Prototyping and Systems Engineering Project Report for a Line-Following Autonomous Vehicle

Muhammad Sayem
Sayed Galib Hossen Rizvi
Ronjon Sarker
Raphael Christian Garfitt Catchpole
Dapsara kapuge

Hamm-Lippstadt university of Applied Sciences
Lippstadt, Germany

***Abstract*—This document presents the prototype developed by our team for a line-following car that is capable of following specific lines, detecting obstacles, identifying the colour of the obstacle and executing tasks based on the colour. In this paper, we will discuss the project's objectives, hardware design, electronic components used, and the algorithms implemented to enable the robot to meet its requirements. The paper will also cover the tasks performed, experiments conducted, and tests carried out throughout the semester to validate the functionality of the system.**

Keywords—RISC (Reduced Instruction Set Computing)

I. INTRODUCTION

The main intention of this project is to develop a small autonomous vehicle capable of following a line and making decisions based on obstacle detection and colour sensing. This vehicle is designed to navigate a test track, detect obstacles, identify their colours, and perform specific actions accordingly (e.g. Remove , Ignore and Park). This paper will detail the project's objectives, hardware design, electronic components, and the algorithms implemented to achieve these functionalities.

II. INITIAL SETUP AND PLANNING

Right at the beginning of the project, after receiving all the necessary components from the laboratory, we moved to connect two motors with a motor driver and two IR sensors with our Arduino Uno and conducted a test run. Our initial objective was to make sure that the motors could be controlled based on the input IR sensors provide.

Subsequent to the successful connection and testing of the motors and IR sensors, we got the laser-cut chassis components. We mounted the motors and IR sensors onto the chassis. In our initial test, we had used a track marked with line-wide black colored on a white surface.

In our first try, both IR sensors were kept inside the black line. After uploading the relative code in the Arduino Uno, the system worked correctly. We soon realized, however, that this sensor placement was not optimal for 90-degree turns and general stability. Moreover, we did not know how much width would be required for the final track.

It was then decided to reposition the IR sensors outside the line, based on these observations, and to see an improved performance in the next lab session. Indeed, this adjustment succeeded and laid down the foundation for further development and testing.

III. COMPONENTS

The electronic components of this robot prototype are given below, along with a description of the functions and some of the properties

Arduino UNO
2 IR sensors
Color sensor
2 DC Motors
Motor Driver (L298N)
Ultrasonic sensor
Rechargeable Polymer Li-Ion Battery
Breadboard and some jumper wires

Arduino UNO

The main microcontroller for this project is shown in Figure (01). The selection of the Arduino Uno was due to its sturdy functionalities and user-friendliness. When regarded as the Arduino Uno, based on the Atmel Atmega328 microcontroller from the megaAVR family, gains its popularity to people as a development board. The microcontroller has a revised 8-bit RISC processor core, which increases performance and efficiency. Besides, the highly extensive community support and available libraries come in high numbers, which make the Arduino Uno an ideal solution for rapid prototyping and development in educational and hobbyist projects. Its versatile operation interfaces with most sensors and modules to perform complex functions in the vehicle.

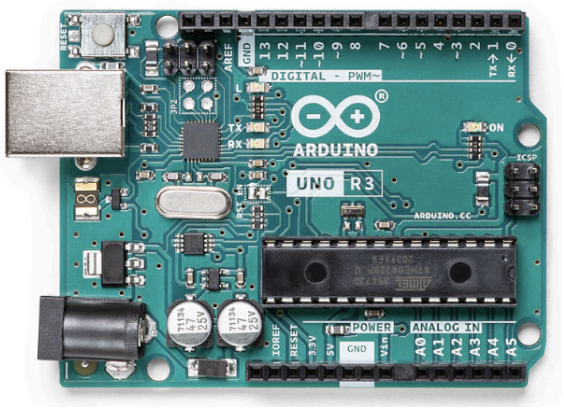


Figure (01) : Arduino UNO

1) Technical Specifications

Microcontroller: ATmega328P
Operating Voltage: 5V
Input Voltage (recommended): 7-12V
Input Voltage (limit): 6-20V
Digital I/O Pins: 14 (of which 6 can provide PWM output)
PWM Digital I/O Pins: 6
Analog Input Pins: 6
DC Current per I/O Pin: 20 mA
DC Current for 3.3V Pin: 50 mA
Flash Memory: 32 KB (ATmega328P) of which 0.5 KB is used by the bootloader
SRAM: 2 KB (ATmega328P)
EEPROM: 1 KB (ATmega328P)
Clock Speed: 16 MHz

IR Sensor

As shown in Figure (02), in this prototype there have been used two line tracking modules (ST1140) as line sensors. This prototype is following a black track with a white boundary, and these sensors can distinguish successfully between reflective (white) and non-reflective (black) surfaces. IR sensors use the principle of light reflection: white reflects infrared rays and black absorbs them. If the reflected infrared ray is detected, the vehicle shall be on the white track; otherwise, it shall be on the black boundary.

In this project, ST1140 sensors are mounted to detect the track's boundaries. These are configured to be able to detect the white track, ensuring that the vehicle is on course. If both of them detect the white, it keeps the vehicle on track. If either one detects the black boundary, it adjusts its path accordingly.

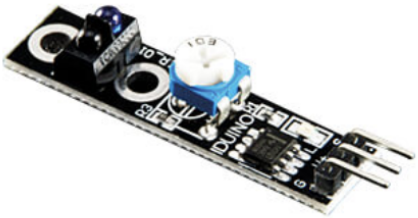


Figure (02) : ST 1140 IR Sensor

Color Sensor

In figure, We used a color sensor In this project to detect the colors red, blue, and green. The color sensor is an important component that gives the capability to the robot to do different things for each color.

- The following actions are triggered by respective color detections:
- Red: When the color sensor detects red, the robot is programmed to ignore obstacles.
- Blue: Upon detection, the robot removes obstructions from its way.
- Green: The robot will automatically park upon detecting the green color.

The ability of the color sensor gives the robot autonomy in navigation and interaction with the environment to make effective decisions in real-time upon color detection.

Figure (03) : Color sensor

D. DC Motors

Figure 04 shows that this project uses two Dc motors, each producing a torque ratio of 1:30, which helps control the movement of the line-following car. The DC motors are what drive the car, and they play an important role in the adjustment of speed. If the speeds of the motors are changed, then the car will either run faster or slower; hence, one controls how the car moves along the line. The application of the DC motors provides efficient and responsive movement, making the car very capable of following accurately in the tracks.

Motor Driver (L298N)

As shown in figure (05), the L298N Motor Driver drove the DC motors that powered the line following car. The L298N motor driver is a dual H-bridge module that can be used to drive two DC motors simultaneously. Distinctively, it provides an effective, precise module used in controlling motor speed that directly receives the command from the microcontroller.

Ultrasonic sensor

Figure 6 shows that an HC-SR04 ultrasonic sensor has been fitted in front of the car to detect any obstacle that may be on its way. The HC-SR04 sensor works by sending out ultrasonic waves and measuring the time it takes for the waves to return after bouncing on an obstacle. Through this, it can get an exact calculation of the distance to the obstacle.

Figure (04) : DC motor

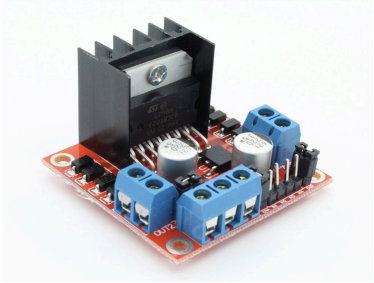


Figure (05) : Motor Driver (L298N) board

The sensor shall be combined with the Arduino board for processing the measurements.

On detecting any object within its detection range, the HC-SR04 sensor sends a signal to the microcontroller, Arduino. Once the signal is received, the Arduino board processes it and does the necessary action as required.

Battery

In figure (07), For the power supply in this project, a battery

with the following specification was used:

Nominal Voltage: 11.1V

Charging Capacity: Maximum 6.4A

Capacity: 3200mAh

Discharging Current: Maximum 128A

This configuration of the battery was to make sure that power is supplied robustly and reliably during the running of the operations of the project. Its nominal voltage of 11.1V provides stability in operation that will support the running of many components such as motors, sensors, and microcontrollers.

.

Figure (06) : Ultrasonic sensor HC-SR04

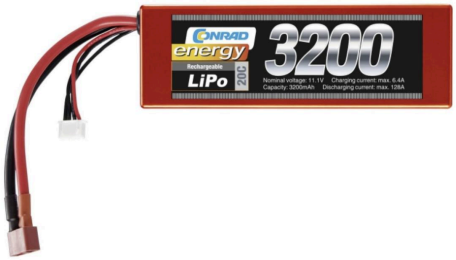


Figure (07) : LiPo 3200 mAh battery

IV. SIMULATIONS

To test the design before its physical realization and functioning, we used Tinkercad for simulations. It provided a flexible environment for developing virtual circuits to envision and debug the system in an appropriate manner.

This chapter will detail the circuit and schematic views of the various simulations that were run, interspaced with integration of components, including Arduino Uno, IR sensors, motors, and other modules integral to it. These simulations have been critical in fine-tuning the design and proving operational reliability associated with the robot.

Circuit View

Figure (08) explains the circuit details of an Arduino Uno interfaced with many devices, including IR sensors, servo motors, RGB LEDs, photodiodes, and an ultrasonic sensor. All these components play a critical role in the functioning of our autonomous car.

1.) IR Sensors: These are digital sensors connected to the Arduino on digital pins 2 and 4. They help the car trace lines around the track by changing at any point in time the direction of the car with respect to the sensor readings.

2.) Servo Motor: It is attached to the Arduino at digital pin 9. The servo motor steers the car.

providing precise rotational control to navigate around obstacles and follow the path.

3.) RGB LED and Photodiode: The RGB LED is controlled through analog pins A5, A4, A3, and A2. The photodiode, connected to analog pin A0, detects the intensity of the reflected light from the RGB LED, allowing the Arduino to determine the colour of objects in its path.

4.) Ultrasonic Sensor (HC-SR04): This sensor uses sound waves to measure the distance to objects in front of the car. The trigger and echo pins are connected to digital pins 6 and 5, respectively. The sensor data is used to stop the car and initiate obstacle avoidance maneuvers when an obstacle is detected within a set distance.

5.) Motor Driver (L298N): The motor driver controls the two DC motors that drive the car's wheels. The enable pins (enA and enB) are connected to PWM pins 3 and 11, while the control pins in1, in2, in3, and in4 are connected to digital pins 13, 8, 7, and 12. This setup allows the Arduino to manage the speed and direction of the motors, ensuring accurate movement.

6.) Power and Ground Connections: The Arduino powers all components through a breadboard, ensuring stable voltage supply.

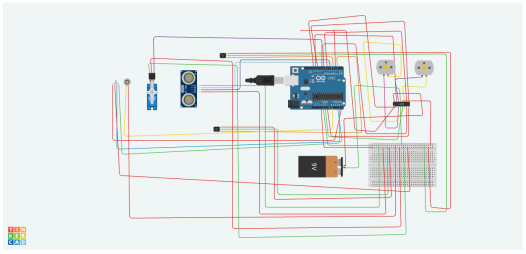


Figure (08) : Thinkercad view

Schematic View

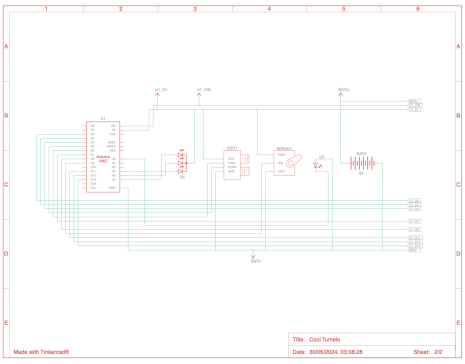


Figure (08.01): Tinkercad schematic view 1

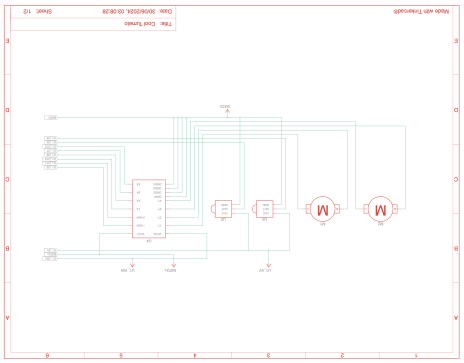
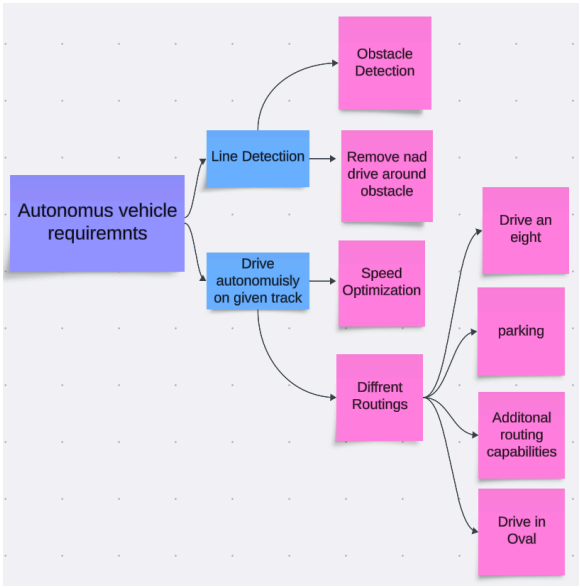


Figure (08.02): Tinkercad schematic view 2

IV. DIAGRAMS

A. Requirement Diagrams:



The project demonstrates advanced autonomous navigation, obstacle detection, and complex routing patterns, satisfying all requirements.

State Machine diagram:

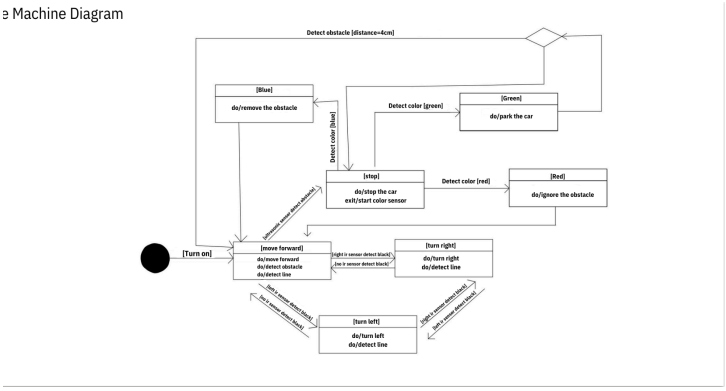


Figure (09) :State Machine Diagram

This state machine diagram indicates the behavior of a line-following robot capable of identifying obstacles and colors. When first powered on, the robot moves forward in the "Move forward" condition. It

In the process, it searches for lines and obstructions. When the line is detected by the right infrared sensor, the robot goes to "Turn right" mode, in attempts to reroute itself. Similarly, upon detection of the line by the left infrared sensor, the state of "Turn left" is turned on. If neither of these corresponding sensors detects the line, then the status of both turns is returned to the "Move forward" status.

At 4 cm, encountering an obstacle, it changes to the "Stop" state and switches on the color sensor. Depending on the color identified, it changes into three more states: "Blue" - the obstacle was taken away and it can continue, "Green" - it has finally parked, which marks the end of the entire operation, and "Red" - it will now ignore the impediment and continue going forward.

Activity diagram:

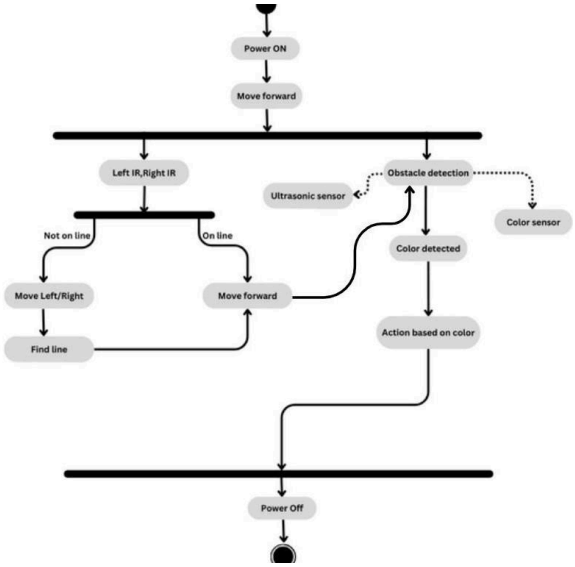


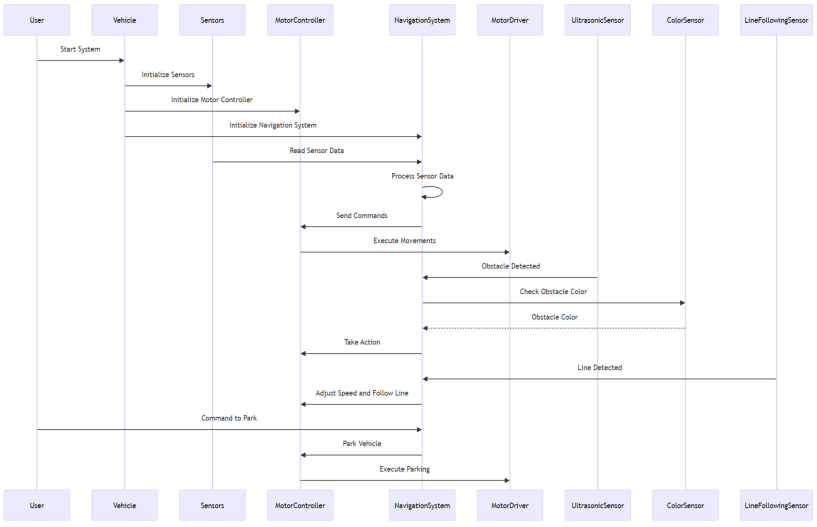
Figure (09.01) :Activity Diagram

In this paper, an activity diagram of a line following robot with obstacle detection and actions colored is given. This process starts with the switched on robot, this being the first step in its operation to which it activates the robot immediately after switching it on. This robot makes use of both left and right infrared It runs on IR sensors, through which the robot constantly measures its position regarding a

line as it travels. If the IR sensors detect that it is not on the line, then it runs on both sides to search for the line to put itself in the line. It keeps running until it finds the line. Then, after the verification by the IR sensors, the robot moves on the course.

As the robot moves forward, it detects an obstacle that lies in front of it with the help of the ultrasonic sensor. With this obstacle sensing, the color sensor is then used to sense the color of the obstacle. In the sensing of any color, it reacts to the color-sense through a pre-programmed path that may involve stopping, turning, or doing any other action per the programmed instruction. It ends by turning off the power in the operational flow, which means stopping the robot. These patterns guarantee robots to follow this path very precisely and to be obstacle-free. Besides, the robots also react impressively to any color of obstacles that come in front of them.

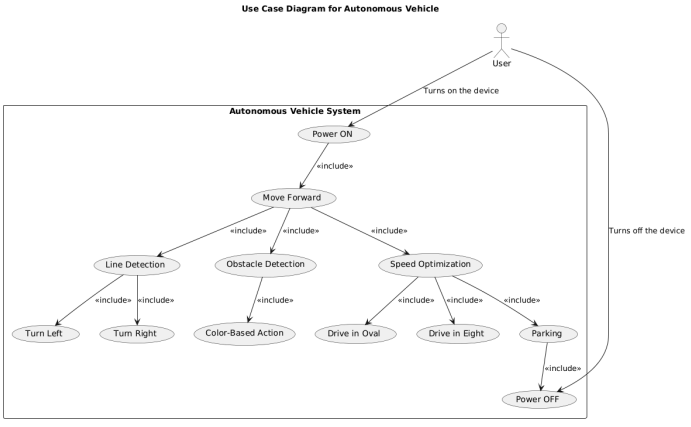
Sequence Diagram:



The sequence diagram shows the important interactions in an autonomous vehicle system. Triggering the system from scratch triggers the initialization of the car's sensors, motor controller, and navigation system simultaneously. This is the first part of the procedure. Sensing conditioned information thereafter interprets environmental inputs to the navigation system. After processing the data it has received, the navigation system instructs the motor controller with the appropriate directives. The motor controller controls how movements are executed, giving the motor driver instructions on how to carry out the required movements.

Among the aspects related to handling obstacles, the navigation system first determines the color of each obstacles it meets before instructing the motor controller on what to do. The ultrasonic sensor detects obstacles in the way. In the case of following lines, this sensor identifies the line and passes this information to the navigation system, which controls the speed and heading of the vehicle to maintain a course aligned with the line. At some point, the user commands the vehicle to park, thus engaging the parking sequence. Subsequently, the navigation system instructs the motor controller to execute the parking maneuver for secure and precise car parking. Via this series of interactions, an autonomous vehicle can handle various tasks and obstacles on its own and function impressively.

Use Case Diagram



Actors

User: The primary actor who communicates with the autonomous vehicle. Operator or any other person using the vehicle to execute a job and travel.

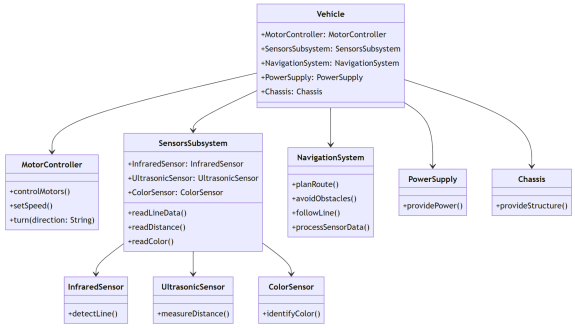
Use Examples

Drive Autonomously on Computer Created Follow up

Description: The automobile drives by itself over a pre-made path with no help by a human.

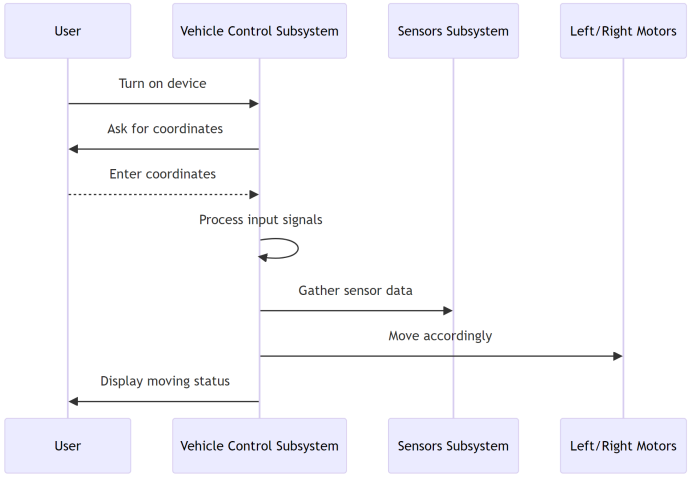
interaction: Upon having this option, the automobile follows its path depending on the understanding of the atmosphere and with the help of the navigation equipment.

Class Diagram

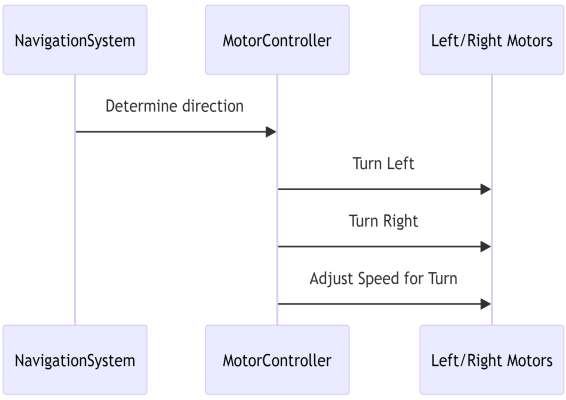


The Class Diagram offers a systematic view of an autonomous vehicle system, outlining most of the principal constituents and how they are related. In this system, the Vehicle class is central. This class welds a number of key subsystems together. The MotorController class is responsible for motor control, including direction and speed control, using techniques such as these: turn(direction: String), setSpeed(), and controlMotors(). In the SensorsSubsystem class, a number of sensors are defined, such as ColorSensor, which identifies the color of the object, UltrasonicSensor, which estimates the distance to obstacles, and InfraredSensor, for line detection. This subsystem contains functions for reading colors, distances, and line data.

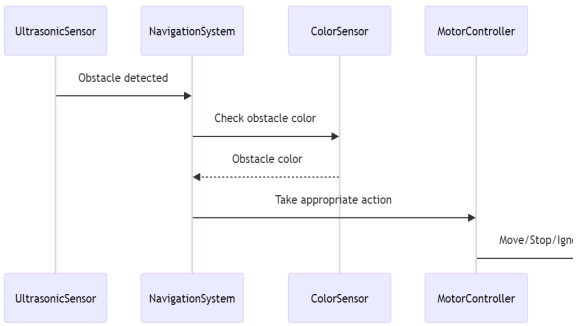
NavigationSystem has the responsibilities of route planning, avoiding obstacles, following lines, and processing sensor data to ensure safe and successful travel by a car. The PowerSupply class provides the necessary power so that all other components have sufficient energy to perform their work. The Chassis class provides for both a structural framework for the vehicle and housing with support for every other component. These elements can interact with each other and provide a coherent and working system to enable the autonomous car to function for the intended mission and demonstrate well-designed and useful architecture.



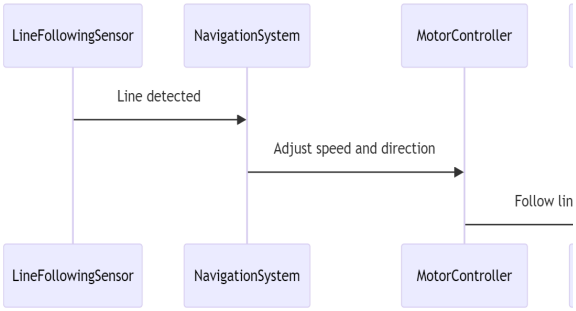
Control subsystem



Turning Subsystem



Object Avoiding Subsystem



Line Following Subsystem

Controll Subsystem

The Control Subsystem is responsible for managing the general operation of the vehicle. It coordinates activities between other subsystems and interacts with the user. Initially, the user inputs the desired coordinates and activates the device. The Control Subsystem processes input signals from the user and collaborates with the control subsystem in the collection of data that will be vital during navigation. It then enables the motors with exact instructions of the motions to be made, ensuring that the vehicle takes the intended route based on the data processed. During any movement, the Control Subsystem updates the user with regard to the progress that the vehicle is making and changes made, thereby communicating the status of the vehicle's movement to the user.

Turning subsystem

Turning Subsystem: It keeps track of changes in the vehicle's direction to ensure that turns, if required, are made accurately and with the closest possible vicinity. First, the Navigation System calculates the direction a turn needs to be made. Having determined the direction, the Motor Controller instructs the motors to execute the left or right turns. The controller further adjusts the motor speeds to facilitate the execution of such maneuvers with ease and accuracy so that the car takes the turn precisely and quickly.

Subsystem for Avoiding Objects

The Object Avoiding Subsystem identifies the obstacles that would lie in the path of the car with the help of color recognition and avoids them by employing due precautionary measures. When the Ultrasonic Sensor first detects an obstacle, it sends a signal to the Navigation System. The Navigation System then determines the color of the obstacle by consulting the Color Sensor. Based on the color it has observed, it decides to halt, maneuver around, or ignore the obstacle. Finally, the Motor Controller goes through the needed motions in order to enact the Navigation System's

judgment, guaranteeing that the automobile effectively avoids that obstacle.

Line-following Subsystem

By following the observed line, the Line Following Subsystem makes sure the car stays on the intended course. The Line Following Sensor initiates the process by identifying the line on the track. Once this line data has been processed, the Navigation System notifies the Motor Controller of the appropriate changes in speed and direction. This constant adjusting makes sure the car stays precisely on course and enables it to follow the track efficiently.

UPPAAL Simulation

Obstacle

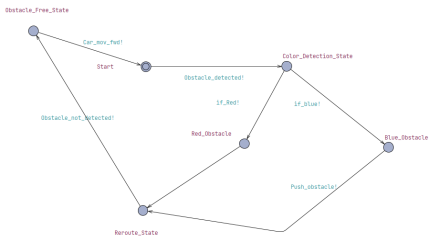


Figure (10.1) Obstacle State

This Figure (10.1) shows how the automated car reacts to a barrier. Initially, the vehicle travels forward while in the Obstacle_Free_State. When it detects an obstruction, it switches to the Color_Detection_State. It reroutes in the Red_Obstacle state. In the Blue_Obstacle state, it tries to push past it.. This simulation effectively simulates how the car would make decisions in different situations, resulting in safe and efficient navigation.

Line

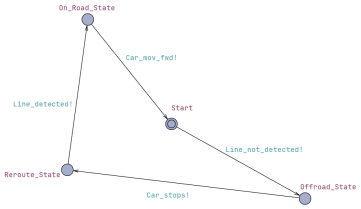


Figure (10.2)

Line State

In this model(10.2) when the condition Car_move_fwd! is satisfied, the car transitions from the Start state to the On_Road_State. When the car detects a line (Line_detected!). it moves from the Reroute_State to On_Road_State. The car enters the Offroad_State if the line is not detected (Line_not_detected!).And the car stops (Car_stops!), it moves from the Offroad_State to the Reroute_State, which enables it to return to the On_Road_State when the line is detected.

CAR

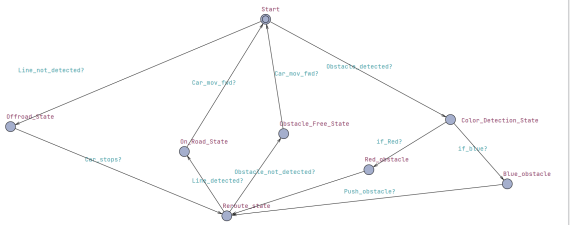


Figure (10.3)

Car System

This Figure(10.3) shows Car looks for a detected line to keep on track or reroutes if no line is detected when it is in the On_Road_State. It pauses and reroutes if it gets off-road.If the vehicle can continue onward, it advances from the Start state to

the Obstacle_Free_State. When an obstacle is found, the system switches to Color_Detection_State, which determines the color of the obstruction and displays either Red_obstacle or Blue_obstacle. After addressing the impediment appropriately, it either continues on or reroutes back to Obstacle_Free_State. This model contributes to safe and dependable navigation by simulating and verifying the behavior of the vehicle under various circumstances.

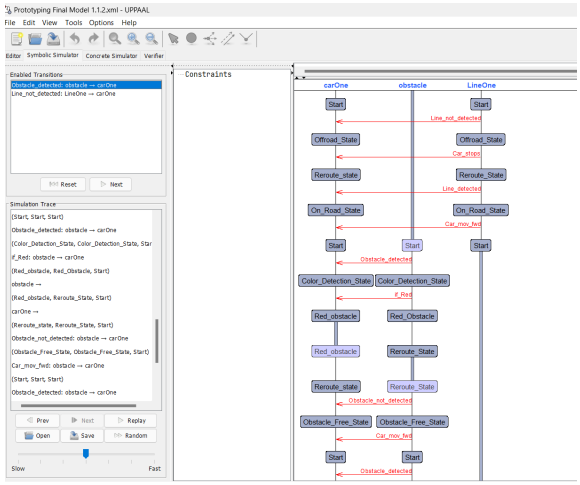


Figure (10.4) Simulation Trace

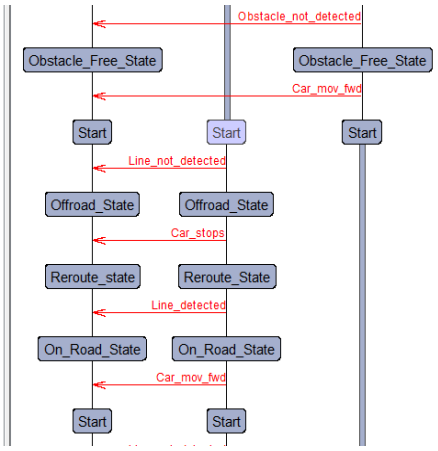


Figure (10.4.1) Simulation Trace

V. PROTOTYPE DESIGN

A. Overview

In this section the design process will be covered. For our prototype we have utilized several parts designed with Solidworks. The vehicle base is composed of two wooden pieces which have been laser cut into shape with a laser cutter and a number of 3-D printed parts each serving one or more distinct purposes. In this chapter each of these parts will be elaborated upon in detail, explaining the thought process behind the design decisions and the specific functionalities which the parts fulfill.

B. Custom Components

1) *Wooden Parts:* The prototypes main structural components are wooden boards which have been laser-cut into shape.

a) *Description:* The back corners of the boards are square shaped, providing a stable base and a large area for mounting larger components. The front half is tapered, creating a narrower profile. This tapering is a design choice based primarily on aesthetics, as it does not significantly affect aerodynamics or maneuverability given the prototype's relatively slow speed. This distinctive shape makes the front of the prototype easily recognizable. Additionally, the bottom board features cutouts allowing the wheels to be placed close to the body. Additional holes for mounting of further components and screws, as well as allowing wires to pass through were added by drilling and filed to correct size.

b) *Design Specifications:* The boards were designed using SolidWorks [1] and cut out with a laser cutter from wood with a thickness of 6mm. Their width is 240mm and their length is 300mm.

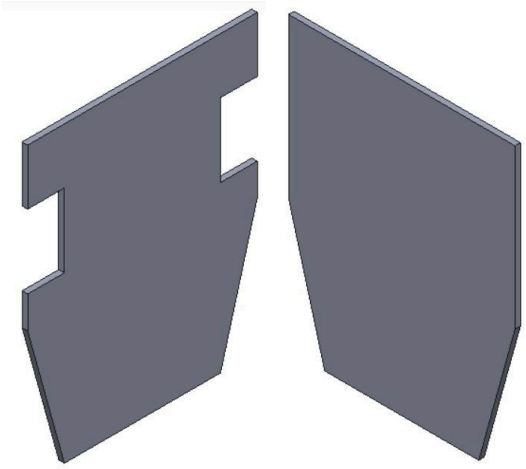


Figure (11.1) Base wooden parts bottom (left) and top (right)

2) *3D- Printed Parts:* A total of six further custom components were developed.

a) Description: the main purpose of the custom 3D-Printed components are to mount the electronic components in the correct positions.The front wheel mounting base (11.2) was designed to position the front wheel, which is a metal ball bearing enclosed in a plastic housing, close to the ground so the vehicle does not tilt towards the front.This in turn allows the IR sensors which are needed to detect the line to be almost parallel to the ground. The IR sensors are also mounted on this part on a long extrusion sticking out beyond the front of the vehicle which allows the IR sensors to be positioned slightly ahead of the Vehicles base.

Another important part is a 3D printed plate which serves as a mounting base for the Colour-, and Ultrasonic Sensors.

Finally, 4 identical spacers were designed in order to separate the two wooden base components creating a space between them for further components and wires to be placed. Their distinct oval shape allows them to be added or removed quickly and easily without the need for screws or any other securing measures.

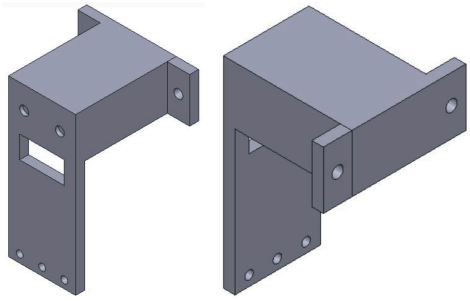


Figure (11.2) Front wheel mounting base, two views

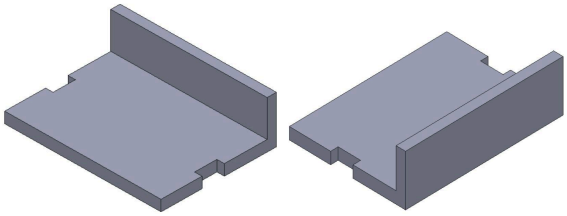


Figure (11.3) Front facing sensor mount plate, two views

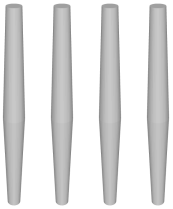


Figure (11.4) Spacers

b) *Design Considerations and Assembly process:* All 3D-Printed components were designed with SolidWorks. The front wheel mounting base (11.2) features a cutout which allows wires to pass through to connect to the IR Sensors which are mounted on the bottom. An additional hole was drilled manually for the wires to pass through the part. Two small divots were added to provide a space for the screws of the front wheel component. The part itself was attached to the bottom board with screws. A spare piece of wood was attached to the front holes of the part to

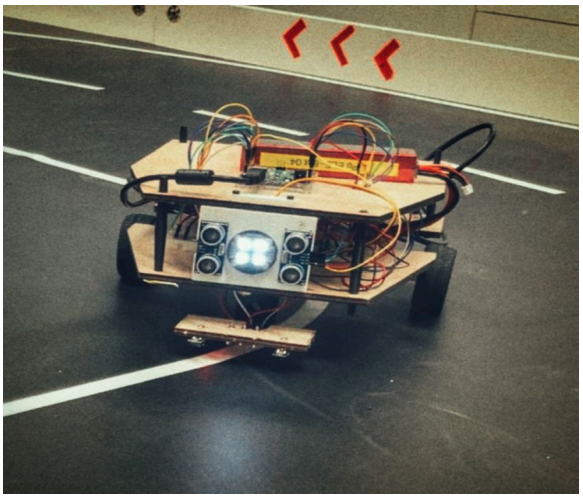


Figure (11.5) Assembled Prototype in action

serve as a base for the mounting of the IR sensors. The Sensors were secured to the wood with M3 screws and nuts to secure the screws and one M3 Nut was additionally used per sensor to serve as a spacer between sensor and the spare wood component.

The colour sensor was added in the center of the sensor mount plate (11.3) with M2 screws. The holes for these were carefully drilled into the component. The part features cutouts on each side for the pinheaders of the Ultrasonic Sensors to be placed. The holes for mounting these Sensors were too small in diameter for any screws available to us, so insead small holes were drilled into the component and a pair of thin rigid metal wires were used to securely mount the sensors.

4 holes were drilled in the corners of each wooden component for the spacers to be placed through them. The spacers allow for a space of approximately 60 to 70mm between the two wooden boards.

VI. SOFTWARE IMPLEMENTATION

Microcontroller, IDE and Programming Language :

The microcontroller “Arduino Uno” was provided, therefore the Arduino IDE was used for programming the prototype. C/C++ programming language is used.

Following The Line :

Fig.(12.1) shows that A simple algorithm is implemented to keep the vehicle in the line,prevent it from getting out of the line and make the turns according to the line The both IR sensors were attached to the front-wheel mount extension underneath the chassis. IR sensors send signals to the microcontroller, these signals indicate whether the surface is black or white. In our setup the sensors provide either High or Low signals. During the calibration process, the IR sensors send a High signal when detecting a black surface, and a Low signal when detecting a white surface to the Arduino. This behavior formed the basis for the control logic used to guide the vehicle along the track, enabling it to move forward and execute turns accordingly.

The track consisted of a black line on a white surface and IR sensors were placed outside this black line. To move forward, both IR sensors needed to detect the white surface, which would result in both sensors giving a low signal.

If the right IR sensor detected the black line (giving a high signal), it means the vehicle had drifted leftward. In response, the vehicle would execute a right turn until both IR sensors returned to giving low signals.

Similarly, if the left IR sensor detected the black line, indicating a rightward drift, the vehicle would execute a left turn until both sensors indicated a return to the white surface, ensuring it stayed on track.

Obstacle Detection :

The Ultrasonic Sensor HC-SR04 is mounted to the front of the vehicle to measure the distance of the obstacle. The measureDistance function calculates distance and sends it to the microcontroller.

```
62  if (rightSensor == 0 && leftSensor == 0) {
79      forward();
80    }
81  } else if (rightSensor == 1 && leftSensor == 0) {
82      turnRight();
83  } else if (rightSensor == 0 && leftSensor == 1) {
84      turnLeft();
85  }
```

Figure (12.1) Parts of code to keep the vehicle in line

```
88 long Ultrasonic_read() {
89   digitalWrite(trigger, LOW);
90   delayMicroseconds(2);
91   digitalWrite(trigger, HIGH);
92   delayMicroseconds(10);
93   long time = pulseIn(echo, HIGH);
94   return time / 29 / 2;
95 }
```

Figure (12.2) ; Function for measuring the distance of obstacles.

Fig.(12.2) shows, It requires triggerPin and echoPin parameters for sensor interfacing. Initially, it sets the trigger pin low to reset the trigger pin for 2 microseconds. The trigger pin is then set HIGH for 10 microseconds. This triggers the ultrasonic sensor to send out a high-frequency sound pulse and the function waits for the echo signal. Using the pulseIn function, it measures the duration during which the echo pin remains HIGH, corresponding to the round-trip travel time of the ultrasonic pulse.The speed of sound in air at room temperature is approximately 343 meters per second. Now convert this time measurement into centimeters using the speed of sound in air is approximately 29 microseconds per centimeter. As we are measuring one-way distance it requires to be divided by 2 again. Now the function accurately calculates and returns the distance in centimeters of the nearest object. The Minimum distance of an obstacle is set to 4 centimeters. If any obstacle is detected within this range the vehicle will stop and take further decisions.

Color Detection :

The Color Sensor TCS3200 is also positioned to the front of the vehicle to identify the colour of the obstacle. The “getcolour()” function reads the colours (R.G.B) and returns the readings to the main loop.

As shown in Fig (12.3), the function is implemented to detect and distinguish between red, green, and blue colours. It operates by configuring the sensor through control pins (S2_PIN and S3_PIN) for each colour channel and subsequently measuring the PWM signals from the sensor's output pin (OUT_PIN). These signals correspond to the intensity of light detected in each colour spectrum. After capturing the PWM values for red, green, and blue light, the function also prints these values to the serial monitor for debugging purposes. During calibration of the colour sensor, an observation happened on the serial monitor: when a red object was placed in front of the sensor, the readings for red colour intensity consistently appeared lower compared to those for blue and green. Similarly, when testing with blue and green objects, their respective colour intensity values also showed this lower trend.

So this logic is used to determine the colour, If redValue is the smallest, it identifies the colour as "Red"; if blueValue is smallest, it identifies as "Blue"; and if greenValue is smallest, it identifies as "Green" and return to the main loop to execute decision based on the colour of the obstacles.

Servo for obstacle elimination :

A micro servo SG-90 is connected to pin 9 and positioned beneath the chassis and above of the front wheel mount-plate. A library was included for the servo to operate as required for the vehicle. Initial position of the servo is set to 180.

```
int getColor() {
  int redValue, greenValue, blueValue;

  digitalWrite(S2_PIN, LOW);
  digitalWrite(S3_PIN, LOW);
  redValue = pulseIn(OUT_PIN, LOW);

  digitalWrite(S2_PIN, HIGH);
  digitalWrite(S3_PIN, HIGH);
  greenValue = pulseIn(OUT_PIN, LOW);

  digitalWrite(S2_PIN, LOW);
  digitalWrite(S3_PIN, HIGH);
  blueValue = pulseIn(OUT_PIN, LOW);

  Serial.print("Red: ");
  Serial.print(redValue);
  Serial.print("\tGreen: ");
  Serial.print(greenValue);
  Serial.print("\tBlue: ");
  Serial.println(blueValue);

  if (redValue < greenValue && redValue < blueValue) {
    Serial.println("Color: Red");
    return 1;
  } else if (blueValue < redValue && blueValue < greenValue) {
    Serial.println("Color: Blue");
    return 2;
  } else if (greenValue < redValue && greenValue < blueValue) {
    Serial.println("Color: Green");
    return 3;
  } else {
    return 0;
  }
}
```

Figure (12.3) : Function for color determination

```
    } else if (color == 2) { //Color blue
      servo.write(0);
      delay(1000);
      servo.write(180);
      delay(1000);
```

Figure (12.4) ; Code for servo operations

In Fig(12.4). It shows how the servo is rotating to

remove the obstacle if the colour of the obstacle is blue. If blue colour is detected then the servo will rotate from its original position 180 degree to 0 degree, wait for 1 second and then it will rotate back to its initial position.

Obstacle Avoidance :

Fig (12.5) demonstrates how the "IgnoreObstacle" function is designed to enable an autonomous obstacle avoidance for the vehicle.

Upon encountering an obstacle, the function initiates a sequence of maneuvers according to this code to navigate around it and get back to the line again. Separate Forward, turnLeft and turnRight motor functions are created with higher motor speed to enable the vehicle to bypass the obstacle quickly.

Main Loop:

The main loop shown above in fig:(12.6) continuously checks all the inputs from the sensors and makes decisions to move safely. The loop starts by measuring how far an object is by ultrasonic_read

```
133 void IgnoreObstacle() {
134   Serial.println("avoiding");
135   backward();
136   delay(400);
137   aturnLeft();
138   delay(500);
139   aforward();
140   delay(1000);
141   aturnRight();
142   delay(500);
143   aforward();
144   delay(700);
145   aturnRight();
146   delay(400);
147 }
```

Figure (12.5) ; Function for obstacle avoidance

function.Then it checks if both the IR sensors are Low and also if there is any obstacle in 4 centimeter distance. If there is no obstacle it will continue to move forward. But if any obstacle is detected within the range it will pause. During this pause, the getColor function is called to identify the color of the obstacle. If the obstacle is red, the IgnoreObstacle function is called to bypass the obstacle . If the obstacle is blue,the servo will rotate from 180 to 0 to remove the obstacle. For a green obstacle, the vehicle parks. The vehicle will turn right and turn left according to the response of the IR sensors.

```
53 void loop() {
54   distance_F = Ultrasonic_read();
55   Serial.print("D F="); Serial.println(distance_F);
56
57   int rightSensor = digitalRead(R_S);
58   int leftSensor = digitalRead(L_S);
59   Serial.print("Right Sensor: "); Serial.println(rightSensor);
60   Serial.print("Left Sensor: "); Serial.println(leftSensor);
61
62   if (rightSensor == 0 && leftSensor == 0) {
63     if (distance_F < Set) {
64       Stop();
65        delay(1000);
66       int color = getColor();
67       if (color == 1) {  //color red
68         IgnoreObstacle();
69       } else if (color == 2) { //Color blue
70         servo.write(0);
71         delay(1000);
72         servo.write(180);
73         delay(1000);
74
75       } else if (color == 3) { //color green
76         ParkCar();
77       }
78     } else {
79       forward();
80     }
81   } else if (rightSensor == 1 && leftSensor == 0) {
82         turnRight();
83   } else if (rightSensor == 0 && leftSensor == 1) {
84         turnLeft();
85   }
```

Figure (12.6) ; Main execution Loop

Future Improvements:

This autonomous vehicle system, in its robust and functional form, offers several opportunities for enhancement to further enhance efficiency, reliability, and versatility. Here are some suggestions of how the system could be improved:

1. Advanced Sensor Integration

Description:

The vehicle has sensors that detect lines, obstacles, and colors. The basic infrared, ultrasonic, and color sensors used to a large extent.

Achievements:

Autonomous Driving: On a predefined track, the vehicle autonomously drives using infra-red sensors for line detection.

Obstacle Detection: Equipped with an ultrasonic sensor for obstacle detection and avoidance for a perfect chiseled path.

Color-Mate Obstacle Handling: Onboard works a color sensor that helps the vehicle recognize the colors of obstacles and hence decide to either ignore, move, or park.

Optimizing Speeds: The vehicle changes the speed dynamically for effective and smooth navigation.

Complex Routing: It gives the handling of complex path patterns, and negotiates ovals and figure-eights patterns before finally taking up a parking task.

This project has also shown a very tight sensor and control integration that forms the base for any real-world application in autonomous navigation and robotics.

Potential Improvements:

Lidar Sensors: Integrating lidar sensors can bring in such improvements in obstacle detection and environment mapping. Lidar will provide high resolution 3D data that shall be helpful in navigation, avoiding obstacles with accurate precision.

Camera Systems: Equip cameras with image processing algorithms for image processing to enhance line detection and object recognition. This would enable the vehicle to move in far more complicated environments and recognize a wide range of other objects.

2. Navigation Algorithms Improvement

Current Implementation:

The vehicle follows a simple line following algorithm and applies simple decision making for obstacle avoidance.

Possible Improvement:

AI and machine learning: AI and machine learning algorithms implemented into the vehicle would make it learn from its surroundings and evolve over time to perfect its navigation. For example, reinforcement learning can make the vehicle optimize a path based on past experiences.

SLAM: Integrate SLAM algorithm to enable the vehicle to create a map of the environment and then further localize itself in that created map. This would improve navigation and obstacle avoidance within unfamiliar terrains.

3. Improved Control System

Current Implementation:

The control system uses a simple method correlating sensor information to motor movement.

Possible Enhancement:

PID Controllers: PID controllers would enhance the accuracy of Motor Controls to provide smooth and precise movements, particularly during turns or speed adjustments.

Redundancy and Fault Tolerance: Through redundancy in the control system, coupled with fault tolerance mechanisms in place, comes the reliability of the system when running without failure due to faults.

4. User Interface and Interaction

Present Implementation:

User interaction with the vehicle is through basic input methods, and in return, the user gets limited feedback about the status of the vehicle.

Potential Improvement:

Mobile App Integration: A mobile application could be used to provide a more convenient and user-friendly interface for driving the vehicle, setting coordinates, and feedback from real-time status situational awareness of the vehicle status and the environment.

Voice Commands: Integration of state of the art voice recognition technology could make vehicle driving and input commands possible through mere voice commands, completely aimed at making it easy for every kind of person to operate the vehicle.

Dynamic Power Management: Implementation of dynamic power management techniques can optimize energy usage at the different operational states of the vehicle in order to extend battery life and reduce the power consumption.

Energy Harvesting: For instance, such solution space exploration on energy harvesting as solar panels offers additional sources of power that would enhance the operational duration and sustainability of the vehicle.

6. Simplified Aesthetics

Current Design:

The vehicle's appearance was utilitarian, less than glamorous in appearance.

Potential Betterment:

Modern Sleek Design: Give the vehicle body a modern sleek outlook. A more futuristic and professional outlook can be given by applying smooth curved surfaces and having a consistent color scheme in the vehicle.

Integrated components: Conceal all wiring and components within the body works that will give a clean polished finish. Allow internal components to be protected and secluded by casings and housing.

Link to the Github

Link : https://github.com/mddsayemm/LFC

Division of Work for the Autonomous Vehicle Project

In our autonomous vehicle project, the tasks were divided between five team members to be comprehensive and well executed at the end. The tasks were collaborative and required teamwork; every member brought along their particular or special skills and helped others in underlined goals. Below is the detailed division of responsibilities, describing the contributions of each member towards the completion of the project and how they supported one another.

Team Members and Their Roles

Muhammad Sayem: Coding Specialist

Primary Responsibility: Worked on all phases of programming the vehicle. Core algorithms developed for line following, obstacle detection, color detection.

Support Role: Collaborated with Raphael C.G Catchpole in designing the prototype.Supplied Suggestions and Trouble shooting support in testing to Ronjon Sarker. Help was required while preparing diagrams from Sayed Galib Hossen Rizvi.

Raphael C. G. Catchpole: Design specialist

Primary Responsibility: Responsible for the overall design and aesthetics of the vehicle, including making the design of a modern, sleek vehicle body and integrating components smoothly.

Support Role: Supported Sayed Galib Hossen Rizvi in all matters concerned with the visuals regarding the documentation, making sure that the diagrams are correct and give an honest image of the design. During assembly, Subhajit worked hand in glove with Ronjon Sarker to be sure that his design was something practical and feasible to implement.

Sayed Galib Hossen Rizvi: Diagram and Documentation Specialist

Primary Responsibility: He was responsible for all UML diagrams such as sequence diagrams and Activity Diagrams diagrams. Documented the whole design and development process.

Collaborative Role: Worked alongside Raphael C. G. Catchpole in the documentation

Ensuring all design elements were captured. Moreover, the coding methodologies

And algorithms used were documented with the help of Muhammad Sayem. Finally, the final

Presentation was prepared by integrating technical details and visual elements with the

Help of Ronjon Sarker.

Ronjon Sarker: Assembly, Tinkercad simulation and Testing Specialist

Primary Responsibility: This was the leading of the physical vehicle assembly, ensuring that all components are rightly installed and configured. Simulated the entire process in Tinkercad and ran it on extensive testing for functionality validation.

Testing: Worked with Raphael C. G. Catchpole to put the design into practice and ensure that it is assembled practically. Worked with Muhammad Sayem in an analyzing problem of software which are found during testing. Worked with Sayed Galib Hossen Rizvi in collecting data and outputs for documentation and final presentation.

Dapsara Kapuge: UPPAAL Specialist

Primary Responsibility: Diffused primary attention in formal verification using UPPAAL, making models and verifying to prove the rightness and dependability of the vehicle's control logic.

Support Role: Collaborated with Muhammad Sayem to ensure verification and qualification of developed algorithms. Provided feedback and support to Sayed Galib Hossen Rizvi for documenting verification process and results.

References:

Mermaid.js. "Mermaid Live Editor." https://mermaid-js.github.io/mermaid-live-editor/

This source was used to create and visualize the UML sequence diagrams for the project documentation.

PlantUML.com. "PlantUML Language Reference Guide." https://plantuml.com/

[1] "Arduino Uno Rev3", Arduino Store. [Online]. Available: https://store.arduino.cc/products/arduino-uno-rev3. [Accessed: 27-Jun-2024].

[2] "IR Sensor, Motor, Motor Driver, Ultrasonic Sensor, and Color Sensor", roboter-bausatz.de. [Online]. Available: https://www.roboter-bausatz.de/. [Accessed: 29-Jun-2024].

[3] "Cool Tumelo", Tinkercad. [Online]. Available: https://www.tinkercad.com/things/j08u1SjPMVG-cool-tumelo/editel?tenant=circuits. [Accessed: 25-Jun-2024].

[4] "Activity Diagram and State Machine Diagram", diagrams.net. [Online]. Available: https://app.diagrams.net/#G1RwNVTiLk0aIhJt-UK53nhdtl8AUBLlXR#%7B%22pageId%22%3A%22qakn04WDkjylHX9NUeI2%22%7D. [Accessed: 30-Jun-2024].

[5] Dassault Systèmes, "SolidWorks 2021," Dassault Systèmes, 2021. [Online]. Available: https://www.solidworks.com/. [Accessed: 26-Jun-2024].