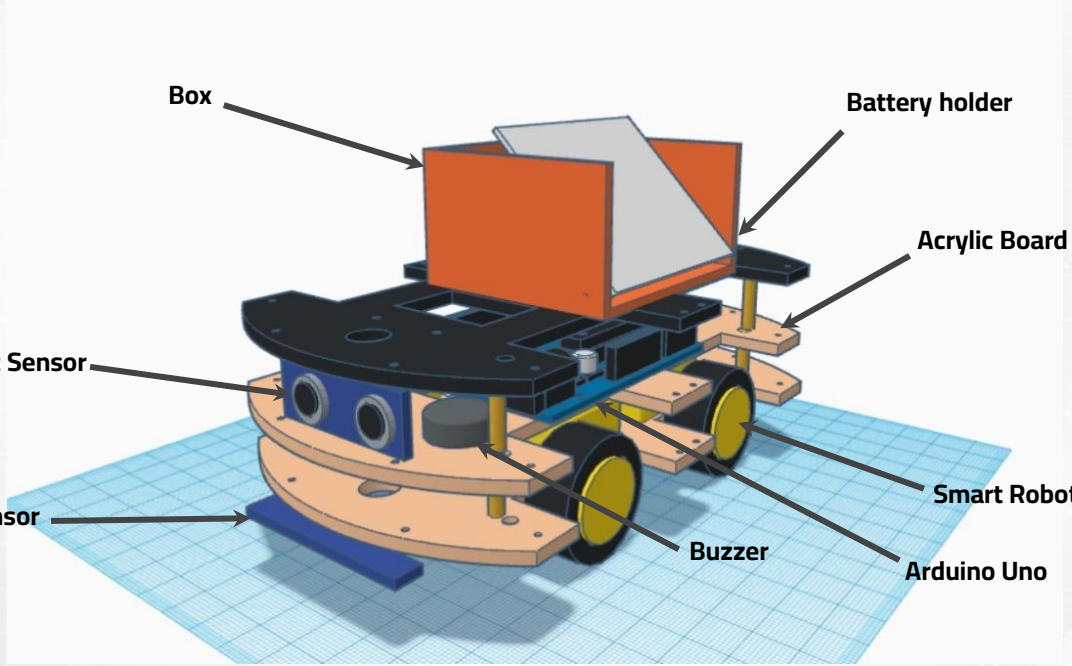
Enable the robot car move

12

Acrylic Board

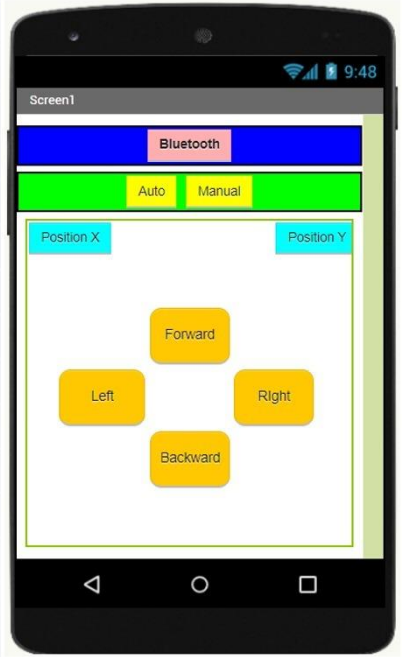
Act as the body of the car

VISUAL DIAGRAM OF DELIVERY ROBOT



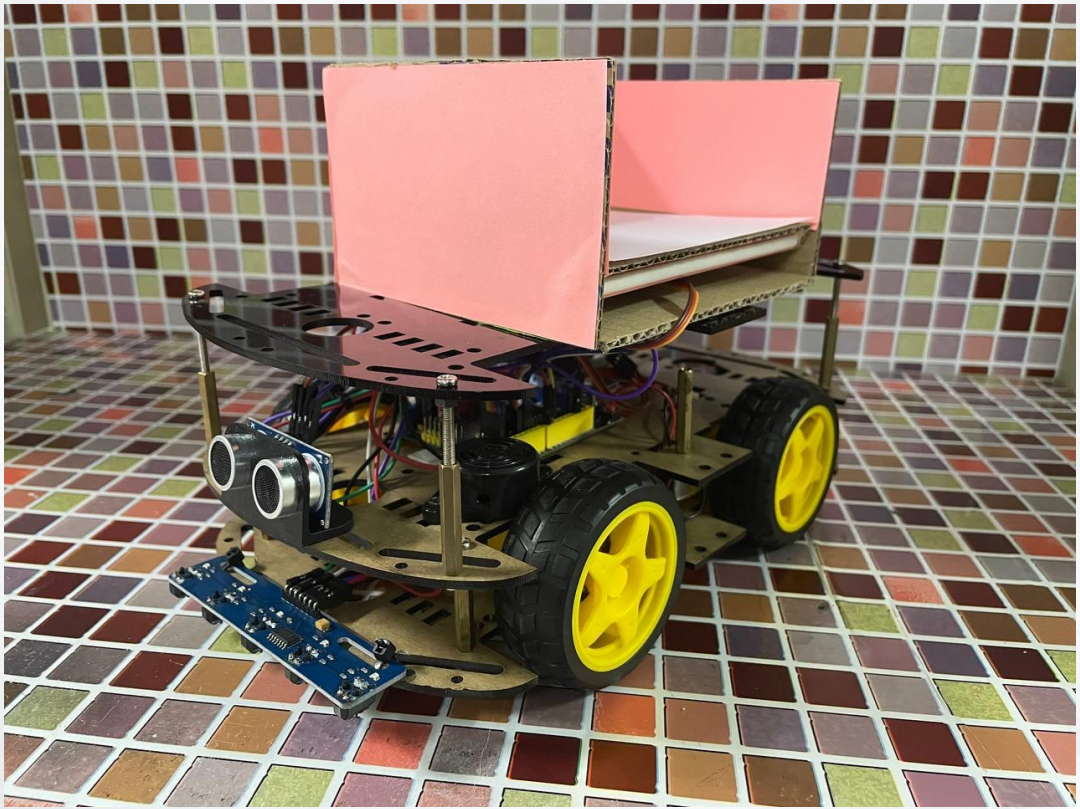
APPLICATION FOR THE SYSTEM

MIT app code block



UI of the Application

REAL MODEL OF DELIVERY ROBOT

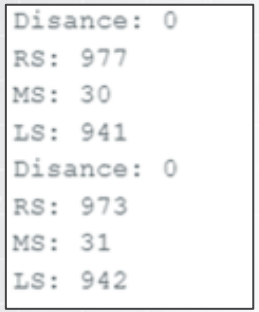


PROJECT DEMONSTRATION

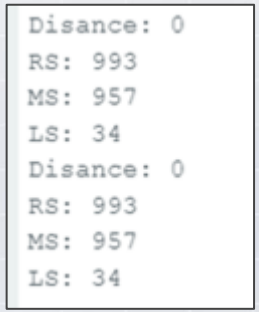
CONTROLLED VARIABLE

The IR sensor will detect the refracted light of the path :

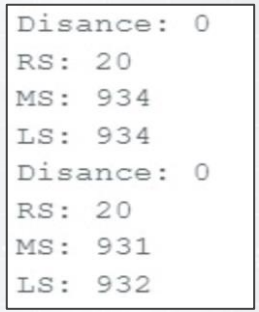
- When the value less than 900 : it will be considered as black line
- When the value more than 900: it will considered as white area



Middle sensor detect the refracted light less than 900 so it is on the black line. Thus it will move forward



Left sensor detect the refracted light less than 900 so it will turn left



Right sensor detect the refracted light less than 900 so it will turn right

CONTROLLER TYPE

Controller type is ON-OFF system, ON is ‘1’ and OFF is ‘0’

Autonomous mode - closed loop system.

- IR sensor read analog inputs and converts to digital values.
- Arduino makes decision based on pre-programmed decision tree.
- The DC motor speed, direction of the wheel and angle of servo motor

Remote control mode - open loop system

- Input data received from Bluetooth. Arduino execute instruction without considering the feedback of the system. Cycle will repeat until turned OFF

CONCLUSION

- All the objectives have been achieved; An autonomous delivery control system was built to transport material to the desired destination and to increase the efficiency of work.
- Limitation of this project and its improvements in the future
  - The robot car cannot move when there is obstacle placed in front of it
    - Improvement: Making an arm that can push the obstacle from the track
  - The robot cannot move on a bumpy surface
    - Improvement:Increasing the wheel height of the robot to its body and adding springs and shock absorbers
  - It requires 3-7 cm broad line track to move and take turns properly
    - Improvement: Making the robot more sensitive to the line on the track