School of Engineering

The University of Jordan, Amman-Jordan

Mechatronics Systems Design Lab (0908557)

Eng. Hisham Mohammad

Mobile robot to obstacle avoidance and traffic light reading.

By:

Khaled Al-Yousef (0193315)

Zuhour Alsaqqa (0195786)

Nada Zein Eddin (0195841)

May 2024

Abstract

The development of an autonomous mobile robot utilizing Arduino for navigation in an environment with both static obstacles and dynamic obstacles like traffic lights is explored in this project. Ultrasonic sensors are employed to detect and avoid static obstacles, while a color sensor is incorporated to identify and stop red objects, potentially representing traffic signals. Informed decisions are made by the robot leveraging its combined sensor data, including moving forward in safe conditions, moving left or right depending on obstacle existence, and prioritizing stopping at red objects detected by the color sensor. This multi-sensor approach enhances the robot's ability to navigate its environment safely and adhere to basic traffic light rules.

The project focuses on sensor integration, obstacle detection algorithm development, traffic light response implementation, and robot control and maneuvering programming. The potential of Arduino-based robots for navigating complex environments while adhering to basic traffic rules will be demonstrated upon successful completion, paving the way for further exploration in autonomous navigation functionalities.

Table of Contents

Table of Figures

Problem statement

Our project aims to address the challenge of building an autonomous robot with advanced navigation capabilities, specifically focusing on obstacle avoidance and traffic light recognition. The primary problem we seek to solve is how to enable the robot to navigate safely and efficiently in some dynamic environments, where it must avoid obstacles to prevent collisions and adhere to traffic signals for regulatory compliance. This involves integrating sensor technologies and some algorithms to ensure the robot can make real-time decisions to navigate independently, minimizing the need for external interventions.

Selection of the components

In this section we with show the selection of the components to build our robot.

1. Selection of the sensors

First, we must choose a suitable sensor for the specific tasks in the project.

- **Object detection.**

we have a variety of choices from sensors that can detect when obstacles appear in front of the car, such as IR sensors and ultrasonic sensors, for our design, we decided to use the ultrasonic sensor because ultrasonic sensors use sound waves (echolocation) to measure how far away you are from an object. On the other hand, IR sensors use infrared light to determine whether or not an object is present. By knowing how far the obstacle is from the car we have more flexibility to control the car and achieve greater accuracy.

Ultrasonic Ranging Module HC - SR04:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules include ultrasonic transmitters, receivers, and control circuits.[1]

Working Voltage	**DC 5 V**
Working Current	**15mA**
Working Frequency	**40Hz**
Max Range	**4m**
Min Range	**2cm**
MeasuringAngle	**15 degree**
Trigger Input Signal	**10uS TTL pulse**
Echo Output Signal	**Input TTL lever signal and the range in proportion**
Dimension	**45*20*15mm**

Figure 1: Parameters of HC - SR04 ultrasonic.[1]

We note that we can supply our sensor with power from Arduino without any driving components.

Principle of operation:

Ultrasonic sensors operate by emitting sound waves at a frequency beyond the range of human hearing. Subsequently, they await the reflection of these sound waves and calculate distance based on the time it takes for the echo to return. This process bears resemblance to radar, which measures the duration it takes for a radio wave to rebound after striking an object, as shown in Figure 2.

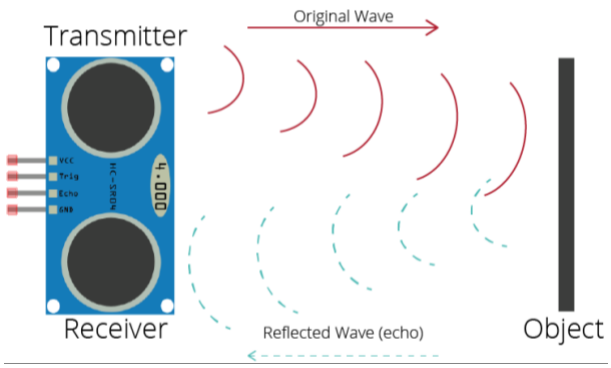

Figure 2: Working principle of the Ultrasonic sensor.[2]

- **Traffic light reading.**

To detect the color Light from the traffic light, automobile cars use a variety of sensors, and for the sake of the project, we are going to use a current-to-frequency converter plus the silicon photodiodes in the same circuit which is called TCS230 programmable color light to frequency converter.

TCS230 programmable color light to frequency converter:

The TCS230 programmable color light-to-frequency converter combines configurable silicon photodiodes and a current-to-frequency converter on single monolithic CMOS integrated circuit. The output is a square wave (50% duty cycle) with frequency directly proportional to light intensity (irradiance). The full-scale output frequency can be scaled by one of three preset values via two control input pins. Digital inputs and digital output allow direct interface to a microcontroller or other logic circuitry.

- High-Resolution Conversion of Light Intensity to Frequency
- Programmable Color and Full-Scale Output Frequency
- Communicates Directly With a Microcontroller
- Single-Supply Operation (2.7 V to 5.5 V)
- Power Down Feature
- Nonlinearity Error Typically 0.2% at 50 kHz
- Stable 200 ppm/°C Temperature Coefficient
- Low-Profile Surface-Mount Package

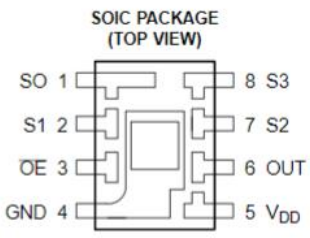

Figure 3:TCS230 specifications and the packaging.[3]

Principle of operation:

The light-to-frequency converter reads an 8×8 array of photodiodes. 16-photodiodes have blue filters, 16 photodiodes have green filters, 16 photodiodes have red filters, and 16 photodiodes are clear with no filters. The four types (colors) of photodiodes are interdigitated to minimize the effect of non-uniformity of incident irradiance. All 16 photodiodes of the same color are connected in parallel and which type of photodiode the device uses during operation is pin-selectable. Photodiodes are $120\ \mu m\ \times 120\ \mu m$ in size and are on $144\ \mu m$ centers.[3]

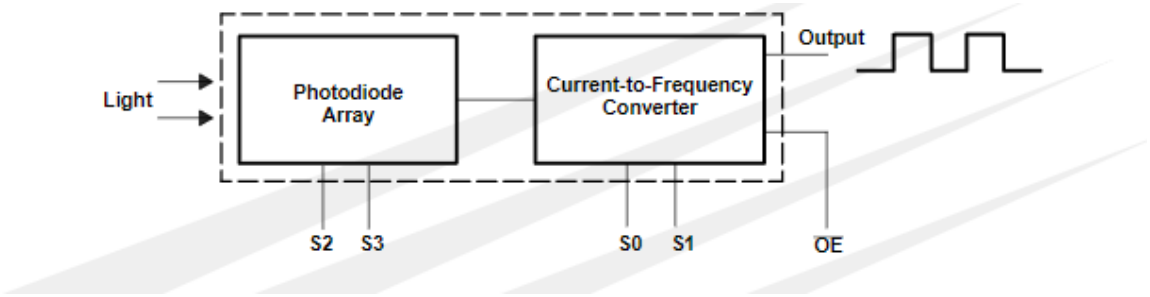

Figure 4:Functional block diagram.[3]

2. Selection of the actuators

Choosing the appropriate actuators plays a critical role in the success of a mobile robot. This report details the selection process for the two primary actuators (motors) in an obstacle-avoiding robot building to interact with traffic lights: a DC motor with a gearbox for main movement and a servo motor for a specific secondary function.

- **DC continuous motor**

The project has opted for a DC motor paired with a right-angled gearbox and rubber tire for several reasons. Firstly, this motor is purposefully designed for small mobile robot and toy vehicle applications, aligning perfectly with the requirements of our project. Its lightweight plastic gearbox contributes to the overall agility of the robot by keeping its weight minimal, thereby enhancing maneuverability and potentially prolonging battery life. Additionally, the motor's bidirectional control capability is crucial for executing obstacle avoidance and navigating around objects efficiently.

With the motor operating within a voltage range of 3-6 VDC, it offers adaptability in power supply configuration, accommodating various setups. Moreover, the inclusion of the right-angled gearbox furnishes the robot with heightened torque, surpassing that of a direct-drive motor, enabling it to manage slightly heavier loads or surmount moderate inclines with ease.

- Voltage: 3-6VDC
- Current: 80-150mA
- No Load Speed: 3V-125 rev/min5V-200 rev/min6V-230 rev/min
- Load Speed: 3V-95 rev/min5V-160 rev/min6V-175 rev/min
- Output Torque: 3V-0.8kg.cm5V-1.0kg.cm6V-1.1kg.cm
- Wheel Diameter: 65mm including tyre
- Wheel Width: 25mm
- Gearbox/Motor Dimensions: 20mm x 22mm x 65mm
- Weight: 50grams

Figure 5: Motor specifications. [4]

However, to drive a DC motor we will need an H-bridge, and an additional supply voltage connected to it.

Calculating the maximum weight the two yellow DC motors can handle:

$@3V$ the output torque is $\mathbf{0.8\ kg \cdot cm}$

- Considering we have two motors:

$$Total\ torque = 2 \times\ 0.8\ kg \cdot cm = 1.6\ kg \cdot cm$$

- Next, we need to convert this torque into force, considering the diameter of the wheel:

The formula to calculate torque is:

$$\boldsymbol{Torque\ = Weight \times Distance}$$

- Given that the wheel diameter is $\mathbf{65mm}$ and the torque is $\mathbf{1.6\ kg \cdot cm}$ we can rearrange the formula to solve for force:

$$\boldsymbol{Weight = Torque\ /\ Distance}$$

$$\rightarrow \boldsymbol{Distance} = \frac{\boldsymbol{Wheel\ Diameter}}{\mathbf{2}} = \frac{65mm}{2} = 32.5mm = 3.25cm$$

$$Weight = \frac{1.6\ kg \cdot cm}{3.25\ cm} \approx 0.492\ kg$$

- **Motor driver**

Most motors require a motor driver circuit for operation to vary the speed and direction of the motors to function as shown in Table 1. In our project, we utilize a dual full-bridge driver (L298N) to efficiently control the motors.

This is the popular L298N Dual H-Bridge Motor Controller, typically used to control motor speed and rotation direction.

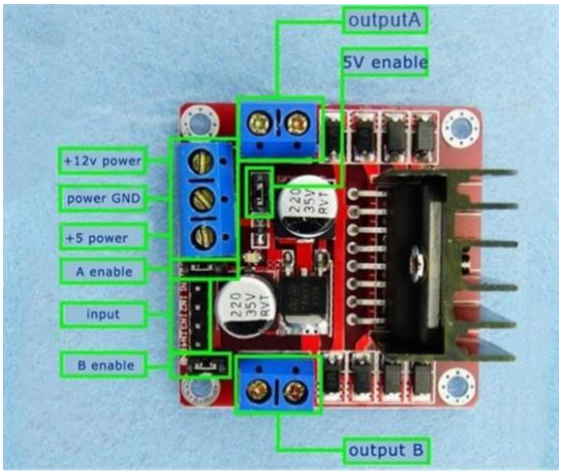

Figure 6: H-bridge photo and pins.[5]

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition	V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0 V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0 V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L V_i = X			6	mA
V_{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V_{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		VSS	V
I_{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L			–10	µA
I_{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	Vi = H ≤ V_{SS} –0.6V		30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)		2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L			–10	µA
I_{en} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{SS} –0.6V		30	100	µA

Figure 7: H-bridge electrical characteristics.[6]

3. Selection of the controller

Within the realm of mobile robotics, the controller plays a pivotal role, akin to the central processing unit (CPU) in a computer.

The core functionalities of the controller in an obstacle-avoiding robot translate to specific technical requirements:

- **Sensor Data Acquisition:** The controller must be equipped with Analog-to-Digital Converters (ADCs) to interface with analog sensors like ultrasonic sensors (typically outputting voltage) and the color sensor. The ADC resolution determines the accuracy of sensor readings. For obstacle avoidance, a resolution of 8-bits (256 discrete values) or 10-bits (1024 values) is often sufficient.
- **Decision Making and Algorithm Execution:** The controller needs adequate processing power (clock speed and memory) to execute the obstacle avoidance algorithm. Simple algorithms based on thresholding sensor readings might run efficiently on an Arduino Uno with an 8-bit AVR microcontroller. More complex path planning algorithms requiring real-time calculations might benefit from a controller with a faster clock speed and more memory, such as the Arduino Mega with a 16-bit AVR architecture.
- **Motor Control:** The controller needs digital output pins (GPIO) to interface with a motor driver. The motor driver then translates these digital signals into appropriate control signals (e.g., PWM) for the robot's motors. The number of GPIO pins required depends on the robot's motor configuration (e.g., two-wheeled differential drive requiring two motor control signals).

Selection Criteria - Matching Arduino Uno to Sensor Requirements

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on the computer, used to write, and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) to load new code onto the board. Finally, Arduino provides a standard form factor that breaks out the functions of the microcontroller into a more accessible package.[7]

Pins of the Arduino Uno

Arduino microcontrollers typically feature a variety of digital and analog pins, each serving specific functions and offering different electrical specifications. Here's an overview of the main Arduino pins and their typical electrical specifications:[7]

I. Digital Pins:

Digital Input Pins: These pins can read digital signals, typically denoted as HIGH (5 volts) or LOW (0 volts).

- Input Voltage Range: 0 to 5 volts.
- Input Impedance: High impedance (greater than 100 kΩ).

Digital Output Pins: These pins can output digital signals, either HIGH (5 volts) or LOW (0 volts).

- Output Voltage Range: 0 to 5 volts.
- Output Current: Up to 20mA per pin (recommended), with a maximum total current of 200mA for all pins.
- Output Impedance: Low impedance (around 25 ohms).

II. Analog Pins:

Analog Input Pins: These pins can read analog voltage levels, typically ranging from 0 to 5 volts.

- Resolution: 10-bit ADC (Analog-to-Digital Converter), providing a range of 0 to 1023.
- Input Impedance: High impedance (greater than 10 kΩ).

III. Power Pins:

5V Pin: Provides a regulated 5-volt output, typically used to power external components.

- Output Current: Varies based on the Arduino board's power source and regulator capabilities.

3.3V Pin : Provides a regulated 3.3-volt output.

- Output Current: Varies based on the Arduino board's power source and regulator capabilities.

Ground (GND) Pins: These pins are common ground connections used to complete circuits and provide a reference voltage.

- Voltage: 0 volts.

Connection and Hardware:

- **Overall project components**

- 1x Arduino UNO R3.
- 1x 2 Wheel Robot Car Chassis (Includes Motors).
- 1x HC-SR04 Ultrasonic sensor.
- 1x TCS230 color sensor.
- 1x (L298N) Motor Driver.
- 1x SG90 Servo Motor.
- 2x 3.5V Battery.
- 1x Power Switch.

- **Connection diagram**

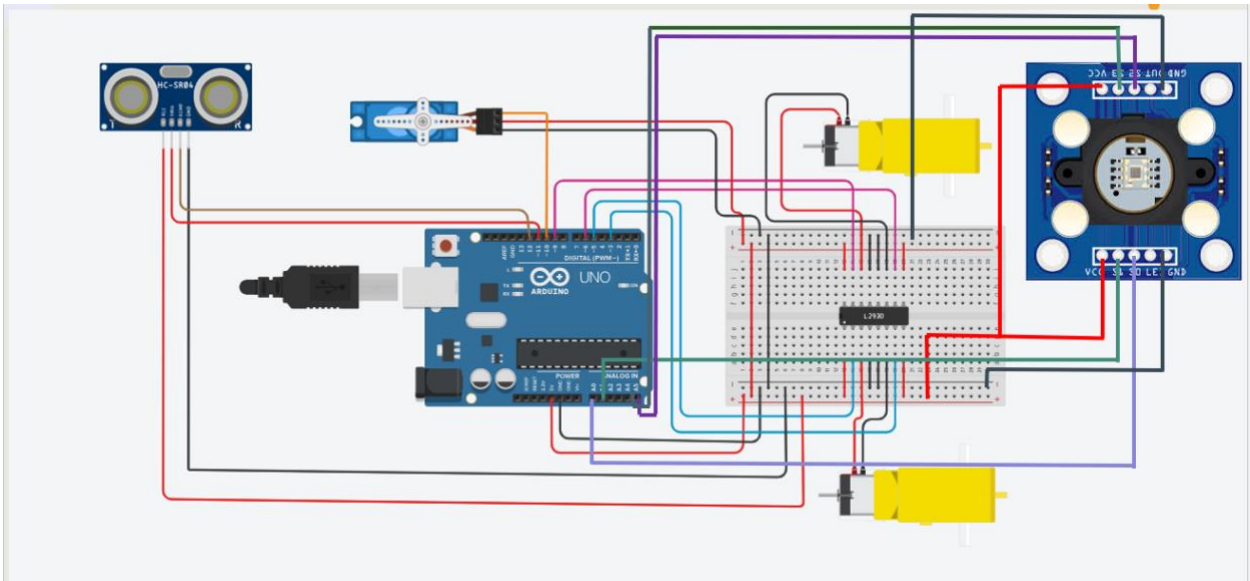

Figure 8 circuit diagram showing the Hardware Implementation of the project.

- **Project design**

In our project details the rationale behind the placement of sensors and servo motors in a robot designed for obstacle avoidance and traffic light detection. The robot is envisioned to operate in street environments, mimicking human navigation and interaction with traffic signals.

Sensor Placement

The primary obstacle avoidance and traffic light detection sensor is positioned high on the robot's body. This placement is inspired by the location of eyes on humans and animals, offering a wider field of view for effective obstacle detection in a human-designed environment. Streets present obstacles from all directions, making a high vantage point advantageous for the sensor.

Servo Motor and Sensor Configuration.

The report proposes a configuration where the servo motor directly rotates itself, rather than the shaft it's mounted on, by fixing the servo on from its shaft. This can be achieved by removing the standard servo horn and attaching the sensor directly to the motor with the help of a wood arm. This allows for providing a wider angle of observation.

Color Sensor Placement for Traffic Light Detection

Traffic lights are typically positioned on the left side of intersections. To optimize detection of these signals, the color sensor responsible for traffic light detection is positioned on the right side of the robot. This placement ensures the sensor has a clear view of the traffic light without triggering the ultrasonic signals.

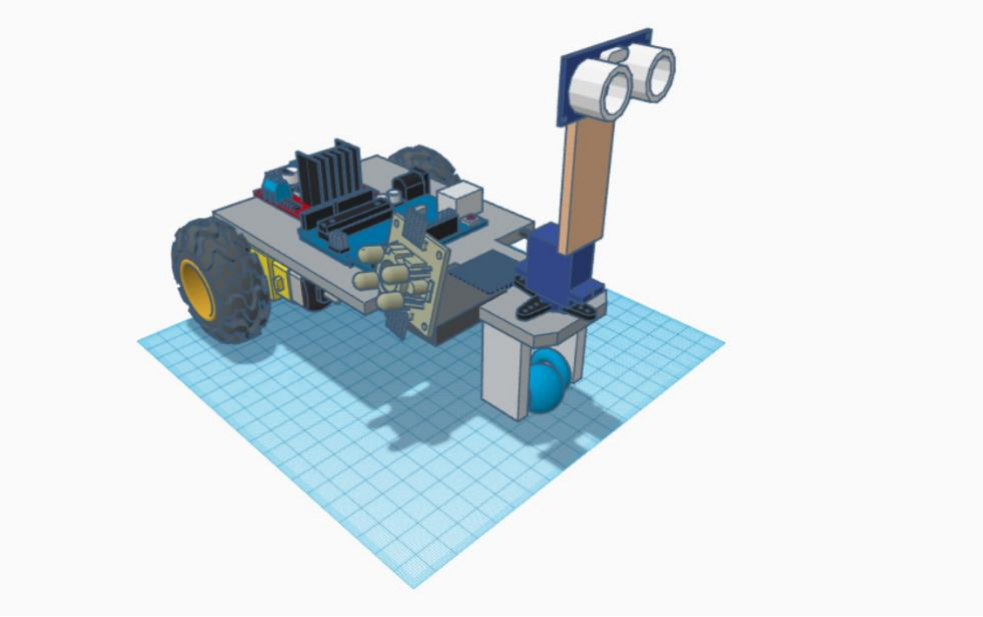

Figure 9: Design of the robot.

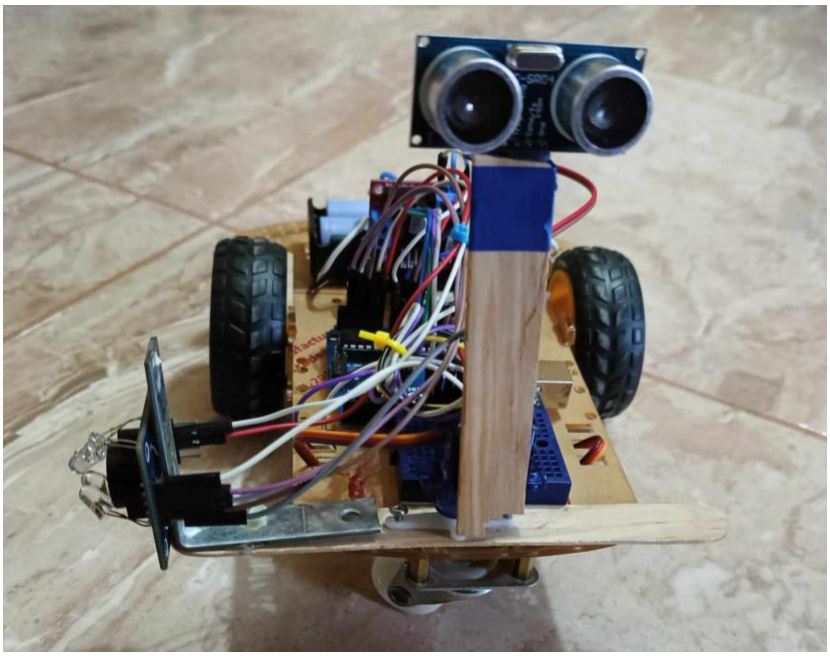

Figure 10: Final assembly.

Figure 11: The robot with the case.

Flow chart and think methodology.

In this section we will represent over all consent for building our robot with to function as we require.

- **Thinking methodology**

We want to accomplish two requirements, first, the traffic light detection, and second, the obstacle avoidance, in order to stop at the red traffic light without consideration off an obstacle presence we need to prioritize the Color sensor output signal over the ultrasonic one.

We have chosen to make our traffic light detection system detects only red color, this decision is based on the assumption that there are only red and green colors on the traffic lights options that operates complementary to each other and no other third option, and thus, when the red is present the green is not present and when the red is not present the green is present. Making the red, not red, a reasonable condition to decide the movement Abon. However, to measure the red color existence we have to compare its signal measurements compared to other signal, thus, we have to measure them, we had to tune them ourselves on specific determined red, green, and blue colors, so the frequency measuring is conditioned into meaningful reading.

Furthermore, for the obstacle avoidance we have made our sequence to check for forward direction obstacle as a priority to move in a straight line and if not applicable because of an obstacle presence to adjust the robot orientation to move in straight line after it has rotated to the right if the right road is open. Furthermore, if right way is also not applicable, we then go to the left if there is no obstacle there. However, if the robot is stuck where there is always an obstacle it will remain at the same position with its ultrasonic oscillating right and left until it finds a way to go with. Additionally, when the robot has sensed a object there is a potential to have an object

also at the right or left of the sensed object, considering a long width of the object, this object can be of any shape, if considering a non-symmetrical concave shape, there is a possibility to hit the object from the left when moving to the right and vise-versa. Hence, every time it changes its orientation it will move backward slightly to eliminate any possibility for hitting an object.

Table 1. motor driving signals giving that D1 is CW and D2 is CCW

Motion/Motor	Left Motor	Right Motor
Move Forward	D1	D1
Move Backward	D2	D2
Turn Left	Stop	D1
Turn Right	D1	Stop

- **Flow chart.**

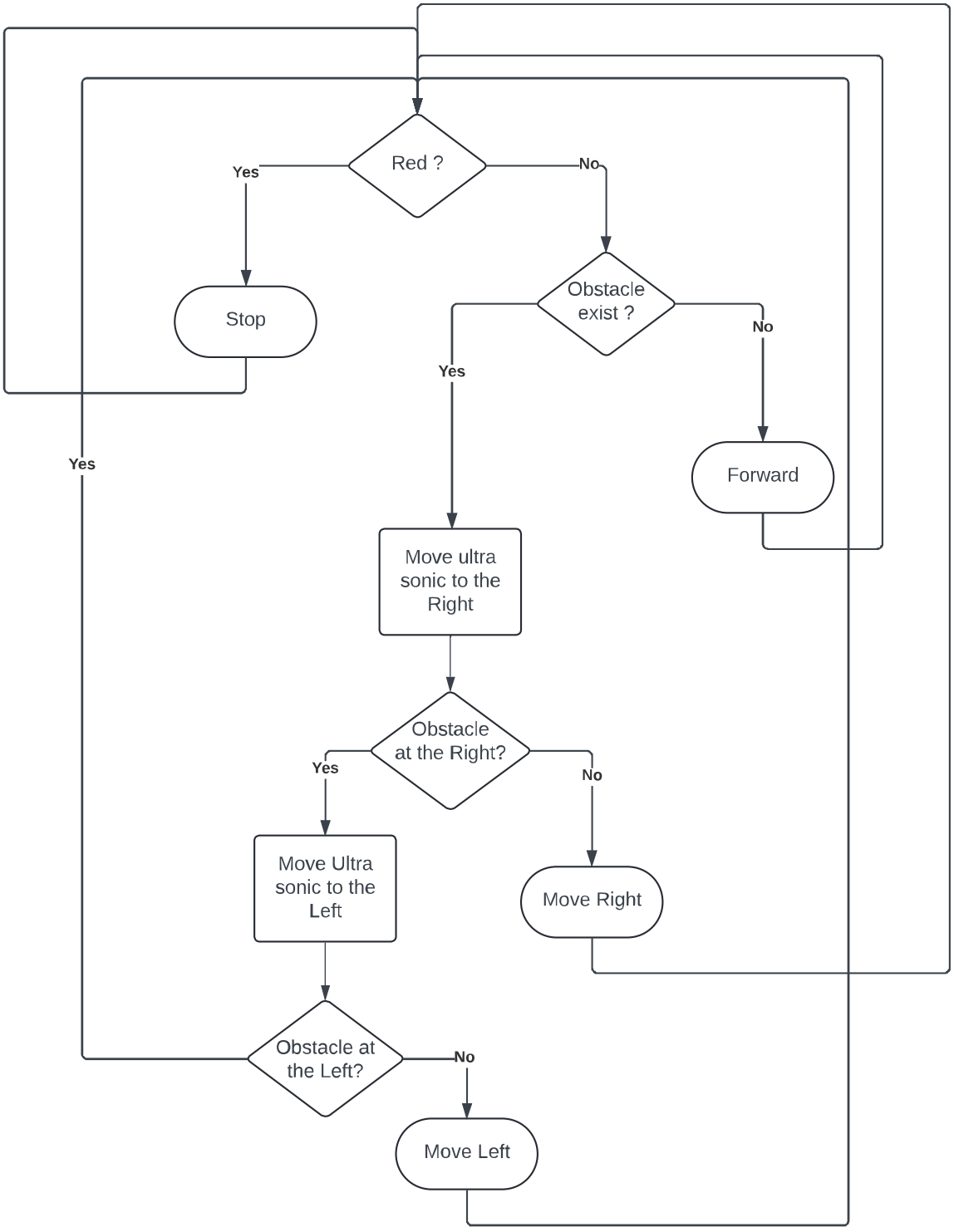

Figure 12: Flow chart of the system.

Conclusion

Building obstacle-avoiding robots with traffic light detection requires knowledge of various technologies, and troubleshooting various errors in hardware, design, and software. Furthermore, exposure to simulation software like Tinker CAD broadens understanding of the development process. By addressing these areas, we can create even more capable and versatile robots for the future. We can test for a sequence where the ultrasonic compares between left and write distances to compare and chose the clearer one and butting a value for a distance limit where it’s not allowed to move even if its better than the other way see how it will vary the performance from the current one .

References

[1] “HCSR04.pdf.” Accessed: May 15, 2024. [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

[2] “Complete Guide for Ultrasonic Sensor HC-SR04 with Arduino | Random Nerd Tutorials.” Accessed: May 15, 2024. [Online]. Available: https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/

[3] alldatasheet.com, “TCS230 PDF.” Accessed: May 15, 2024. [Online]. Available: http://pdf1.alldatasheet.com/datasheet-pdf/view/96470/ETC/TCS230.html

[4] “DC Motor Gearbox Wheel and Tyre - Mikroelectron MikroElectron is an online electronics store in Amman.” Accessed: May 04, 2024. [Online]. Available: https://mikroelectron.com

[5] “Dual H-Bridge DC & Stepper Motor Driver L298N - Mikroelectron MikroElectron is an online electronics store in Amman.” Accessed: May 04, 2024. [Online]. Available: https://mikroelectron.com

[6] “L298N.pdf.” Accessed: May 04, 2024. [Online]. Available: https://www.tech.dmu.ac.uk/~mgongora/Resources/L298N.pdf

[7] “What is an Arduino? - SparkFun Learn.” Accessed: May 15, 2024. [Online]. Available: https://learn.sparkfun.com/tutorials/what-is-an-arduino/all

Appendix A: Arduino C code:

```
#include <Servo.h>
Servo Myservo;

//ultra sonic
#define trigPin 11          // Trig Pin Of HC-SR04
#define echoPin 12          // Echo Pin Of HC-SR04

//motors on PWM pins
#define MLa 9               // Right motor 1st pin
#define MLb 6               // Right motor 2nd pin
#define MRa 3               // Left motor 1st pin
#define MRb 5               // Left motor 2nd pin

//color sensor
#define S0 A0
#define S1 A1
#define S2 A5
#define S3 A4
#define sensorOut A2

long duration, distance;
int sp = 0.5*255;           //speed in terms of PWM

//initialize color sensor readings
int redFrequency = 0;
int greenFrequency = 0;
int blueFrequency = 0;

void setup() {
// Color sensor pins
  pinMode(sensorOut, INPUT);
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);

// Setting frequency scaling to 20%
  digitalWrite(S0,1);
  digitalWrite(S1,0);

// Motor pins
  pinMode(MLa, OUTPUT);
  pinMode(MLb, OUTPUT);
  pinMode(MRa, OUTPUT);
  pinMode(MRb, OUTPUT);

//ulta-sonic pins
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

//servo motor signal pin
  Myservo.attach(10);

  Serial.begin(9600);
```

```
}
void loop()  {
  // Setting RED (R) filtered photodiodes to be read
  digitalWrite(S2, LOW);
  digitalWrite(S3, LOW);
  redFrequency = pulseIn(sensorOut, LOW);

  // Setting GREEN (G) filtered photodiodes to be read
  digitalWrite(S2, HIGH);
  digitalWrite(S3, HIGH);
  greenFrequency = pulseIn(sensorOut, LOW);

  // Setting BLUE (B) filtered photodiodes to be read
  digitalWrite(S2, LOW);
  digitalWrite(S3, HIGH);
  blueFrequency = pulseIn(sensorOut, LOW);

  // Convert frequency to color intensity
  int redIntensity = redFrequency * 100 * 1.234;
  int greenIntensity = greenFrequency * 90 * 0.99;
  int blueIntensity = blueFrequency * 100;
  // Print color intensities
  Serial.print("R = ");
  Serial.print(redIntensity);
  Serial.print(" G = ");
  Serial.print(greenIntensity);
  Serial.print(" B = ");
  Serial.println(blueIntensity);

  // Check if red color is dominant
  if (redIntensity < greenIntensity && redIntensity < blueIntensity) {
    Serial.println(" - RED detected!");
  // Stop the robot
    digitalWrite(MRb, LOW);
    digitalWrite(MRa, LOW);
    digitalWrite(MLb, LOW);
    digitalWrite(MLa, LOW);
    Serial.println("Stop");
    delay(1000);
  } else {
  // Proceed with obstacle avoidance
//measuring the distance
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);       // Transmit Waves For 10us
    delayMicroseconds(10);
    duration = pulseIn(echoPin, HIGH); // Receive Reflected Waves
    distance = duration / 58.2;        // Get Distance
    Serial.println(distance);
    delay(10);
  // Obstacle avoidance logic...
    if (distance > 25)                 // Condition For Absence Of Obstacle
      {
        Myservo.write(90);
        analogWrite(MRb, sp);          // Move Forward
        digitalWrite(MRa, LOW);
        analogWrite(MLb, sp);
```

```
        digitalWrite(MLa, LOW);
      }
      else if ((distance < 25 )&&(distance > 0)){
// Condition For Presence Of Obstacle
        digitalWrite(MRb, LOW);     //Stop
        digitalWrite(MRa, LOW);
        digitalWrite(MLb, LOW);
        digitalWrite(MLa, LOW);
        delay(100);

// measure right side distance
        Myservo.write(0);                  //move Ultrasonnic to the Right
        delay(500);
        digitalWrite(trigPin, LOW);
        delayMicroseconds(2);
        digitalWrite(trigPin, HIGH);       // Transmit Waves For 10us
        delayMicroseconds(10);
        duration = pulseIn(echoPin, HIGH); // Receive Reflected Waves
        distance = duration / 58.2;        // Get Distance
        Serial.println(distance);
        delay(10);
        if (distance > 25)  //Right road is clear
        {
            Myservo.write(0);
            delay(500);
            digitalWrite(MRb, LOW);        // Move Backward
            analogWrite(MRa, sp);
            digitalWrite(MLb, LOW);
            analogWrite(MLa, sp);
            delay(100);
            digitalWrite(MRb, LOW);        //Stop
            digitalWrite(MRa, LOW);
            digitalWrite(MLb, LOW);
            digitalWrite(MLa, LOW);
            delay(100);
            digitalWrite(MRb, LOW);        // Move Right
            digitalWrite(MRa, HIGH);
            digitalWrite(MLa, LOW);
            digitalWrite(MLb, LOW);
            delay(300);
        }
        else if ((distance < 25)&&(distance > 0)){
// right and forward roads are blocked
//check for the left road distance

          Myservo.write(90);                 //move ultrasonic to the Left
          delay(500);
          digitalWrite(trigPin, LOW);
          delayMicroseconds(2);
          digitalWrite(trigPin, HIGH);       // Transmit Waves For 10us
          delayMicroseconds(10);
          duration = pulseIn(echoPin, HIGH); // Receive Reflected Waves
          distance = duration / 58.2;        // Get Distance
          Serial.println(distance);
          delay(10);
          if (distance > 25)                 //Left road is clear
          {
```

```
            Myservo.write(45);
            delay(500);
            digitalWrite(MRb, LOW);           // Move Backward
            analogWrite(MRa, sp);
            digitalWrite(MLb, LOW);
            analogWrite(MLa, sp);
            delay(100);
            digitalWrite(MRb, LOW);      //Stop
            digitalWrite(MRa, LOW);
            digitalWrite(MLb, LOW);
            digitalWrite(MLa, LOW);
            delay(100);
            digitalWrite(MRb, LOW);      // Move Left
            digitalWrite(MRa, LOW);
            digitalWrite(MLa, HIGH);
            digitalWrite(MLb, LOW);
            delay(300);
          }
        }
      }

  }
}
```