PennState

CMPSC 442: Artificial Intelligence (Fall 2022)

CMPSC 442: Artificial Intelligence

Hansi Seitaj, Parth Shah, Almaz Akhunbaev, Bohan Dong

Final Project Report
November 10, 2022

Project Name: Factory Robot

Dr. Vinayak Elangovan

Submitted On:
12/12/22

TABLE OF CONTENTS

1. INTRODUCTION

In this project, we are programming a 4WD and Arm robot that will accomplish an objective. The objective we would like the robot to do is to retrieve an object and deliver it to the destination. When the robot goes to the location to get an object, it should avoid obstacles and follow a line. To accomplish this we intend to utilize several sensors. We are going to use a tracking sensor to follow the line and an ultrasonic module to avoid obstacles. Finally, for the retrieval, we would like to use a robotic arm to put on top of the robot in order for it to carry an object and deliver it to the desired destination.

Peas for Factory Robot:

Agent: Robot 'Pet'
P: successful retrieval of an object.
E: floors of various types, multitudes of ground types.
A: Wheels, servo motors.
S: Ultrasonic module, tracking sensor, freenove control board, freenove 4WD extension board.

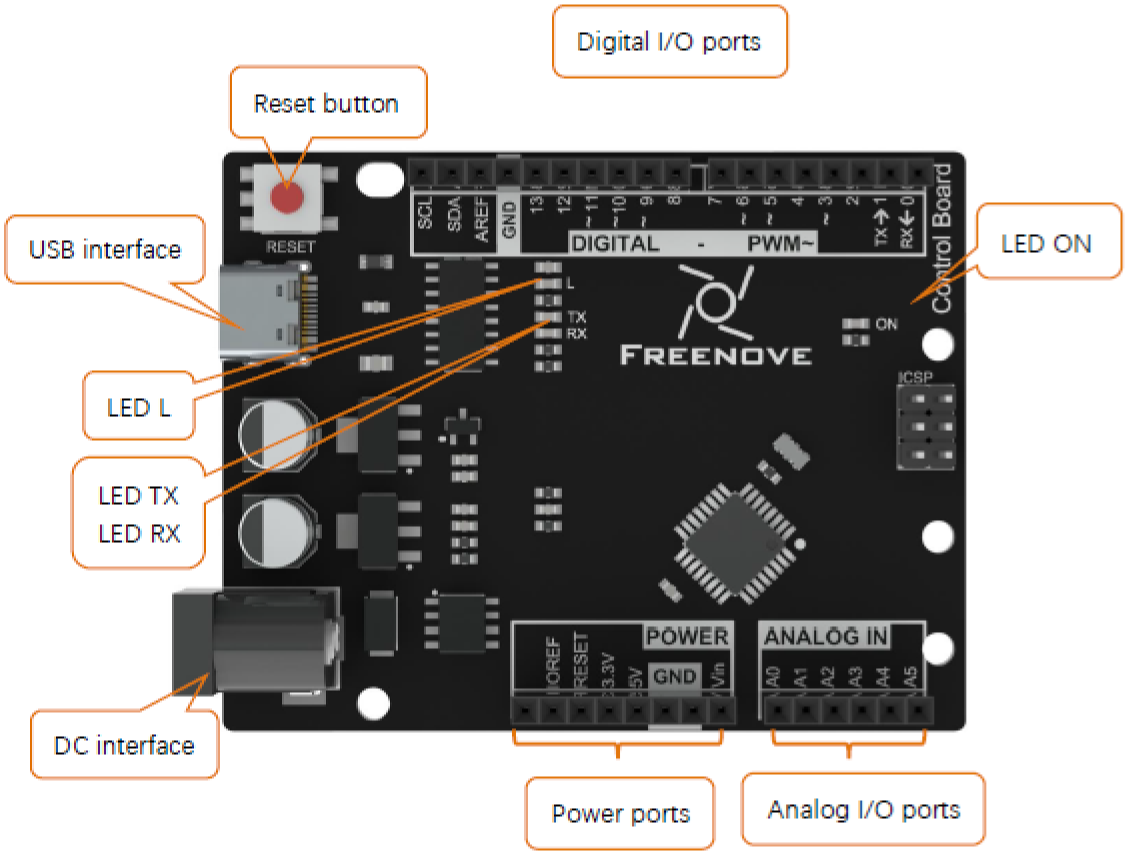
Peas:
Agent: Mechanical arm
P: successful retrieval and drop off of an object.
E: floors of various types, multitudes of ground types.
A: Wheels, servo motors, robot clamps.
S: Person's eyes.

2. BACKGROUND

The instructions that we found based on the research for the arm connections are the map below. In addition, the map shows the way to connect the Arm robot to two relays and then to an Arduino board.

Control Board:

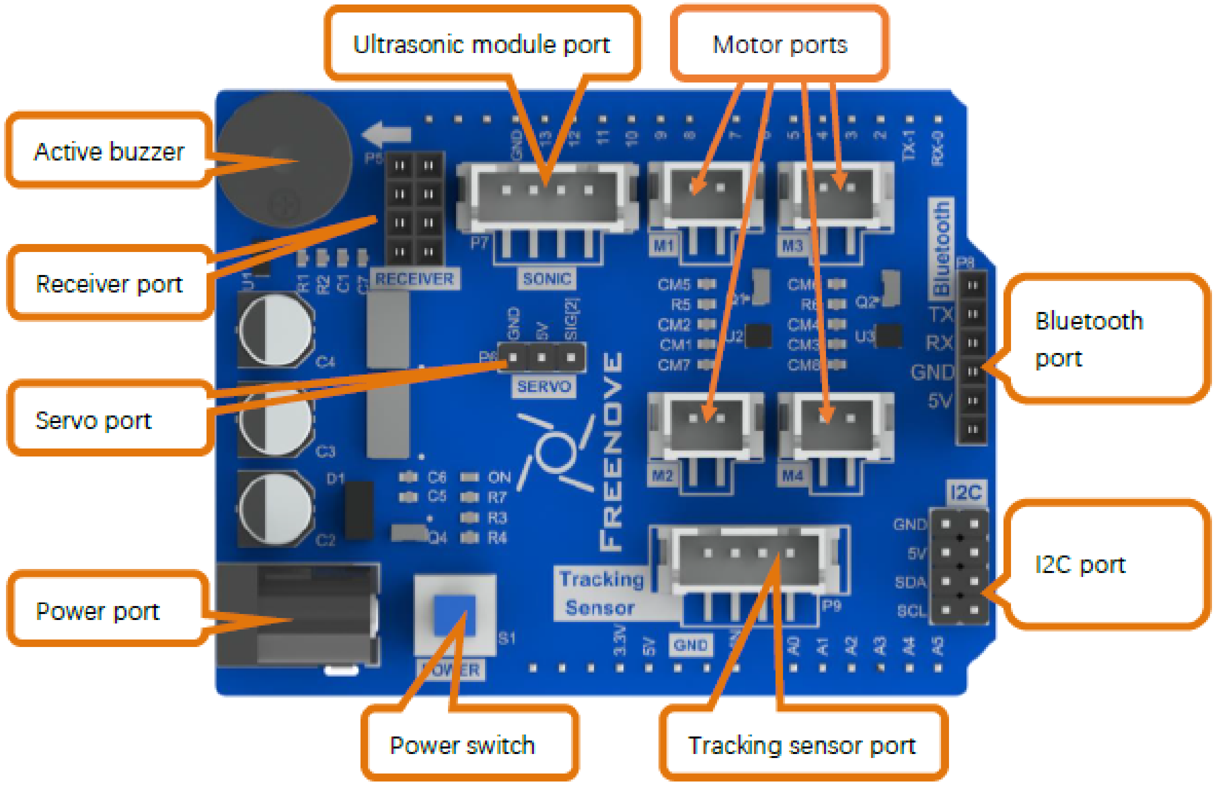
Diagram of Freenove Control board is shown below:



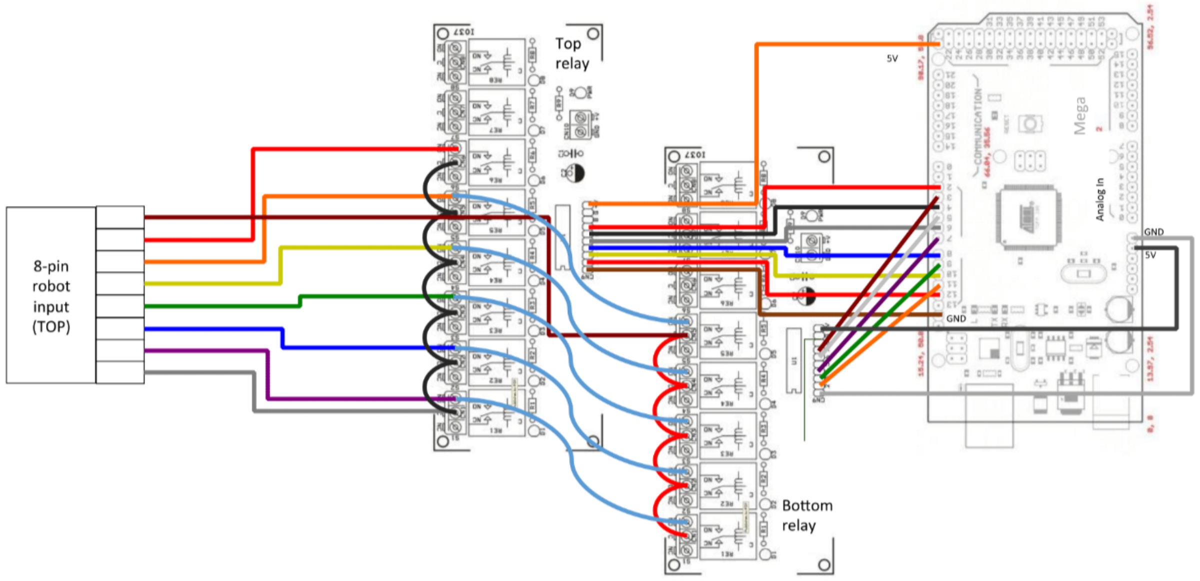
- Digital I/O ports is used to connect to other components or modules, to receive an input signal, or to
- send a control signal. Usually, we name it by adding a "D" in front of the number, such as D13.
- Only digital I/O ports with “~” can output PWM, Pin 3, 5, 6, 9, 10, 11.
- USB interface is used to provide power, upload code or communicate with PC.
- LED L is connected to digital I/O port 13 (D13).
- LED TX, RX is used to indicate the state of the serial communication.
- DC interface is connected with DC power to provide power for the board.
- Power ports can provide power for electronic components and modules.
- Analog I/O ports can be used to measure analog signals.
- LED ON is used to indicate the power state.

Extension Board:

Freenove 4WD Extension board is below:



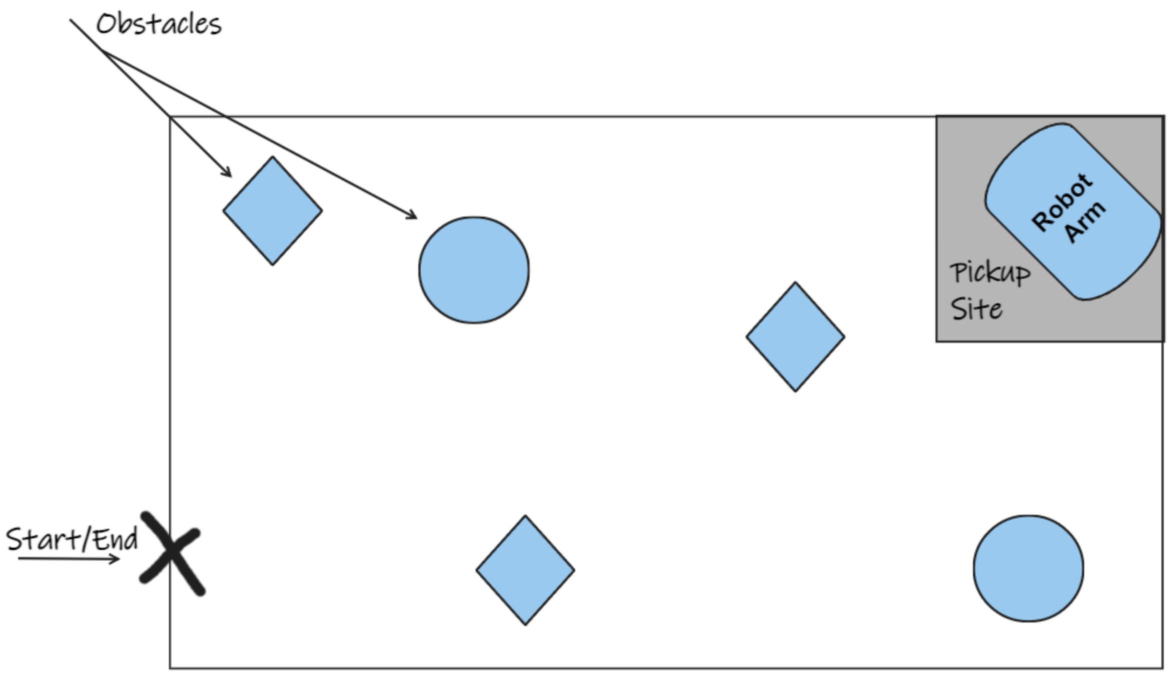
Receiver port can connect to RF module or IR module.



3. DESIGN & IMPLEMENTATION

For this project, our goal is to have a factory robot "car" that will perform fetch tasks based on a designated area. The robot will traverse an obstacle course, reach a pickup spot, wait for the object to be loaded onto the robot, and then carry the object to a goal area.

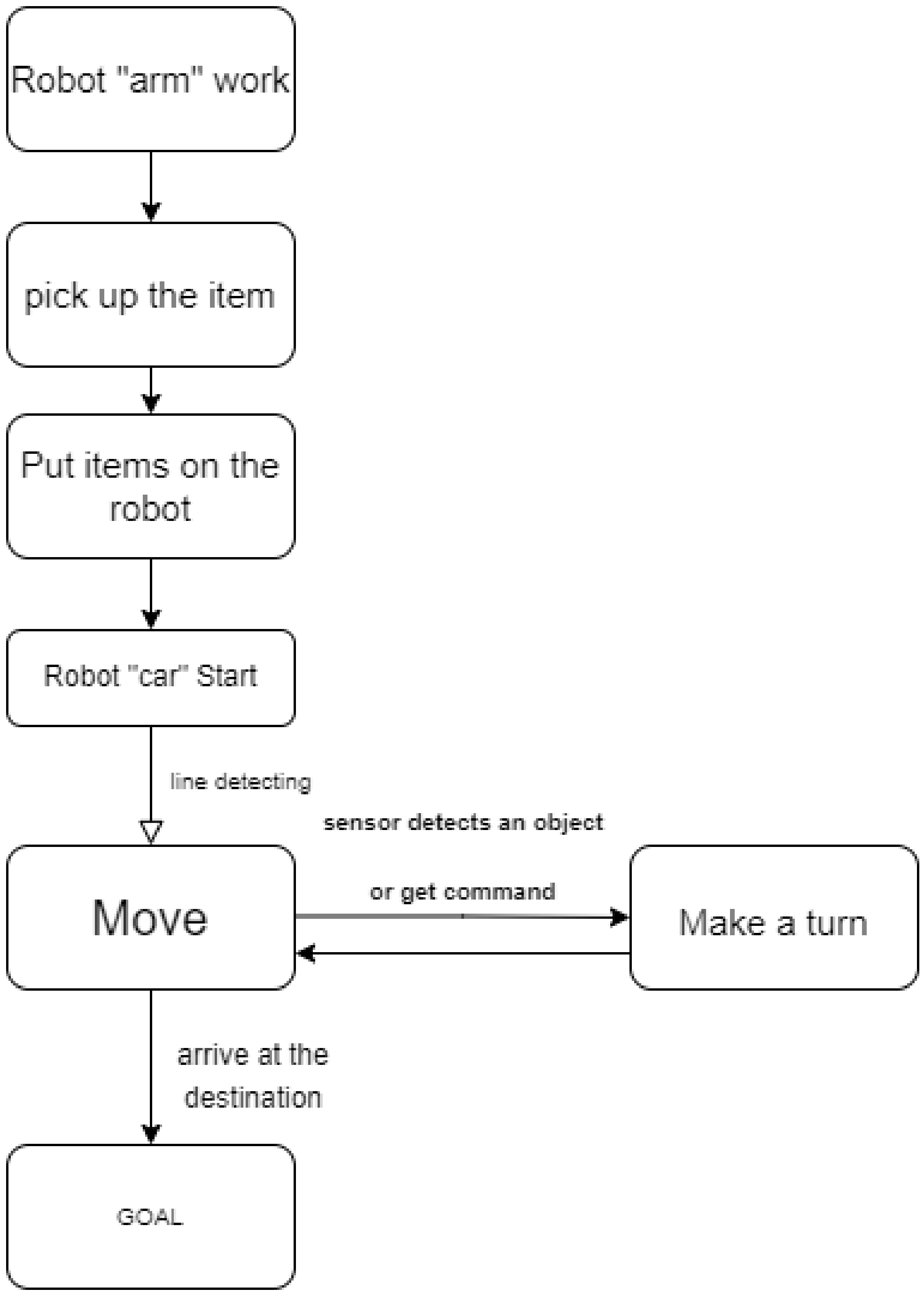
Below we have an example of the layout of our final solution:



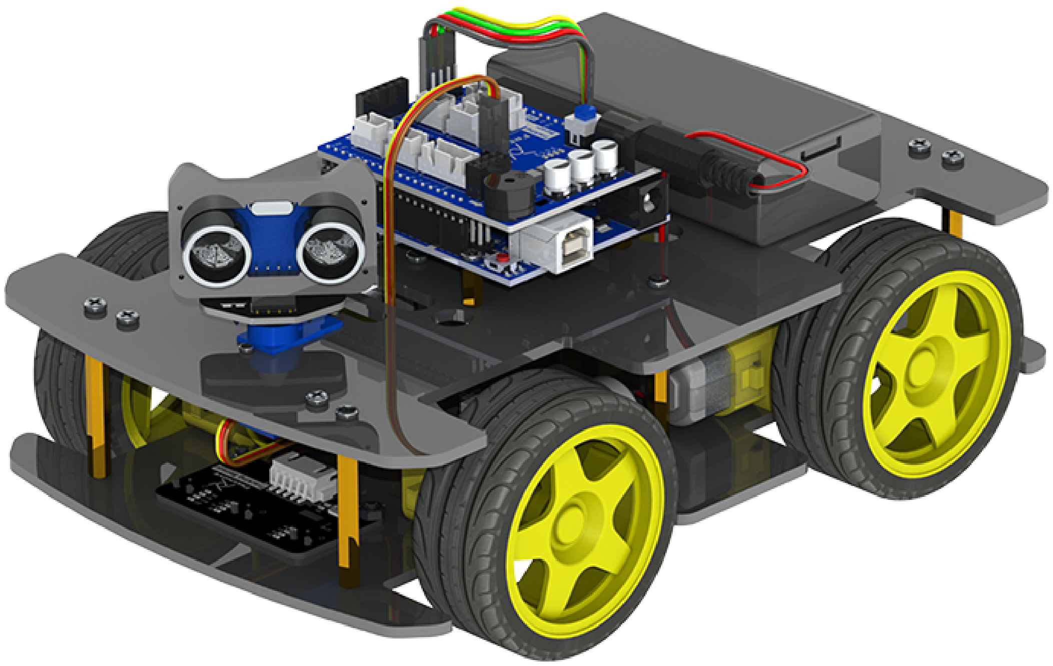
Objectives:

- To utilize the tracking sensor to perform line detecting to get to the destination.
- To train and test the robot to use the information received to perform the object avoiding
- To program the arm robot to utilize several steps to pick up an object and drop it off.

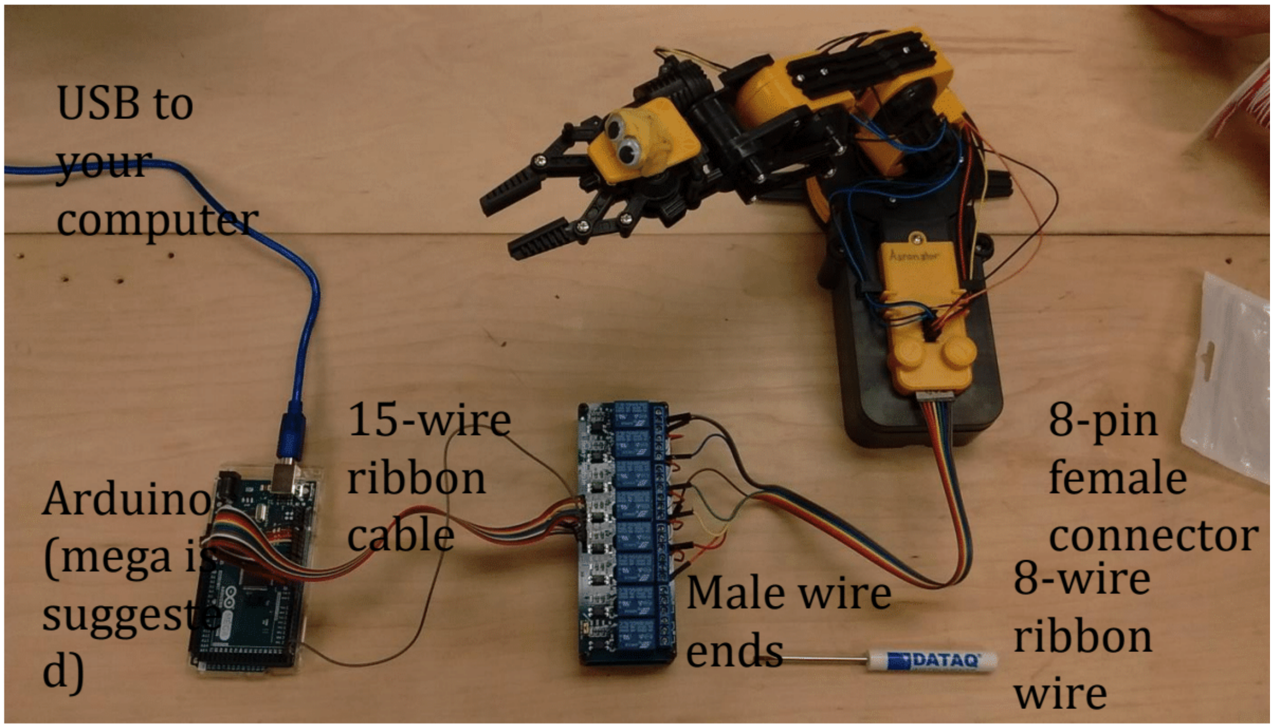
Class diagram:



The 4WD robot:



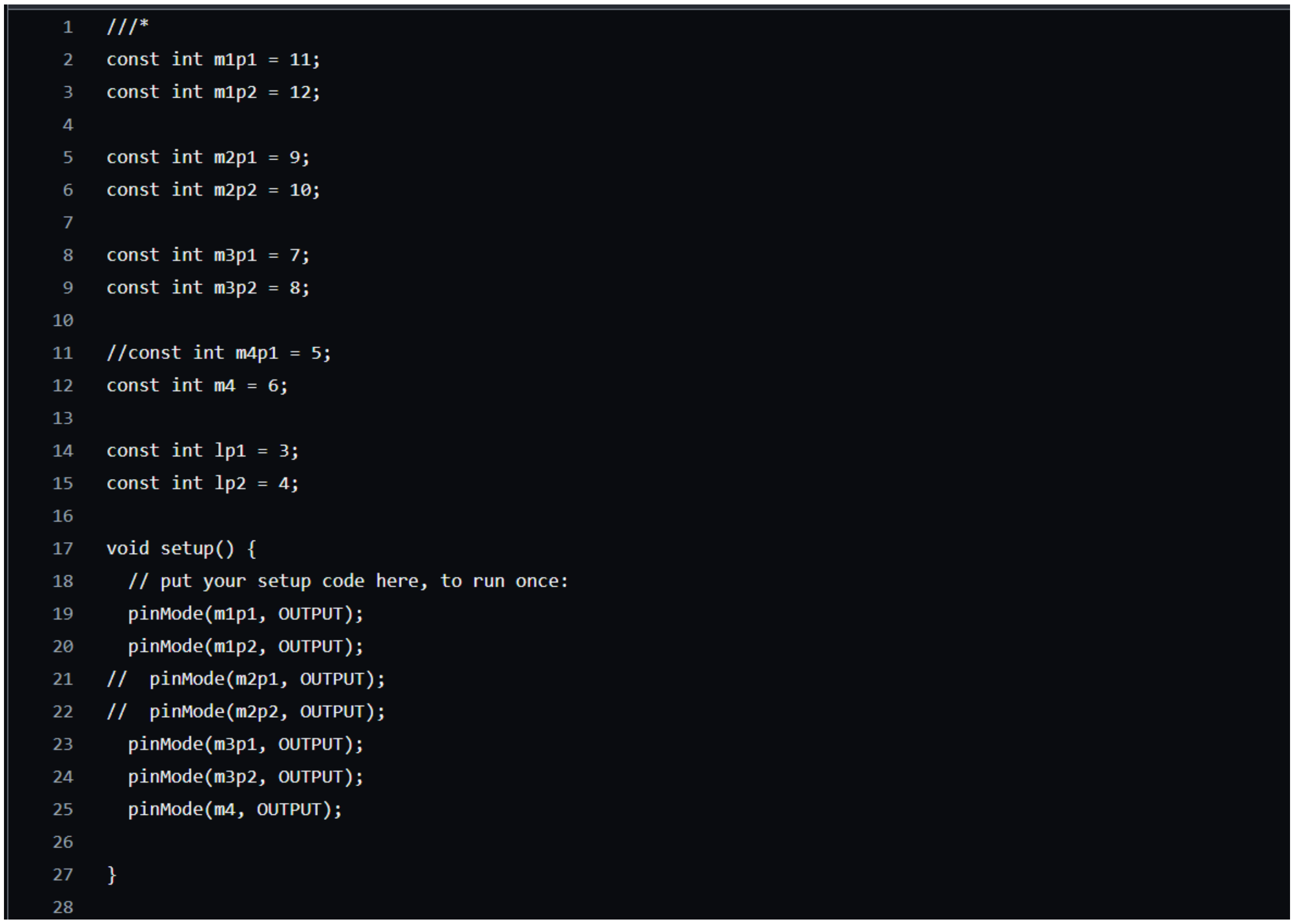
The mechanical Arm robot:



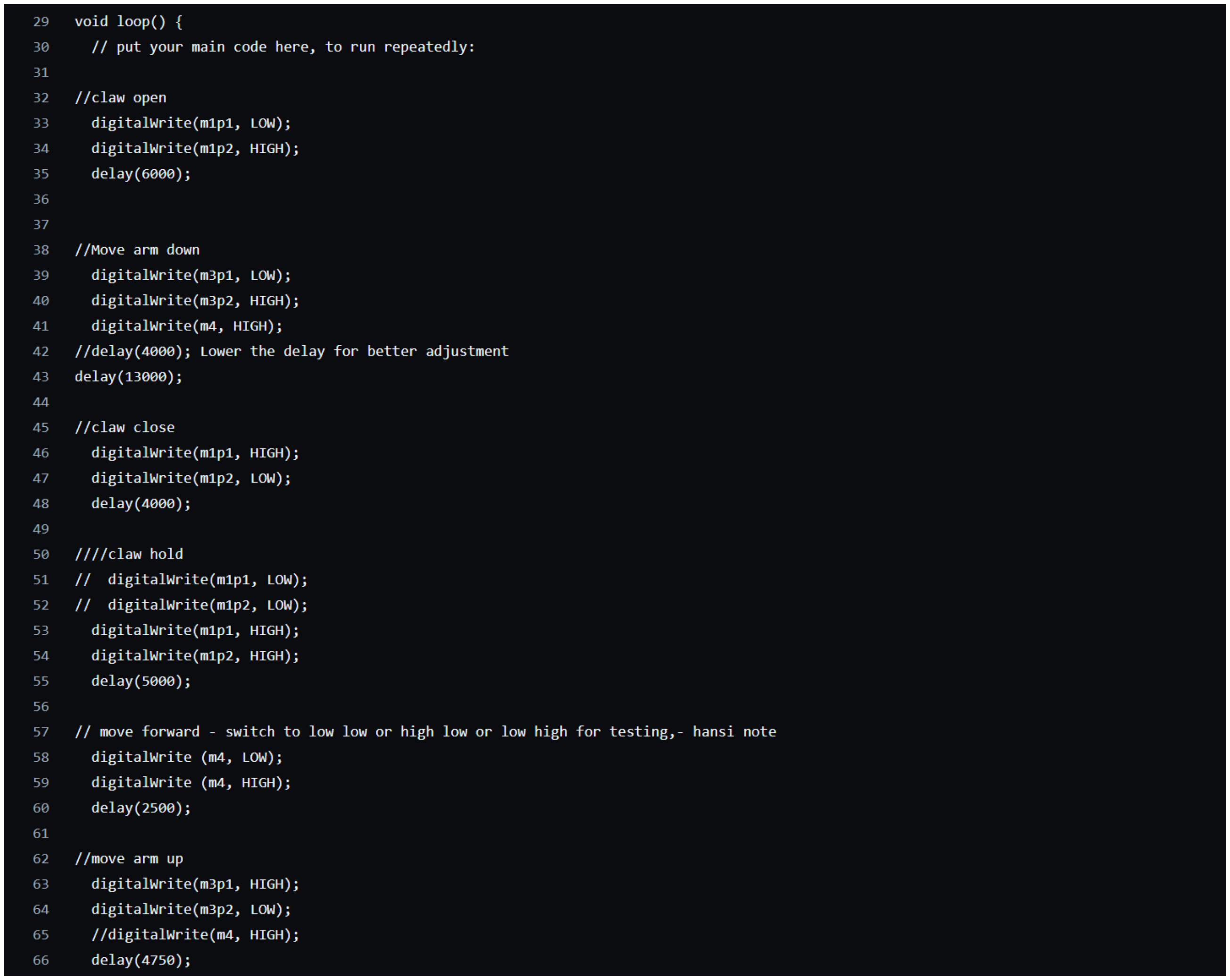
4. RESULTS / SAMPLE OUTPUTS

Sample Outputs:

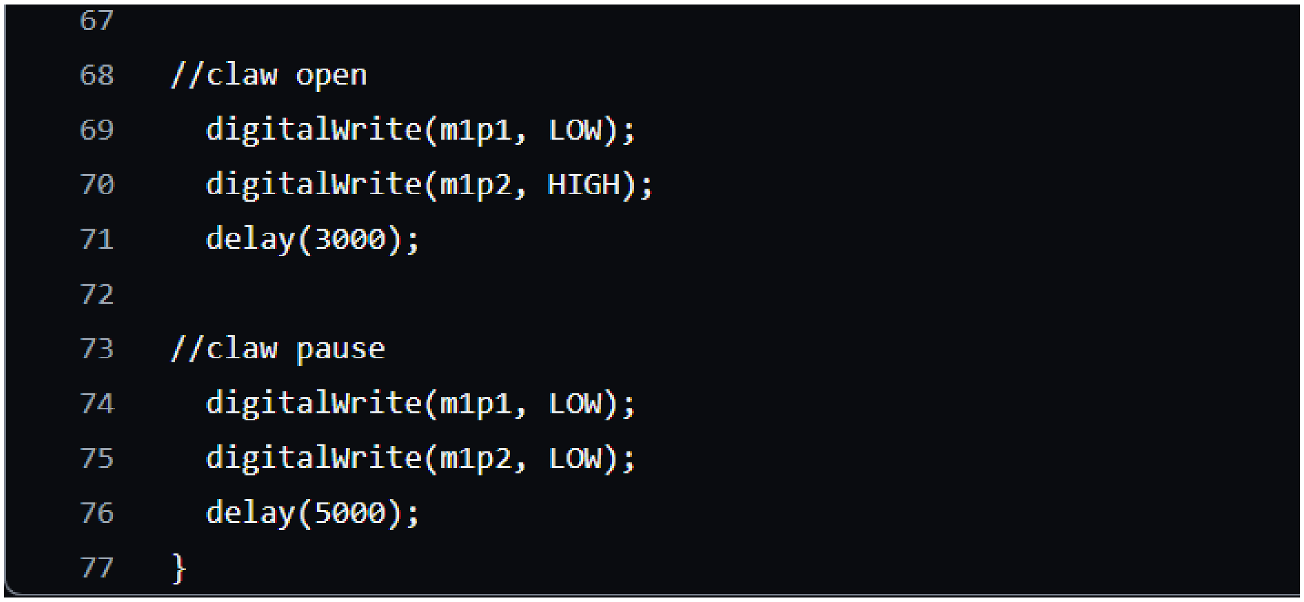
- **Robotic arm code:**



```
///*
const int m1p1 = 11;
const int m1p2 = 12;

const int m2p1 = 9;
const int m2p2 = 10;

const int m3p1 = 7;
const int m3p2 = 8;

//const int m4p1 = 5;
const int m4 = 6;

const int lp1 = 3;
const int lp2 = 4;

void setup() {
  // put your setup code here, to run once:
  pinMode(m1p1, OUTPUT);
  pinMode(m1p2, OUTPUT);
//  pinMode(m2p1, OUTPUT);
//  pinMode(m2p2, OUTPUT);
  pinMode(m3p1, OUTPUT);
  pinMode(m3p2, OUTPUT);
  pinMode(m4, OUTPUT);

}

```



```
void loop() {
  // put your main code here, to run repeatedly:

//claw open
  digitalWrite(m1p1, LOW);
  digitalWrite(m1p2, HIGH);
  delay(6000);


//Move arm down
  digitalWrite(m3p1, LOW);
  digitalWrite(m3p2, HIGH);
  digitalWrite(m4, HIGH);
//delay(4000); Lower the delay for better adjustment
delay(13000);

//claw close
  digitalWrite(m1p1, HIGH);
  digitalWrite(m1p2, LOW);
  delay(4000);

////claw hold
//  digitalWrite(m1p1, LOW);
//  digitalWrite(m1p2, LOW);
  digitalWrite(m1p1, HIGH);
  digitalWrite(m1p2, HIGH);
  delay(5000);

// move forward - switch to low low or high low or low high for testing,- hansi note
  digitalWrite (m4, LOW);
  digitalWrite (m4, HIGH);
  delay(2500);

//move arm up
  digitalWrite(m3p1, HIGH);
  digitalWrite(m3p2, LOW);
  //digitalWrite(m4, HIGH);
  delay(4750);
```



```

//claw open
  digitalWrite(m1p1, LOW);
  digitalWrite(m1p2, HIGH);
  delay(3000);

//claw pause
  digitalWrite(m1p1, LOW);
  digitalWrite(m1p2, LOW);
  delay(5000);
}
```

The program was created in Arduino IDE and we had to frequently verify and upload the code to the board so the robot can follow the instructions based on its programming. In addition, the code for the arm is in C++, C# format. Furthermore, we are setting the pins that are connected to the

Arduino board as global variables. There are two pins per motor in the arm robot, and one pin for the LED light.

- The setup() method:

The setup() function is called when a sketch starts. Use it to initialize variables, pin modes, start using libraries, etc. The setup() function will only run once, after each powerup or reset of the Arduino board.

- The digitalWrite() method:

Write a HIGH or a LOW value to a digital pin.

If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.

If the pin is configured as an INPUT, digitalWrite() will enable (HIGH) or disable (LOW) the internal pullup on the input pin. It is recommended to set the pinMode() to INPUT_PULLUP to enable the internal pull-up resistor. See the Digital Pins tutorial for more information.

If you do not set the pinMode() to OUTPUT and connect an LED to a pin, when calling digitalWrite(HIGH), the LED may appear dim. Without explicitly setting pinMode(), digitalWrite() will have enabled the internal pull-up resistor, which acts like a large current-limiting resistor.

- The delay() method:

While it is easy to create a blinking LED with the delay() function and many sketches use short delays for such tasks as switch debouncing, the use of delay() in a sketch has significant drawbacks. No other reading of sensors, mathematical calculations, or pin manipulation can go on during the delay function, so in effect, it brings most other activity to a halt. For alternative approaches to controlling timing see the Blink Without Delay sketch, which loops, polling the millis() function until enough time has elapsed. More knowledgeable programmers usually avoid the use of delay() for the timing of events longer than 10's milliseconds unless the Arduino sketch is very simple.

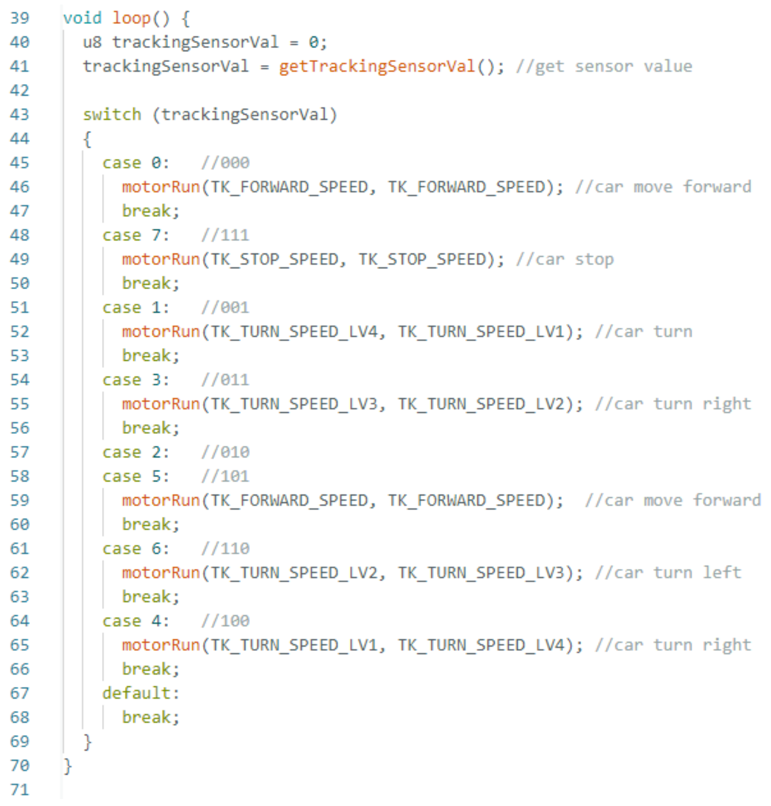
Certain things do go on while the delay() function is controlling the Atmega chip, however, because the delay function does not disable interrupts. Serial communication that appears at the RX pin is recorded, PWM (analogWrite) values and pin states are maintained, and interrupts will work as they should.

- The loop() method

The method loop() is the method that gives the instructions to every motor based on the preferred movement and keeps running it in a loop repeatedly after each code is executed. Part of this method is to calibrate each of the motor movements correctly by the delay() method.
The delay method accepts the amount in milliseconds and will perform the movement of each motor based on the delayed time given.

Car robot code:

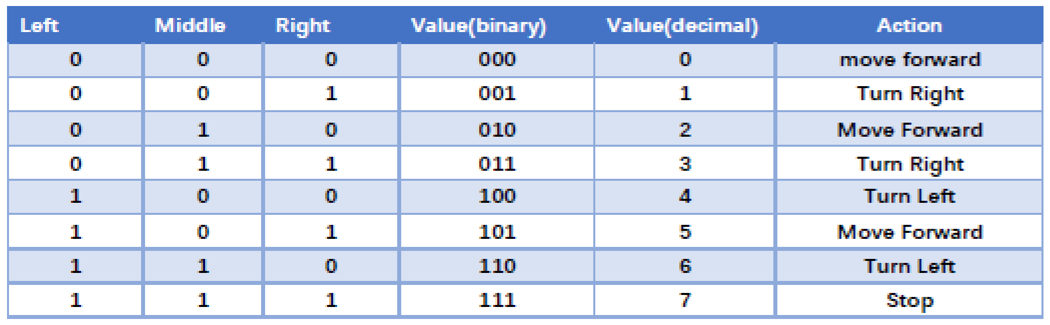
Line Tracking:



```
void loop() {
  u8 trackingSensorVal = 0;
  trackingSensorVal = getTrackingSensorVal(); //get sensor value

  switch (trackingSensorVal)
  {
    case 0:   //000
      motorRun(TK_FORWARD_SPEED, TK_FORWARD_SPEED); //car move forward
      break;
    case 7:   //111
      motorRun(TK_STOP_SPEED, TK_STOP_SPEED); //car stop
      break;
    case 1:   //001
      motorRun(TK_TURN_SPEED_LV4, TK_TURN_SPEED_LV1); //car turn
      break;
    case 3:   //011
      motorRun(TK_TURN_SPEED_LV3, TK_TURN_SPEED_LV2); //car turn right
      break;
    case 2:   //010
    case 5:   //101
      motorRun(TK_FORWARD_SPEED, TK_FORWARD_SPEED);  //car move forward
      break;
    case 6:   //110
      motorRun(TK_TURN_SPEED_LV2, TK_TURN_SPEED_LV3); //car turn left
      break;
    case 4:   //100
      motorRun(TK_TURN_SPEED_LV1, TK_TURN_SPEED_LV4); //car turn right
      break;
    default:
      break;
  }
}
```

The car will make different actions according to the value transmitted by the line-tracking sensor.

When

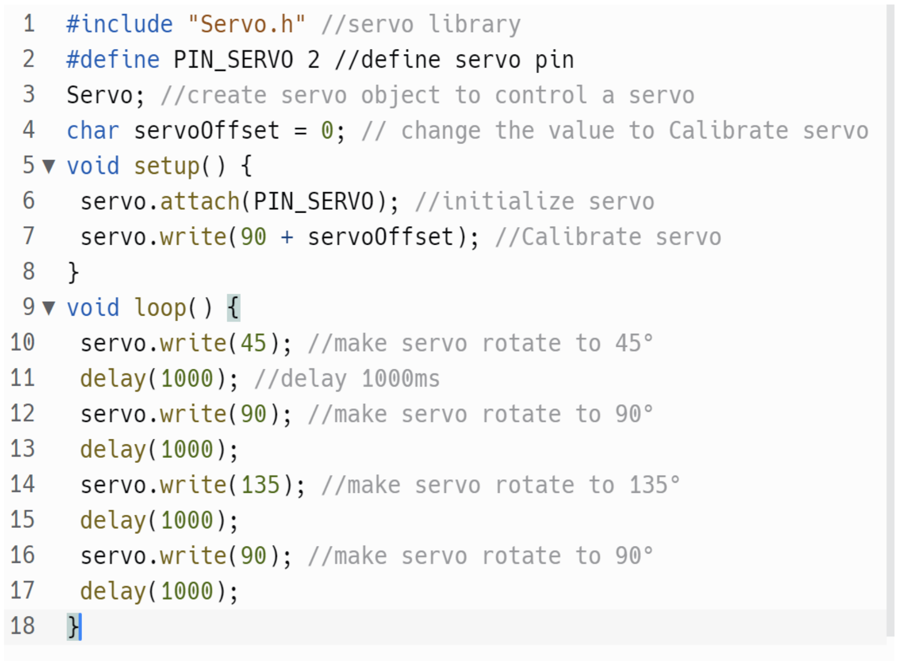


Left	Middle	Right	Value(binary)	Value(decimal)	Action
0	0	0	000	0	move forward
0	0	1	001	1	Turn Right
0	1	0	010	2	Move Forward
0	1	1	011	3	Turn Right
1	0	0	100	4	Turn Left
1	0	1	101	5	Move Forward
1	1	0	110	6	Turn Left
1	1	1	111	7	Stop

- The loop() method

The trackingSensorVal will be a number from 0 to 7. The robot's wheels will rotate based on the the number. Line sensors detect the presence of a black line by emitting infrared (IR) light and detecting the light levels that return to the sensor. They do this using two components: an emitter and a light sensor (receiver). A black line will not reflect as much light, so the output will be set to high (1) when a black surface is underneath. If the sensor does not receive enough light to surpass the threshold value, the digital output will be high (1). If enough light is received and the threshold value is surpassed, the pin will be set to low (0). For example, when the number is 0, that means we have a binary value of 000.That means there is no black line detected and the car will keep moving forward. When the number is 1, that means we have a binary value of 001. It means the right most light of the line tralking sensor is on and the car should turn right until the middle light of the sensor is on. It means that the car is in the center of the line.

Obstacle Avoidance:



```
#include "Servo.h" //servo library
#define PIN_SERVO 2 //define servo pin
Servo; //create servo object to control a servo
char servoOffset = 0; // change the value to Calibrate servo
void setup() {
 servo.attach(PIN_SERVO); //initialize servo
 servo.write(90 + servoOffset); //Calibrate servo
}
void loop() {
 servo.write(45); //make servo rotate to 45°
 delay(1000); //delay 1000ms
 servo.write(90); //make servo rotate to 90°
 delay(1000);
 servo.write(135); //make servo rotate to 135°
 delay(1000);
 servo.write(90); //make servo rotate to 90°
 delay(1000);
}
```

- Servo Method

Servo is a compact package which consists of a DC Motor, a set of reduction gears to provide torque, a sensor and control circuit board. Servos can output higher torque than a simple DC Motor alone and they are widely used to control motion in model cars, model airplanes, robots, etc. We use a 50Hz PWM signal with a duty cycle in a certain range to drive the servo. We will use a 50Hz PWM signal with a duty cycle in a certain range to drive the servo.

High level time	Servo angle
0.5ms	0 degree
1ms	45 degree
1.5ms	90 degree
2ms	135 degree
2.5ms	180 degree

- Ultrasonic Ranging Module

The Ultrasonic Ranging Module uses the principle that ultrasonic waves will reflect when they encounter any obstacles. This is possible by counting the time interval between when the ultrasonic wave is transmitted to when the ultrasonic wave reflects back after encountering an obstacle.
The ultrasonic ranging module integrates an ultrasonic transmitter and an ultrasonic receiver. The function of a transmitter is to convert electrical signals into high-frequency sound waves. The function of a receiver is the opposite.

5. CONCLUSION

Contribution plan:

- We assembled the 4WD robot.
- We connected the Arm robot to two relays and an Arduino board.

- We connected the 4WD and Arm robot to our computer.
- We utilized code to make the 4WD and Arm robot move and operate a few steps.
- We have been gathering every week twice after class and elaborating on the ideas regarding the project.
- Mainly, Almaz and Bohan had more focused work on the 4WD.
- Mainly, Hansi and Parth had more focused work on the Arm robot, however, we all have been in touch and worked on both robots periodically.

Meetings here:
We meet every week after class to

- Design the project plan
- To add and improve our robots
- We spent about 1 to 2 hours every week on this project

We specifically worked on these tasks:

- Hansi worked with the arm robot and programmed it.
- Parth worked on the arm robot and programmed it.
- Bohan worked on testing the 4WD robot.
- Almaz worked on the 4WD robot assembly and programmed it.
- Hansi and Parth are working on making the arm robot move accordingly to the plan.
- Almaz and Bohan are working on completing the destination program and implementation for the 4WD robot.
- However, we mostly worked during our meetings. So we all worked on both robots.

- What have you learned from this project?

In this project, we gained experience in assembling a 4WD and Arm robot and implementing Arduino boards. Since we were able to connect the robot to our devices, we learned how to train and test the two robots. In addition, we gained experience with robot sensors implementation and how to implement Artificial intelligence in two robots to move autonomously and follow our plan. Part of this project's learning phase was to make the agent read the environment and learn from it by implementing line detecting and obstacle avoidance. Overall, we learned how to research and program from scratch the Arm Robot. Finally, worked in a team to create an agile framework project.

- In what ways can you expand this project?

This project had some really intrusive and tedious steps toward accomplishing it for ex. no programming instructions on the Arm robot. Since now we have implemented all these instructions and projects I believe that other groups in the future can use our work as a reference to build something in combination with other robots.

6. REFERENCES

The link for the 4wd instructions
https://www.youtube.com/watch?v=m8Pq-4ecAwQ
Robot arm reference:
https://www.dropbox.com/sh/m6rnbvdpgu4nkuj/AACgWZc4fSYMhyYOOW3_2FRja/Using%20Relays?dl=0&preview=RobotDemo2.docx&subfolder_nav_tracking=1
https://www.arduino.cc/reference/en/