

ARDUINO FIRE FIGHTING ROBOT

A report submitted in partial fulfillment of the requirements
for the award of

Bachelor of Technology

In

CSO (Computer Science and Engineering (IoT))

By

BANDLA SRISAI SANKAR
(20BQ1A4909)

VANAMA NIKHIL
(20BQ1A4953)

ANUMALASETTY CHANDRA
MANIKANTA
(20BQ1A4904)

SENSORS AND ACTUATORS FOR IoT
(Skill Oriented Course (Course Code: 20CO4C01))



DEPARTMENT OF CSO (Computer Science and Engineering (IoT))
VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY
(AUTONOMOUS)

(Approved by AICTE and permanently affiliated to JNTUK, Accredited by NBA and NAAC)

NAMBUR (V), PEDAKAKANI (M), GUNTUR-522 508

JUNE 2022

DEPARTMENT OF CSO (Computer Science and Engineering (IoT))
VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY: NAMBUR
(AUTONOMOUS)



CERTIFICATE

This is to certify that the Mini Project titled “**ARDUINO FIRE FIGHTING ROBOT**” is a bonafide record of work done by **B.SRISAI SANKAR (20BQ1A4909), V.NIKHIL (20BQ1A4953), A.CHANDRA MANIKANTA (20BQ1A4904), B.KALYAN KRISHNA (20BQ1A4907)** as part of the Skill Oriented Course **Sensors and Actuators for IoT (Course Code: 20CO4C01)** in partial fulfillment of the requirement of the degree for Bachelor of Technology in CSO (CSE (IoT)) during the academic year 2021–22.

Mr. S. KRISHNA PRASAD
Course Coordinator

DR. CH. V. SURESH
Head of Department

TABLE OF CONTENTS

TITLE OF CONTENT	PAGE NO
Abstract	6
Chapter 1 :Introduction	7-8
Chapter 2 :Hardware and software Requirements	9-19
Chapter 3 :Working and Implementation	20-25
Chapter 4 :Result and Discussion	26-27
Chapter 5 : Summary and Future Scope	28-30
References	31
Appendix (include code)	

ABSTRACT

The purpose of this project is to design, build, and test a robot capable of extinguishing building and basement fires and effectively replacing a firefighter in highly dangerous situations. The robot will allow for firefighters to not only put out a fire remotely, but allow rescuers to scout a burning building before sending any firefighters inside. The implementation of this robot will increase the safety of firefighters and therefore help mitigate deaths from unsafe conditions. The robot will be controlled by an operator via wireless remote control. The operator will have visual feedback from the robot through the use of an onboard camera with normal imaging and infrared imaging for low light situations. While the robot design will be as robust as possible, the main goal of the project is a proof of concept. The design of the robot has been separated into several major components: the mobility base, the extinguishing system, the frame, the base station, and internal control. Size and weight constraints were determined utilizing the New Jersey Building Codes and the weight of a fully equipped firefighter. Multiple design options were investigated for each component of the robot. The final designs were chosen based on effectiveness, practicality, and available resources. Ultimately, the robot should have full mobility on flat and inclined surfaces and have the ability to traverse a staircase. The robot should be able to fully extinguish a test fire in a safe and controlled manner, while sending visual feedback to the operator. A fully functioning base station, acting as a wireless hotspot for the feedback system, should be developed and the operator should be able to fully control the robot via a remote interface. All of these goals should be met while following the size and weight constraints.

Fire accidents are very common and sometimes it becomes very difficult for a fireman to save someone's life. It is not possible to appoint a person to continuously observe for accidental fire where a robot can do that. Therefore in such cases a fire fighting robot comes in picture. A robot will detect fire remotely. These robots are mostly useful in industries where the probability of accidental fire is more.

1. Introduction

This advanced firefighting robotic system independently detects and extinguishes fire. In the age of technology, the world is slowly turning towards the automated system and self-travelling vehicles, fire fighters are constantly at a risk of losing their life. Fire spreads rapidly if it is not controlled. In case of a gas leakage there even may be an explosion. So, in order to overcome this issue, safe guard live of our hero, our system comes to the rescue. This firefighting robotic system is powered by Arduino Uno development board it consists of the HC-SR04 ultra-sonic sensor mounted on a servo motor for obstacles detection and free path navigation, it is also equipped with the fire flame sensor for detecting and approaching fire it also makes use of water tank and spray mechanism for extinguishing the fire. Water spraying nozzle is mounted on servo motor to cover maximum area. Water is pumped from the main water tank to the water nozzle with the help of water pump. Fire fighting is the act of extinguishing destructive fires. A firefighter must be able to stop fire quickly and safely extinguish the fire, preventing further damage and rescue victims to a safer location from the hazard. Technology has finally bridged the gap between fire fighting and machines allowing for a more efficient and effective method of fire fighting. Robots were designed to find a fire, before it rages out of control. The robots could one day work with fire fighters in reducing the risk of injury to victims. As a result of a fire outbreak (or) fire explosion, we are demanding that we use human resources that are not secure to put out the fire. It is very much possible to replace human work in putting out a fire in a dangerous environment by using higher technology, specifically robotics. This strategy would free firefighters from dangerous tasks, increase their efficiency, and reduce the number of fires. Moreover, it will discourage human lives from being jeopardized. Forth is, we'll create an Arduino based firefighting robot that will detect the fire and it will begin to pump water on the fire detected area using sprinklers.

2 . Hardware and software Requirements

Hardware Requirements

1. Arduino UNO
2. Fire sensor or Flame sensor (3 Nos)
3. Servo Motor (SG90)
4. L293D motor Driver module
5. Mini DC Submersible Pump
6. Small Breadboard
7. Robot chassis with motors (4) and wheels(4) (any type)
8. A small can
9. Connecting wires

1.Arduino UNO

Arduino Uno Technical Specifications:

Microcontroller	ATmega328P – 8 bit AVR family microcontroller
Operating Voltage	5V
Recommended Input Voltage	7-12V
Input Voltage Limits	6-20V
Analog Input Pins	6 (A0 – A5)
Digital I/O Pins	14 (Out of which 6 provide PWM output)
DC Current on I/O Pins	40 mA
DC Current on 3.3V Pin	50 mA
Flash Memory	32 KB (0.5 KB is used for Bootloader)
SRAM	2 KB
EEPROM	1 KB
Frequency (Clock Speed)	16 MHz

Table.1



Fig.1

Arduino Uno Pinout Configuration:

Power	Vin, 3.3V, 5V, GND	Vin: Input voltage to Arduino when using an external power source. 5V: Regulated power supply used to power microcontroller and other components on the board. 3.3V: 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA. GND: ground pins.
Reset	Reset	Resets the microcontroller.
Analog Pins	A0 – A5	Used to provide analog input in the range of 0-5V
Input/Output Pins	Digital Pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(Tx)	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.
PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.
Inbuilt LED	13	To turn on the inbuilt LED.
TWI	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide reference voltage for input voltage.

Table.2

2.Fire sensor or Flame sensor:

IR Infrared Flame Module Pin Diagram:



Fig.2

3 pins Flame Sensor Module :

Pin Number	Pin Name	Description
1	VCC	+5 v power supply
2	GND	Ground (-) power supply
3	OUT	Digital Output (0 or 1)

Table.3

Working Principle:

This sensor/detector can be built with an **electronic circuit** using a receiver like electromagnetic radiation. This sensor uses the infrared flame flash method, which allows the sensor to work through a coating of oil, dust, water vapor, otherwise ice.

Different Types:

Flame-sensors are classified into four types

- IR single frequency
- IR multi-spectrum
- UV flame detectors
- UV/ IR flame detectors

Features & Specifications:

The features of this sensor include the following.

- Photosensitivity is high
- Response time is fast
- Simple to use
- Sensitivity is adjustable
- Detection angle is 600,
- It is responsive to the flame range.
- Accuracy can be adjustable
- Operating voltage of this sensor is 3.3V to 5V
- Analog voltage o/ps and digital switch o/ps
- The PCB size is 3cm X 1.6cm
- Power indicator & digital switch o/p indicator
- If the flame intensity is lighter within 0.8m then the flame test can be activated, if the flame intensity is high, then the detection of distance will be improved.

Applications:

These sensors are used in several dangerous situations which include the following.

- Hydrogen stations
- Industrial heating
- Fire detection
- Fire alarm
- Fire fighting robot
- Drying systems
- Industrial gas turbines
- Domestic heating systems
- Gas-powered cooking devices

3.Servo Motor (SG90):

Servo motor pinout(wires):

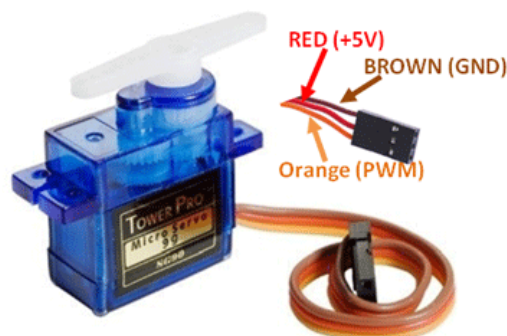


Fig.3

Servo Motor Wire Configuration:

Wire Number	Wire Colour	Description
1	Brown	Ground wire connected to the ground of system
2	Red	Powers the motor typically +5V is used
3	Orange	PWM signal is given in through this wire to drive the motor

Table.4

TowerPro SG-90 Features:

- Operating Voltage is +5V typically
- Torque: 2.5kg/cm
- Operating speed is 0.1s/60°
- Rotation : 0°-180°
- Weight of motor : 9gm
- Package includes gear horns and screws

Working of servo motor:

Servo motors are high torque motors which are commonly used in robotics and several other applications due to the fact that it's easy to control their rotation. Servo motors have a geared output shaft which can be electrically controlled to turn one (1) degree at a time. For the sake of control, unlike normal DC motors, servo motors usually have an additional pin besides the two power pins (Vcc and GND) which is the signal pin. The signal pin is used to control the servo motor, turning its shaft to any desired angle.

Applications:

- Used as actuators in many robots like Biped Robot, Hexapod, robotic arm etc..
- Commonly used for steering system in RC toys
- Robots where position control is required without feedback
- Less weight hence used in multi DOF robots like humanoid robots

4.L293D motor Driver module:

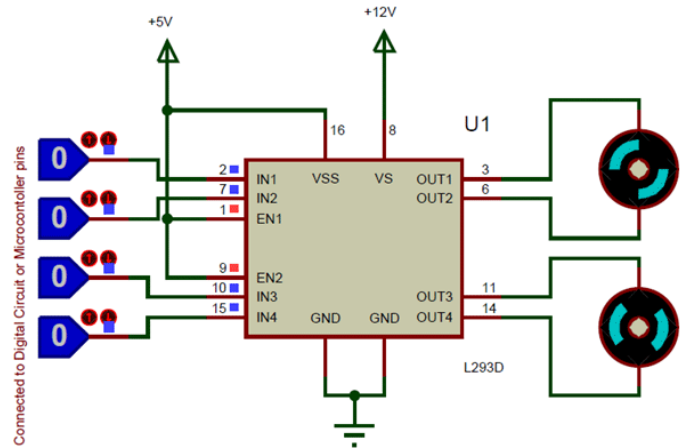


fig.4

Working of L293D motor driver:

Using this L293D motor driver IC is very simple. The IC works on the principle of Half H-Bridge, let us not go too deep into what H-Bridge means, but for now just know that H bridge is a setup which is used to run motors both in clock wise and anticlockwise direction. As said earlier this IC is capable of running two motors at the any direction at the same time, the circuit to achieve the same is shown above. All the Ground pins should be grounded. There are two power pins for this IC, one is the Vss(Vcc1) which provides the voltage for the IC to work, this must be connected to +5V. The other is Vs(Vcc2) which provides voltage for the motors to run, based on the specification of your motor you can connect this pin to anywhere between 4.5V to 36V, here I have connected to +12V.

table.5

Input 1 = HIGH(5v)	Output 1 = HIGH	Motor 1 rotates in Clock wise Direction
Input 2 = LOW(0v)	Output 2 = LOW	
Input 3 = HIGH(5v)	Output 1 = HIGH	Motor 2 rotates in Clock wise Direction
Input 4 = LOW(0v)	Output 2 = LOW	
Input 1 = LOW(0v)	Output 1 = LOW	Motor 1 rotates in Anti-Clock wise Direction

Input 2 = HIGH(5v)	Output 2 = HIGH	
Input 3 = LOW(0v)	Output 1 = LOW	Motor 2 rotates in Anti -Clock wise Direction
Input 4 = HIGH(5v)	Output 2 = HIGH	

table.6

Input 1 = HIGH(5v)	Output 1 = HIGH	Motor 1 stays still
Input 2 = HIGH(5v)	Output 2 = HIGH	
Input 3 = HIGH(5v)	Output 1 = LOW	Motor 2 stays still
Input 4 = HIGH(5v)	Output 2 = HIGH	

table.7

Applications

- Used to drive high current Motors using Digital Circuits
- Can be used to drive stepper motors
- High current LED's can be driven
- Relay Driver module (Latching Relay is possible)

5. Mini DC Submersible Pump:



fig.5

Specifications:

- Input Voltage: 2.5V to 6V DC
- Working Current - 130-250mA
- Power 0.4 - 1.5W
- max Lift - 40-110 cm
- Flow Rate - 80-120 L/H
- Outside Dia of Water Outlet - 7.5mm
- Inside Dia of Water Outlet - 4.7mm
- Diameter - 24mm
- Length - 45mm
- Height - 33mm
- Wire Length - 15-20cm
- Material - Engineering Plastic

Working of Mini DC Submersible Pump:

A mini submersible water pump is a centrifugal water pump, which means that it uses a motor to power an impeller that is designed to rotate and push water outwards. The motor is located in a waterproof seal and closely connected to the body of the water pump which it powers.

6.Small Breadboard:

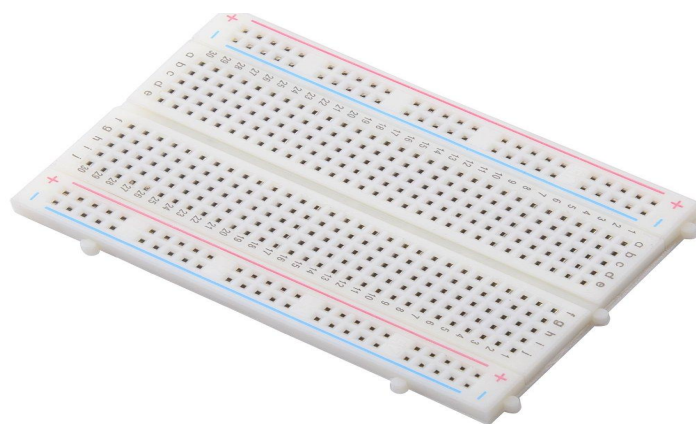


fig.6

Working of Small BreadBoard:

A breadboard is used to build and test circuits quickly before finalizing any circuit design. The breadboard has many holes into which circuit components like ICs and resistors can be inserted. The bread board has strips of metal which run underneath the board and connect the holes on the top of the board. The metal strips are laid out as shown below. Note that the top and bottom rows of holes are connected horizontally while the remaining holes are connected vertically. Because the solderless breadboard does not require [soldering](#), it is reusable. This makes it easy to use for creating temporary prototypes and experimenting with circuit design. For this reason, solderless breadboards are also popular with students and in technological education. Older breadboard types did not have this property. A [stripboard](#) ([Veroboard](#)) and similar prototyping [printed circuit boards](#), which are used to build semi-permanent soldered prototypes or one-offs, cannot easily be reused. A variety of electronic systems may be prototyped by using breadboards, from small analog and digital circuits to complete [central processing units](#) (CPUs).

7.Robot chassis with BO motors (4) and wheels(4) :



fig.7

8. A Small Can:



fig.8

9.Connecting Wires:



fig.9

Software Requirements:

Arduino IDE

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

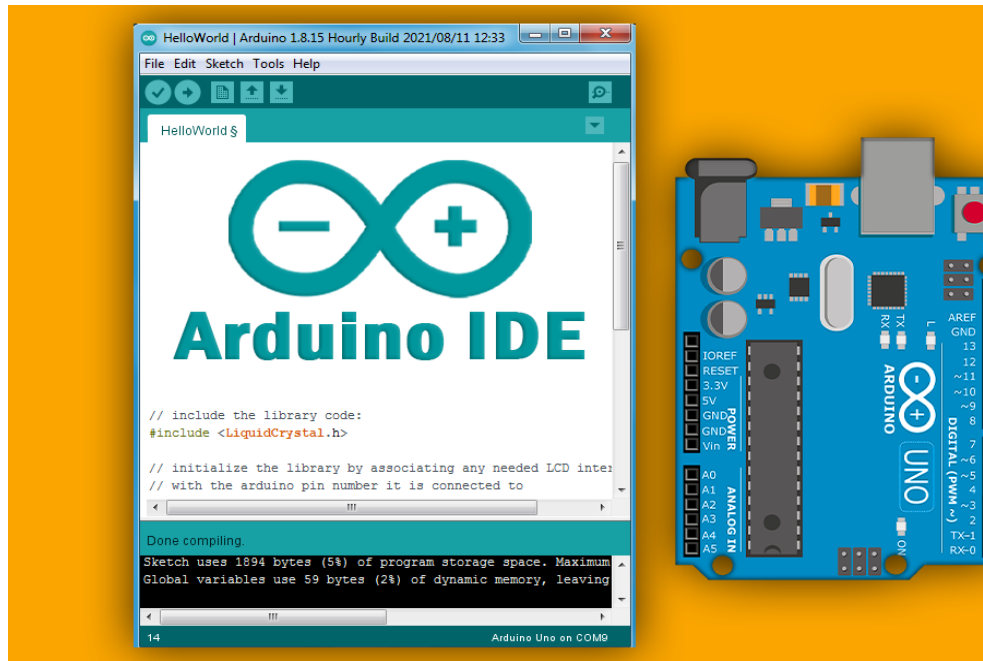


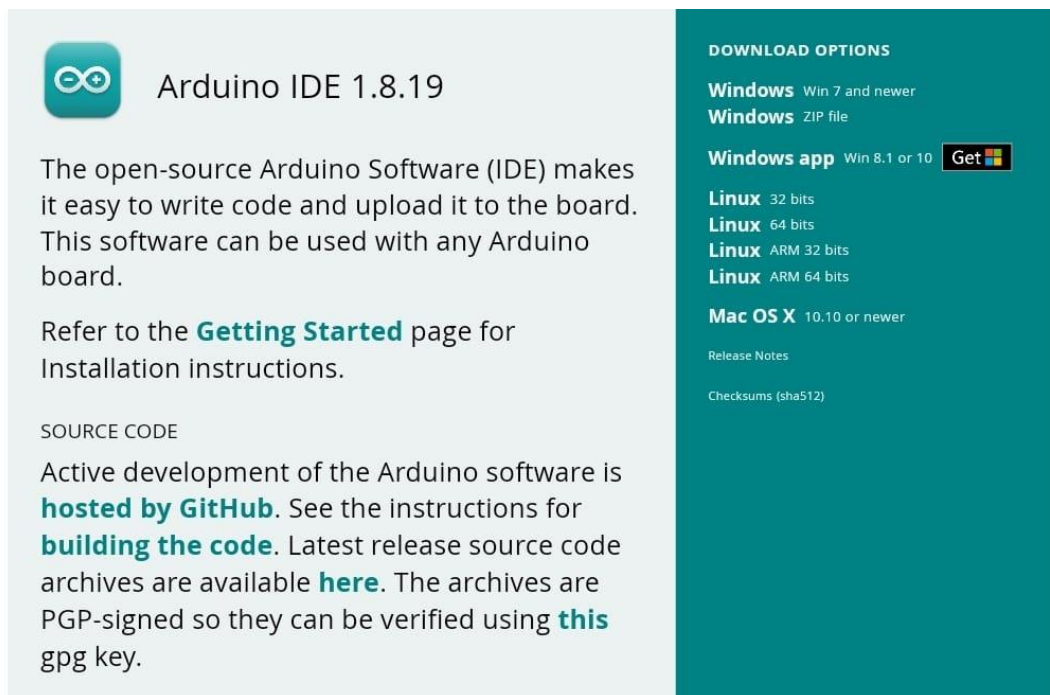
fig.10


Its hardware products are licensed under a CC BY-SA license, while software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL) permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official website or through authorized distributors.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ('shields') or breadboards (for prototyping) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs. The microcontrollers can be programmed using the C and C++ programming languages, using a standard API which is also known as the Arduino language, inspired by the Processing language and used with a modified version of the Processing IDE. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) and a command line tool developed in Go. aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats and motion detectors.

The Arduino [integrated development environment](#) (IDE) is a [cross-platform](#) application (for [Microsoft Windows](#), [macOS](#), and [Linux](#)) that is written in the [Java](#) programming language. It originated from the IDE for the languages [Processing](#) and [Wiring](#). It includes a

code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, [brace matching](#), and [syntax highlighting](#), and provides simple *one-click* mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the [GNU General Public License](#), version 2. The Arduino IDE supports the languages [C](#) and [C++](#) using special rules of code structuring. The Arduino IDE supplies a [software library](#) from the [Wiring](#) project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable [cyclic executive](#) program with the [GNU toolchain](#), also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.



 **Arduino IDE 1.8.19**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file
Windows app Win 8.1 or 10 [Get](#)

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

Mac OS X 10.10 or newer

[Release Notes](#)
[Checksums \(sha512\)](#)

fig.11

3. WORKING AND IMPLEMENTATION

Working Concept of Fire Fighting Robot:

As you can see these sensors have an **IR Receiver (Photodiode)** which is used to detect the fire. How is this possible? When fire burns it emits a small amount of Infra-red light, this light will be received by the IR receiver on the sensor module. Then we use an Op-Amp to check for change in voltage across the IR Receiver, so that if a fire is detected the output pin (DO) will give 0V(LOW) and if there is no fire the output pin will be 5V(HIGH).

So, we place three such sensors in three directions of the robot to sense on which direction the fire is burning.

We detect the direction of the fire we can use the motors to move near the fire by driving our motors through the **L293D module**. When near a fire we have to put it out using water. Using a small container we can carry water, a 5V pump is also placed in the container and the whole container is placed on top of a **servo motor** so that we can control the direction in which the water has to be sprayed. Let's proceed with the connections now.

You can either connect all the shown connections for uploading the program to check the working or you can assemble the bot completely and then proceed with the connections. Both ways the connections are very simple and you should be able to get it right.

Based on the robotic chassis that you are using you might not be able to use the same type of container that I am using. In that case use your own creativity to set up the pumping

system. However the code will remain same. I used a small aluminium can (cool drinks can) to set the pump inside it and poured water inside it. I then assembled the whole can on top of a servo motor to control the direction of water. My robot looks something like this after assembly.

As you can see, I have fixed the servo fin to the bottom of the container using got glue and have fixed the servo motor with chassis using nuts and bolts. We can simply place the container on top of the motor and trigger the pump inside it to pump water outside through the tube. The whole container can then be rotated using the servo to control the direction of the water.

Circuit Diagram: The complete circuit diagram for this Fire Fighting Robot is given below

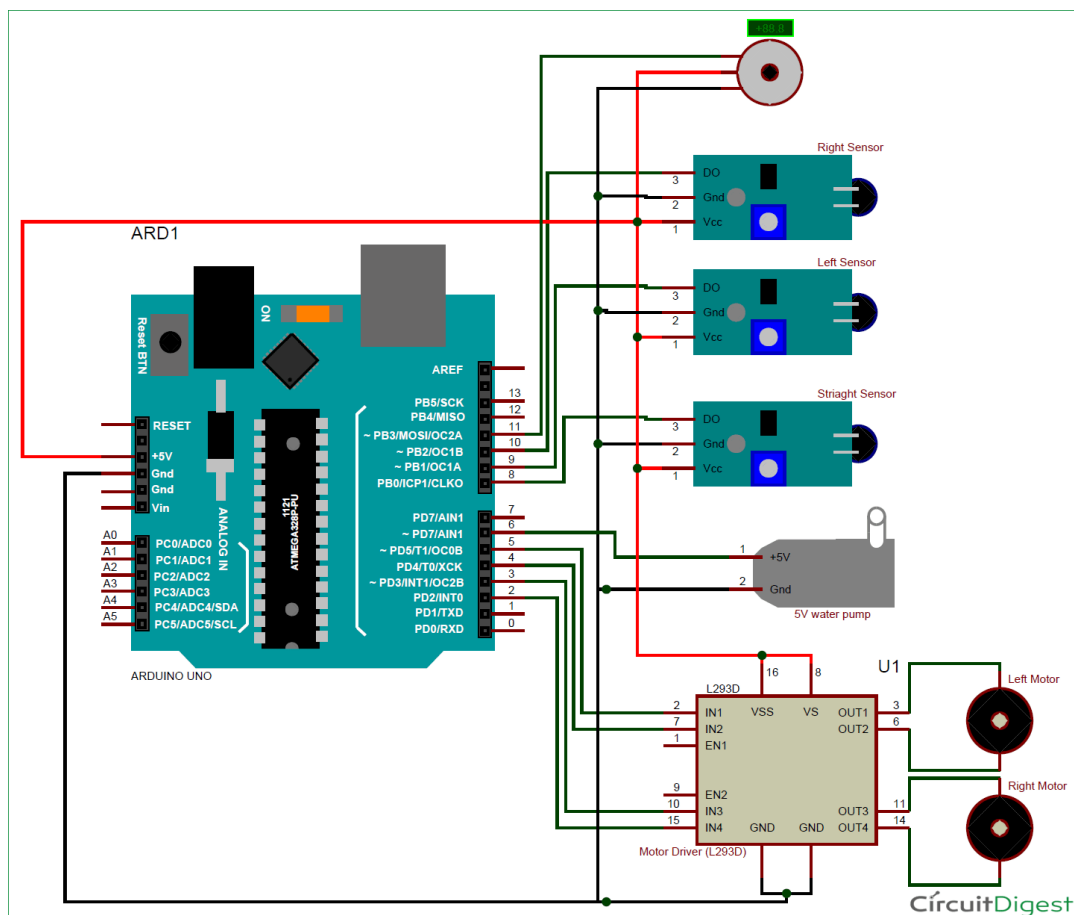


fig.12

We can either connect all the shown connections for uploading the program to check the working or you can assemble the bot completely and then proceed with the connections. Both ways the connections are very simple and you should be able to get it right.

Programming Arduino:

Once you are ready with your hardware, you can upload the Arduino code for some action. The complete program is given at the end of this page. However I have further explained few important bits and pieces here.

As we know the fire sensor will output a HIGH when there is fire and will output a LOW when there is no fire. So we have to keep checking these sensors if any fire has occurred. If no fire is there we ask the motors to remain stop by making all the pins high as shown below

```
if      (digitalRead(Left_S)==1      &&      digitalRead(Right_S)==1      &&
digitalRead(Forward_S) ==1)      //If Fire not detected all sensors are zero

{

//Do not move the robot

digitalWrite(LM1, HIGH);

digitalWrite(LM2, HIGH);

digitalWrite(RM1, HIGH);

digitalWrite(RM2, HIGH);

}
```

Similarly, if there is any fire we can ask the robot to move in that direction by rotating the respective motor. Once it reaches the fire the left and right sensor will not detect the fire as it would be standing straight ahead of the fire. Now we use the variable named “*fire*” that would execute the function to put off the fire.

```
else if (digitalRead(Forward_S) == 0) //If Fire is straight ahead

{

    //Move the robot forward

    digitalWrite(LM1, HIGH);

    digitalWrite(LM2, LOW);

    digitalWrite(RM1, HIGH);

    digitalWrite(RM2, LOW);

    fire = true;

}
```

Once the variable *fire* becomes true, the fire fighting robot arduino code will execute the *put_off_fire* function until the fire is put off. This is done using the code below.

```
while (fire == true)

{

    put_off_fire();

}
```

Inside the *put_off_fire()* we just have to stop the robot by making all the pins high. Then turn on the pump to push water outside the container, while this is done we can also use the servo motor to rotate the container so that the water is split all over uniformly. This is done using the code below

```
void put_off_fire()

{

    delay (500);

    digitalWrite(LM1, HIGH);

    digitalWrite(LM2, HIGH);

    digitalWrite(RM1, HIGH);

    digitalWrite(RM2, HIGH);

    digitalWrite(pump, HIGH); delay(500);

    for (pos = 50; pos <= 130; pos += 1) {

        myservo.write(pos);

        delay(10);

    }

    for (pos = 130; pos >= 50; pos -= 1) {

        myservo.write(pos);
```

```
    delay(10);  
  
}  
  
digitalWrite(pump,LOW);  
  
myservo.write(90);  
  
fire=false;  
  
}
```

ROBOT DIAGRAM:

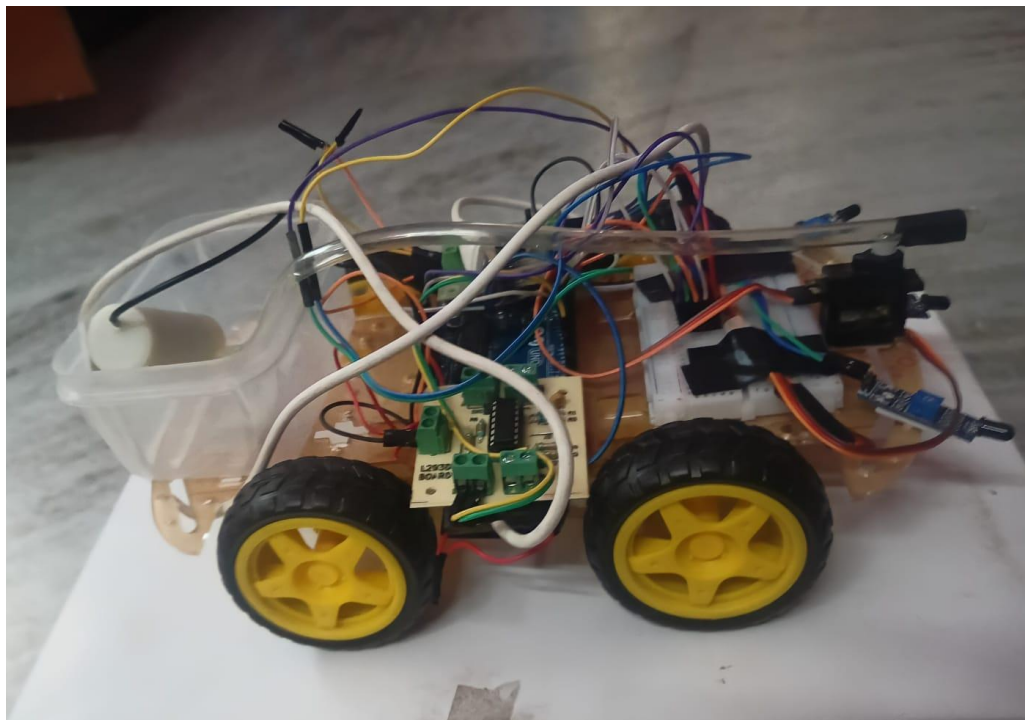


fig.13

4.RESULT AND DISCUSSION

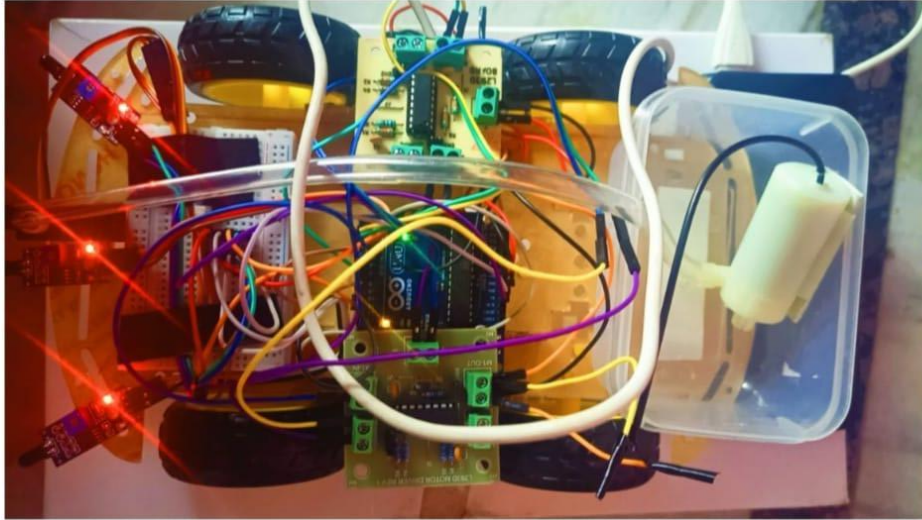


fig.14

In this project, an autonomous Firefighting Robot has been implemented which is capable of detecting flames & smokes and extinguishing them successfully. This robot can move forward, move left & right flawlessly. The motors and Arduino code work together to control the movement of the robot. If any of the flame sensors or smoke sensor are triggered, then buzzer will start to buzz & warning about the danger environment will be displayed on the Virtual Terminal & safe environment will be shown in case of no such detection. The motor will start to rotate & move the robot to the danger point upon receiving a signal about the danger environment & start to pump the water with the help of servo motor. This process will be continued until the fire or smoke has been extinguished completely. Then it will display about the safe environment. After successfully building the project, the simulation was run and the desired output was obtained. Proper snapshots of the results were attached. Thus, an autonomous firefighting robot has been built to achieve the objectives of this project successfully.

5.SUMMARY AND FUTURE SCOPE

The Fire Fighting Robot is effective enough to fight against fire on a small scale. It can sense fire flame better at darker places. It is made as a preventer robot. Because it can detect fire instantly and can extinguish it before spreading. This multisensory based robot may be a solution to all fire hazards. Various sensors like flame, smoke sensors have

been incorporated in this robot. If the fire is detected, a water spraying mechanism is triggered to extinguish the fire. Sound alert is also issued upon all events to alert the operator. With enough funding and scope, this design of robot can also fight against large fire with larger reserving capacity and an improved sensing unit can provide even an earlier detection of fire at all circumstances. As a conclusion, the project entitled “Fire Fighting Robot” has achieved its aim and objective successfully.

Future work can include the transformation of the experimental robot prototype into a practical robot, which requires improvement in its overall performance. Face detection system are developed for a fire-fighting robot to rescue human beings caught in fire. The face detection system alerts the presence of human beings caught in fire to facilitate their rescue. The ultrasonic sensor can also be attached to detect any object around the robot to avoid any collision with other objects. Wireless remote-control idea can be introduced to this system so that human can control the mechanism of the robot by their own needs. and its performance can be enhanced by interfacing it with a wireless higher Resolution zooming Camera so that the person controlling it can view the operation of the robot remotely on a screen.

References

- [1]. Automated robot fire extinguisher-2011 8th International Ubiquitous Robots and Ambient Intelligence Conference (URAI)
- [2]. Development and implementation of dual mode fire extinguishing robot based on arduino microcontrollers: 2017 IEEE International Conference on Intelligent Control, Optimization and Signal Processing Techniques (INCOS) (INCOS) (INCOS)
- [3]. Cease Fire: The Fire Fighting Robot: 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)
- [4]. Fire extinguisher robot based on multiple sensors based on DTMF, bluetooth and GSM technology with multiple modes of operation: 2016 International Computational Intelligence Workshop 2016 (IWCI)
- [5]. Intelligent Fire Fighting Robot(2010) Kristi Kosasih, E. Merry Sartika, M. Jimmy Hasugin, dan MulidyElectric Motorering Department, Maranatha Christian University J1.Prof.Drg. Sumantri Suria.
- [6]University Malaysia Perlis, UNIMAP, “Fire Fighting Robot Competition, Theme & Rules”, 2009.
- [7]Vesely, “Implementation of Micromouse Class Robot”. William Dubel, Hector Gongora, Kevin Bechtold and Daisy Diaz, “An Autonomous Firefighting Robot”.
- [8]John Iovine, “PIC Robotics: A Beginner’s Guide to Robotics Projects Using the PICmicro”, Mc Graw Hill, 2004.

Appendix

Source code:

```
/*----- Arduino Fire Fighting Robot Code----- */
#include <Servo.h>
Servo myservo;
int pos = 0;
boolean fire = false;
/*-----defining Inputs-----*/
#define Left_S 9 // left sensor
#define Right_S 10 // right sensor
#define Forward_S 8 //forward sensor
/*-----defining Outputs-----*/
#define LM1 2 // left motor
#define LM2 3 // left motor
#define RM1 4 // right motor
#define RM2 5 // right motor
#define pump 6
void setup()
{
  pinMode(Left_S, INPUT);
  pinMode(Right_S, INPUT);
  pinMode(Forward_S, INPUT);
  pinMode(LM1, OUTPUT);
  pinMode(LM2, OUTPUT);
  pinMode(RM1, OUTPUT);
  pinMode(RM2, OUTPUT);
  pinMode(pump, OUTPUT);
  myservo.attach(11);
  myservo.write(90);
}
void put_off_fire()
{
  delay (500);
```

```

digitalWrite(LM1, HIGH);
digitalWrite(LM2, HIGH);
digitalWrite(RM1, HIGH);
digitalWrite(RM2, HIGH);
digitalWrite(pump, HIGH); delay(500);
for (pos = 50; pos <= 130; pos += 1) {
    myservo.write(pos);
    delay(10);
}
for (pos = 130; pos >= 50; pos -= 1) {
    myservo.write(pos);
    delay(10);
}
digitalWrite(pump, LOW); myservo.write(90);
fire=false;
}void loop()
{
    myservo.write(90); //Sweep_Servo();
    if (digitalRead(Left_S) ==1 && digitalRead(Right_S)==1 &&
digitalRead(Forward_S) ==1) //If Fire not detected all sensors are zero
    {
        //Do not move the robot
        digitalWrite(LM1, HIGH);
        digitalWrite(LM2, HIGH);
        digitalWrite(RM1, HIGH);
        digitalWrite(RM2, HIGH);
    }
    else if (digitalRead(Forward_S) ==0) //If Fire is straight ahead
    {
        //Move the robot forward
        digitalWrite(LM1, HIGH);
        digitalWrite(LM2, LOW);
        digitalWrite(RM1, HIGH);
        digitalWrite(RM2, LOW);
        fire = true;
    }
    else if (digitalRead(Left_S) ==0) //If Fire is to the left
    {
        //Move the robot left

```

```

digitalWrite(LM1, HIGH);
    digitalWrite(LM2, LOW);
    digitalWrite(RM1, HIGH);
    digitalWrite(RM2, HIGH);
}
else if (digitalRead(Right_S) ==0) //If Fire is to the right
{
    //Move the robot right
    digitalWrite(LM1, HIGH);
    digitalWrite(LM2, HIGH);
    digitalWrite(RM1, HIGH);
    digitalWrite(RM2, LOW);
}
delay(300); //Slow down the speed of robot
    while (fire == true)
    {
        put_off_fire();
    }
}

```