Students Details	
Module Name:	Robotics Application Development
Module Lecturer/ Course Coordinator:	R.GVimukthi Pathriana
Department:	School of Computing
Submission Due on :	21 December 2023
Type of Coursework:	Individual
Title of the Coursework:	Automated Firefighting robot

Students Details :

	Student No.	**Student Name**
01	COHNDSE232F-058	P.A.T.T. PERERA
02		
03		
04		
05		
06		
07		
08		
09		
10		

Office use only :
Date Stamp Required of the Department

NATIONAL INSTITUTE OF BUSINESS MANAGEMENT
HIGHER NATIONAL DIPLOMA IN SOFTWARE ENGINEERING
COURSEWORK TWO

ROBOTICS APPLICATION DEVELOPMENT

Automated Firefighting robot

SUBMITTED BY

P.A.T.T. PERERA

P.A.T.T. PERERA	**COHNDSE232F-058**

Date of Submission: …21/12/2023….

DECLARATION

I, P.A.T.T. PERERA declare that the work presented in this project report, titled "[firefighting robot]," was carried out under the supervision of Mr. R.G. Vimukthi Pathirana. I affirm that this work represents my original contribution to the field of [Your Project Field]. No portion of this report has been submitted in full or in part to any university or other institution for another Degree/Diploma**.**

Date: 21 December 2023

P.A.T.T.PERERA COHNDSE232F-058 *tharun*
Name with Initials Index Number Signature

R.G Vimukthi Pathirana

Name of the Lecturer

Supervisor

Consultant/ Lecturer Matara Branch

National Institute of Business Management

Date:21th December 2023

SUMMARY

Emergency situations involving fires can be fatal and damaging, necessitating quick action and effective remedies. In this project, we'll investigate how to use Arduino to build an automatic fire-fighting robot. This robot is a vital tool for fire safety and prevention as it can automatically identify and put out flames. This robot can recognize fires and react quickly to put them out by combining sensors, motors, and smart control.

TABLE OF CONTENT

CHAPTER 01

1.1 Introduction for firefighting robot

In response to the pressing need for efficient firefighting solutions in emergency situations, this project explores the development of an Arduino-based automatic firefighting robot. This innovative robot serves as a crucial tool for fire safety and prevention by autonomously identifying and extinguishing flames. By seamlessly integrating sensors, motors, and intelligent control mechanisms, the robot demonstrates a proactive approach to fire management.

1.2 Features of the Product

The firefighting robot integrates several cutting-edge features, emphasizing its effectiveness in tackling fire emergencies:

1.Versatile Power Options:

The inclusion of a power switch allows users to easily toggle the robot on and off, enhancing user control.

A female DC connector for a 9-volt adaptor provides flexibility in power supply options, enabling the robot to run on either an adapter or batteries.

2.Intelligent Flame Detection:

Equipped with three strategically positioned flame sensors, the robot can detect fires from multiple angles.

The arrangement includes sensors on the back, front, and left side, ensuring comprehensive coverage for effective fire detection.

3.Arduino UNO Microcontroller:

The central processing unit is an Arduino UNO microcontroller, serving as the brain of the firefighting robot.

Real-time analysis of sensor data is facilitated, allowing for prompt decision-making and precise control over motor and servo movements.

4.Enhanced Mobility with SG90 Servo Motor:

The robot's mobility is managed by a compact SG90 servo motor, providing the ability to adjust its position dynamically.

The servo motor enables the robot to react swiftly to detected flames, aligning itself for efficient firefighting.

5.Efficient Fire Extinguishing Mechanism:

A DC water pump is employed for extinguishing flames, adding a practical firefighting component to the robot.

Safety measures include a filter capacitor and a diode connected in parallel with the water pump's power supply, ensuring correct operation and preventing reverse biasing.

6.NPN Transistor and TIP 122:

Control over the water pump is achieved using an NPN transistor, specifically the TIP 122, in conjunction with a 1k ohm biasing resistor.

This transistor allows precise control over the water pump, enabling it to be turned on or off based on the current flow.

7.L298 Motor Driver for Controlled Movement:

Robot movement is efficiently managed using the L298 motor driver, working in tandem with the Arduino UNO.

The L298 motor driver facilitates clockwise and counterclockwise control of DC motors, ensuring precise and controlled movement.

CHAPTER 02

2.1 Problem Identification

The project addresses the challenges posed by emergency situations involving fires, emphasizing the need for quick and effective remedies. Identifying delayed response times and limitations in traditional firefighting methods, the project seeks to design an Automatic Fire-Fighting Robot using Arduino technology. The key problems identified include:

Delayed Response Time: Traditional firefighting methods may experience delays in reaching the site of a fire, leading to increased damage and potential loss of life.

Limited Automation: Conventional firefighting equipment often requires human intervention, which can be hindered by various factors such as accessibility and safety concerns.

Inadequate Detection: The ability to accurately detect the presence of flames in diverse environments is crucial for effective firefighting. Conventional methods may fall short in providing precise and timely flame detection.

Risk to Human Safety: Firefighters are exposed to high-risk environments, and their safety is paramount. Automating certain firefighting tasks can reduce the risk to human life in hazardous situations.

Efficiency and Precision: Enhancing the efficiency and precision of firefighting operations is essential for minimizing damage and ensuring successful fire containment.

The project's Problem Identification phase recognizes the limitations of existing firefighting practices and seeks to introduce an innovative solution that combines Arduino technology, sensors, and automation to create an Automatic Fire-Fighting Robot. This robot aims to overcome the identified problems by providing a rapid, automated, and precise response to fire emergencies, thereby improving overall disaster and fire safety measures.

2.2 Research and Design

1. Problem Definition:

The project addresses the challenges posed by emergency situations involving fires, emphasizing the need for quick and effective remedies.

Identifying delayed response times and limitations in traditional firefighting methods, the project seeks to design an Automatic Fire-Fighting Robot using Arduino technology.

2. Objectives:

Develop an autonomous robot capable of identifying and extinguishing flames.

Utilize Arduino UNO, flame sensors, motors, and a water pump to create a comprehensive firefighting solution.

4. Circuit Design:

The power supply comprises two 3.7V batteries connected in series with an 12V total voltage.

Arduino UNO serves as the microprocessor, analyzing data from three flame sensors strategically placed on the robot.

Robot mobility is managed by an SG90 servo motor, and a DC water pump, controlled by a TIP 122 transistor, is used for firefighting.

L298 Motor Driver facilitates motor control, and the circuit includes appropriate connections for effective operation.

5. Arduino Code:

The code initializes motor driver and flame sensor pins, allowing precise control over the robot's movements.

Flame sensors are crucial for fire detection, and servo motor control ensures the robot aligns with the fire source.

Water pump activation is based on flame sensor inputs, providing an automated fire extinguishing mechanism.

The duty cycle variable, "Speed," allows for precise control over the robot's speed, balancing accuracy and responsiveness.

6. Testing and Calibration:

Hardware testing involves assembling the robot as per the schematic and programming the Arduino with the provided code.

Simulate a fire using one of the flame sensors to observe the robot's actions in locating, approaching, and extinguishing the flames.

Adjustments may be made based on testing results to optimize the robot's real-world performance.

2.3 Building the Robot

Required Components and Materials:

1. Solderless Breadboard
2. 4-wheel robot car kit
3. Arduino UNO
4. Flame Sensor x 3
5. L298 Motor Driver
6. Mini Servo Motor SG90
7. 4Pcs Smart Robot Car Tyers Wheels
8. TIP-122 Transistor
9. 1k Resistor
10. 104pf Capacitor
11. PVC Water pipe 8mm
12. 12V DC Water Pump
13. Water Tank
14. Male to Male Jumper Wires
15. Male to Female Jumper Wires
16. Hard Jumper Wire
17. Male Header
18. On/Off Switch
19. 18650 Battery Holder – 3 Cell
20. 18650 Battery Cell 3.7V x 3

Building Steps

Phase 1: Robot Assembly and Basic Mobility

- Assemble the robot chassis using a 5mm card sheet.
- Attach two gear motors to the chassis for mobility.
- Mount the L298 motor driver on the chassis.
- Securely place the Arduino Uno board on the chassis.
- Connect the gear motors to the L298 motor driver for motor control.
- Attach four wheels to the gear motors for movement.
- Install a caster wheel at the front for stability.
- Connect to three 18650 lithium-ion batteries (3.7V each).
- Securely attach the battery holder to the chassis.
- Connect the batteries to power the motor driver and Arduino.

Phase 2: Flame Detection and Mobility Control

- Mount three flame sensors strategically on the robot (front, back, and left).
- Connect the flame sensors to the Arduino for fire detection.
- Integrate a mini servo motor (SG90) for robot orientation based on flame detection.
- Use the L298 motor driver to control the movement of the robot.
- Implement a DC water pump and connect it to the TIP-122 transistor for firefighting.
- Connect a water tank with a PVC water pipe (8mm) to the water pump.
- Design and install a mechanism for water discharge to extinguish flames.

- Ensure proper safety measures, such as a diode and filter capacitor, for the water pump.

Phase 3: Circuit Connection and Arduino Programming

- Connect all components using male-to-male, male-to-female, and hard jumper wires.
- Incorporate an on/off switch for easy control.
- Use male headers for secure connections.
- Design and create a circuit based on the provided circuit diagram.
- Upload the provided Arduino code to the Arduino Uno board.
- Configure the motor driver and flame sensor pins in the Arduino code.
- Adjust the servo motor control in the code for precise orientation.

Phase 4: Testing and Calibration

- Assemble the robot according to the schematic.
- Program the Arduino with the provided code.
- Simulate a fire using one of the flame sensors.
- Observe the robot's actions in detecting, approaching, and extinguishing the flames.
- Make any necessary adjustments to optimize the robot's real-world performance.

2.4 Programming

Flow chart for firefighting_robot.

Start

-- Setup --
  |
  V
[Initialize]
  |
  V
[Set Servo Position]
  |
  V
[Set Motor Speed]
  |
  V
[Delay 500 ms]

-- Loop --
  |
  V
[Read Flame Sensors (s1, s2, s3)]
  |
  V
[Print Sensor Values]
  |
  V
-- Auto Control --
  |
  V
[Check s1 < 250]
  |
  V
[Stop, Activate Pump, Move Servo]
  |
  V
[Delay 100 ms]
  |
  V
[Turn Off Pump, Move Servo]
  |
  V
[Check s2 < 350]
  |
  V
[Stop, Activate Pump, Move Servo]

|
V
[Delay 100 ms]
|
V
[Turn Off Pump, Move Servo]
|
V
[Check s3 < 250]
|
V
[Stop, Activate Pump, Move Servo]
|
V
[Delay 100 ms]
|
V
[Turn Off Pump, Move Servo]
|
V
[Check s1, s2, s3 for specific ranges]
|
V
-- Movement Control --
|
V
[Execute Forward, Backward, Right, Left Movements]
|
V
[Stop]

End

Arduino code for my project

```
#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);  Speed
  analogWrite(enB, 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);
  }

}

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
}
```

CHAPTER 03

3.1 Results of the operation

The automatic fire-fighting robot demonstrated commendable performance during various test scenarios, showcasing its ability to detect and respond to fire hazards effectively.

1. Flame Detection and Localization

The robot employs three strategically positioned flame sensors—located at the front, back, and left sides—to accurately detect the presence of flames. During testing, the sensors consistently provided reliable readings, allowing the robot to identify the location of the fire source.

2. Autonomous Movement and Fire Suppression

Upon detecting a fire, the robot executed precise and controlled movements to approach the fire source. The autonomous control algorithm successfully maneuvered the robot to position itself optimally for fire suppression.

3. Servo Motor and Water Pump Control

The integrated servo motor effectively controlled the orientation of the robot, allowing it to align with the detected fire. The water pump, triggered upon flame detection, demonstrated efficient water discharge for fire suppression.

4. Stability and Error Handling

The robot exhibited stability in its movements, even in dynamic environments. Additionally, the implemented error-handling mechanisms proved effective during testing, allowing the robot to recover from unexpected situations.

5. Serial Monitoring for Debugging

The serial communication interface facilitated real-time monitoring of sensor values, aiding in debugging and fine-tuning the robot's behavior. Serial print statements were strategically placed in the code to provide insightful information during testing.

CHAPTER 04

4.1 Limitations, Recommendations and Conclusion

Limitations,

Despite the successful demonstration of the automatic fire-fighting robot, certain limitations were identified during the testing phase:

1. Sensor Range and Sensitivity
The infrared (IR) flame sensors exhibited limitations in detecting flames beyond a certain distance. Further calibration and testing are recommended to optimize sensor sensitivity for varying fire intensities and distances.

2. Environmental Conditions
The robot's performance may be affected by adverse environmental conditions such as heavy smoke or low visibility. Future iterations should explore sensor technologies and algorithms that are resilient to challenging environments.

3. Obstacle Navigation
The current version of the robot lacks advanced obstacle avoidance capabilities. In scenarios with obstacles, the robot's movements may be hindered. Implementing obstacle detection and avoidance algorithms is a recommended area for improvement.

Recommendations,

To enhance the overall functionality and robustness of the automatic fire-fighting robot, the following recommendations are proposed:

1. Sensor Calibration and Testing
Conduct extensive calibration and testing of flame sensors to determine optimal sensitivity levels. This includes testing the robot's performance under various fire intensities and distances to improve detection accuracy.

2. Integration of Additional Sensors
Explore the integration of additional sensors, such as temperature and gas sensors, to provide comprehensive environmental data. This additional information can contribute to more informed decision-making during fire emergencies.

3. Advanced Navigation Algorithms
Incorporate advanced navigation algorithms to enable the robot to navigate complex environments with obstacles more efficiently. This could involve the use of ultrasonic sensors or computer vision techniques for improved spatial awareness.

Conclusion,

In conclusion, the automatic fire-fighting robot represents a significant step towards leveraging technology for fire safety and prevention. The successful integration of sensors, actuators, and control algorithms has demonstrated its potential in autonomously detecting and suppressing fires.

While certain limitations exist, the robot's performance during testing indicates a promising foundation for future developments. Addressing the identified limitations and implementing the recommended enhancements will contribute to the robot's effectiveness in real-world firefighting scenarios.

The project not only showcases the capabilities of robotics in emergency response but also serves as a platform for ongoing research and innovation in the field of autonomous firefighting technology.

REFERENCES

- (Using Infrared sensors for distance measurement in mobile robots) https://www.researchgate.net/publication/222398145_Using_Infrared_sensors_for_distance_measurement_in_mobile_robots

- https://www.arduino.cc/

- https://ss-valpovo.hr/wp-content/uploads/2020/01/arduinoprojecthandbook.pdf

-
- https://circuitdigest.com/microcontroller-projects/interface-single-channel-relay-module-with-arduino

APPENDIX