Abstract

Fire hazards pose significant threats to both life and property, requiring quick and efficient response mechanisms. This project presents the development of an autonomous fire-fighting robot capable of detecting and extinguishing fires while navigating its environment. The robot is equipped with an L298N motor driver for movement control, ultrasonic sensors for obstacle detection, and infrared fire sensors to identify fire sources. It operates independently, using real-time sensor data to make navigation and firefighting decisions.

The ultrasonic sensors continuously measure distances from obstacles, allowing the robot to avoid collisions by adjusting its direction accordingly. The fire sensors detect the presence of flames or high temperatures, triggering the firefighting mechanism when a fire is identified. The firefighting system consists of a servo-controlled water nozzle, which scans the affected area and sprays water until the fire is extinguished. A buzzer alarm alerts nearby individuals when a fire is detected, enhancing safety measures.

The robot's decision-making process follows a hierarchical control logic: it prioritizes fire suppression while ensuring safe movement. If an obstacle is detected in its path, it either turns or moves backward to find an alternate route. When a fire is detected, it moves towards the source, activates the water pump, and rotates the nozzle to cover the affected area. After ensuring the fire is extinguished, it resumes its normal patrolling routine.

This project demonstrates the potential of autonomous robots in firefighting applications, particularly in hazardous environments where human intervention is risky. It can be deployed in industrial areas, warehouses, and remote locations to provide an initial response to fires, reducing damage and improving emergency response efficiency. Future improvements could include integrating thermal cameras, wireless communication, and advanced AI algorithms for enhanced fire detection and navigation.

Acknowledgements

We would like to express our deepest gratitude to Mr. Jitendra Kumar Manandhar (JKM) and Mr. Suresh Jha (SJ) for their invaluable guidance, encouragement, and support throughout our academic journey. Their passion for teaching and unwavering dedication have been a source of motivation for us, helping us grow not only as students but also as problem-solvers.

From the very beginning, Mr. Manandhar and Mr. Jha have been more than just instructors; they have been mentors who instilled in us the importance of critical thinking, perseverance, and curiosity. Their ability to simplify complex concepts and create an engaging learning environment has made a significant impact on our understanding of Electrical and Electronics Engineering. Their patience and encouragement have given us the confidence to explore new ideas, ask questions, and push beyond our limits.

This project has been a challenging yet rewarding experience, and we owe much of our success to the foundational knowledge and problem-solving skills that they have helped us develop. Their continuous support and belief in our abilities have been instrumental in our ability to complete this work with dedication and confidence.

We would also like to extend our gratitude to our peers and lab staff, who have provided support, collaboration, and encouragement throughout the project. Their willingness to work together, share knowledge, and assist whenever needed has made this journey even more enriching.

A special thanks to the Robotics Club and our seniors, whose mentorship and guidance played a crucial role in our project's success. Their support in providing soldering tools, a workspace, and technical insights helped us overcome many challenges and refine our work. Their expertise and willingness to help made a significant difference in the development of this project.

Above all, we are most grateful for the genuine belief and trust that Mr. Manandhar and Mr. Jha have shown in us. Their guidance has shaped not only our academic growth but also our approach to problem-solving and learning. We will carry their teachings and inspiration with us as we continue our journey beyond this project.

Thank you, Mr. Manandhar and Mr. Jha, for your support, patience, and encouragement. Your impact on our learning experience is something we will always cherish.

Table of Contents

Introduction

Firefighting is one of the most dangerous professions in the world, requiring firefighters to risk their lives while rescuing victims and extinguishing fires. They often face hazardous environments, including extreme heat, smoke, and collapsing structures, making their job physically and mentally demanding. Despite advancements in safety equipment and firefighting techniques, there are still many situations where human intervention remains extremely risky.

To address these challenges, autonomous fire-fighting robots offer a promising solution. These robots can be deployed in dangerous areas where human firefighters might struggle to reach, such as burning buildings, industrial sites, or remote locations. They can navigate through hazardous environments, detect fires, and take immediate action to control or extinguish them. By doing so, they reduce the risks faced by human firefighters and help minimize property damage and loss of life.

Additionally, fire-fighting robots can be highly beneficial in households. A small, automated fire-extinguishing robot can detect fire outbreaks in their early stages and respond before they escalate into life-threatening situations. Household fires often start from kitchen mishaps, electrical faults, or unattended flames, and a quick response can prevent them from spreading. Having an intelligent fire-fighting robot at home could provide an additional layer of safety and peace of mind.

The development of such robots involves the integration of sensors, actuators, and intelligent control systems to detect fire, assess its severity, and take appropriate action. With further advancements in artificial intelligence, robotics, and IoT connectivity, fire-fighting robots could become an essential tool in fire departments, industries, and homes worldwide.

By automating fire detection and suppression, these robots have the potential to save lives, protect property, and support firefighters, making fire response faster, safer, and more efficient.

Goals And Deliverables

The main goal of our project is to develop an autonomous fire-fighting robot capable of detecting and extinguishing fires efficiently. This robot is designed to navigate its surroundings while avoiding obstacles and continuously monitor for fire in three directions—left, front, and right. Upon detecting a fire, the robot will move toward the source and activate its water-spraying mechanism until the fire is extinguished, ensuring a fully automated response without human intervention.

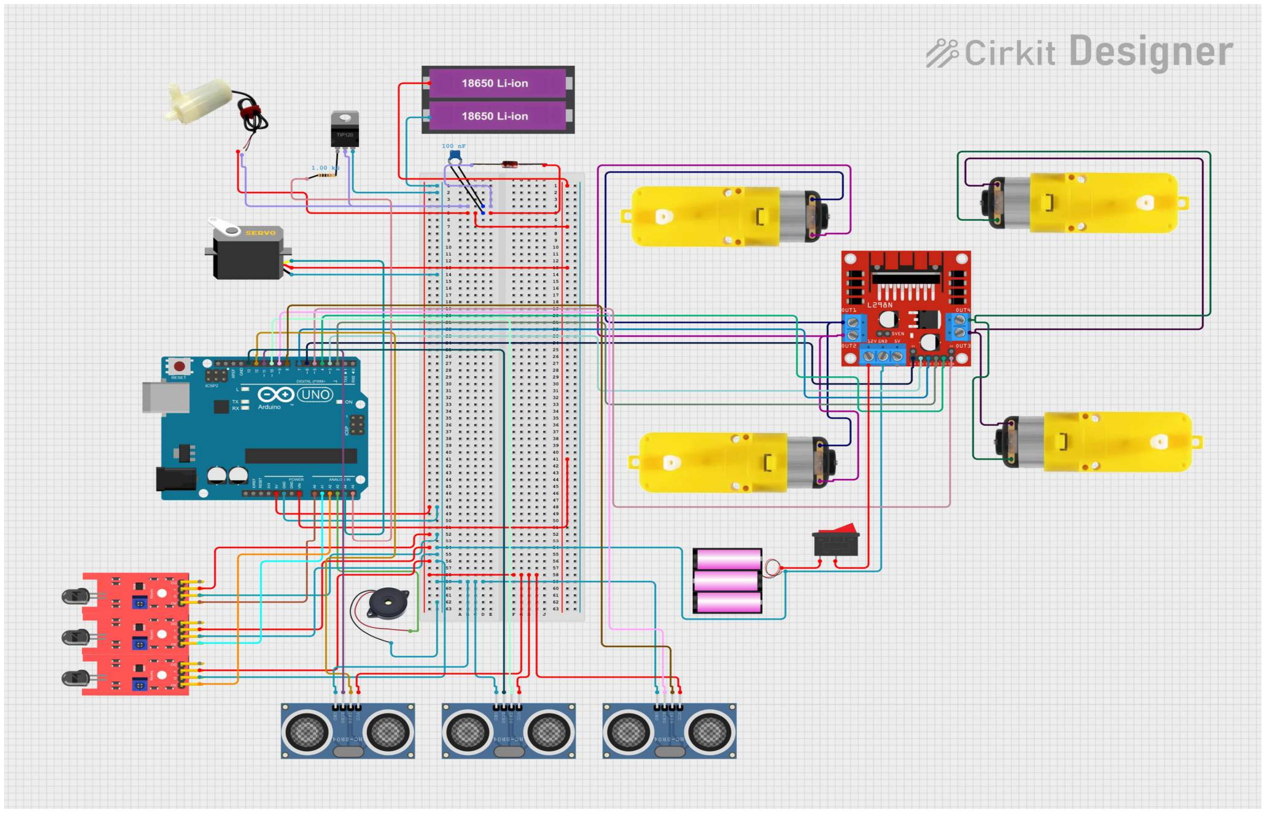
The robot employs ultrasonic sensors for obstacle detection, enabling it to make real-time decisions to avoid collisions while patrolling an area. Its fire detection system consists of infrared sensors that recognize the presence of flames. The servo-controlled water nozzle ensures precise targeting of the fire source, while a buzzer alarm alerts nearby individuals when a fire is detected.

In addition to firefighting, the robot's autonomous navigation and obstacle avoidance features make it adaptable for other household and industrial applications, such as automated surveillance, vacuum cleaning, and smart home assistance. With future improvements, it can be enhanced with AI-based decision-making, remote monitoring, and advanced fire detection technologies, increasing its efficiency and potential use in various environments.

Equipment Required

- Arduino Uno
- Ultrasonic Range Finder Sensor - HC - SR04(3)
- Motor Driver IC - L298N
- Robot Chassis
- Bottle s
- Servo
- Flame Sensors (3)
- Geared DC motors (4)
- Battery – 3.7V (6)
- Voltage Regulator
- Buzzer
- Wheels(4)
- TIP-122 Transistor
- 104 pf capacitor
- 1K Resister
- Solder-less Breadboard
- Water Pump
- Jumper wires

Circuit Diagram And Working



L298N Motor Driver

The L298N motor driver is used to control the movement of the robot. It allows you to drive the left and right motors, controlling both their speed and direction. For the left and right motors, the L298N requires specific pins for controlling the rotation and speed:

- enA (PWM) → Pin 6 on Arduino: This pin is responsible for controlling the speed of the left motor using PWM (Pulse Width Modulation). The higher the PWM value, the faster the motor will rotate.
- enB (PWM) → Pin 5 on Arduino: Similarly, this pin controls the speed of the right motor using PWM.
- in1 → Pin 3 on Arduino: This pin, along with in2, controls the direction of the left motor. If in1 is high and in2 is low, the motor will rotate in one direction; if in1 is low and in2 is high, the motor will rotate in the opposite direction.
- in2 → Pin 7 on Arduino: Works in conjunction with in1 to control the direction of the left motor.
- in3 → Pin 2 on Arduino: This pin, together with in4, controls the direction of the right motor. The same logic applies to in3 and in4 as it does for in1 and in2 for the left motor.
- in4 → Pin 4 on Arduino: Works with in3 to control the direction of the right motor.

The motors are powered by a separate battery source:

- Motor Power (4x3.7V) → Battery positive terminal.
- GND → Common ground shared by both the Arduino and the motor driver.

The L298N motor driver enables the robot to move forward, backward, and make turns by controlling the speed and direction of the motors.

Ultrasonic Sensors (HC-SR04)

The robot utilizes HC-SR04 ultrasonic sensors for obstacle detection. These sensors work by emitting a pulse and measuring how long it takes for the pulse to return after hitting an object. The sensor calculates the distance based on the time taken for the echo to return. The robot has three ultrasonic sensors:

- Front Ultrasonic Sensor:
  - Trig → Pin 12 on Arduino: This pin sends the trigger signal that activates the ultrasonic pulse.
  - Echo → Pin 11 on Arduino: This pin receives the returning pulse to calculate the distance.
- Left Ultrasonic Sensor:
  - Trig → Pin 13 on Arduino: The trigger pin for the left ultrasonic sensor.
  - Echo → Pin 10 on Arduino: The echo pin for the left ultrasonic sensor.
- Right Ultrasonic Sensor:
  - Trig → Pin 8 on Arduino: The trigger pin for the right ultrasonic sensor.
  - Echo → Pin 9 on Arduino: The echo pin for the right ultrasonic sensor.

Each of these sensors is connected to the Arduino using two pins: one for triggering the ultrasonic pulse (Trig) and one for receiving the returning signal (Echo). The Arduino uses the pulseIn() function to measure the time between the trigger pulse and the return echo, which is then converted into a distance in centimeters. The ultrasonic sensors allow the robot to detect obstacles and avoid collisions as it moves.

Fire Sensors (IR Flame Sensors)

The IR flame sensors are designed to detect the presence of fire by sensing infrared radiation emitted by flames. These sensors are placed on the robot to detect fires in three key positions: the front, left, and right. The sensors output an analog signal, which the Arduino reads to detect the intensity of the flame.

- Fire Right → Pin A0 on Arduino: This sensor detects fire on the right side of the robot.
- Fire Front → Pin A1 on Arduino: This sensor detects fire in front of the robot.
- Fire Left → Pin A2 on Arduino: This sensor detects fire on the left side of the robot.

Each fire sensor is connected to an analog input pin on the Arduino (A0, A1, A2) so the robot can read the intensity of infrared radiation. When the sensor detects a fire, the analog value is low, triggering the robot to take action, such as stopping and activating the water pump. The fire sensors are crucial in allowing the robot to identify and extinguish small fires autonomously.

Water Pump (DC Pump)

The DC water pump is responsible for spraying water to extinguish fires. The pump is connected to the Arduino through a transistor that acts as a switch. The transistor allows the Arduino to control the water pump without overloading the board, as the pump requires higher current than the Arduino can supply directly.

- Pump + → Battery positive terminal (connected to the same battery that powers the Arduino).
- Pump – → Collector/Drain of an NPN transistor.
- Transistor Base → Pin A3 on Arduino: The Arduino controls the transistor, thus controlling the pump.
- Transistor Emitter → GND (Ground).

The transistor acts as a switch, turning the pump on or off depending on the signal received from the Arduino. A flyback diode (e.g., 1N4007) is placed across the pump terminals to protect the circuit from voltage spikes caused by the inductive load. The water pump is activated when a fire is detected, spraying water to put out the flames.

Servo Motor (Nozzle)

The servo motor controls the angle of the water nozzle, allowing the robot to aim the water spray at the fire. This servo motor is connected to the Arduino as follows:

- Signal → Pin A4 on Arduino: The control signal sent to the servo to move it to a specific position.
- VCC → 5V on Arduino: Provides power to the servo motor.
- GND → GND on Arduino: Completes the circuit for the servo.

The servo motor is controlled using the Servo library, and the Arduino adjusts its angle to aim the water nozzle in the direction of the detected fire. The nozzle can rotate back and forth to extinguish the fire effectively.

Buzzer

The buzzer provides an audible alert when a fire is detected. It is connected as follows:

- Signal → Pin A5 on Arduino: The Arduino sends a control signal to turn the buzzer on or off.
- VCC → 5V on Arduino: Powers the buzzer.
- GND → GND on Arduino: Completes the circuit for the buzzer.

The buzzer is activated when the robot detects a fire, providing an audible alert to notify nearby people. This is especially useful in situations where immediate attention is required to prevent the fire from spreading.

Ground Connections

For the system to function correctly, all components need a common ground. The GND connections from various components are tied together:

- The Arduino GND connects to the motor driver GND, ultrasonic sensor GND, fire sensor GND, water pump GND, and buzzer GND through ground rail in breadboard

The common ground ensures that all components in the system share a unified reference point, allowing them to work together seamlessly.

Power Supply Overview

The robot is powered by two 3.7V batteries in series, providing 7.4V to the Arduino. The motors, servo motor, and water pump are powered by four 3.7V batteries, providing higher voltage for these components. The 5V components, such as the sensors and buzzer, are powered by the 5V output from the Arduino through voltage rail in breadboard. This ensures that each component receives the correct voltage to operate efficiently.

Algorithm

The code for this fire-fighting robot begins with the initialization of components in the setup() function. In this step, we set up the motor driver pins (enA, enB, in1, in2, in3, in4) as output to control the motors for movement. The ultrasonic sensor pins (frontTrig, frontEcho, leftTrig, leftEcho, rightTrig, rightEcho) are also configured to measure distance for obstacle avoidance. The fire sensors (fireRight, fireFront, fireLeft) are set as inputs to detect flames. Additionally, the water pump and the buzzer are set as outputs for activating the extinguishing process and alerting when a fire is detected. The servo motor, controlling the water spray nozzle, is attached to its pin (servoPin) and initialized to its default position (75°). The Serial.begin(9600) function enables communication with the serial monitor to output sensor readings and other messages.

In the loop() function, the robot constantly measures distances using the ultrasonic sensors with the getDistance() function, which emits a pulse and measures the duration it takes for the pulse to return. This duration is then converted to a distance in centimeters. If the robot detects an obstacle (distance < 20 cm), it stops moving, backs up briefly, and then turns either left or right based on the available space. The robot uses moveForward(), moveBackward(), turnLeft(), and turnRight() functions to control the movement of the motors, ensuring it navigates without colliding with objects. These functions set the appropriate motor pins to either rotate the motors forward or backward, controlling the robot's movement.

The robot also continuously monitors the fire sensors (fireRight, fireFront, fireLeft) for fire detection. If any sensor reading goes below the threshold (indicating the presence of fire), the robot will immediately stop moving and call the extinguishFire() function. This function stops the robot and activates the water pump. The servo motor is then commanded to sweep the water nozzle back and forth between 0° and 150° with small delays in between. The nozzle continues to spray water until the sensor readings indicate that the fire is extinguished. Once the fire is extinguished, the water pump is turned off, and the servo is returned to its default position (75°). During the fire detection, the digitalWrite(buzzer, HIGH) function is used to sound an alarm, indicating that the fire has been detected and the robot is working on extinguishing it.

The robot then resumes its normal operation of navigation by continuously measuring distances and checking for obstacles and fires. If no fire is detected, it continues to move forward, avoiding obstacles using the ultrasonic sensors and adjusting its path accordingly. This cycle repeats, ensuring the robot autonomously detects fires and avoids obstacles while performing the necessary actions to extinguish any detected flames.

This explanation outlines the core functions of the robot and how they interrelate to create an autonomous system capable of detecting and extinguishing fires while navigating around obstacles. The functions moveForward(), moveBackward(), turnLeft(), turnRight() control the movement of the robot, getDistance() measures distances, and extinguishFire() handles the fire extinguishing process. The fire sensors and ultrasonic sensors work in tandem to detect environmental changes and trigger appropriate actions, such as stopping to spray water or changing direction when an obstacle is encountered.

Source Code

```
#include <Servo.h>

// Motor driver L298N pins
#define enA 6
#define in1 3
#define in2 7
#define in3 2
#define in4 4
#define enB 5

// Ultrasonic sensor pins
#define frontTrig 12
#define frontEcho 11
#define leftTrig 13
#define leftEcho 10
#define rightTrig 8
#define rightEcho 9

// Fire sensor pins
#define fireRight A0
#define fireFront A1
#define fireLeft A2

// Other components
#define waterPump A5
#define servoPin A4
#define buzzer A3
int threshold = 100;
Servo nozzle;  // Servo motor for water spray

// Function to measure distance using ultrasonic sensor
long getDistance(int trig, int echo) {
   digitalWrite(trig, LOW);
   delayMicroseconds(2);
   digitalWrite(trig, HIGH);
   delayMicroseconds(10);
   digitalWrite(trig, LOW);
   long duration = pulseIn(echo, HIGH);
   return duration * 0.034 / 2;  // Convert to cm
}

// Function to move forward
```

```
void moveForward() {
  analogWrite(enA, 100); // Motor speed
  analogWrite(enB, 100);
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
}

// Function to move backward
void moveBackward() {
  analogWrite(enA, 170);
  analogWrite(enB, 170);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}

// Function to turn right
void turnRight() {
  analogWrite(enA, 170);
  analogWrite(enB, 170);
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
}

// Function to turn left
void turnLeft() {
  analogWrite(enA, 170);
  analogWrite(enB, 170);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
}

// Function to stop movement
void stopMoving() {
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
```

```
}

// Function to extinguish fire
void extinguishFire() {
  stopMoving();
  digitalWrite(waterPump, HIGH);
  while(analogRead(fireFront)<threshold||analogRead(fireRight)<threshold||analogRead(fireLeft)<thres
hold){
  int pos;
  Serial.print("Fire Right: ");
  Serial.print(analogRead(fireLeft));
  Serial.print("  Fire Front: ");
  Serial.print(analogRead(fireFront));
  Serial.print("  Fire Left: ");
  Serial.println(analogRead(fireRight));
  for (pos = 0; pos <= 150; pos += 1) { // goes from 0 degrees to 180 degrees
  // in steps of 1 degree
  nozzle.write(pos);              // tell servo to go to position in variable 'pos'
  delay(10);                      // waits 10 ms for the servo to reach the position
 }
 for (pos = 150; pos >= 0; pos -= 1) { // goes from 150 degrees to 0 degrees
  nozzle.write(pos);              // tell servo to go to position in variable 'pos'
  delay(10);                      // waits 10 ms for the servo to reach the position
 }}
 nozzle.write(75);
  digitalWrite(waterPump, LOW);
}

void setup() {
  // Motor driver setup
  pinMode(enA, OUTPUT);\


  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
  pinMode(enB, OUTPUT);

  // Ultrasonic sensor setup
  pinMode(frontTrig, OUTPUT);
  pinMode(frontEcho, INPUT);
  pinMode(leftTrig, OUTPUT);
  pinMode(leftEcho, INPUT);
  pinMode(rightTrig, OUTPUT);
  pinMode(rightEcho, INPUT);

  // Fire sensor setup
  pinMode(fireRight, INPUT);
  pinMode(fireFront, INPUT);
  pinMode(fireLeft, INPUT);

  // Other components setup
  pinMode(buzzer, OUTPUT);
  pinMode(waterPump, OUTPUT);
  nozzle.attach(servoPin);
  nozzle.write(75); // Default position
  digitalWrite(waterPump, LOW);
  digitalWrite(buzzer, LOW);
  Serial.begin(9600);
}

void loop() {
  long frontDist = getDistance(frontTrig, frontEcho);
  long leftDist = getDistance(leftTrig, leftEcho);
  long rightDist = getDistance(rightTrig, rightEcho);

  int fireR = analogRead(fireRight);
  int fireF = analogRead(fireFront);
  int fireL = analogRead(fireLeft);

  // Check for fire
  if (fireF < 200 || fireR < 200 || fireL < 200) {
    Serial.println("Fire Detected!");
    moveForward();
    if (fireF < 50 || fireR < 50 || fireL < 50) {
      digitalWrite(buzzer, HIGH); // Buzzer ON
      delay(500);
      digitalWrite(buzzer, LOW); // Buzzer OFF
      extinguishFire();
      return; // Resume navigation after extinguishing
    }
  } else {
    // Obstacle Avoidance Logic
    if (frontDist < 20) {  // If an object is too close in front
      stopMoving();
      moveBackward();  // Move back briefly
      delay(200);  // Reduced delay (was 500ms)
      stopMoving();
      if (leftDist > rightDist) {
        Serial.println("Turning Left");
        turnLeft();
        delay(800);
      } else {
        Serial.println("Turning Right");
        turnRight();
        delay(800);
      }
    } else if (leftDist < 20) {  // Avoid left obstacle
      Serial.println("Obstacle on Left, Turning Right");
      turnRight();
      delay(800);
    } else if (rightDist < 20) {  // Avoid right obstacle
      Serial.println("Obstacle on Right, Turning Left");
      turnLeft();
      delay(800);
    } else {
      Serial.println("Moving Forward");
      moveForward();
    }
  }

  delay(100);
}
```

Problems During The Project

- Poor Quality Flame Sensors:
  At the start, we faced issues with the flame sensors not being able to properly detect fire. The sensors we received were of poor quality, which led to inaccurate fire detection. This significantly hindered our testing and required us to rework the sensor setup multiple times before achieving reliable results.
- Faulty Ultrasonic Sensor:
  One of our ultrasonic sensors malfunctioned and displayed a distance of -1 in the serial monitor, rendering it completely useless. This issue caused problems in navigation, as the robot was unable to accurately gauge the distance to obstacles on one side, affecting its obstacle avoidance behavior.
- Broken Servo Motor:
  During the course of the project, one of our team members accidentally broke the servo motor. This motor controlled the water spray nozzle, and its failure meant that we could no longer properly direct the water to extinguish the fire. This required us to replace the motor and re-calibrate the entire system.
- Inadequate Power Supply for L298N Motor Driver:
  We initially used a 9V battery to power the L298N motor driver, but this wasn't powerful enough to provide the necessary current to run the motors efficiently. As a result, the car struggled to move and lacked the required torque to drive the robot, causing delays in project progress and requiring us to switch to a better power source.
- Rapid Battery Drain:
  The lithium-ion battery we used for the project drained very quickly, especially when the motors, water pump, and sensors were all in use. This significantly limited the operational time of the robot and required frequent recharging, which disrupted testing and prolonged the development process.
- Incorrect Fire Detection Behavior:
  Our code had a bug where it mistakenly treated fire as an obstacle. As a result, the robot tried to avoid the fire by turning away, rather than extinguishing it. This misbehavior required us to adjust the fire detection logic and refine the sensor readings to ensure the robot correctly identified and acted on fire.
- Short Circuit Risk:
  At one point during the project, we encountered a short circuit due to improper wiring. This issue nearly damaged the entire system, including the Arduino and other connected components. Thankfully, we identified the problem in time and fixed the wiring, preventing any significant damage to the project.

Limitations

- LimitedFire Detection Range:
  The flame sensors used in the project had a short detection range, which limited the robot’s ability to detect fire from a distance. This was problematic for larger spaces, as the robot could only react to nearby fires, delaying its response time.
- Power Consumption Issues:
  The robot's power consumption was high, especially with the motors and water pump in use. The lithium-ion battery drained quickly, limiting the robot's operational time and requiring frequent recharging, which is impractical for long-term use.
- Obstacle Avoidance Challenges:
  The ultrasonic sensors sometimes failed to detect objects accurately, especially in cluttered or complex environments. This led to occasional collisions or missed obstacles, reducing the robot’s effectiveness in real-world conditions.
- Speed and Agility:
  The robot was slow in movement, limiting its efficiency in emergency situations. The motor power and design restricted its ability to quickly navigate the environment or react promptly to fire and obstacles.
- Limited Fire Extinguishing Capacity
  The water pump's pressure was insufficient for extinguishing larger fires. The robot could only handle small fires, like those caused by candles, and its nozzle movement range was limited, reducing its fire-extinguishing efficiency.
- Environmental Sensitivity:
  The flame sensors were prone to false positives from other light sources, such as sunlight or bright lights, leading to incorrect fire detection. This affected the robot's accuracy in various lighting conditions.
- Simplistic Fire Detection Logic:
  The fire detection system was basic, triggering the water pump whenever a fire was detected, regardless of its size or danger. The robot lacked the ability to assess fire severity or prioritize larger fires.
- No Real-time Control or Feedback:
  There was no interface for real-time control or feedback, making it difficult for users to intervene or monitor the robot’s status during operation.
- Limited Mobility:
  The robot’s design limited its ability to navigate through tight spaces or uneven terrain. It struggled with larger obstacles, which hindered its ability to operate in diverse real-world environments.

Conclusion And Discussion

This project has been a highly enriching and valuable learning experience, offering us the chance to explore various aspects of robotics, fire safety, and automation. The development of an autonomous fire-extinguishing robot allowed us to work with multiple components such as flame sensors, ultrasonic sensors, motors, and Arduino programming. Despite facing several obstacles along the way, including faulty sensors, power supply issues, and code bugs, we were able to overcome these challenges and create a functional prototype capable of detecting and extinguishing fire automatically.

Throughout the project, we encountered several technical problems that tested our problem-solving abilities. For instance, we initially struggled with poor-quality flame sensors that failed to detect fire properly. Additionally, one of our ultrasonic sensors malfunctioned, showing -1 distance on the serial monitor, and one of the servo motors was broken during the process. Another critical issue was related to the power supply, where the 9V battery used for the L298N motor driver wasn't powerful enough to run the car efficiently, and the lithium-ion battery we used drained quickly. We also faced issues with our code detecting fire as an obstacle and trying to avoid it. A short circuit nearly destroyed the project, but we were fortunate to spot the issue before it caused more damage.

Despite these setbacks, we learned valuable lessons about troubleshooting, wiring, and modifying our system. The iterative process of testing, debugging, and adjusting our approach allowed us to refine the system and improve the performance of the robot. By the end of the project, we successfully developed a robot that could move around, detect fire, and extinguish it autonomously, marking a significant milestone in our understanding of robotics and automation.

This project also highlighted the importance of using high-quality components, ensuring a reliable power supply, and fine-tuning code to avoid misinterpretations of sensor data. Going forward, improvements could be made by incorporating better sensors, a more efficient power management system, and enhanced fire detection algorithms. The concept of autonomous fire-extinguishing robots holds great potential, particularly in household and industrial safety, and with further development, this technology could play an important role in fire prevention and management.

In conclusion, although the project presented several challenges, it also provided invaluable learning opportunities and insights into the practical application of robotics. We are proud of our final prototype and excited about the potential of such technology in real-world fire safety solutions. The project has not only improved our technical skills but also demonstrated the importance of perseverance and teamwork in overcoming difficulties and achieving success.

References

- GitHub Repository:
  - Suthar, P. (2021). Automatic Fire Fighting Robot. GitHub. Retrieved from https://github.com/pdsuthar10/Automatic-Fire-Fighting-Robot
- YouTube Video Reference:
  - Automatic Fire Fighting Robot | Arduino Based Project. Retrieved from https://www.youtube.com/watch?v=jsvAL9ogFBw&t=3s
  - How dc gear motor works with arduino and l298n. Retrieved from https://www.youtube.com/watch?v=GPVC84D5ULw&t=449s
- Arduino Documentation:
  - Arduino. (n.d.). Arduino Documentation. Retrieved from https://docs.arduino.cc/