TRASH BOT: THE LINE FOLLOWER

Drishya Dahal
ID: @01524334
drishyadahal@yahoo.com

Joseph Galloway
ID: @01437523
arr466@my.utsa.edu

Dr. Pranav A. Bhounsule
Mentor
210-458-6570

ABSTRACT

1. NOMENCLATURE

Artificial Intelligence- AI
Infrared- IR

2. INTRODUCTION

The prominent use of robotics in the foreseeable future is imminent because mechanical robots, machines, and instruments are rapidly outperforming humans. Similar to the revolutionary concept of automatic computers, which is to make machines do the calculations, AI is a radical concept of machines simulating and learning human-like traits such as abstraction, randomness, and creativity. This project aims to create a prototype of a mobile robot that can autonomously navigate an area in the library and move past students' tables for trash collection, contributing to environment conservation while embedding basic programming concepts of AI.

3. METHODS

The autonomous movement of Trash Bot is based on the property of light and its degree of absorbance of different colors. Because black surfaces absorb light and white surfaces reflect, the robot's movement is fixated around a black line. Eight linearly placed sensors, which continuously emit and receive infrared light, send a signal to an Arduino Mega 2650 that describes the color of the surface below the robot. The Arduino then manipulates the motors in such a way that the robot strictly follows a black line.

4. RESULTS

The robot follows a strict set of protocols to stay on track. If it senses deviation from the path, it backs up until it senses the black line again. As soon as a sensor detects a black line, the wheels turn until Trash Bot is back on track.

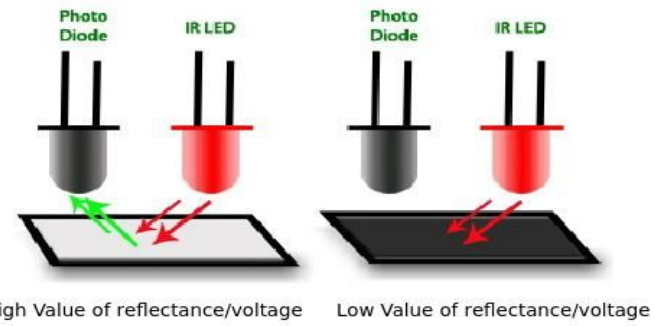

Fig 1. Reflectance and Absorbance of light by an IR Sensor above different surfaces

5. DISCUSSION

We included libraries created by Adafruit and Arduino to simplify our code. Each sensor displayed a value after each time it received its infrared transmission. The higher the value, the less infrared light the sensor received from its transmission. The highest possible value was 1,023, which indicated a black surface. The lowest possible value was 0, which indicated a white surface. Trash Bot needed to move backward when all of the sensors read less than 500.

We used a motor shield to switch the direction of the motors when the robot needed to move forward or backward. The arduino is powered by a single 9V battery. The motor shield draws power from the arduino through a direct connection.

6. CONCLUSION AND FUTURE WORK

One limitation was the amount of time our schedules allowed us to work on Trash Bot. We managed to accomplish our task while working four to six hours a week, but we could not always meet some weeks because of class, work, or other responsibilities.

Our prototype is a proof-of-concept that can serve as a foundation for future complex robots. An upgraded version can include a camera to localize images and receive verbal commands, which can help students navigate the campus. This concept can be further

upgraded to build navigation and disaster relief robots.

REFERENCES

"Arduino Project 05: Line Follower Robot". Online Video Clip. Youtube.com.15 Feb 2017.

"Line Follower using Arduino".Circuitdigest.com. Image. 23 March 2017.

"Ping Ultrasonic Range Finder".Arduino.com. Web. 20 Jan 2017.

"QTR- 8RC Reflectance Sensor Array".Pololu.com. Web. 20 Dec 2016.

APPENDIX

Section A. Code

```
#include <AFMotor.h>
#include<Wire.h>
#include <QTRSensors.h>
#define NUM_SENSORS 8// number of sensors used
#define NUM_SAMPLES_PER_SENSOR 4// average 4 analog samples per sensor reading
#define EMITTER_PIN A11
// sensors 0 through 5 are connected to analog inputs 0 through 5, respectively
QTRSensorsAnalog qtra((unsigned char[]) {A8, A9, A10, A11, A12, A13, A7, A15}, NUM_SENSORS, NUM_SAMPLES_PER_SENSOR, EMITTER_PIN);
unsigned int sensorValues[NUM_SENSORS];
AF_DCMotor right_motor(2);
AF_DCMotor left_motor(3);
const int trigPin = 43;
const int echoPin = 45;
long duration;
int distance;

void setup(){
  delay(500);
  int i;
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(47, OUTPUT);   //LED
  digitalWrite(47, HIGH);
   for (i = 0; i < 200; i++)  // make the calibration take about 5 seconds
  {
     qtra.calibrate();       // reads all sensors 10 times at 2.5 ms per six sensors (i.e. ~25 ms per call)
    delay(20);
  }
   // print the calibration minimum values measured when emitters were on
  Serial.begin(9600);
  digitalWrite(47, LOW);
  for (i = 0; i < NUM_SENSORS; i++)
  {
    Serial.print(qtra.calibratedMinimumOn[i]);
    Serial.print(' ');
  }
  Serial.println();
   // print the calibration maximum values measured when emitters were on
  for (i = 0; i < NUM_SENSORS; i++)
  {
    Serial.print(qtra.calibratedMaximumOn[i]);
    Serial.print(' ');
  }
  Serial.println();
  Serial.println();
  delay(1000);
  }

void loop()
{
  digitalWrite(trigPin, LOW);       //Clears the trigPin
  delay(2);
   digitalWrite(trigPin, HIGH);      //sets the trigPin on HIGH for 10 microseconds
  delay(10);
  digitalWrite(trigPin, LOW);
     duration = pulseIn(echoPin, HIGH); //Reads the echoPin, returns the sound wave travel time in microseconds
  distance = duration*0.034/2;       //cm
  //Serial.print("Distance: ");
    //Serial.println(distance);                //End of Ultrasonic Sensor portion
  qtra.read(sensorValues);
  unsigned char i;
  for (i = 0; i < NUM_SENSORS; i++)
  {
    Serial.print(sensorValues[i]);
    Serial.print(' ');
  }
  Serial.println("    ");
  delay(25);
  movement();
  }

  int x = 1;
  if (distance <= 10)
  {
   right_motor.run(RELEASE);
   left_motor.run(RELEASE);
   Serial.println("Stop!");
  }

                        else            if((sensorValues[0,1,2,5,6,7]
```

```
<=450)&&(sensorValues[4] >=500))  //can do ! statement here as well
 {
  right_motor.setSpeed(190*x);
  left_motor.setSpeed(190*x);
  Serial.println("Forward!");
  right_motor.run(FORWARD);
  left_motor.run(FORWARD);
  delay(200);
 }
                else if ((sensorValues[0,1,2,4] <=450)&&((sensorValues[7,6] >=500) || (sensorValues[6,5] >= 500) || (sensorValues[5] >= 500) || (sensorValues[5] >= 500) || (sensorValues[7] >= 500)))
 {
   right_motor.run(RELEASE);
   Serial.println("Turn Right");
   left_motor.setSpeed(160*x);
   left_motor.run(FORWARD);
   delay(160);
 }
 else if ((sensorValues[0,1,2] <=450)&&(sensorValues[4] >= 450))
  {
  right_motor.run(RELEASE);
   Serial.println("Turn Slightly Right");
   left_motor.setSpeed(160*x);
   left_motor.run(FORWARD);
   delay(160);
}
            else if ((sensorValues[7,6,5,4] <= 450)&&((sensorValues[0] >=500) || (sensorValues[1,0] >= 500)) ||(sensorValues[2,1] >= 500) || (sensorValues[2] >= 590))
 {
   left_motor.run(RELEASE);
   Serial.println("Turn Left");
   right_motor.setSpeed(160*x);
   right_motor.run(FORWARD);
   delay(160);
 }
else if (sensorValues[0,1,2,4,5,6,7] <= 500)
{
  Serial.print("Backward");
  right_motor.setSpeed(190*x);
  left_motor.setSpeed(190*x);
  right_motor.run(BACKWARD);
  left_motor.run(BACKWARD);
  delay(200);
}
else
{
Serial.print("Nothing");
right_motor.run(RELEASE);
 left_motor.run(RELEASE);
 }
}
```

ACKNOWLEDGEMENTS

We would like to acknowledge the guidance and support given by Kyle Lamoureux for helping us improve our code and sensor setup. We are also grateful to Dr. Bhounsule for giving us this project, guiding us to solutions, and for letting us use the Robotics and Motion laboratory to assemble Trash Bot.