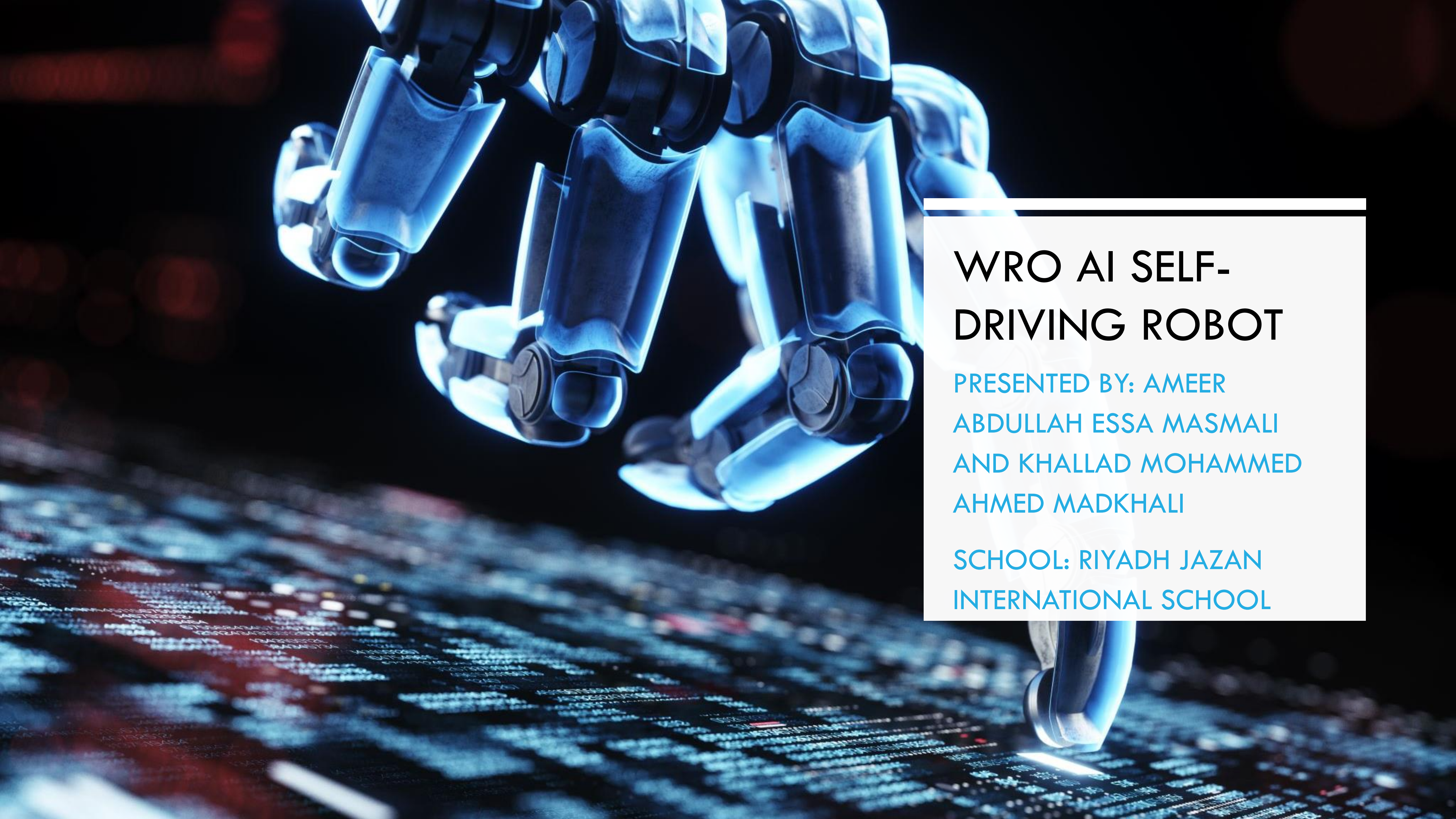


WRO AI SELF-DRIVING ROBOT

PRESENTED BY: AMEER ABDULLAH ESSA MASMALI AND KHALLAD MOHAMMED AHMED MADKHALI

SCHOOL: RIYADH JAZAN INTERNATIONAL SCHOOL

THE ISSUE IN REAL-TIME

- Robots today have some serious issues that can and need to be solved.

1. Human Error and Safety

   Human error is a leading cause to accidents on the road. With self-driving cars, we aim to reduce accidents significantly by minimizing or eliminating the role of human drivers.

2. Environmental Impact

   Cars have always given off pollution. Self-driving cars have the potential to reduce greenhouse gas emissions by reducing traffic congestion, optimizing driving patterns, and promoting the adoption of electric and alternative fuel vehicles.

OUR MISSION

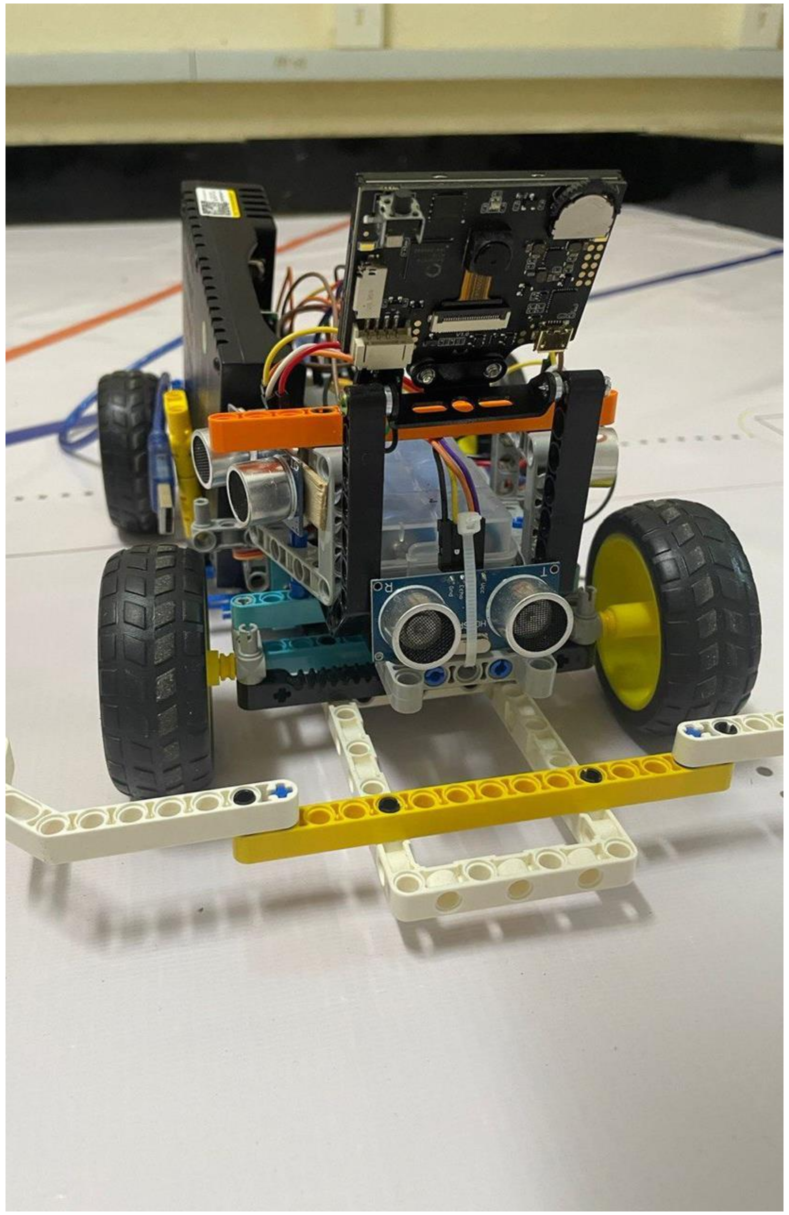
- As students, we envision our self-driving robot to as perfect as possible. For this, we have key goals for our robot.

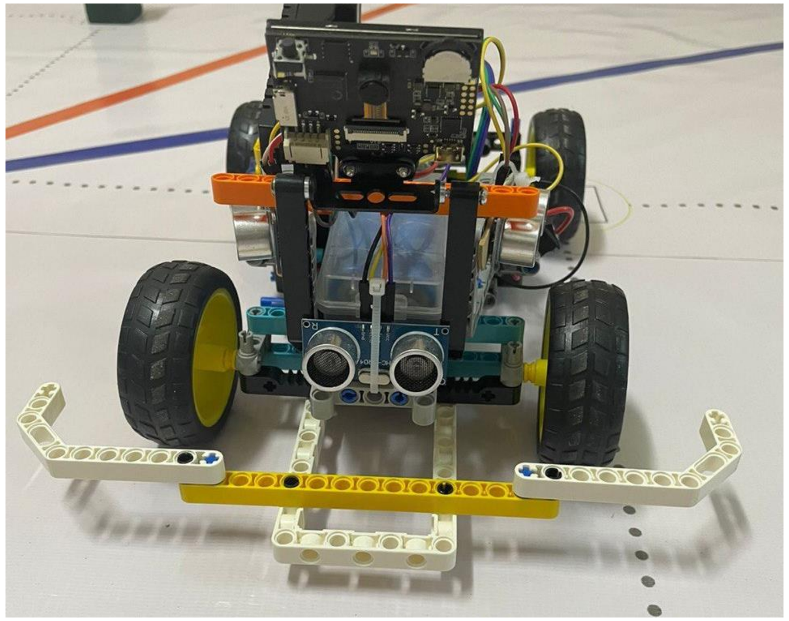
  Safe Operation

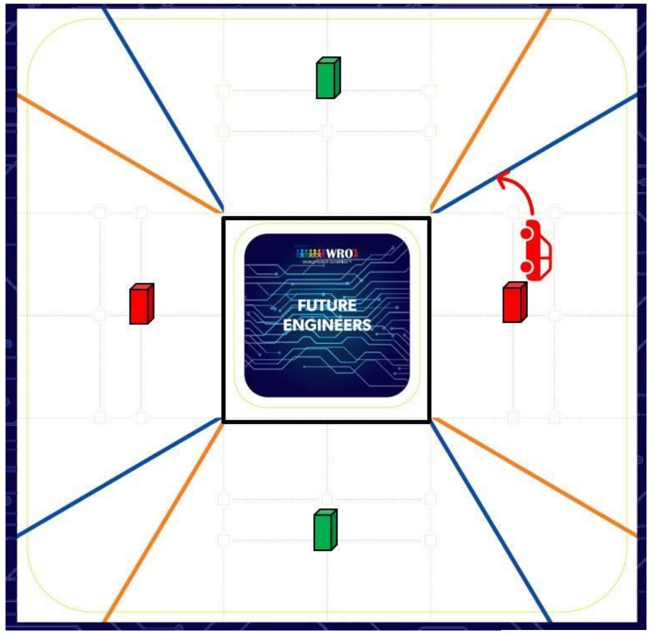
  The mission of a self-driving robot is to operate safely and the risk of accidents or collisions.

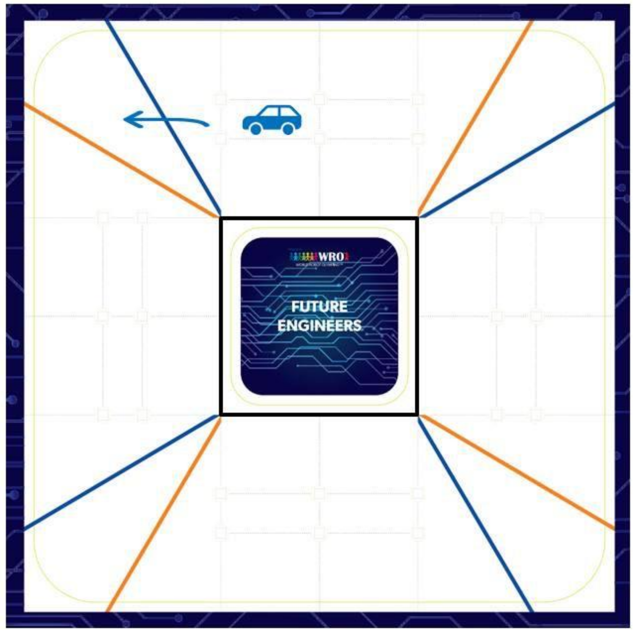
  Efficient Navigation

  Self-driving cars aim to navigate efficiently from one location to another.









STRATEGY

- Our robot is a self-driving car. The robot is powered by AI using HuskyLens. The robot stays on the right side of the wall. We're using color recognition so that the robot can detect the green, red and blue blocks scattered around the map.

HUSKYLENS

- As you already know, we have three different colors that the lens detects. The colors consist of blue, red, and green. The blue color is a blue line on the track. The red and green colors are blocks on the track that are used as obstacles to indicate where the robot needs to go. The HuskyLens is a camera sensor that helps the robot find its destination using the blocks and line. Each block has a different role for the HuskyLens camera sensor.

Blue line: The robot uses this block to count each lap that it completes.

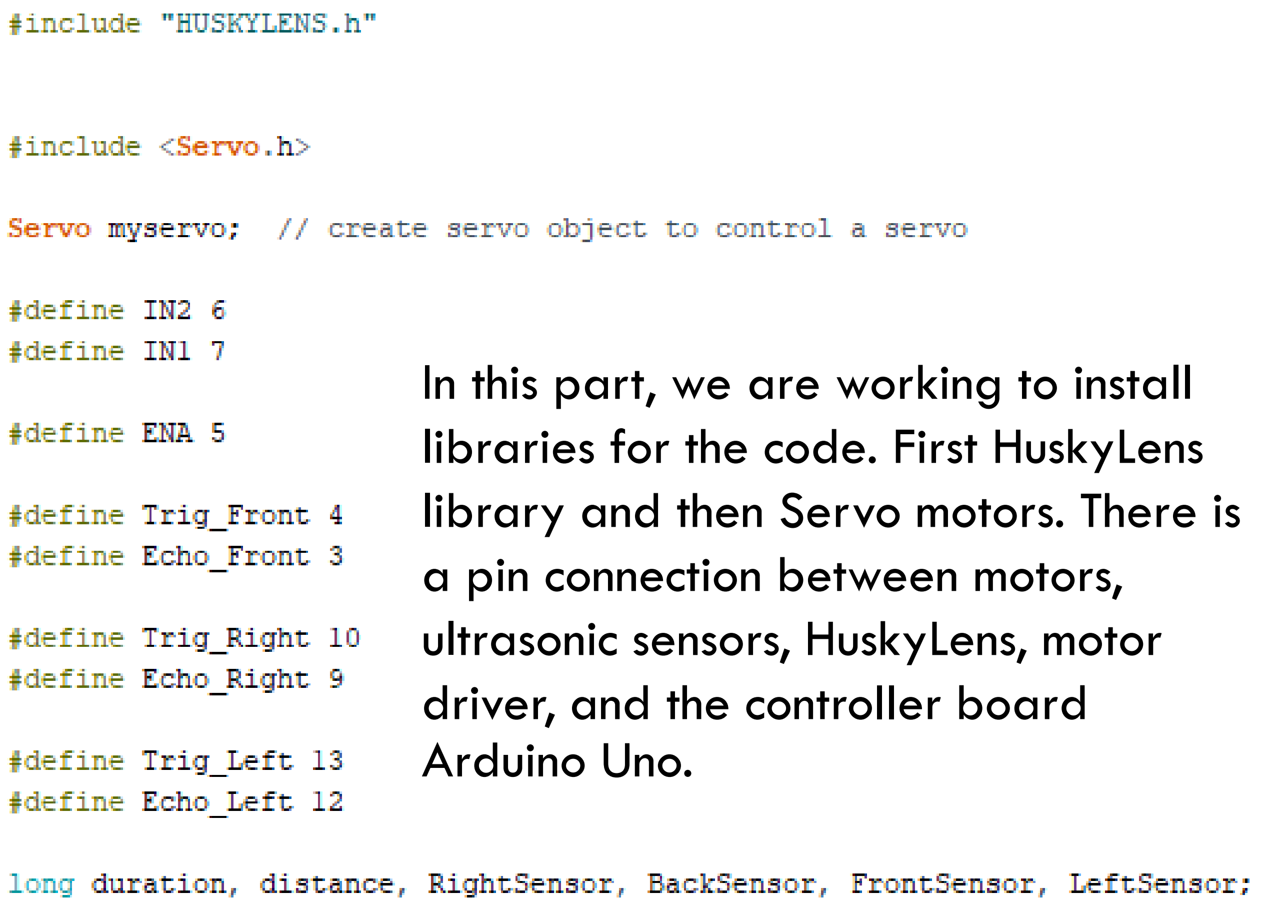
Red block: The red block indicates the robot to turn and move right.

Green block: The green block indicates the robot to turn and move left.

EFFECT ON THE WORLD

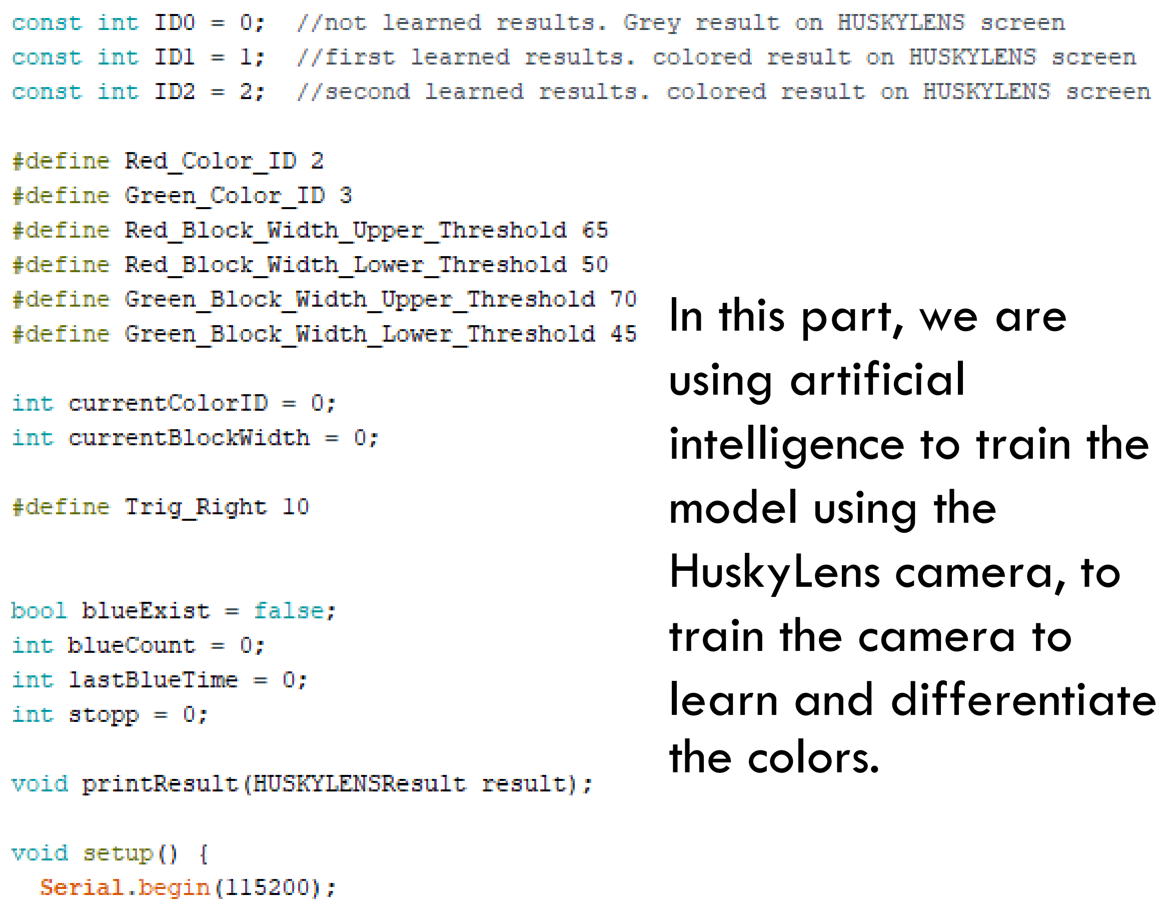
- Our project can improve the future of the way we do things. With the first small steps into this vision, we believe our project can simplify jobs and execute tasks without error.

THE ALGORITHM



```
#include "HUSKYLENS.h"


#include <Servo.h>

Servo myservo;  // create servo object to control a servo

#define IN2 6
#define IN1 7

#define ENA 5

#define Trig_Front 4
#define Echo_Front 3

#define Trig_Right 10
#define Echo_Right 9

#define Trig_Left 13
#define Echo_Left 12

long duration, distance, RightSensor, BackSensor, FrontSensor, LeftSensor;
```

In this part, we are working to install libraries for the code. First HuskyLens library and then Servo motors. There is a pin connection between motors, ultrasonic sensors, HuskyLens, motor driver, and the controller board Arduino Uno.

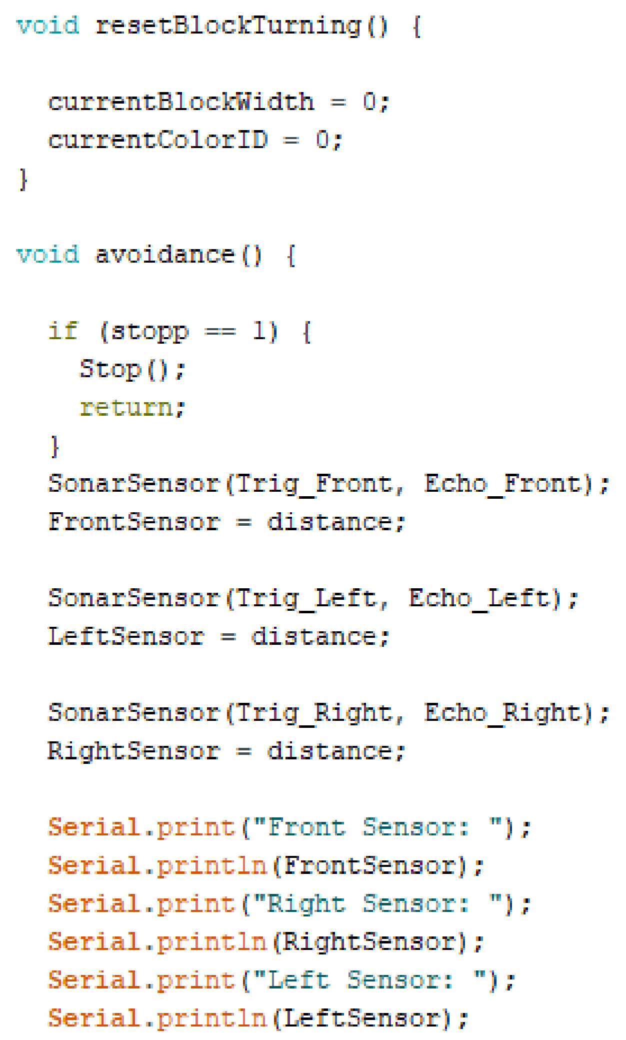


```
const int ID0 = 0;  //not learned results. Grey result on HUSKYLENS screen
const int ID1 = 1;  //first learned results. colored result on HUSKYLENS screen
const int ID2 = 2;  //second learned results. colored result on HUSKYLENS screen

#define Red_Color_ID 2
#define Green_Color_ID 3
#define Red_Block_Width_Upper_Threshold 65
#define Red_Block_Width_Lower_Threshold 50
#define Green_Block_Width_Upper_Threshold 70
#define Green_Block_Width_Lower_Threshold 45

int currentColorID = 0;
int currentBlockWidth = 0;

#define Trig_Right 10


bool blueExist = false;
int blueCount = 0;
int lastBlueTime = 0;
int stopp = 0;

void printResult(HUSKYLENSResult result);

void setup() {
  Serial.begin(115200);
```

In this part, we are using artificial intelligence to train the model using the HuskyLens camera, to train the camera to learn and differentiate the colors.

VOID FUNCTION

- The obstacle avoidance function is what we use to prevent the robot from making accidents.

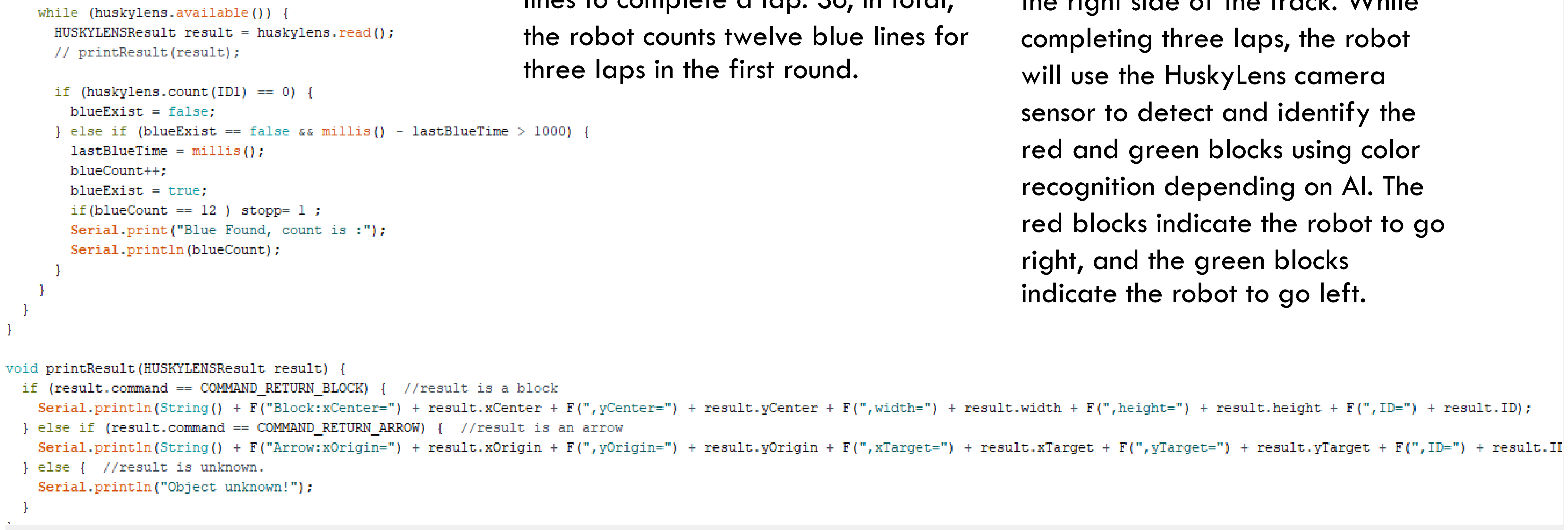


```
void resetBlockTurning() {

  currentBlockWidth = 0;
  currentColorID = 0;
}

void avoidance() {

  if (stopp == 1) {
    Stop();
    return;
  }
  SonarSensor(Trig_Front, Echo_Front);
  FrontSensor = distance;

  SonarSensor(Trig_Left, Echo_Left);
  LeftSensor = distance;

  SonarSensor(Trig_Right, Echo_Right);
  RightSensor = distance;

  Serial.print("Front Sensor: ");
  Serial.println(FrontSensor);
  Serial.print("Right Sensor: ");
  Serial.println(RightSensor);
  Serial.print("Left Sensor: ");
  Serial.println(LeftSensor);
```

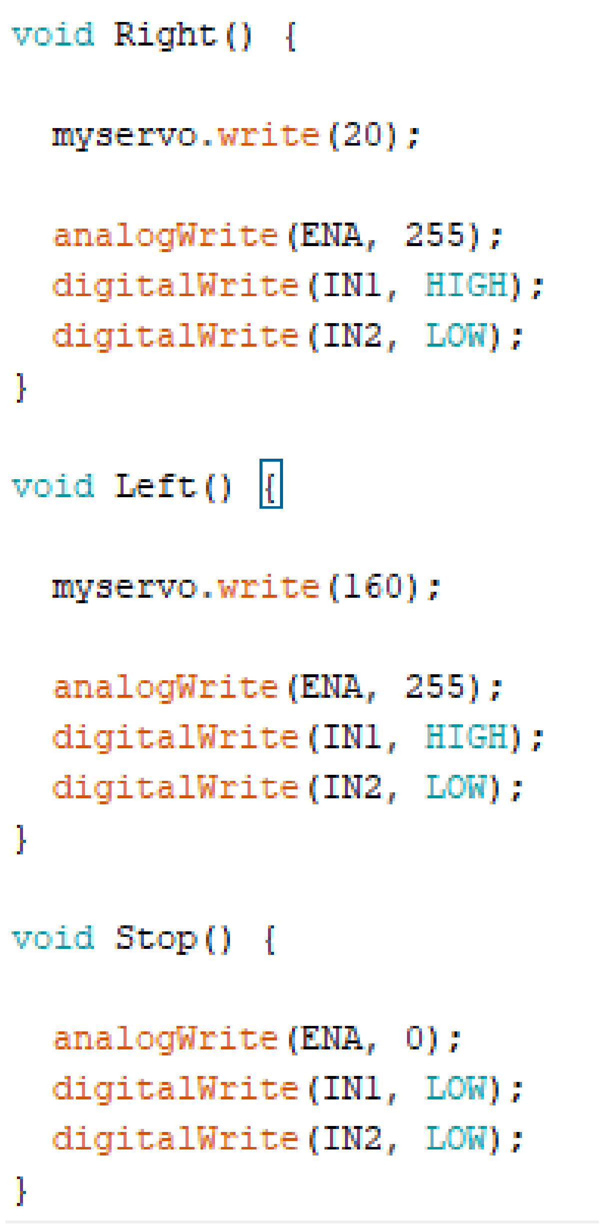
LAB FUNCTION

In the first lap, the Robot takes three laps around the track for each round. It follows the right wall, and for each round it counts four blue lines to complete a lap. So, in total, the robot counts twelve blue lines for three laps in the first round.

In the second round, the concept will be similar the first round. We will add red and green blocks to the track. The robot will start on the right side of the track. While completing three laps, the robot will use the HuskyLens camera sensor to detect and identify the red and green blocks using color recognition depending on AI. The red blocks indicate the robot to go right, and the green blocks indicate the robot to go left.



```
  while (huskylens.available()) {
    HUSKYLENSResult result = huskylens.read();
    // printResult(result);

    if (huskylens.count(ID1) == 0) {
      blueExist = false;
    } else if (blueExist == false && millis() - lastBlueTime > 1000) {
      lastBlueTime = millis();
      blueCount++;
      blueExist = true;
      if(blueCount == 12 ) stopp= 1 ;
      Serial.print("Blue Found, count is :");
      Serial.println(blueCount);
    }
  }
 }
}

void printResult(HUSKYLENSResult result) {
  if (result.command == COMMAND_RETURN_BLOCK) {  //result is a block
    Serial.println(String() + F("Block:xCenter=") + result.xCenter + F(",yCenter=") + result.yCenter + F(",width=") + result.width + F(",height=") + result.height + F(",ID=") + result.ID);
  } else if (result.command == COMMAND_RETURN_ARROW) {  //result is an arrow
    Serial.println(String() + F("Arrow:xOrigin=") + result.xOrigin + F(",yOrigin=") + result.yOrigin + F(",xTarget=") + result.xTarget + F(",yTarget=") + result.yTarget + F(",ID=") + result.I
  } else {  //result is unknown.
    Serial.println("Object unknown!");
  }
```



```
void Right() {

  myservo.write(20);

  analogWrite(ENA, 255);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
}

void Left() {

  myservo.write(160);

  analogWrite(ENA, 255);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
}

void Stop() {

  analogWrite(ENA, 0);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
}
```

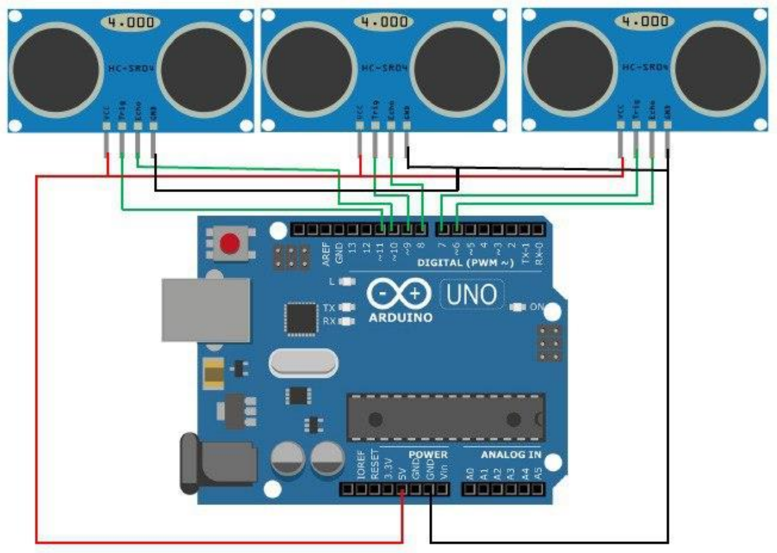
SERVO MOTOR CODE

- This code lets the servo motor turn right or left depending on the distance. We need this so that the robot can dodge the red and green blocks and continue its path.

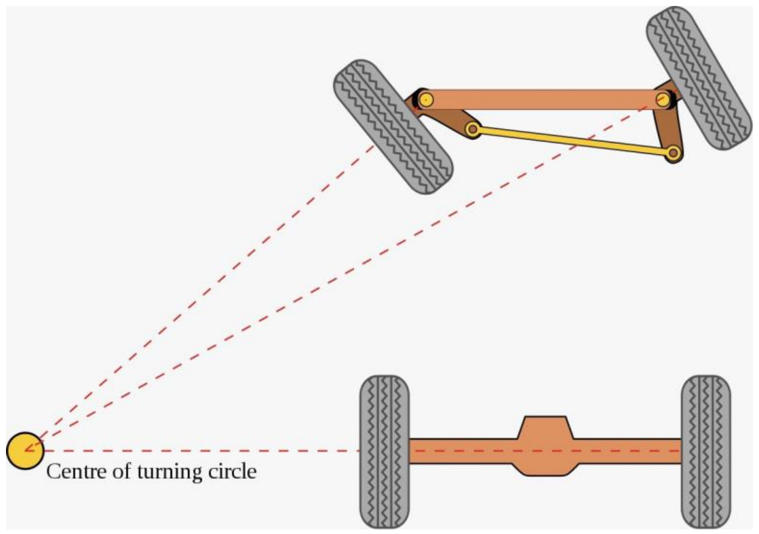
WIRING DIAGRAM

- Wiring for each components.

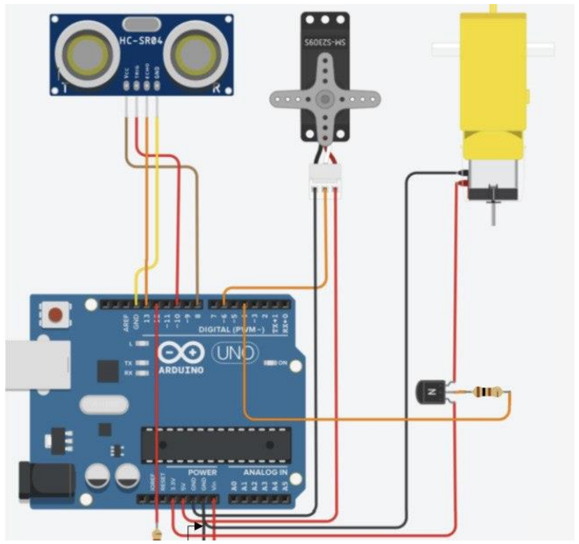
Wiring diagram for obstacle avoiding.



Wiring diagram for driving and steering control.

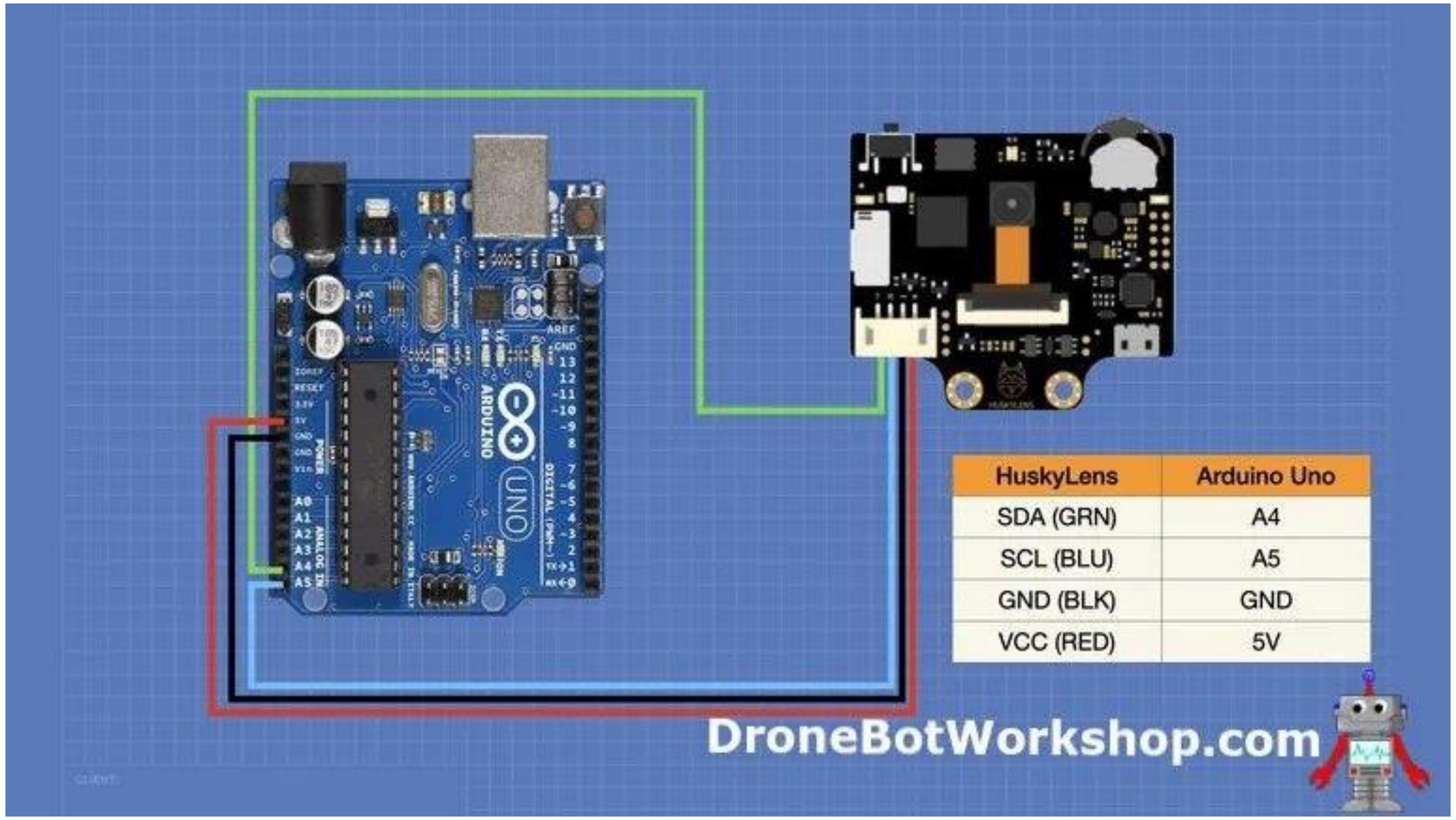


Mechanical parts and steering mechanism.

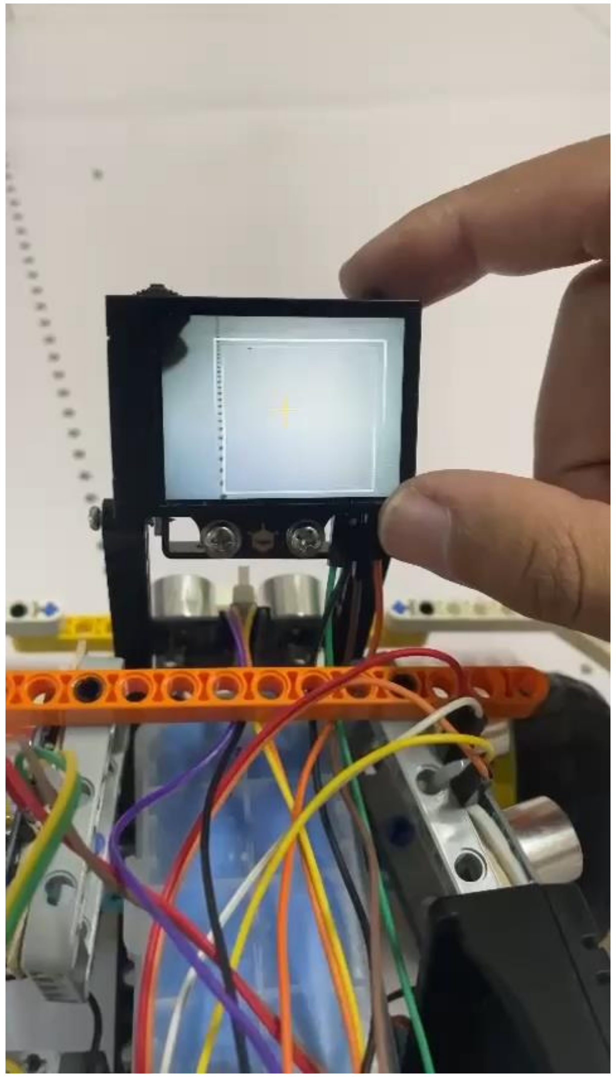


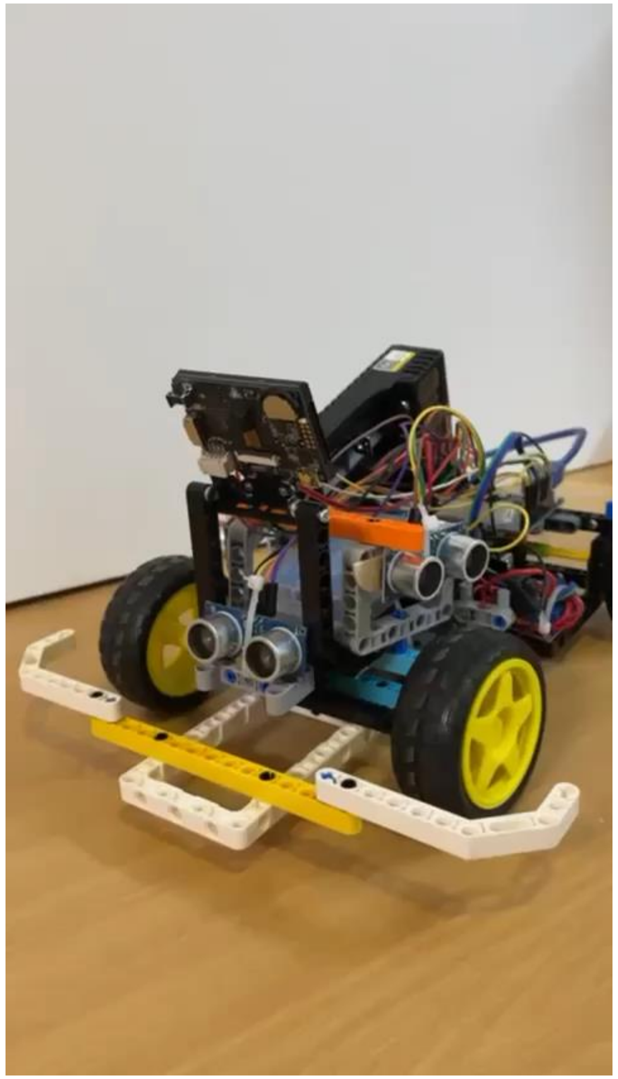
HUSKYLENS WIRING DIAGRAM

- HuskyLens Arduino Hookup – I2C Mode
- You can also use the HuskyLens in I2C mode. To do this, you'll need to change your wiring slightly.



HuskyLens	Arduino Uno
SDA (GRN)	A4
SCL (BLU)	A5
GND (BLK)	GND
VCC (RED)	5V





SHORT VIDEOS

REFERENCE

DRONEBOTWORKSHOP.COM – HUSKYLENS VISION SENSOR

SCIENCEDIRECT.COM – ISSUE IN REAL TIME

DFROBOT.COM – MANUFACTURE FOR HUSKYLENS