Report

Group 3 (Chaware Abadi, Harwin Noori, Telman Martirosian, Walid Iqbal)

For this assignment we have chosen to work on a robot following a line and adapt to sudden shifts and/or disturbances that may occur. The robot will follow a black line through a circuit and will follow it. Until it is turned off, or by some reasons it cannot scan the black line ahead. Meaning it just takes a left turn or right turn, assuming there is no black line on the sides. The robot contains 6 IR sensors, two for each of the sides and one for stopping. The system runs on a 9V battery, assuming that all of the components require that total energy. We have a 5v regulator so that we get a clean DC signal, sending power to the gearmotors via a H-bridge motor driver. The H-bridge's main function is to switch the polarity of the voltage to a load (the gearmotors) making them spin. Some stuff that might happen is that the 9V battery is not sufficient but then we just change to a new 9v battery assuming the old one is faulty or just a higher voltage battery (15V). the code for all of the functions is compiled in the Arduino down under will you see an equipment list and screenshots and some links to the circuit

Equipment list

5 IR sensors

1 breadboard small

1 Arduino Uno R3/Arduino Nano

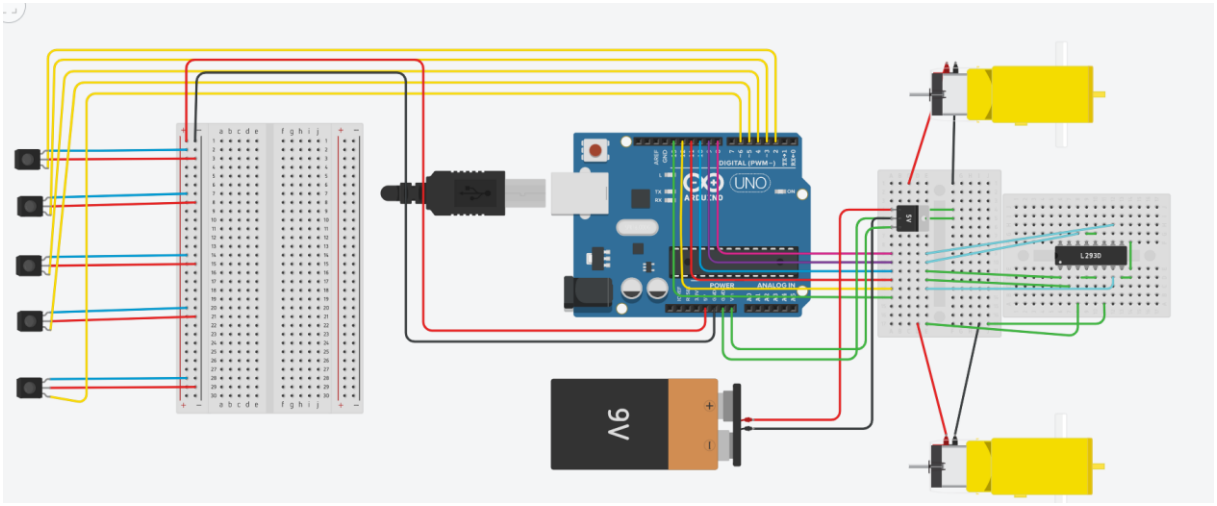
1 9V Battery

2 hobby gearmotors/torque

2 breadboards mini

1 H-Bridge motor driver

5V regulator




Link to the circuit

https://www.tinkercad.com/things/1uPOO2VZxfp-line-follower-robot/editel?sharecode=0K_YQXNjddzff0NwX7d6HNz9jiyjytTGhjOMV1liqO8

```
void setup()
{
  pinMode(13, OUTPUT); // Venstre motor +
  pinMode(12, OUTPUT); // Venstre motor -
  pinMode(11, OUTPUT); // aktivering 1&2
  pinMode(10, OUTPUT); // aktivering 3&4
  pinMode (9, OUTPUT); // Høyre Motor B -
  pinMode (8, OUTPUT); // Høyre Motor B +
  pinMode(6, INPUT);   // V1
  pinMode(5, INPUT);   // V2
  pinMode(4, INPUT);   // S
  pinMode(3, INPUT);   // H1
  pinMode(2, INPUT);   // H2
}

void loop()
{
  if((digitalRead(6 )== 1) && (digitalRead(5) == 1) && (digitalRead(4) == 0) && (digitalRead(3)== 1) && (digitalRead(3)
== 1));
  {beveger_fremover();} // calling functions
  if((digitalRead(6 )== 1) && (digitalRead(5) == 1) && (digitalRead(4) == 1) && (digitalRead(3)== 1) && (digitalRead(3)
== 1));
  {beveger_bakover();}
  if((digitalRead(6 )== 1) && (digitalRead(5) == 0) && (digitalRead(4) == 0) && (digitalRead(3)== 1) && (digitalRead(3)
== 1));
  {venstre_a();}
  if((digitalRead(6 )== 0) && (digitalRead(5) == 0) && (digitalRead(4) == 0) && (digitalRead(3)== 1) && (digitalRead(3)
== 1));
  {venstre_b();}
  if((digitalRead(6 )== 1) && (digitalRead(5) == 0) && (digitalRead(4) == 1) && (digitalRead(3)== 1) && (digitalRead(3)
== 1));
  {venstre();}
  if((digitalRead(6 )== 1) && (digitalRead(5) == 1) && (digitalRead(4) == 0) && (digitalRead(3)== 0) && (digitalRead(3)
== 1));
  {hoyre_a();}
  if((digitalRead(6 )== 1) && (digitalRead(5) == 1) && (digitalRead(4) == 0) && (digitalRead(3)== 0) && (digitalRead(3)
== 0));
  {hoyre_b();}
```

```
  if((digitalRead(6 )== 1) && (digitalRead(5) == 1) && (digitalRead(4) == 1) && (digitalRead(3)== 0) && (digitalRead(3) == 1));
  {hoyre();}
  if((digitalRead(6 )== 0) && (digitalRead(5) == 0) && (digitalRead(4) == 0) && (digitalRead(3)== 0) && (digitalRead(3) == 0));
  {brems();}
}

// Declaring Functions
void beveger_fremover()
{
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);
  analogWrite(11, 80);
  analogWrite(10, 80);

}
void beveger_bakover()
{
  digitalWrite(13, LOW);
  digitalWrite(12, HIGH);
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);
  analogWrite(11, 60);
  analogWrite(10, 60);
}
void venstre_a()
{
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);
  analogWrite(11, 70);
  analogWrite(10, 60);
}
void venstre_b()
{
```

```
  digitalWrite(13, LOW);
  digitalWrite(12, HIGH);
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);
  analogWrite(11, 60);
  analogWrite(10, 60);
}
void venstre()
{
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);
  analogWrite(11, 70);
  analogWrite(10, 50);
}
void hoyre_a()
{
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);
  analogWrite(11, 60);
  analogWrite(10, 70);
}
void hoyre_b()
{
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(9, HIGH);
  digitalWrite(8, LOW);
  analogWrite(11, 60);
  analogWrite(10, 60);
}
void hoyre()
{
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
```

```
  digitalWrite(9, LOW);
  digitalWrite(8, HIGH);
  analogWrite(11, 50);
  analogWrite(10, 70);
}
void brems()
{
  digitalWrite(13, LOW);
  digitalWrite(12, LOW);
  digitalWrite(9, LOW);
  digitalWrite(8, LOW);
  analogWrite(11, 0);
  analogWrite(10, 0);
}
```