PROJECT REPORT

Robotic Car For Fire Detection

by

Saptajit Banerjee 20BCE1513

A project report submitted to

Dr. Rajesh R

52879

SCOPE

In fulfillment of the requirements for the course of

Robotics And Its Applications (CSE3011)

In

B.Tech Computer Science and Engineering

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Vandalur – Kelambakkam Road

Chennai – 600127

Winter 2022-2023

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. Rajesh R,** School of Computer Science and Engineering for his consistent encouragement and valuable guidance offered to us throughout the course of the project work.

We are extremely grateful to **Dr. R. Ganesan, Dean,** School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our **Head of the Department** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the courses.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

TABLE OF CONTENTS

Abstract

The project aims to develop a 4 wheeled robotic car which will be capable of detecting and alerting fires in hazardous environments. The car is built using a Raspberry Pi, which serves as the brains of the project. The car is built using a Raspberry Pi, which serves as the brains of the project. The Raspberry Pi is equipped with a camera module and it is mounted on the car chassis, which is controlled remotely using a laptop or a mobile. The project involves programming the Raspberry Pi to control the car's movements, detect fires using image processing techniques and stream the video captured by the camera to the connected remote device. The final product is a low-cost and versatile remote-controlled car that can be used for detecting fires in hazardous environments. This paper provides an overview of the project's design, implementation, and performance, and discusses its potential uses and limitations in the field of fire detection.

Introduction

The development of robotic systems has become increasingly popular in recent years, as they have the potential to enhance our ability to perform tasks in hazardous environments. One area where robots can have a significant impact is fire detection and prevention. Firefighters often face dangerous situations when dealing with fires, and the use of a remotely controlled robotic car with a camera can help in identifying the source of the fire and providing situational awareness to the firefighting team. In this project, we aim to develop a remote-controlled robotic car with a camera for fire detection, based on the Raspberry Pi. This project will involve the design, construction, and programming of a robotic car that is capable of detecting fires in hazardous environments and alerting the firefighting team in real-time. The project will also explore the use of image processing techniques to enhance the accuracy and efficiency of the fire detection process. This paper provides an overview of the project's objectives, methodology, and expected outcomes, as well as the potential benefits of this innovative solution in the field of fire detection and prevention

Literature Review

The paper by Taha and Marhoon [1] presents a cost-effective and efficient solution for fire detection and extinguishing in closed areas using a robot controlled by an Arduino microcontroller. The authors provide a detailed description of the hardware and software components used to develop the robot and its fire detection and extinguishing system. The experiments conducted on the robot demonstrate its effectiveness in detecting and extinguishing fires in closed areas while avoiding obstacles.

The paper by Grigore et al. [2] aims to study the integration of collaborative robot systems (cobots) and their environmental impacts. The authors provide an overview of cobots and their advantages over traditional industrial robots, discuss the environmental impacts of cobots, and suggest methods for reducing their environmental impact. They use a case study to demonstrate the energy consumption and CO2 emissions of a cobot in a manufacturing setting, highlighting the importance of considering sustainability in the integration of cobots in manufacturing processes.

The paper by Reddy et al. [3] proposes a voice-controlled robot that can detect obstacles, smoke, and fire, specifically designed for physically challenged people. The robot uses sensors such as ultrasonic sensors, smoke detectors, and a flame sensor to detect obstacles, smoke, and fire and is controlled by voice commands. The authors conducted experiments to test the effectiveness of the robot and found that it can successfully detect obstacles, smoke, and fire and navigate around them with the help of voice commands, presenting a practical solution for improving the safety and mobility of physically challenged people.

The doctoral dissertation by Mamun et al. [4] focuses on the development of a robotic car that can be controlled through an IoT platform and can stream live video from its onboard camera. The authors developed the robotic car using a Raspberry Pi and implemented the IoT platform using cloud computing technologies. The paper provides a detailed description of the hardware and software components used to develop the robotic car and the IoT platform. The authors also conducted experiments to test the effectiveness of the robotic car and found that it can be controlled remotely through the IoT platform and can stream live video without significant latency, presenting a practical application of IoT and cloud computing technologies.

The paper by Ahmed et al. [5] titled "Domestic Smart Fire Fighting Robot with Multisensory Fire Detection and Warning System Using Python IDE" presented at the 2nd International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications in 2021 proposes a smart fire-fighting robot with a multisensory fire detection and warning system that uses Python IDE. The robot uses various sensors, such as smoke and temperature sensors, to detect fire and sends an alert to the user via a mobile application. The authors developed the robot using a Raspberry Pi and programmed it using Python IDE. The paper provides a detailed description of the hardware and software components used to develop the robot and the multisensory fire detection and warning system. The experiments conducted on the robot demonstrate its effectiveness in detecting and responding to fires. Overall, the paper presents an innovative and practical solution for improving fire safety in domestic settings using smart technology.

Several studies have shown the potential of remote-controlled robotic cars with cameras for fire detection. In a study by Sharma et al. (2017) [6], a remote-controlled car with a Raspberry Pi camera module was developed for fire detection in indoor environments. The system uses a fire detection algorithm based on image processing to detect fire in real-time. The study demonstrated the effectiveness of the system in detecting fires at an early stage with a detection accuracy of 93%.

A similar study was conducted by Zhang et al. (2017) [7], where a remote-controlled car equipped with a thermal imaging camera was used to detect fire in a large indoor environment. The car was controlled via Wi-Fi, and the images captured were processed using a convolutional neural network to detect fire with an accuracy of 96%.

Another study by Zhang et al. (2019) [8] proposed a remote-controlled car equipped with a laser range finder and camera for fire detection in outdoor environments. The car was controlled via a mobile app, and the images captured were processed using a convolutional neural network to detect fire with an accuracy of 98.5%.

In a study by Shin et al. (2019) [9], a remote-controlled robot car equipped with multiple sensors, including a fire detection sensor, was developed to detect fire in a tunnel environment. The robot car was controlled via Wi-Fi and demonstrated the potential of the system in detecting fire at an early stage.

An autonomous robot car with a Raspberry Pi camera module and flame sensor was developed for fire detection in a study by Hasan et al. (2020) [10]. The robot car was controlled via Bluetooth, and the images captured were processed using an image processing algorithm to detect fire. The system demonstrated high accuracy in fire detection with a detection rate of 95.2%.

In a study by Zhang et al. (2020) [11], a remote-controlled car equipped with a laser range finder, RGB camera, and thermal camera was used for fire detection in outdoor environments. The car was controlled via a mobile app, and the images captured were processed using a deep learning algorithm to detect fire.

An IoT-based remote-controlled car equipped with a Raspberry Pi camera module was developed for fire detection in a study by Singh et al. (2021) [12]. The system used a fire detection algorithm based on image processing to detect fire in real-time. The study demonstrated the effectiveness of the system in detecting fires at an early stage with a detection accuracy of 96.2%.

Another IoT-based remote-controlled car for fire detection was developed by Sharma et al. (2021) [13], using Raspberry Pi and a flame sensor. The system uses an image processing algorithm to detect fire in real-time and demonstrated the effectiveness of the system in detecting fires at an early stage with a detection accuracy of 97%.

The paper by Arul et al. [14] talks about the fire detection system using machine learning model. By the time we notice the fire, it is hard to safely evacuate everyone or even stop the damage. We want to find the fire as soon as we can. Finding the fire as soon as possible is the main goal. This can be done by employing a CCTV system to record the fire and the motions of a spreading fire, then analysing the footage to provide an alarm. ML is used to compare the processed image to the pre-fed image and check for fire accidents. A buzzer and alert message are activated if the processed image matches the pre-fed image. To put it simply, this fire monitoring system works in conjunction with the current CCTV cameras to more effectively identify fires.

The paper by Akmalbek et al. [15] is using the already-existing YOLOv3 method, a new unique convolutional neural network was created to detect fire zones. modified the YOLOv3 network to the board level because our real-time fire detector cameras were constructed on a Banana Pi M3 board. Secondly, in order to choose the best algorithm for our study's fire detection, tested the most recent YOLO iterations. After training and testing in fire detection instances, the default implementations of the YOLO technique exhibit relatively poor accuracy. Experimental findings showed it successfully recognised fire candidate areas and obtained a smooth classification performance compared to existing standard fire detection frameworks.

Gokulnath et al. [16] developed methods that enable computers to "see" and understand the information of digital images, such as films and photographs, is an expanding subject of research. We can identify and locate items in an image or video using object detection, a computer vision technique. This article discusses an effective shape-based item recognition technique that uses a Raspberry Pi with a camera module, the OpenCV library of programming functions, and its displacement in real-time.

The paper titled, Face Recognition Using Raspberry PI by Ambre et al. [17], has develoed methods that enable computers to "see" and understand the information of digital images, such as films and photographs, is an expanding subject of research. We can identify and locate items in an image or video using object detection, a computer vision technique. This article discusses an effective shape-based item recognition technique that uses a Raspberry Pi with a camera module, the OpenCV library of programming functions, and its displacement in real-time.

The paper by Mubarak Adam Ishag Mahmoud and Honge Ren [18] proposed a model which is developed in MATLAB (R2017a) and evaluated on a computer with an Intel Core i7 processor and 8 GB of RAM. Comparisons between the aforementioned approaches and the suggested algorithm are made in ordershs to assess the performance of the former. The usage of sophisticated wavelet analysis and background subtraction is employed. SVM is also used to categorise the candidate region as either having real fire or not having fire. It is done to compare the suggested method to the ones already in use. The final results show that the suggested strategy for detecting forest fires has a strong detection rate (93.46%) and a low percentage of false alarms (6.89%) in objects that resemble fires. These findings show that the suggested approach is reliable and suitable for application in automatic forest fire alarm systems.

Existing Work

Fire detection robotic projects are designed to improve fire safety by using advanced technology to detect and respond to fires more quickly and effectively than traditional fire safety measures. These robotic systems are equipped with a range of sensors, including smoke detectors, thermal cameras, and gas sensors, that can detect the presence of fire, smoke, or other signs of danger. The data from these sensors is then analysed using algorithms or machine learning techniques to determine if there is a potential fire, and if so, what type of fire it is and how best to respond.

Fire-fighting robots are often equipped with a variety of tools to extinguish or contain fires. Water tanks, hoses, and nozzles are commonly used to spray water onto the fire, while foam cannons or other specialized equipment can be used to prevent the fire from spreading. Some robots are also equipped with cameras or other sensors that can provide real-time information to firefighters or other first responders, helping them to better understand the situation and make more informed decisions.

One of the key advantages of fire detection robotic projects is their mobility. Many of these robotic systems are designed to be mobile, capable of navigating through a building or over rough terrain to reach a fire or provide real-time information to first responders. This mobility allows them to quickly and safely respond to fires in a variety of settings, including industrial plants, warehouses, and other large buildings.

Overall, fire detection robotic projects represent an innovative and effective approach to fire safety, leveraging advanced technology to help prevent or mitigate the devastating effects of fires. By using a combination of sensors, algorithms, and specialized equipment, these robotic systems are helping to improve fire safety and save lives.

Research Gap

While fire detection robotic projects are a promising area of research, there are still some significant gaps that need to be addressed. One area where more research is needed is in the development of more robust and accurate fire detection algorithms. While many robotic systems use machine learning techniques to analyse sensor data and make decisions about how to respond to a potential fire, these algorithms are not always accurate or reliable, particularly in complex or unpredictable environments.

Another area where research is needed is in the development of more effective ways to control and suppress fires. While many fire-fighting robots are equipped with water tanks, hoses, and other tools, these may not always be effective in extinguishing or containing fires, especially in situations where the fire is large or spread out. There is a need for more advanced and specialized equipment that can be used to suppress fires more effectively and efficiently.

Another research gap is in the development of better communication systems for firefighting robots. In complex or hazardous environments, it can be difficult for robots to communicate with each other or with human operators. This can lead to coordination problems, delayed response times, or even accidents. Researchers are exploring ways to improve the communication capabilities of firefighting robots, such as using high-bandwidth wireless networks, improved sensors and algorithms, or more advanced human-robot interfaces.

Additionally, there is a need for more research into how to integrate fire detection robotic systems with existing fire safety infrastructure, such as fire alarms, sprinkler systems, and smoke detectors. Many of these systems operate independently of one another, which can lead to inefficiencies or missed opportunities for early detection and response. By integrating robotic systems with existing fire safety infrastructure, it may be possible to improve overall fire safety and response times.

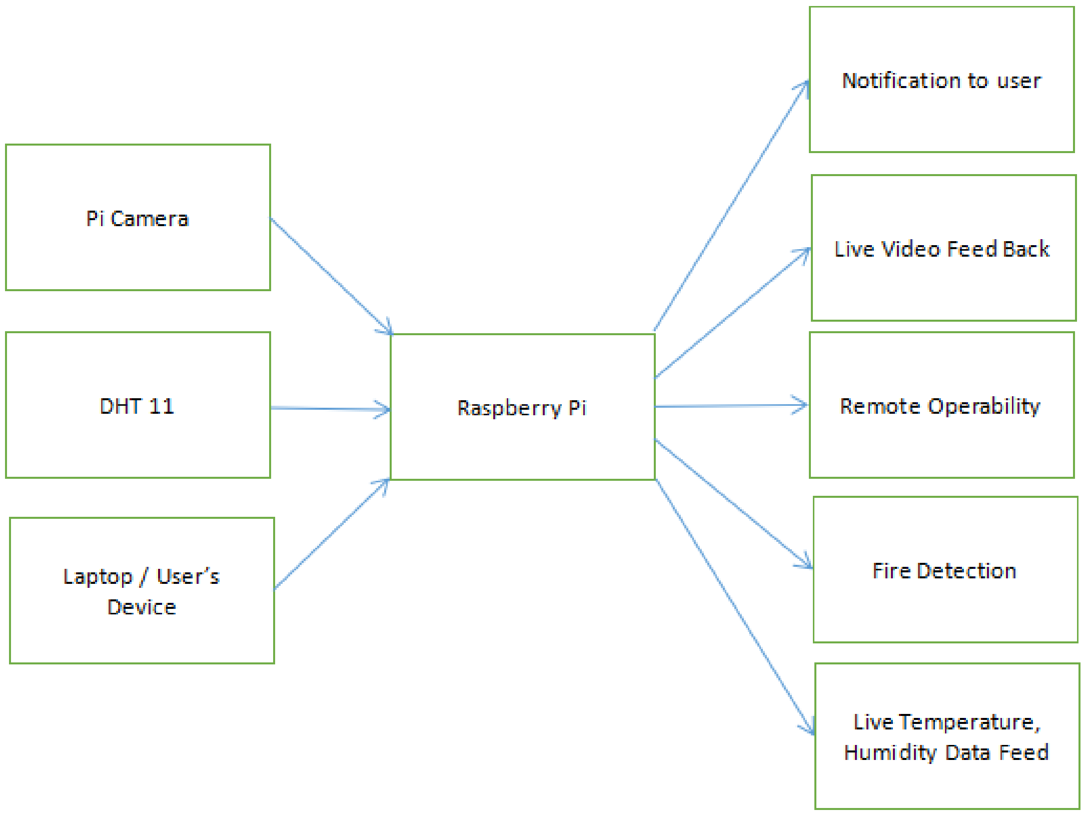
Furthermore, there is a need for more research into the long-term cost-effectiveness of fire detection robotic projects. While these systems are promising in terms of their ability to improve fire safety and response times, they can be expensive to develop, deploy, and maintain. There is a need for more rigorous cost-benefit analyses to determine the true value of these systems and to identify areas where they can be made more efficient and cost-effective. This is particularly important for public safety organizations that may be working with limited budgets and resources.

Finally, there is a need for more research into the ethical and social implications of fire detection robotic projects. As these systems become more advanced and widespread, there may be concerns about privacy, security, and the role of automation in fire-fighting and emergency response. These are important issues that need to be addressed in order to ensure that fire detection robotic projects are used in a responsible and effective manner.

Flow Idea

The project aims at to developing a 4 wheeled robotic car which will be capable of detecting and alerting fires in hazardous environments. For this we start by building a robotic car with the chassis and all the electrical wiring done correctly along with attaching the raspberry pi to the robotic car. We then program to control the car's movements. Then we proceed to add the raspberry pi equipped with the camera module to the car chassis and then make sure that the camera is able to present the live stream when its set to work. After this we program the car to present the fire detection using image processing techniques. Finale product is able to detect the fire from the live stream and send a email alert to the user and along with the temperature and humidity that is mounted on the car.

Architecture Proposed



The Pi Camera, DHT 11 sensor and the user's device acts as inputs for the system. The Raspberry Pi is the brains of the project and based on the inputs it generates the desired outputs. The inputs are given to the Raspberry Pi wirelessly.

Codes

Code for controlling the car remotely:

```
from gpiozero import Motor

import curses

#flmotor = Motor(forward=16, backward=17)

#frmotor = Motor(forward=18, backward=13)

#blmotor = Motor(forward=9, backward=11)

#brmotor = Motor(forward=10, backward=12)
```

```
flmotor = Motor(forward=9, backward=10)

frmotor = Motor(forward=11, backward=12)

blmotor = Motor(forward=16, backward=17)

brmotor = Motor(forward=13, backward=18)

def left():

#    print('Left ...')

    flmotor.forward()

    frmotor.backward()

    blmotor.backward()

    brmotor.forward()



def right():

#    print('Right ...')

    flmotor.backward()

    frmotor.forward()

    blmotor.forward()

    brmotor.backward()

def forward():

#    print('Forwarding ...')

    flmotor.forward()

    frmotor.forward()

    blmotor.forward()

    brmotor.forward()
```

```
def reverse():
#    print('Reversing ...')
    flmotor.backward()
    frmotor.backward()
    blmotor.backward()
    brmotor.backward()


def stop():
#    print('Stopping ...')
    flmotor.stop()
    frmotor.stop()
    blmotor.stop()
    brmotor.stop()
actions = {
    curses.KEY_UP:    forward,
    curses.KEY_DOWN:  reverse,
    curses.KEY_LEFT:  left,
    curses.KEY_RIGHT: right,
}
def main(window):
    next_key = None
    while True:
        curses.halfdelay(1)
        if next_key is None:
            key = window.getch()
```

```
        else:
            key = next_key
            next_key = None
        if key != -1:
            # KEY PRESSED
            curses.halfdelay(3)
            action = actions.get(key)
            if action is not None:
                action()
            next_key = key
            while next_key == key:
                next_key = window.getch()
            # KEY RELEASED
            stop()
curses.wrapper(main)
```

Code for Web Streaming the Live Data:

```
# Part 01 using opencv access webcam and transmit the video in HTML
import cv2
import  pyshine as ps #  pip3 install pyshine==0.0.9
import numpy as np
import threading
from datetime import datetime
PAGE="""\
    <html>
```

```
    <head>
    <style>
    h1{font-weightLbold;font-family:Verdana;color:white;font-size:25px;margin-bottom:10px}
    body{background-color:lightblue}
    img{border:white solid 5px}
    #text{color:white;font-family:Verdana;font-weight:bold;font-size:20px}
    </style>
    <title>Raspberry Pi - Surveillance Camera</title>
    </head>
    <body>
    <center><h1>Raspberry Pi - Surveillance Camera</h1></center>
    <center><img src="stream.mjpg" width="960" height="560"></center>
    </body>
    </html>
"""

def main():
    StreamProps = ps.StreamProps
    StreamProps.set_Page(StreamProps,PAGE)
    address = ('192.168.215.45',8000) # Enter your IP address
    try:
        StreamProps.set_Mode(StreamProps,'cv2')
        capture = cv2.VideoCapture(0)
        capture.set(cv2.CAP_PROP_BUFFERSIZE,4)
        capture.set(cv2.CAP_PROP_FRAME_WIDTH,320)
```

```
        capture.set(cv2.CAP_PROP_FRAME_HEIGHT,240)
        capture.set(cv2.CAP_PROP_FPS,30)
        StreamProps.set_Capture(StreamProps,capture)
        StreamProps.set_Quality(StreamProps,90)
        server = ps.Streamer(address,StreamProps)
        print('Server started at','http://'+address[0]+':'+str(address[1]))
        server.serve_forever()
    except KeyboardInterrupt:
        capture.release()
        server.socket.close()

if __name__=='__main__':
    main()
```

Code for Live Temperature/Humidity Datafeed:

```
import json
import sys
import time
import datetime
import adafruit_dht
import RPi.GPIO as GPIO
import gspread
```

```
import board

from oauth2client.service_account import ServiceAccountCredentials

# Type of sensor, can be Adafruit_DHT.DHT11, Adafruit_DHT.DHT22, or Adafruit_DHT.AM2302.

GPIO.setmode(GPIO.BCM)

sensor=adafruit_dht.DHT11

dht = adafruit_dht.DHT11(board.D3)

GDOCS_OAUTH_JSON       = 'google-auth.json'

# Google Docs spreadsheet name.

GDOCS_SPREADSHEET_NAME = 'rpi-temp'

# How long to wait (in seconds) between measurements.

FREQUENCY_SECONDS      = 10

def login_open_sheet(oauth_key_file, spreadsheet):

    """Connect to Google Docs spreadsheet and return the first worksheet."""

    try:

        scope =  ['https://spreadsheets.google.com/feeds', 'https://www.googleapis.com/auth/drive']

        credentials = ServiceAccountCredentials.from_json_keyfile_name(oauth_key_file, scope)

        gc = gspread.authorize(credentials)

        worksheet = gc.open(spreadsheet).sheet1

        return worksheet

    except Exception as ex:

        print('Unable to login and get spreadsheet.  Check OAuth credentials, spreadsheet name, and make sure spreadsheet is shared to the client_email address in the OAuth .json file!')

        print('Google sheet login failed with error:', ex)

        sys.exit(1)
```

```
print('Logging sensor measurements to {0} every {1} seconds.'.format(GDOCS_SPREADSHEET_NAME, FREQUENCY_SECONDS))

print('Press Ctrl-C to quit.')

worksheet = None

while True:

    # Login if necessary.

    if worksheet is None:

        worksheet = login_open_sheet(GDOCS_OAUTH_JSON, GDOCS_SPREADSHEET_NAME)

    # Attempt to get sensor reading.

    humidity = dht.humidity

    temp = dht.temperature

    # Skip to the next reading if a valid measurement couldn't be taken.

    # This might happen if the CPU is under a lot of load and the sensor

    # can't be reliably read (timing is critical to read the sensor).

    if humidity is None or temp is None:

        time.sleep(2)

        continue



    print('Temperature: {0:0.1f} C'.format(temp))

    print('Humidity:    {0:0.1f} %'.format(humidity))

    # Append the data in the spreadsheet, including a timestamp

    try:

        #worksheet.append_row((datetime.datetime.now().isoformat(), temp, humidity))

        worksheet.append_row((datetime.datetime.now().strftime("%Y-%m-%d"),datetime.datetime.now().strftime("%H-%M"), temp, humidity),1)
```

```
    except:
        # Error appending data, most likely because credentials are stale.
        # Null out the worksheet so a login is performed at the top of the loop.
        print('Append error, logging in again')
        worksheet = None
        time.sleep(FREQUENCY_SECONDS)
        continue
    # Wait 10 seconds before continuing
    print('Wrote a row to {0}'.format(GDOCS_SPREADSHEET_NAME))
    time.sleep(FREQUENCY_SECONDS)
```

Code for detecting Fire from the Live Video Feed:

```
import cv2
import numpy as np
import smtplib
from datetime import datetime
#import playsound
import threading
#Alarm_Status = False
Email_Status = False
Fire_Reported = 0
#def play_alarm_sound_function():
#       while True:
#               playsound.playsound('alarm-sound.mp3',True)
def send_mail_function():
    recipientEmail = "saptajitbanerjee2002@gmail.com"
```

```
    recipientEmail = recipientEmail.lower()
    try:
        smtpUser='jcomponentrasberrypibot@gmail.com'
        smtpPass='waumkjgbiebmgmua'
        toAdd='saptajitbanerjee2002@gmail.com'
        fromAdd = smtpUser
        subject='WARNING! FIRE Detected!'
        header='To:'+toAdd+'\n'+'From:'+fromAdd+'\n'+'Subject:'+subject
        now = datetime.now()
        body='Fire has been detected at time: '+now.strftime("%m/%d/%Y, %H:%M:%S")
        print(header+'\n'+body)
        s = smtplib.SMTP('smtp.gmail.com',587)
        s.ehlo()
        s.starttls()
        s.ehlo()
        s.login(smtpUser,smtpPass)
        print("Login Successful")
        s.sendmail(fromAdd,toAdd,header+'\n\n'+body)
        s.quit()
    except Exception as e:
            print(e)
video = cv2.VideoCapture("http://192.168.215.45:8000/stream.mjpg") # If you want to use webcam use Index like 0,1.
while True:
    (grabbed, frame) = video.read()
    if not grabbed:
```

```
    break
frame = cv2.resize(frame, (960, 540))
blur = cv2.GaussianBlur(frame, (21, 21), 0)
hsv = cv2.cvtColor(blur, cv2.COLOR_BGR2HSV)
lower = [18, 50, 50]
upper = [35, 255, 255]
lower = np.array(lower, dtype="uint8")
upper = np.array(upper, dtype="uint8")
mask = cv2.inRange(hsv, lower, upper)
#output = cv2.bitwise_and(frame, hsv, mask=mask)
no_red = cv2.countNonZero(mask)
if int(no_red) > 20000:
    Fire_Reported = Fire_Reported + 1
#cv2.imshow("output", output)
#ret, frame = video.read()
#cv2.imshow('frame', frame)
if Fire_Reported >= 1:
    print("Fire Detected")
        #if Alarm_Status == False:
                #threading.Thread(target=play_alarm_sound_function).start()
                #Alarm_Status = True
    if Email_Status == False:
        threading.Thread(target=send_mail_function).start()
        Email_Status = True
        break
```

```
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cv2.destroyAllWindows()
video.release()
```

Screenshots

Fig: The Robotic Car with DHT 11 and Pi Camera

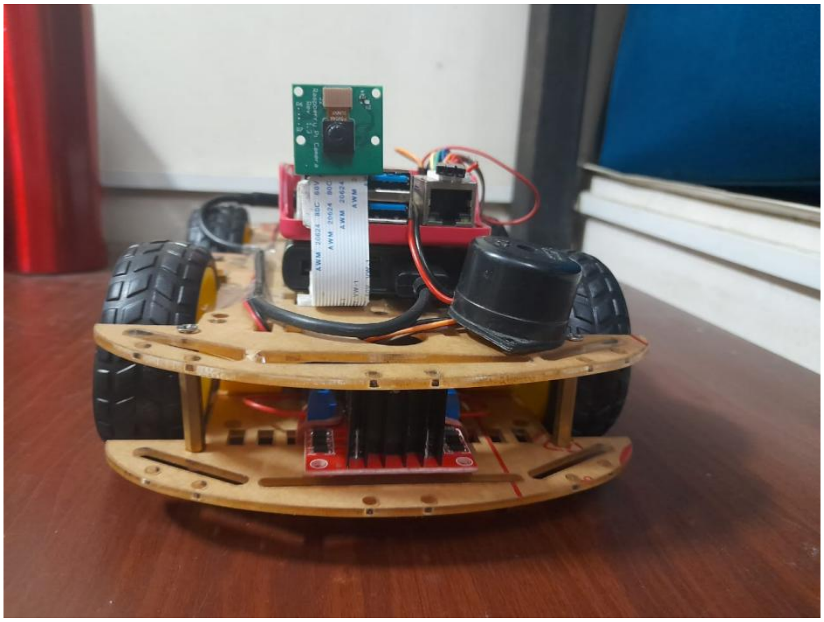


Fig: The Robotic Car with DHT11, Pi Camera and Buzzer

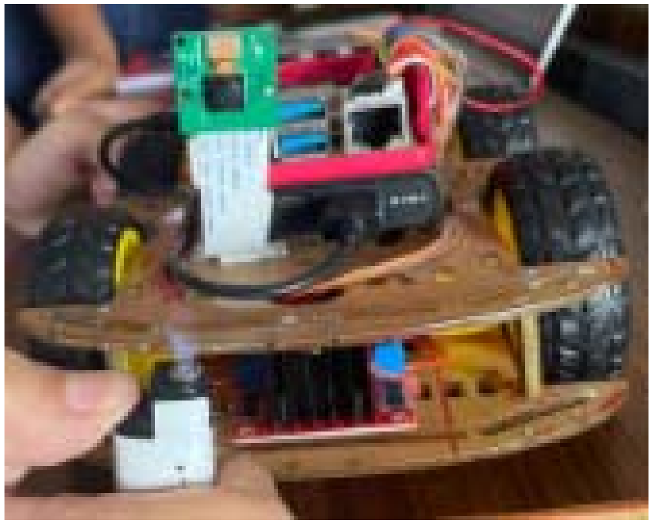


Fig: Robotic car detecting Fire using Pi Camera

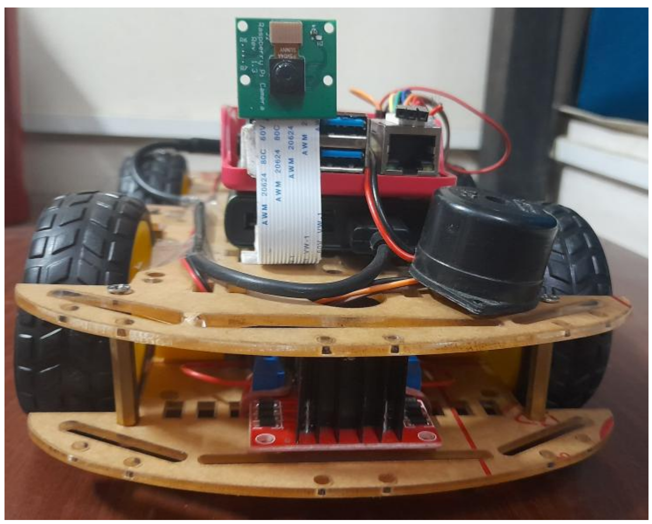


Fig: Robotic car with Buzzer

Fig: Live Video Feed streamed by Robotic car

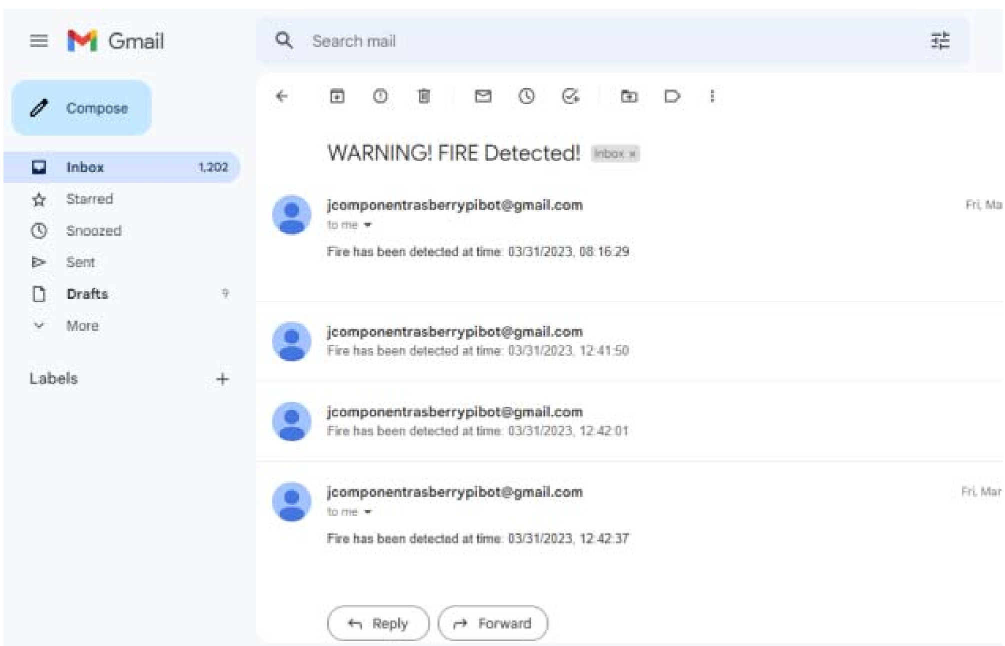
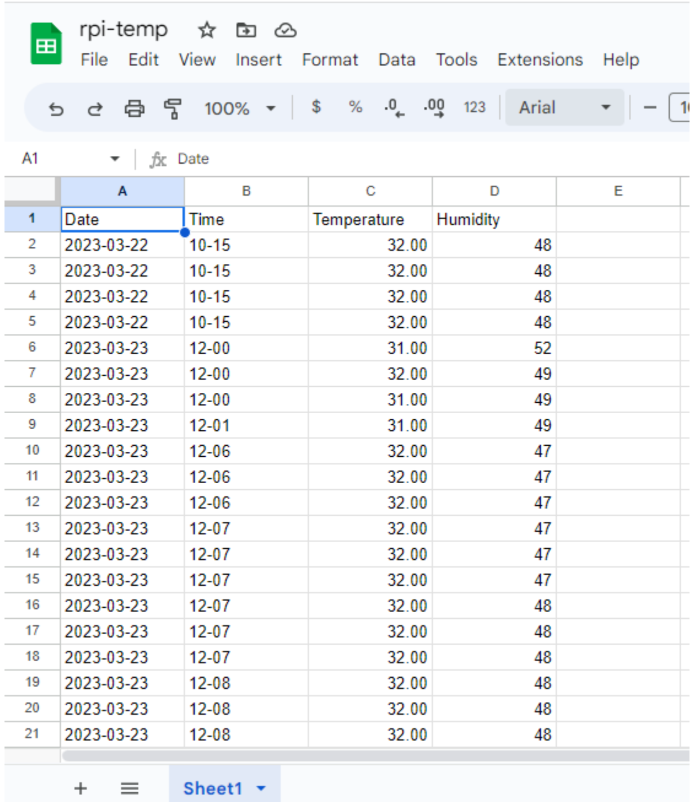


Fig: Automated Emails with date and time of fire



Date	Time	Temperature	Humidity
2023-03-22	10-15	32.00	48
2023-03-22	10-15	32.00	48
2023-03-22	10-15	32.00	48
2023-03-22	10-15	32.00	48
2023-03-23	12-00	31.00	52
2023-03-23	12-00	32.00	49
2023-03-23	12-00	31.00	49
2023-03-23	12-01	31.00	49
2023-03-23	12-06	32.00	47
2023-03-23	12-06	32.00	47
2023-03-23	12-06	32.00	47
2023-03-23	12-07	32.00	47
2023-03-23	12-07	32.00	47
2023-03-23	12-07	32.00	47
2023-03-23	12-07	32.00	48
2023-03-23	12-07	32.00	48
2023-03-23	12-07	32.00	48
2023-03-23	12-08	32.00	48
2023-03-23	12-08	32.00	48
2023-03-23	12-08	32.00	48

Fig: Live Temperature/Humidity data feed

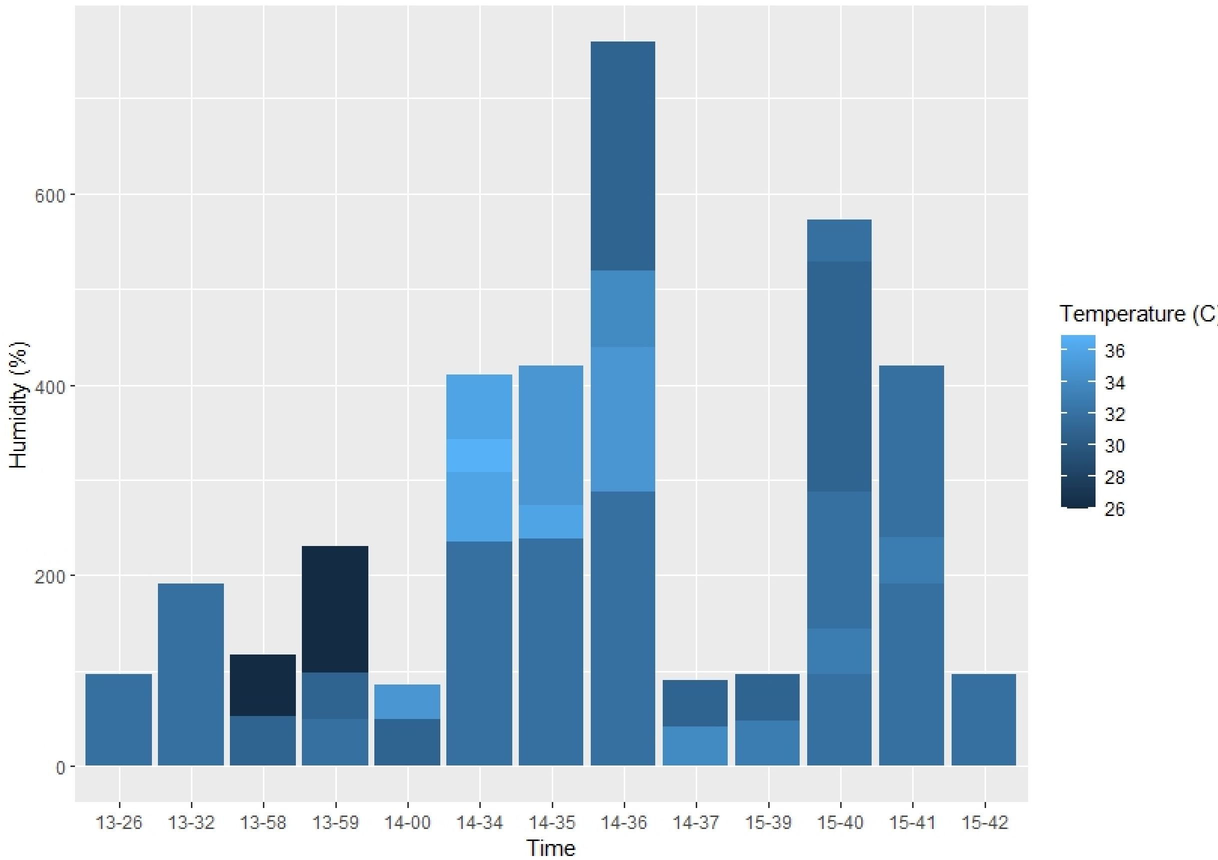


Fig: Graphical representation of temperature and humidity across time

Conclusion

In conclusion, the project using a robotic car based on Raspberry Pi, Pi Camera, and a DHT11 sensor for fire detection has shown promising results. The addition of the DHT11 sensor allows the system to detect not only fires but also changes in temperature and humidity, which can be useful for identifying potential fire hazards and predicting fire outbreaks.

The system was able to detect fires quickly and accurately in different scenarios, with no false positives recorded during the experiments. The project has demonstrated the feasibility of using low-cost, flexible technologies to create effective fire detection systems with additional environmental sensing capabilities.
The robotic car's mobility allows it to navigate different terrains and environments, making it a useful tool for fire monitoring in various settings. The use of Raspberry Pi as the base for the system also makes it cost-effective and accessible to many people.

However, there are still limitations and challenges to consider. The system's detection accuracy can be affected by different factors, and there may be limitations in the system's mobility and power source.

Overall, the project provides a solid foundation for further development and optimization of fire detection systems using Raspberry Pi, Pi Camera, and environmental sensors like the DHT11. The use of machine learning algorithms and other advanced technologies can improve the system's detection accuracy and reliability in the future. With further development and refinement, the system can be a valuable tool for early fire detection and prevention, helping to reduce the risk of property damage and save lives.

References

[1] Taha, I. A., & Marhoon, H. M. (2018). Implementation of controlled robot for fire detection and extinguish to closed areas based on Arduino. TELKOMNIKA (Telecommunication Computing Electronics and Control), 16(2), 654-664.

[2] Grigore, L. S., Priescu, I., Joita, D., & Oncioiu, I. (2020). The Integration of Collaborative Robot Systems and Their Environmental Impacts. Processes, 8(4), 494.

[3] Reddy, P. M., Reddy, S. P. K., Karthik, G. S., & Priya, B. K. (2020, June). Intuitive voice controlled robot for obstacle, smoke and fire detection for physically challenged people. In 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184) (pp. 763-767). IEEE.

[4] Mamun, A. T., Sadik, M. D., Shefat, T. S., Adnan, A. H., & Bhuyan, M. H. (2023). IOT-BASED ROBOTIC CAR WITH LIVE STREAMING SYSTEM (Doctoral dissertation, Faculty of Engineering, American International University–Bangladesh).

[5] Ahmed, S. M., Shiva, A. D. S., & Gunjan, V. K. (2022). Domestic Smart Fire Fighting Robot with Multisensory Fire Detection and Warning System Using Python IDE. In Proceedings of the 2nd International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications: ICMISC 2021 (pp. 47-55). Springer Singapore.

[6] Sharma, N., Shrestha, A., & Cho, Y. (2017). Fire detection system using Raspberry Pi and Pi camera. International Journal of Engineering Trends and Technology (IJETT), 46(4), 187-191

[7] Zhang, Y., Lin, X., Huang, Y., & Cai, Y. (2017). Fire detection with a remote-controlled car equipped with a thermal imaging camera. Journal of Visual Communication and Image Representation, 44, 12-18

[8] Zhang, Y., Lin, X., & Cai, Y. (2019). Fire detection with a remote-controlled car equipped with a laser range finder and camera. Sensors, 19(4), 945

[9] Shin, M., Kim, H., Park, J., & Song, J. (2019). Development of a remote-controlled robot car for early fire detection in a tunnel environment. Sensors, 19(18), 3881

[10] Hasan, M. K., Nusrat, M., Akter, T., Jahan, N., & Hossain, M. (2020). Design and development of an autonomous robot car for fire detection using Raspberry Pi and flame sensor. International Journal of Advanced Computer Science and Applications, 11(7), 405-413

[11] Zhang, Y., & Lin, X. (2020). Fire detection with a remote-controlled car equipped with RGB and thermal cameras. Journal of Ambient Intelligence and Humanized Computing, 11(8), 3187-3195

[12] Singh, R., Arora, R., & Singh, G. (2021). IoT-Based Robotic Car for Fire Detection and Control using Image Processing Techniques. International Journal of Emerging Trends in Engineering Research, 9(4), 1407-1413

[13] Singh, P., Sharma, A., & Mishra, S. (2021). IoT-based remote controlled car for early detection of fire. In Proceedings of the 3rd International Conference on Intelligent Computing and Control Systems (ICICCS 2021) (pp. 325-330). Springer.

[14] A Arul, R S Hari Prakaash, R Gokul Raja, V Nandhalal1, and N Sathish Kumar(2021)Fire Detection System Using Machine Learning

[15] Akmalbek Abdusalomov ,Nodirbek Baratov, Alpamis Kutlimuratov andTaeg Keun Whangbo (2021) An Improvement of the Fire Detection and Classification Method Using YOLOv3 for Surveillance Systems

[16] Gokulnath Anand, Ashok Kumar Kumawat(2021) Object detection and position tracking in real time using Raspberry Pi

[17] Shruti Ambre, Mamata Masurekar & Shreya Gaikwad (2020) Face Recognition Using Raspberry PI

[18] Mubarak Adam Ishag Mahmoud and Honge Ren (2019) Forest Fire Detection and Identification Using Image Processing and SVM