iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** X **Month of publication:** October 2026

DOI: https://doi.org/10.22214/ijraset.2026.85516

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

Design and Development of an Arduino-Based Autonomous Fire Fighter Robot

Aakriti Kumari[1], Vishnupriya Kumari[2], Arpita Kumari[3], Chhoti Kumari[4], Ankita Sinha[5], Mansha Kumari[6], Er. Shwetabh[7]

[1, 2, 3, 4, 5, 6]*Diploma in Electrical Engineering, Government Women's Polytechnic, Patna State Board of Technical Education Bihar, Patna*

[7]*Lecturer, Department of Electrical Engineering, Government Women's Polytechnic, Patna*

Abstract: ***Fire incidents can expose people to high temperatures, smoke, and other hazards. This paper presents the design and operating principle of a compact Arduino-based fire fighter robot developed as an educational prototype for flame detection, directional movement, and water spraying. The project uses three flame sensors positioned to observe the left, front, and right directions; an Arduino UNO as the main controller; an L298 motor driver; four BO geared DC motors; a servo-operated nozzle; and a small water pump supplied from an onboard tank. Sensor readings are processed by the controller to select a movement response. When a flame is detected, the robot stops or turns according to the programmed thresholds, positions the nozzle, and activates the pump. Related work has examined Arduino-based firefighting robots, multi-robot assistance, and sensor-fusion-based fire detection [1], [7]–[9], [11]. The controller and motor-driver choices are supported by the relevant hardware documentation [2], [3], while the sensor-reading method uses Arduino's analog input functionality [12]. No quantitative experimental results were provided; therefore, detection range, response time, extinguishing time, success rate, and battery endurance are not claimed. The robot is an educational demonstrator, not a certified firefighting appliance, and must not be used on structural, electrical, chemical, or spreading fires.***

Index Terms: ***Fire fighter robot, Arduino UNO, flame sensor, L298 motor driver, autonomous mobile robot, water pump, servo motor, embedded systems.***

I. INTRODUCTION

Fire accidents can cause serious damage to human life, property, and the environment. Entering a hazardous area to inspect or approach a fire can expose people to heat, smoke, and unstable surroundings. Mobile robotic systems provide a way to demonstrate remote sensing and limited intervention while keeping people farther from immediate hazards [1], [7], [8], [10].

A fire fighter robot combines sensors, a controller, a movement mechanism, and an action mechanism. In this project, flame sensors detect infrared radiation associated with a flame and send signals to an Arduino UNO. The Arduino processes these signals and controls the movement of a four-wheel platform through an L298 motor driver [2], [3]. A small servo motor adjusts the water nozzle, while a pump delivers water from a mounted container. Similar systems incorporating flame sensors, mobile platforms, and pump-based fire suppression have been discussed in previous firefighting robot studies [7], [8], [11].

The project is designed as a low-cost educational prototype that demonstrates sensor input, programmed decision-making, motor control, and a basic suppression mechanism. It provides practical learning in embedded programming, circuit connections, power distribution, and robotics. Arduino documentation explains the UNO's input/output capabilities [2], while the L298 datasheet describes its function as a dual full-bridge driver for motor loads [3]. However, sensor-based systems may be affected by calibration and environmental conditions. Research on multi-sensor fire detection and flame localization explores methods to improve detection reliability [9]–[11]. The present prototype does not include advanced navigation, obstacle avoidance, thermal imaging, or remote telemetry.

Manual firefighting can be dangerous when people must approach a fire source. Therefore, this project explores whether a compact mobile platform can detect the approximate direction of a small flame and perform a basic water-spraying response using readily available electronic components. The paper documents the prototype, its hardware components, circuit connections, operating principles, and original source code. It also discusses the system's limitations and proposes an evaluation plan. Since no experimental dataset was provided, no claims are made regarding measured safety or firefighting effectiveness.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

II. OBJECTIVES AND RELATED WORK

The main objective of the Fire Fighter Robot project is to develop a low-cost robotic prototype capable of detecting a flame and performing a basic water-spraying response. The system uses three directional flame-sensor modules to estimate whether a flame is located to the left, in front, or to the right of the robot. An Arduino UNO processes the sensor signals and controls a four-wheel platform through an L298 motor driver [2], [3]. A servo motor adjusts the direction of the water nozzle, while a small pump supplies water for the suppression mechanism. The project also aims to demonstrate embedded programming, sensor integration, motor control, and basic robotic automation.

Previous firefighting robot designs have combined sensors, microcontrollers, motor drivers, mobile platforms, and suppression mechanisms. Prakash et al. reviewed firefighting robot architectures and discussed sensing, control, and suppression technologies [1]. Arduino-based projects have also demonstrated the integration of flame sensors, mobile chassis systems, and water pumps [4]–[8]. Research on flame detection and mobile firefighting platforms explores different approaches to improving fire detection and response [7]–[9]. The Arduino UNO provides digital and analog interfaces for connecting sensors and actuators [2], [11].

The L298 is a dual full-bridge motor driver designed to control loads such as brushed DC motors [3]. It provides an interface between microcontroller control signals and the motor power circuit. However, proper power-supply selection, suitable wiring, and current and temperature considerations are necessary for reliable operation.

The proposed robot uses a threshold-based control method in which sensor readings are repeatedly compared with predefined values. Depending on the readings, the robot can move forward, turn, stop, or activate the pump while adjusting the nozzle. Although this approach is simple to implement, its performance may be affected by sensor calibration, ambient infrared sources, flame distance, and environmental conditions.

The component arrangement and pin assignments presented in this paper are based on the supplied project thesis. Since no experimental dataset was provided, detection range, response time, success rate, and endurance have not been reported as measured results. The system is therefore presented as an educational prototype, with further testing required to evaluate its performance and identify possible improvements.

III. COMPONENTS AND SPECIFICATIONS

The prototype is built from readily available electronic and mechanical parts [2], [3], [7], [8]. The quantities and functions below follow the supplied project thesis; exact ratings should be confirmed from the physical modules and their datasheets.

No.	Component	Quantity	Function in the prototype
1	Arduino UNO	1	Reads sensors and coordinates movement, servo and pump commands.
2	4-pin flame sensor module	3	Detects flame-related infrared radiation from left, front and right directions.
3	L298 motor driver	1	Controls direction and speed signals for the drive motors.
4	BO geared DC motor	4	Provides wheel movement.
5	Wheels	4	Allow the chassis to move.
6	Mini servo motor	1	Changes the direction of the nozzle.
7	DC water pump	1	Pumps water from the tank through the outlet.
8	Water tank/bottle	1	Stores water for the demonstration.
9	Breadboard	1	Provides a convenient platform for low-current interconnections.
10	Battery / power supply	As required	Powers controller and actuators as appropriate.
11	Robot chassis	1	Supports and secures the components.
12	Jumper wires	As required	Provide electrical interconnections.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

A. Arduino UNO

The Arduino UNO is the central controller [2], [11]. It provides analog inputs for reading sensor values and digital outputs for controlling the motor driver and other devices. Arduino documentation describes the UNO R3 as an ATmega328P-based board with 14 digital input/output pins and six analog inputs [2].

B. Flame Sensors

Three flame sensor modules are mounted to view different directions. Flame-detection performance can depend on sensor characteristics and calibration [9], [10], [12]. In the source thesis, the right sensor is connected to A0, the front sensor to A1, and the left sensor to A2. The program reads these analog values and compares them with fixed thresholds.

C. Motor Driver and Motors

The L298 driver receives logic-level direction signals from the Arduino and supplies drive current from the motor power rail [3]. Four BO motors provide movement. The supplied program uses two enable pins for motor speed and four input pins for motor direction [3].

IV. SYSTEM ARCHITECTURE

The system can be separated into sensing, decision-making, mobility, and suppression functions, a common structure in mobile fire-detection platforms [7]–[10]. The sensor modules provide directional information. The Arduino executes the programmed decision logic. The L298 driver operates the drive motors, while the servo and pump are used for water delivery.

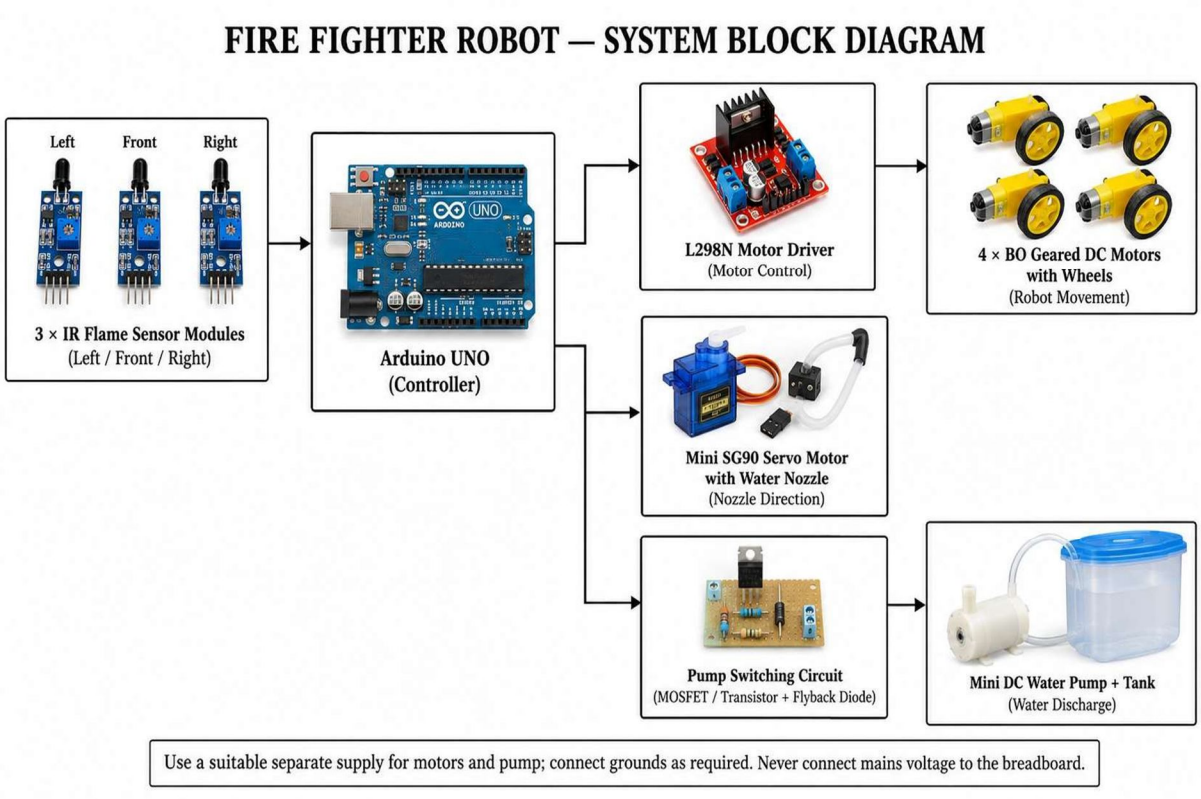


Fig. 1. Functional block diagram of the fire fighter robot.

A. Sensing Layer

The three flame sensors monitor the left, front, and right directions. Multi-sensor fire-detection studies highlight the importance of calibration and combining sensor information [9], [10], [12]. Their outputs are read using analogRead(). The code assumes that lower sensor readings correspond to a stronger flame indication at the configured thresholds; this polarity and threshold behaviour must be checked on the actual modules.

B. Processing Layer

The Arduino UNO reads the sensor channels, prints values to the serial monitor, and selects a branch of the control logic using analog input readings [2], [11]. The decision is threshold-based rather than based on machine learning or image processing.

C. Mobility Layer

The L298 motor driver controls two motor channels, consistent with the device's dual full-bridge design [3]. Each channel can control the direction of a motor pair, depending on the actual wiring. The code defines forward, backward, left-turn, right-turn, and stop functions.

D. Suppression Layer

A mini servo motor changes the nozzle angle. The pump must be switched through a suitably rated driver rather than directly from a microcontroller I/O pin [2], [3]. A water pump moves water from the mounted bottle or tank through a pipe. The pump is controlled by an output signal reported as A5 in the source thesis. A suitable external switching circuit must be used if the pump current exceeds the safe capability of a controller output.

E. Power Distribution

The motor and pump loads must not be powered directly from an Arduino I/O pin. Use a power supply suited to each component, verify polarity and current ratings, and provide a common ground where required by the circuit. Keep water and wet tubing away from exposed electronics.

V. CIRCUIT DIAGRAM AND PIN CONNECTIONS

Figure 2 reproduces the circuit diagram included in the supplied project thesis. The pin mapping and power connections should be checked against the component documentation before operation [2], [3], [11]. The drawing shows the Arduino UNO, three flame sensors, motor driver, four DC motors, servo, pump, and power connections. The exact wiring should be checked against the assembled robot before applying power.

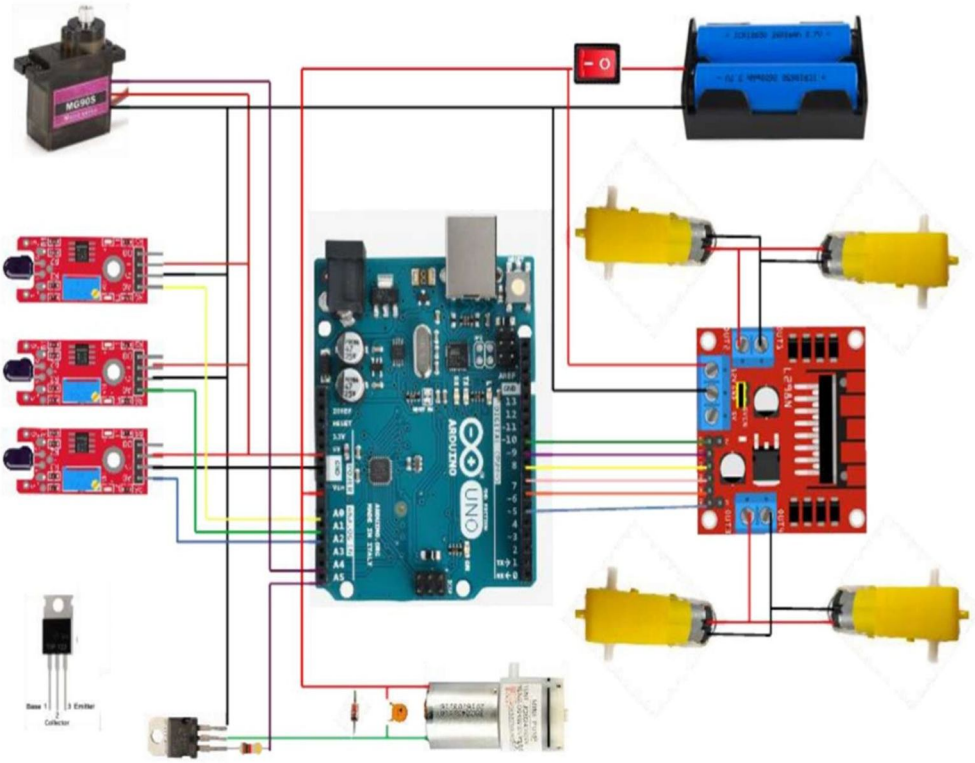


Fig. 2. Circuit diagram reproduced from the supplied Fire Fighter Robot thesis.

Device / signal	Arduino pin or connection	Purpose
Right flame sensor	A0	Right-side flame input
Front flame sensor	A1	Forward flame input
Left flame sensor	A2	Left-side flame input
L298 IN1	D9	Motor direction control
L298 IN2	D8	Motor direction control
L298 IN3	D7	Motor direction control
L298 IN4	D6	Motor direction control
L298 ENA	D10	Motor channel A enable / PWM
L298 ENB	D5	Motor channel B enable / PWM
Servo signal	A4 (as reported)	Nozzle angle control
Pump control	A5 (as reported)	Pump switching signal

VI. WORKING PRINCIPLE AND CONTROL ALGORITHM

The robot follows a repeated sense–decide–act cycle, a basic control pattern used in sensor-driven mobile robots [7]–[10]. When powered on, it initializes the sensor and actuator pins and performs a short servo movement. It then reads the three flame sensor values and prints them to the serial monitor. The values are compared against thresholds in the program. Depending on which sensor condition is met, the robot stops and aims the nozzle while running the pump, or turns/moves with the pump switched off.

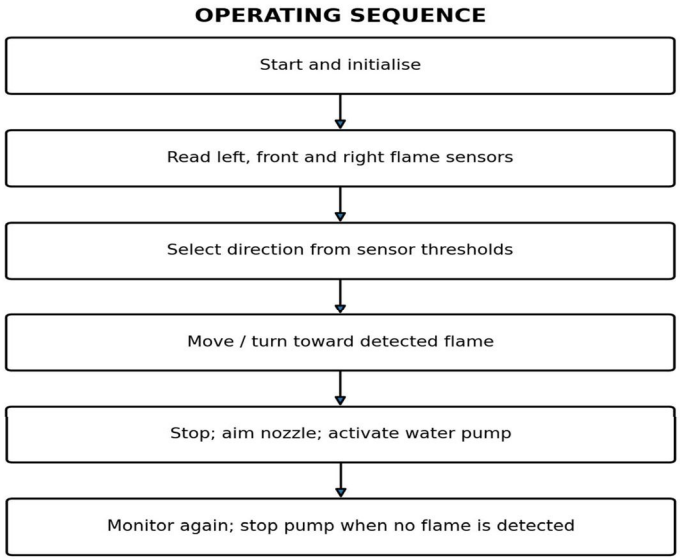


Fig. 3. Simplified operating sequence of the fire fighter robot.

A. Sensor Decision

The program reads the right, front, and left sensors into variables s1, s2, and s3. The threshold values should be calibrated for the actual modules and conditions [9]–[12]. It checks s1 first, then s2, then s3. This order matters: if multiple sensors meet their thresholds at the same time, the first matching condition takes priority.

B. *Movement Response*

The functions forward(), backward(), turnRight(), turnLeft(), and Stop() set the motor-driver input pins. In the original source code, the function is spelled “forword()”. That spelling is preserved in the reproduced program because the user selected the original code without editing.

C. *Nozzle and Pump*

The program calls servoPulse() to generate pulses for the servo. It moves the nozzle across selected angle ranges while the pump output is enabled. When the logic enters movement branches, it switches the pump off before moving or turning.

D. *Operational Limitations*

The source program does not establish a measured safe approach distance and does not document obstacle avoidance or a manual emergency stop. Thresholds are hard-coded. The robot should be tested in a controlled, supervised setting only; water should never be sprayed onto energized electrical equipment or unsuitable fire classes.

VII. PROTOTYPE ASSEMBLY AND IMPLEMENTATION

The robot is assembled on a compact wheeled chassis. The Arduino UNO and motor driver are mounted on the platform, while the flame sensors face the directions that the robot is intended to monitor. The water container is fixed vertically, and the pump feeds a pipe or nozzle positioned by the servo motor.



Fig. 4. Photograph of the assembled fire fighter robot prototype supplied by the user.

E. *Mechanical Assembly*

The chassis must support the weight of the controller, motor driver, motors, power supply, water tank, pump, and nozzle; electrical ratings and mechanical stability should be checked before testing [2], [3]. The water tank and pump should be secured so that movement does not loosen the pipe or wiring. The servo should be able to move through its programmed range without pulling on wires or tubing.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

F. Electrical Assembly

Before powering the robot, check the sensor orientation, motor polarity, enable-pin wiring, pump switching stage, and ground connections. The pump must be driven through a suitable transistor or relay module rated for its voltage and current, with appropriate protection against inductive transients where needed.

G. Software Implementation

The source program uses the Arduino language and standard functions such as pinMode(), analogRead(), analogWrite(), digitalWrite(), delay(), and Serial.print(). It does not require an external library in the reproduced listing. Because the program was transcribed from the project thesis, it is presented as the original source rather than as a verified, compiled release.

H. Assembly Checks

Recommended checks include confirming that each sensor responds to a safe test source, that the wheels rotate in the intended directions with the wheels lifted clear of the floor, that the servo moves without binding, and that the pump is controlled by its driver circuit. Do not run the pump dry for long periods.

VIII. ORIGINAL ARDUINO PROGRAM (PART I)

```
#define enA 10//Enable1 L298 Pin enA #define in1 9 //Motor1 L298 Pin in1 #define in2 8 //Motor1 L298 Pin in2 #define in3 7
//Motor2 L298 Pin in3 #define in4 6 //Motor2 L298 Pin in4 #define enB 5 //Enable2 L298 Pin enB #define ir_R A0
#define ir_F A1 #define ir_L A2 #define servo A4 #define pump A5
int Speed = 160; // Write The Duty Cycle 0 to 255 Enable for Motor Speed int s1, s2, s3;
void setup(){ // put your setup code here, to run once Serial.begin(9600); // start serial communication at 9600bps pinMode(ir_R,
 INPUT);// declare fire sensor pin as input pinMode(ir_F, INPUT);// declare fire sensor pin as input pinMode(ir_L, INPUT);//
 declare fire sensor pin as input pinMode(enA, OUTPUT); // declare as output for L298 Pin enA pinMode(in1, OUTPUT); //
 declare as output for L298 Pin in1 pinMode(in2, OUTPUT); // declare as output for L298 Pin in2 pinMode(in3, OUTPUT); //
 declare as output for L298 Pin in3 pinMode(in4, OUTPUT); // declare as output for L298 Pin in4 pinMode(enB, OUTPUT); //
 declare as output for L298 Pin enB pinMode(servo, OUTPUT);
 pinMode(pump, OUTPUT);
 for (int angle = 90; angle <= 140; angle += 5) { servoPulse(servo, angle);
 }
 for (int angle = 140; angle >= 40; angle -= 5) { servoPulse(servo, angle);
 }
 for (int angle = 40; angle <= 95; angle += 5) { servoPulse(servo, angle);
 }
 analogWrite(enA, Speed); // Write The Duty Cycle 0 to 255 Enable Pin A for Motor1 Speed analogWrite(enB, Speed); // Write
 The Duty Cycle 0 to 255 Enable Pin B for Motor2 Speed delay(500);
}
void loop(){
 s1 = analogRead(ir_R); s2 = analogRead(ir_F); s3 = analogRead(ir_L);
 //=========================================================
 // Auto Control
 //=========================================================
 Serial.print(s1); Serial.print("\\t"); Serial.print(s2); Serial.print("\\t"); Serial.println(s3); delay(50); if(s1<250){
  Stop(); digitalWrite(pump, 1);
  for(int angle = 90; angle >= 40; angle -= 3){ servoPulse(servo, angle);
  }
  for(int angle = 40; angle <= 90; angle += 3){ servoPulse(servo, angle);
  }
 }
```

©IJRASET: All Rights are Reserved

IX. ORIGINAL ARDUINO PROGRAM (PART II)

```
  else if(s2<350){ Stop(); digitalWrite(pump, 1);
   for(int angle = 90; angle <= 140; angle += 3){ servoPulse(servo, angle);
   }
   for(int angle = 140; angle >= 40; angle -= 3){ servoPulse(servo, angle);
   }
   for(int angle = 40; angle <= 90; angle += 3){ servoPulse(servo, angle);
   }
  }
  else if(s3<250){ Stop();
   digitalWrite(pump, 1);
   for(int angle = 90; angle <= 140; angle += 3){ servoPulse(servo, angle);
   }
   for(int angle = 140; angle >= 90; angle -= 3){ servoPulse(servo, angle);
   }
  }
  else if(s1>=251 && s1<=700){ digitalWrite(pump, 0); backword();
   delay(100); turnRight(); delay(200);
  }
  else if(s2>=251 && s2<=800){ digitalWrite(pump, 0); forword();
  }
  else if(s3>=251 && s3<=700){ digitalWrite(pump, 0); backword();
   delay(100); turnLeft(); delay(200);
  }else{ digitalWrite(pump, 0); Stop();
  }
  delay(10);
}
void servoPulse (int pin, int angle){
  int pwm = (angle*11) + 500; // Convert angle to microseconds digitalWrite(pin, HIGH);
  delayMicroseconds(pwm); digitalWrite(pin, LOW);  delay(50); // Refresh cycle of servo
}
void forword(){ //forword
  digitalWrite(in1, HIGH); //Right Motor forword Pin digitalWrite(in2, LOW); //Right Motor backword Pin digitalWrite(in3,
  LOW); //Left Motor backword Pin digitalWrite(in4, HIGH); //Left Motor forword Pin
}
void backword(){ //backword
  digitalWrite(in1, LOW); //Right Motor forword Pin digitalWrite(in2, HIGH); //Right Motor backword Pin digitalWrite(in3,
  HIGH); //Left Motor backword Pin digitalWrite(in4, LOW); //Left Motor forword Pin
}
void turnRight(){ //turnRight
  digitalWrite(in1, LOW); //Right Motor forword Pin digitalWrite(in2, HIGH); //Right Motor backword Pin digitalWrite(in3,
  LOW); //Left Motor backword Pin digitalWrite(in4, HIGH); //Left Motor forword Pin
}
void turnLeft(){ //turnLeft
  digitalWrite(in1, HIGH); //Right Motor forword Pin digitalWrite(in2, LOW); //Right Motor backword Pin digitalWrite(in3,
  HIGH); //Left Motor backword Pin digitalWrite(in4, LOW); //Left Motor forword Pin
}
void Stop(){ //stop
  digitalWrite(in1, LOW); //Right Motor forword Pin digitalWrite(in2, LOW); //Right Motor backword Pin digitalWrite(in3,
  LOW); //Left Motor backword Pin digitalWrite(in4, LOW); //Left Motor forword Pin
```

X. EVALUATION PLAN AND DISCUSSION

The supplied thesis describes the robot's intended operation but does not provide a reproducible experimental dataset. Published fire-detection robot studies report explicit test conditions and measured performance, which would be useful models for future evaluation [9], [10]. No numerical performance results are reported in this paper. A future evaluation should measure the following variables under controlled, repeatable conditions.

Metric	Suggested method
Detection success rate	Number of successful flame detections divided by total trials × 100%.
Direction accuracy	Correct left/front/right classifications divided by total detections × 100%.
Response time	Elapsed time between presenting the flame and the first appropriate robot response.
Extinguishing time	Time from water discharge to flame extinction, only in a safe, supervised test.
Water consumption	Tank volume before and after each trial.
Battery endurance	Operating time under a defined battery and load condition.
False activation	Unnecessary movement or pump activation per test duration.

The prototype should be evaluated on a stable, non-combustible surface to assess its sensing, movement, and water-spraying functions. Initially, the flame sensors, motor driver, servo motor, and pump should be checked individually to verify their proper operation. The robot's movement should be tested with the wheels raised before conducting any controlled demonstration. During testing, flame direction, distance, and angle should be varied while recording sensor readings, movement responses, and power conditions through the serial monitor. Each test condition should be repeated to identify inconsistencies and improve the reliability of the observations. The main advantage of the prototype is its integration of low-cost sensing, embedded control, mobility, and a pump-based response using readily available electronic components [7], [8]. However, its performance may be affected by sensor thresholds, changing illumination, flame distance, limited water capacity, battery life, nozzle alignment, water flow, and environmental conditions such as airflow. The absence of documented obstacle avoidance and advanced navigation also limits its practical application. Further improvements could include additional sensors, obstacle detection, better power management, and wireless monitoring. The prototype is intended for educational and demonstration purposes and is not a certified firefighting device. Testing must never be performed on energized electrical equipment, flammable liquids, gas fires, or chemical fires. Since water is unsuitable for certain types of fires, any controlled flame demonstration must be conducted under qualified supervision with appropriate safety precautions and emergency equipment available. The robot should not be considered a replacement for trained firefighters or professional firefighting equipment.

XI. LIMITATIONS AND FUTURE SCOPE

The Fire Fighter Robot prototype has certain limitations that may affect its performance and practical application. The control program uses fixed sensor thresholds, which may require calibration for different flame-sensor modules, distances, and lighting conditions [9]–[12]. The current system does not include documented obstacle detection, collision avoidance, remote control, or wireless monitoring. Furthermore, the limited capacity of the onboard water tank restricts the duration of the water-spraying operation. No verified experimental data regarding detection range, response time, extinguishing time, success rate, or battery endurance was provided. The reproduced source code is based on the supplied project thesis and has not been independently confirmed as compiled or tested.

Future improvements could include obstacle-detection sensors, a manual emergency stop, battery-voltage monitoring, pump dry-run protection, and wireless status reporting. A camera or thermal sensor could provide additional information about the fire's location, although these additions would increase processing complexity and power consumption. Further development should also include proper sensor calibration, a documented power budget, and systematic evaluation under repeatable conditions. Before operating the

robot near any flame, the actual sensor polarity and thresholds must be verified, the servo signal generation on pin A4 must be checked, and the pump driver connected to pin A5 must be confirmed to be appropriately rated. The motor-driver wiring and power connections should also be inspected, and the program should be compiled and bench-tested.

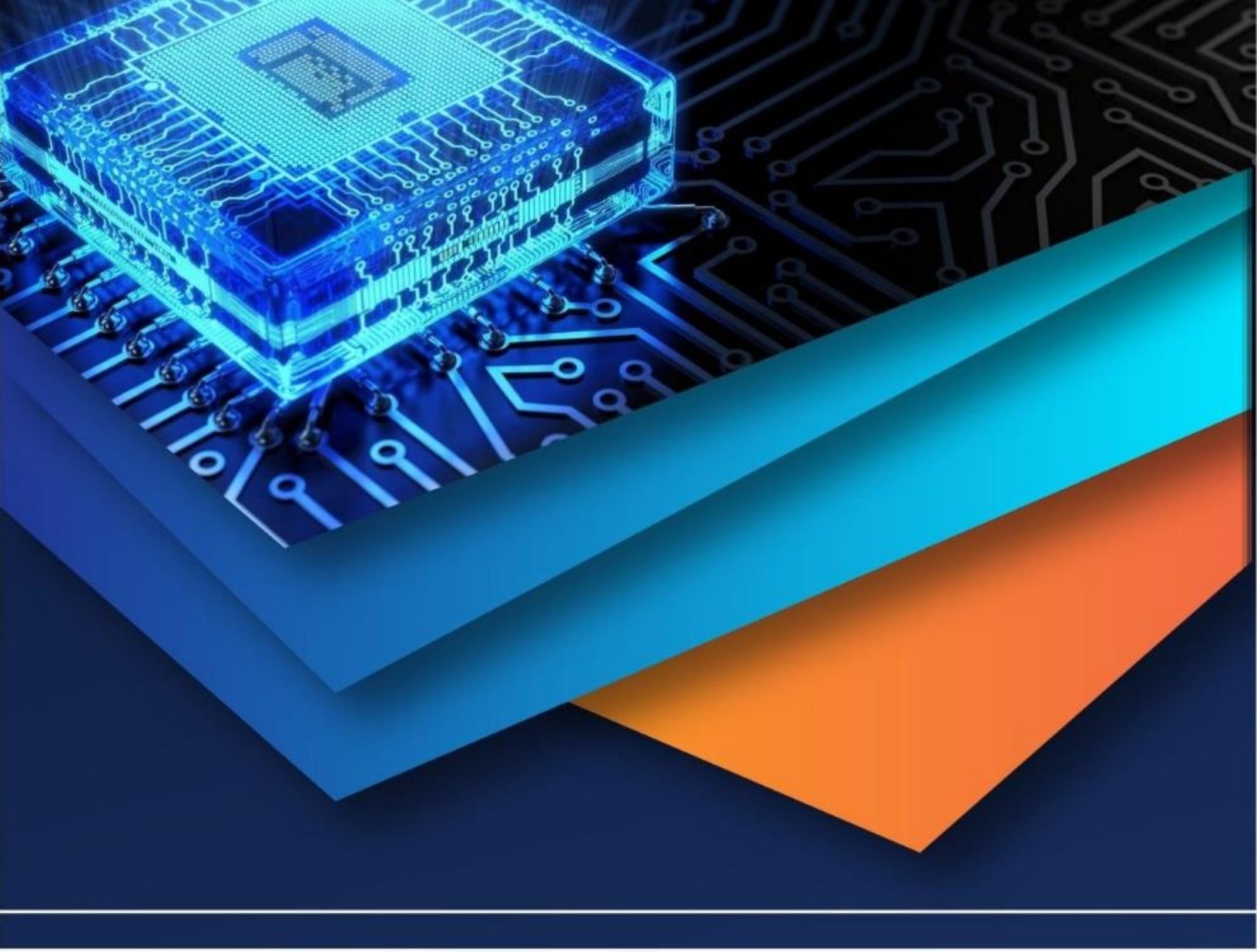
Despite these limitations, the project demonstrates the basic principles of embedded control, sensor integration, motor actuation, and automated response. Its research value can be improved through systematic experiments, recorded sensor readings, documented circuit connections, verified component ratings, and comparisons of different sensor-threshold settings. These improvements would provide a stronger basis for evaluating the robot's reliability and identifying opportunities for further development. The prototype remains an educational model and should not be considered a replacement for professional firefighting equipment or trained firefighters.

XII. CONCLUSION

This paper documented an Arduino-based fire fighter robot using three directional flame sensors, an Arduino UNO, an L298 motor driver, four BO geared DC motors, a servo-operated nozzle, and a water pump. The architecture, circuit connections, operating sequence, physical prototype, and original program were presented. The design demonstrates how sensor inputs can be used to control movement and a simple water-spraying mechanism [2], [3], [7]–[10]. Because no experimental measurements were provided, quantitative performance claims have been intentionally omitted. Improved safety interlocks, sensor calibration, obstacle detection, power management, and repeatable testing are recommended for future development [9]–[12].

REFERENCES

[1] V. Prakash, O. P. Sao, S. K. Sahu, and S. Wadhwa, “A Review Paper on Fire Fighter Robot,” International Journal for Research in Applied Science and Engineering Technology, vol. 13, no. VI, June 2025, doi: 10.22214/ijraset.2025.72561.
[2] Arduino, “UNO R3,” Arduino Documentation. [Online]. Available: https://docs.arduino.cc/hardware/uno-rev3/. [Accessed: Oct. 16, 2026].
[3] STMicroelectronics, “L298: Dual Full-Bridge Driver,” Datasheet DS0218, Rev. 5, Oct. 2023. [Online]. Available: https://www.st.com/resource/en/datasheet/l298.pdf. [Accessed: Oct. 16, 2026].
[4] Arduino Forum, “Firefighter robotic car using Arduino Uno,” Community discussion. [Online]. Available: https://forum.arduino.cc/t/firefighter-robotic-car-using-arduino-uno/1184578. [Accessed: Oct. 16, 2026].
[5] Quartz Components, “Build Fire Fighting Robot Using Arduino,” Project article. [Online]. Available: https://quartzcomponents.com/blogs/electronics-projects/fire-fighting-robot-using-arduino. [Accessed: Oct. 16, 2026].
[6] TechaTronics, “Fire Fighter Robot Using Arduino,” Project article. [Online]. Available: https://techatronic.com/fire-fighter-robot-using-arduino-fire-fighting-robot/. [Accessed: Oct. 16, 2026].
[7] K. Chenchireddy, R. Dora, V. Jagan, G. B. Mulla, V. Jegathesan, and S. A. Sydu, “Design of a prototype firefighting robot based on an Arduino microcontroller using machine learning technique,” International Journal of Robotics and Automation, vol. 14, no. 1, pp. 31–37, 2025, doi: 10.11591/ijra.v14i1.pp31-37.
[8] “A flame sensor-based firefighting assistance robot with simulation based multi-robot implementation,” in Proceedings of the 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), 2022, doi: 10.1109/ACCAI53970.2022.9752470.
[9] “Development of mobile robot with sensor fusion fire detection unit,” IFAC-PapersOnLine, vol. 51, no. 30, pp. 430–435, 2018, doi: 10.1016/j.ifacol.2018.11.324.
[10] T. Yang, W. Ding, Y. Wang, T. Chen, and D. Mu, “Real-time flame detection and localization for firefighting robots,” Robotics and Autonomous Systems, 2025, doi: 10.1016/j.robot.2025.105147.
[11] Arduino, “analogRead(),” Arduino Language Reference. [Online]. Available: https://docs.arduino.cc/language-reference/en/functions/analog-io/analogRead/. [Accessed: Oct. 24, 2026].
[12] F. Riedel and Z. Stössel, “A fail-safe sensor for flame detection,” Sensors and Actuators A: Physical, vols. 37–38, pp. 534–539, 1993, doi: 10.1016/0924-4247(93)80092-U.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)