CHAPTER 1

INTRODUCTION

The security system designed in this project is a simple and easily installable device built using Raspberry Pi 3, Pi camera and some sensors. The Pi camera captures the images and these images are temporarily stored on Raspberry Pi and pushed to Google Cloud from where they are sent as email alert to the owner.

The system detects fire and motion. It has a camera which is activated in correlation with detection. So, the user gets the images of any visitor immediately on email which he can check from his smartphone. This project is focused on developing a surveillance system that detects motion and to respond speedily by capturing an image and relaying it to an administrator device through the internet platform. The system will require Raspberry Pi module, motion detection sensor, camera and internet connection. It will come up with an implementation of a surveillance system which presents the idea of monitoring a particular place in remote areas. The system can be monitored by the user form anywhere in the world. However, this project will not attempt to design the motion detection device, camera or the Raspberry PI. It will therefore use these systems together with a suitable program script to accomplish a real time surveillance system as desired.

Nowadays, people attach much more importance to home security. Home security is related to both the security hardware in place on a property and personal security practices. Security hardware includes doors, locks, alarm systems, lighting, motion detectors, security camera systems, etc. that are installed on a property. However, current home security devices need to update the personal data to the cloud, which brings users the risk of losing privacy. What’s more, since all data are processed in the cloud server, it is inconvenient for users to change control conditions.

CHAPTER 2

LITERATURE SURVEY

2.1 Home Based Security System using Raspberry Pi and GSM

ABSTRACT: Home security and automation are becoming increasingly prominent features on mobile devices. The aim of this paper is to design and implement affordable, flexible and fast monitoring home security system using Raspberry pi with GSM technology. The system is designed to detect burglary, the image of person is captured by camera and sends to mobile and email as well as alarm gets on; and leaking in harmful gas, the smoke caused by fire as such suspicious activity is detected. Also the user can activate all the alarm system while going outside through the mobile. The messenger has the feasibility of activating and deactivating the alarm system with the additional control for some home appliances switching using relays. For controlling, raspberry pi module, GSM technology is used along with camera, PIR sensor, Vibration sensor, air quality sensor. It is only applicable for magnetic door lock system.

CONCLUSION: This paper presents the design and implementation of a low cost but secure home security system for general users. The security level is increased due to the usage of Raspberry pi which sends the images to the user, has in built capabilities and is easily connectible to external devices. Raspberry pi proves to be smart economic and efficient platform for implementing the home security system. Two advantages provided by the system is that, Necessary action can be taken in short span of time in the case of emergency condition and design of a PCB board which is also small in size. Reduced size makes it more applicable for commercial manufacturing and distribution. A raspberry pi and open-source applications with its ever-growing community and development provides a great hope in the near future.

2.2 IoT Based Security System using Raspberry Pi and Mail Server

ABSTRACT: Personal security and surveillance is a hot topic in the field of research and computer vision. The CCTV is used for surveillance and security purpose. The crime detection is possible with surveillance but if the face recognition is integrated with surveillance the performance of existing surveillance and security system can be enhanced. The smart surveillance system using face recognition can be used for a verity of application such as attendance system for students and employees at private and government offices, research labs, ATM etc. In this paper, one application of smart surveillance called hostel attendance system is demonstrated.

CONLUSION: The technology used can be performed in a variety of applications that require a sensor and a device. The project has been successfully developed to connect a mobile device, such as a smartphone or laptop using the Raspberry Pi, for controlling the door and switches of light and camera to broadcast live video, but there are many possible applications that can take advantage of this. The ultimate goal of this file - to create a home security system using raspberry slices. So it can help people feel safe in their home, whether they are home. This project is based on the modern type of the Raspberry model, and the C language. It is low cost and can be easily managed. Even though our homes will experience their own home conversion, constant interaction with networks that constantly seek to improve energy management and full home automation to ensure comfort, safety and privacy.

2.3 Real Time Smart Surveillance System using Raspberry Pi

ABSTRACT: Personal security and surveillance is a hot topic in the field of research and computer vision. The CCTV is used for surveillance and security purpose. The crime detection is possible with surveillance but if the face recognition is integrated with surveillance the performance of existing surveillance and security system can be enhanced. The smart surveillance system using face recognition can be used for a verity of application such as attendance system for students and employees at private and government offices, research labs, ATM etc. In this paper, one application of smart surveillance called hostel attendance system is demonstrated.

CONCLUSION: A face is an important identity of human beings. The proposed system uses a webcam to capture images for training and record video for testing. The system uses one image per user for training purpose. It reduces memory consumption as well as the processing power of Raspberry Pi. The system can work with little environmental change and pose variation as explained in the result section. The system gives 100% and about 95 % of face detection and faces recognition accuracy assuming 1-meter distance between camera and user. After the recognition .CSV file is and stored onto the cloud to maintain attendance records of the day. The future thought is to use a high-resolution camera. The system performance will be improved by integrating more facial features and by applying a more efficient algorithm. Also to add some elements to the system so that it can be controlled and monitored through Android phone. In future this system will be used at the traffic signal to recognize vehicles and pedestrians and face recognition could be used ATM password, bank locker password or at research labs etc.

Author	**Source**	**Paper**	**Findings**
[1] Akash V. Bhatkal, Ulhas B. Shinde, Shrinivas R. Zanwar (September 2016)	Home Based Security Control System using Raspberry Pi and GSM	IJIRCCE,Vol. 4,2016.	The system uses sensors for security system and sends the alerts using GSM.
[2] S. Snigdha, K. Haribabu (September 2019)	IoT based Security System using Raspberry PI and Mail Server	IJIRCCE,Vol. 8,2019.	The system uses sensors for security system and sends the alerts using Mail server.
[3] Dhanshri Mali, R.T. Patil, Nagaraj Dharwadkar, Chaitanya R. Devale, Omprakash Tembhurne	Real-Time Smart Surveillance System Using Raspberry Pi	Rajarambapu Institute, Rajaramnagar,24 Feb 19.	The surveillance system uses PIR Sensor with Raspberry Pi.
[4] Deepak. S. Kumbar, H.C. Chaudhari, Shubhangi M.Taur, Shubhangi S. Bhatambekar	IOT Based Home Security System Using Raspberry Pi-3	IJRAR, Vol 6, January 2019.	The system uses raspberry pi 3 for the security system.

Table 2.1 Consolidated Survey Report

CHAPTER 3

OBJECTIVES

- The main aim of this project is to design and develop a security system that includes features such as motion detection, image processing and emailing to facility owner. The system is to be based on Raspberry Pi SBC.
- The specific objectives are: To study and describe how the Raspberry Pi can be interfaced with a motion detector and Pi camera. To study how a Raspberry Pi can be programmed so as to be able to send an email to a prescribed mail hub. To design and implement a motion detecting, gas detection and flame detection.
- The security system to be designed in this project can be used extensively to monitor facilities by owners. The owner shall be able to monitor their property from wherever they are in the world. It will not replace the use of CCTV and camera surveillance systems but reduce the cost of implementation of a basic security system.
- To study and describe how the Raspberry Pi can be interfaced with a motion detector and Pi camera.

3.1 Scope of the project

This project is focused on developing a surveillance system that detects motion and to respond speedily by capturing an image and relaying it to an administrator device through the internet platform. The system will require Raspberry Pi module, motion detection sensor, camera and internet connection. It will come up with an implementation of a surveillance system which presents the idea of monitoring a particular place in remote areas. The system can be monitored by the user form anywhere in the world. However, this project will not attempt to design the motion detection device, camera or the Raspberry PI. It will therefore use these systems together with a suitable program script to accomplish a real time surveillance system as desired.

3.2 Related Works on security Systems

In the present day, researchers and developers have come up with a wide range of surveillance systems that are used for remote monitoring, alerting as well as controlling tasks through affordable and easy to implement hardware systems. Some have so far been realized while others still remain a proposition. An embedded home surveillance system which assesses the implementation of a cost-effective alerting system based on small motion detection was presented by Padmasree A. Shake and Sumida S. Borde. They worked on implementing cheap in price, low power consumption; well utilize resources and efficient surveillance system using a set of various sensors. Their system helps to monitor the household activities in real time from anywhere and based on microcontroller which is considered nowadays as a limited resource and an open source solution compared to SBC. D. Jeevanand worked on designing of a networked video capture system using Raspberry Pi. The proposed system works on capturing video and distributing with networked systems besides alerting the administration person via SMS alarm as required by the client. Their system was designed to work in a real-time situation and based on Raspberry Pi SBC. Contrasting to other embedded systems their real-time application offers client video monitor with the help of alerting module and SBC platform. Sneha Singhd and his team described IP Camera Video Surveillance system using Raspberry Pi technology. The Researchers aimed at developing a system which captures real time images and displays them in the browser using TCP/IP. The algorithm for face detection is being implemented on Raspberry Pi, which enables live video streaming along with detection of human faces. The research did not include any of surveillance reactions. The design and develop a real time video surveillance system based on embedded web server Raspberry PI B+ Board.

Their system has low cost, good openness and portability and is easy to maintain and upgrade. Thus, this application system provides better security solutions. This system can be used to effect security in banking halls, industry, environment and in military

3.3 Evolution of Security

With the invention of electricity, the art of home protection was greatly improved. In 1853, the first patent on electro-magnetic alarms meant that businesses and wealthy residents could secure valuables. Magnetic contacts were installed on the windows and doors that, when tripped, would send a signal through the electromagnetic wiring and sound an alarm. These ground breaking security systems were effective in deterring break-ins from occurring. According to Cisco Expo, major strides have been made with regards to surveillance systems. After the alarm system, analogy video camera with Video Cassette Recorder evolved. It had poor imaging and no remote access. To overcome the drawbacks of this system, digital video recorders evolved. They gave good quality pictures and enable for transmission of video signals through data networks and thus allowed for remote monitoring. Network Video Recorder then emerged. They have the advantages of the DVRs but have other merits over DVRs. They give more storage options and network connection. The most superior version is the type that uses Cisco Video Surveillance Platform. They give secure remote access and control from anywhere, fail-safe redundant storage, easy integration with other systems and enterprise class storage and support.

CHAPTER 4

METHODOLOGY

The raspberry-pi microcontroller is used for this project as there are a number of tasks to be handled which call for a more sophisticated and robust processing power. Figure shows a pictorial overview of the system functions. The system detects fire and motion. It has a camera which is activated in correlation with detection.

The raspberry-pi then process the detection signals and output to the phone, speaker and drop box accordingly. The system checks if there is motion, upon detection an intrusion detection is sent to the user. The video of the scene is recorded and then uploaded in the cloud. If there is no longer motion the system records the last video and goes back to sensing motion else it loops back to the intrusion outputs. The system checks if there is fire detected, upon detection a fire detection alert is sent to the user. The video of the scene is recorded and then uploaded in the cloud. If there is no longer motion the system records the last video and goes back to sensing fire else it loops back to the fire detection outputs.

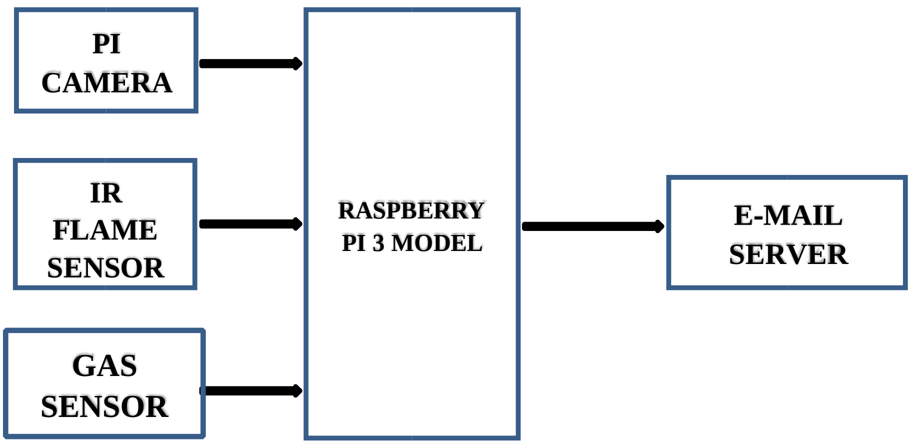

Figure 4.1: Raspberry Pi-based smart home security system

4.1 Hardware Requirements

- Raspberry Pi 3 Model B Board
- Pi Camera
- Gas Sensor and IR Flame Sensor

❖ **Raspberry Pi**: Raspberry Pi a series of small-board computer developed in the United Kingdom by the Raspberry Pi foundation in association with Broadcom. Several generations of Raspberry Pi have been released. All models feature a Broadcom system on a chip (SOC) with an integrated ARM-compatible central processing unit (CPU) and on-chip graphics processing unit (GPU).

Raspberry Pi is a small single board computer. By connecting peripherals like Keyboard, mouse, display to the Raspberry Pi, it will act as a mini personal computer. Raspberry Pi is popularly used for real time Image/Video Processing, IoT based applications and Robotics applications. Raspbian OS is official Operating System available for free to use. This OS is efficiently optimized to use with Raspberry Pi. Raspbian have GUI which includes tools for Browsing, Python programming, office, games, etc. We should use SD card (minimum 8 GB recommended) to store the OS (operating System). It is more than computer as it provides access to the on-chip hardware i.e. GPIOs for developing an application. By accessing GPIO, we can connect devices like LED, motors, sensors, etc and can control them too.

- The Raspberry Pi-3 Model B features:
  - Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
  - 1GB RAM
  - BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board
  - 100 Base Ethernet
  - 40-pin extended GPIO
  - 4 USB 2 ports
  - 4 Pole stereo output and composite video port
  - Full size HDMI

- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touch screen display
- Micro SD port for loading your operating system and storing data
- Upgraded switched Micro USB power source up to 2.5A

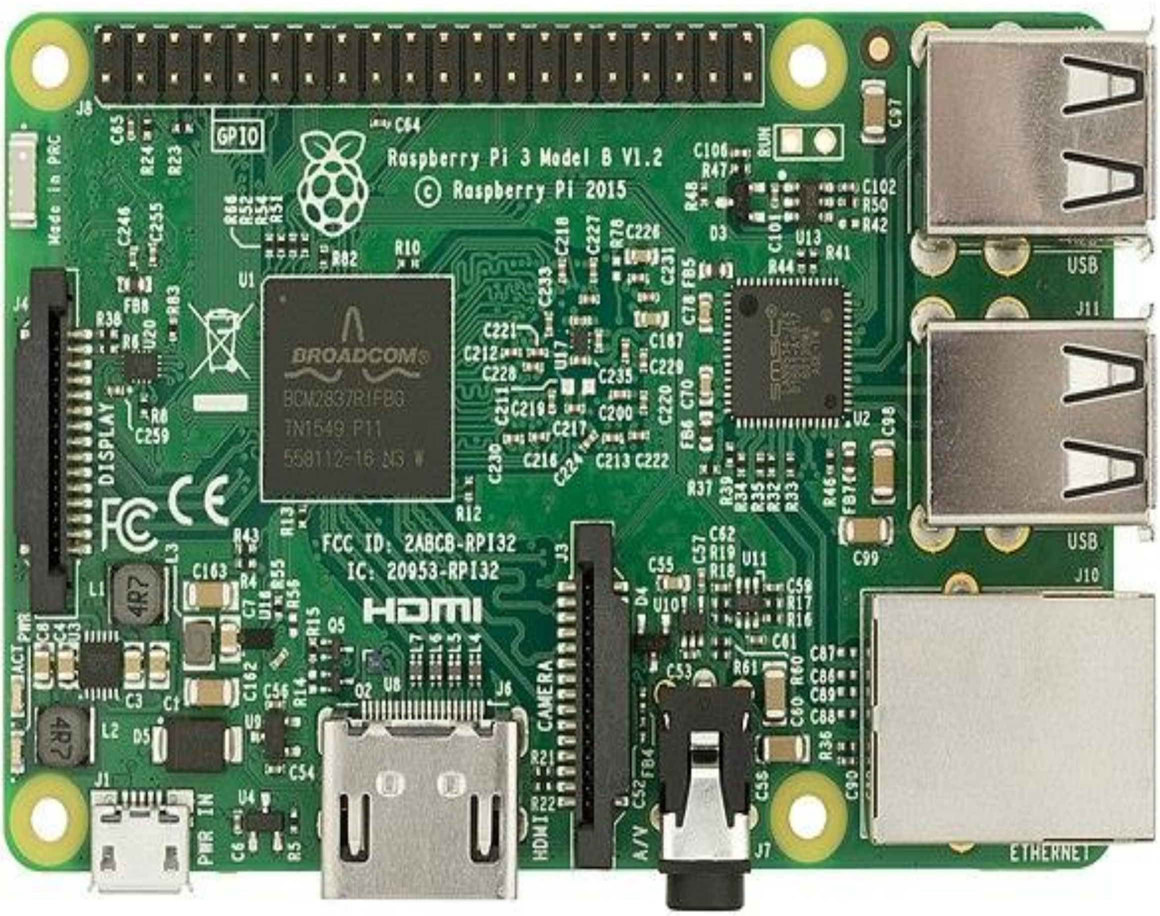

Figure 4.2: Raspberry Pi 3 Model B Board

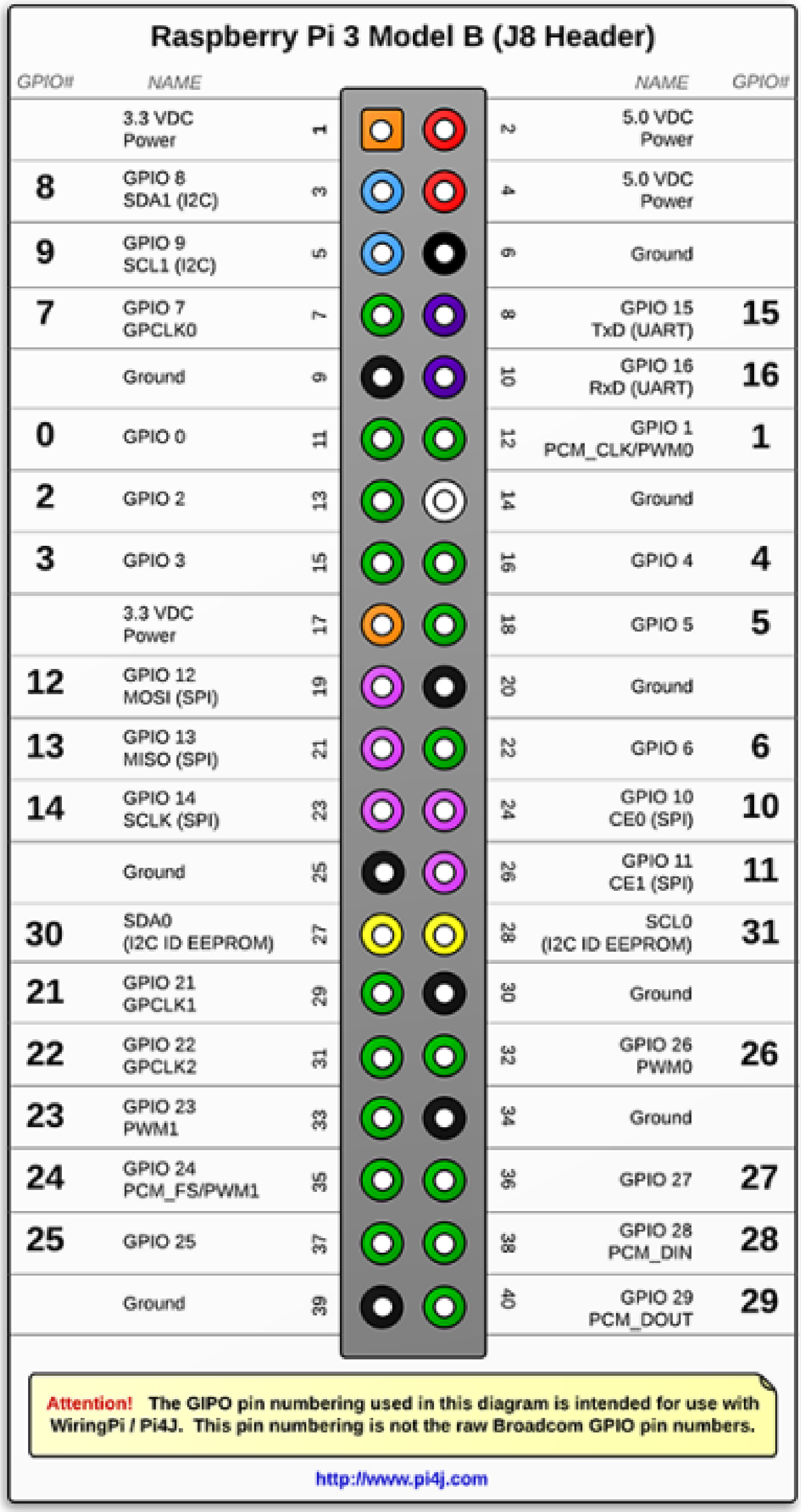

Figure 4.3: GPIO pin

❖ Pi Camera Model

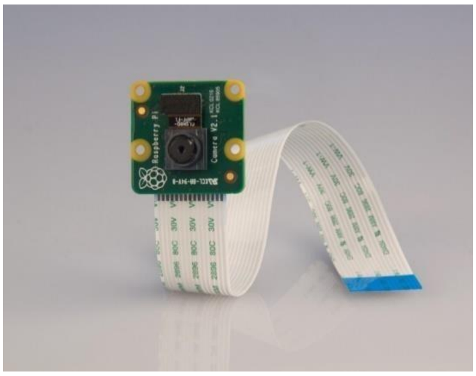

Figure 4.4: Pi Camera

This 5mp camera module is capable of 1080p video and still images and connects directly to your Raspberry Pi. Connect the included ribbon cable to the CSI (Camera Serial Interface) port on your Raspberry Pi, boot up the latest version of Raspbian and you are good to go. The board itself is tiny, at around 25mm x 20mm x 9mm, and weighing in at just over 3g, making it perfect for mobile or other applications where size and weight are important. The sensor has a native resolution of 5 megapixels and has a fixed focus lens onboard. In terms of still images, the camera is capable of 2592 x 1944-pixel static images, and also supports 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 video.

The Raspberry Pi Camera Board Features:

- Fully Compatible with Both the Model A and Model B
- 5MP Omni vision Camera module.
- Still Picture Resolution: 2592 x 1944
- Video: Supports 1080p @ 30fps, 720p @ 60fps and • 640x480p 60/90 Recording 15 pin MIPI Camera Serial Interface - Plugs Directly into the Raspberry Pi Board.

❖ Gas Sensor

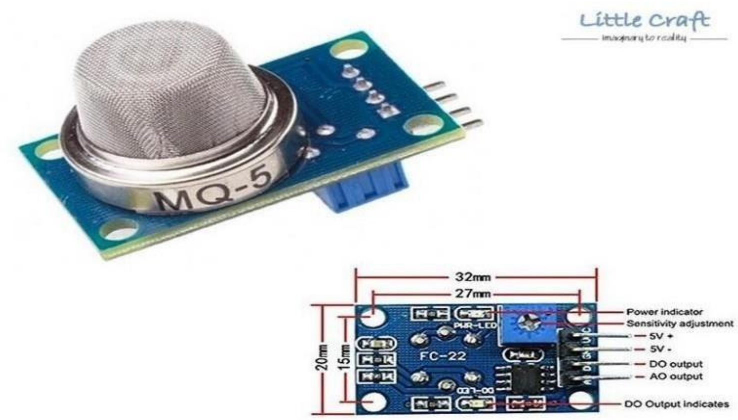

Figure 4.5: Gas Sensor

Gas sensor, as one of the most important devices to detect noxious gases, provides a vital way to monitor the concentration and environmental information of gas in order to guarantee the safety of production. Therefore, researches on high sensitivity, high selectivity, and high stability have become hot issues. Since the discovery of the nanomaterial, it has been increasingly applied to the gas sensor for its distinguishing surface performances. However, 0-D and 1-D nanomaterials, with limited electronic confinement effect and surface effect, cannot reach the requirement for the production of gas sensors. This paper gives an introduction about the current researching progress and development trend of 2-D nanomaterials, analyses the common forms of 2-D nanoscale structure, and summarizes the mechanism of gas sensing. Then, widely concerned factors including morphological properties and crystalline structure of 2-D nanomaterial, impact of doped metal on the sensibility of gas sensors, impact of symmetry, and working temperature on the selectivity of gas sensors have been demonstrated in detail. In all, the detailed analysis above has pointed out a way for the development of new 2-D nanomaterial and enhancing the sensibility of gas sensors.

- High sensitivity to LPG, natural gas, town gas.
- Small sensitivity to alcohol, smoke.
- Stable and long life.
- Simple drive circuit.
- Fast response.

❖ IR Flame Sensor

Figure 4.6: IR Flame Sensor

A Flame Sensor module or Fire Sensor module is a small size electronics device that can detect a fire source or any other bright light sources. This sensor basically detects IR (Infrared) light wavelength between 760 nm – 1100 nm that is emitted from the fire flame or light source. The flame sensor comes with a YG1006 Phototransistor sensor which is a high speed and high sensitivity. Two types of IR Infrared Flame Sensor Module available in the market one having three pins (D0, Gnd, Vcc) and another one having four pins (A0, D0, Gnd, Vcc) both are can be easily used with Arduino and other microcontroller boards.

- 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

❖ Humidity Sensor

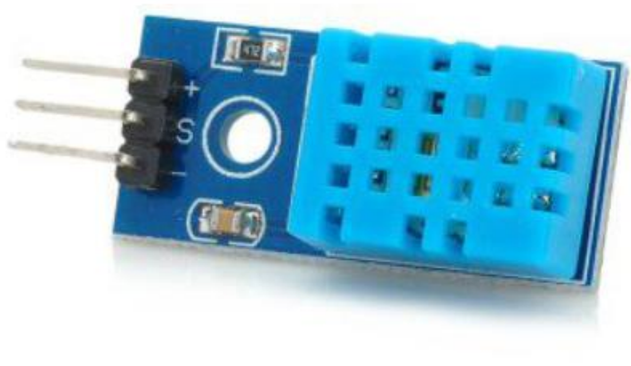

Figure 4.7: Humidity Sensor

The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). It’s fairly simple to use, but requires careful timing to grab data. The only real downside of this sensor is you can only get new data from it once every 2 seconds, so when using our library, sensor readings can be up to 2 seconds old.

- Low cost
- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Good for 20-80% humidity readings with 5% accuracy
- Good for 0-50°C temperature readings ±2°C accuracy
- No more than 1 Hz sampling rate (once every second)
- Body size 15.5mm x 12mm x 5.5mm
- 4 pins with 0.1″ spacing

4.2 Software Configuration

➢ SD Card Preparation

- To download Raspberry Pi and to write code, we need SD card.
- First, we need to download raspberry pi imager, it is a software which is downloaded by its official website https://www.raspberrypi.org
- After the download we need to manually install the operating system for Raspberry Pi imager.
- There is many OS for Raspberry Pi imager
- Raspberry Pi OS with desktop
- Raspberry Pi with desktop and recommended software
- Raspberry Pi OS Lite, etc.
- Here we are using Raspberry Pi OS with desktop.
- Selection of OS depends upon the size of the SD card.
- Before installing any software, first we need to execute these 2 commands
- sudo apt-get update
- sudo apt-get upgrade
- Then we have to write the code into Thony IDE editor.

➢ Camera Enabling

- We have to enable sudo raspi-config to enable camera.
- After enabling the camera, we need to reboot to update for new updates/changes.
- After enabling camera, we need to insert the Raspberry Pi camera on raspberry pi, through CSI (Camera Serial Interfacing) port.

4.3 Installation of Libraries

Open CV

OpenCV (Open-Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then It seez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source Apache 2 License. Starting with 2011, OpenCV features GPU acceleration for real-time operations.

NumPy

NumPy is a library for the Python programming language, adding support for large, multidimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Num array into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy is a Num FOCUS fiscally sponsored project.

4.4 Basic Linux Commands

- cd-change directory.
- ls-list directory content.
- sudo -Super User Do.
- mv-Move/rename a file.
- rm-Delete a file.
- mkdir -Create a directory.
- apt-Install and manage software.
- ping-Check that we are connected.
- hostname-Get the IP address of your Raspberry pi.
- curl-Transfer data over a network.
- history. pwd -print working directory.

4.5 Email Notification

In order to allow for email notifications to send, the OS needs a program that allows for emails to be sent. Simple Mail Transfer Protocol (SMTP) is a program that allows a system to deliver an email from a local computer to a mail host. It does not receive mail but can send out mail. SMTP is ideal for situations where alerts are needed to be sent, therefore it is useful when sending notifications. A python script can be used to achieve this. It may just send a notification without image or can be modified to send an attachment along with the alarm message. For SMTP to support transmission of an attached file, Multipurpose Internet Mail Extension (MIME) is required.

CHAPTER 5

WORKING

Working of this Project is very simple. A PIR sensor is used to detect the presence of any person and a Pi Camera is used to capture the images when the presence it detected. Whenever anyone or intruder comes in range of PIR sensor, PIR Sensor triggers the Pi Camera through Raspberry Pi. Raspberry pi sends commands to Pi camera to click the picture and save it. After it, Raspberry Pi creates a mail and sends it to the defined mail address with recently clicked images. The mail contains a message and picture of intruder as attachment. Here we have used the message “Please find the attachment”, you can change it accordingly in the Code given at the end.

The IoT device built on Raspberry Pi 3 in this project has a simple and straight forward operation. The device detects motion by the PIR sensor and as it detects motion, it starts capturing images. The images are stored on the MicroSD card and sent on the registered email of the user. All of this is managed by a python script running over the Raspbian Operating System. Before running the python script, it is essential to install operating system on the Pi 3 and install the required libraries ie, OpenCV on the operating system. While installing the operating system, installing the libraries and the python script, the Raspberry Pi should be connected to a display monitor using HDMI cable.

CHAPTER 6

APPLICATIONS

- The images are temporarily stored on the Raspberry Pi and sent as an e-mail alert to the owner from Google Cloud.
- The Flammable Gas sensor/MQ2 sensor is used in smoke detection.
- The Fire detection system is used to detect fire in air by IR flame sensor and it is captured by Pi camera in real time monitoring system based on Raspberry Pi.

CHAPTER 7

RESULTS AND DISCUSSION

Executing the Python script in appendix successfully yielded the results indicated. It can be depicted that the range of detection of the IR sensor varies at different sensitivity levels. The higher the sensitivity of the sensor the higher the range of detection Referring again to the results in tables 1-3, it can be safely said that the PIR sensor formed the primary motion detector and hence the first line of defines. All the other modules solely depended on the PIR GPIO pin to go HIGH. In case this sensor fails, the whole security procedure as designed shall fail. The send alerts from this part are provided in appendix section. They show the email send to a mail host through SMTP protocol. The attached file is a 10 second video that shall be analysed in order to explain the concept of image processing in real time video. Running the Python-OpenCV code provided did not yield successful results in the RPI. This was because a large video file or otherwise live video streams require large processor speed. The RPI runs at 700MHz. This script was thus implemented differently on the laptop PC. The results in table 4 were obtained. From those results in table 4, it can be deduced that varying the pixel threshold of the camera achieves the action of detection and tracking. However, detection occurs only within some limits. This can be shown by comparing the captions indicated in Figures 18 - 20. At a pixel threshold = 25, Motion detection and tracking was achieved, while at 100 pixels (Figure 20), no detection nor tracking was possible.

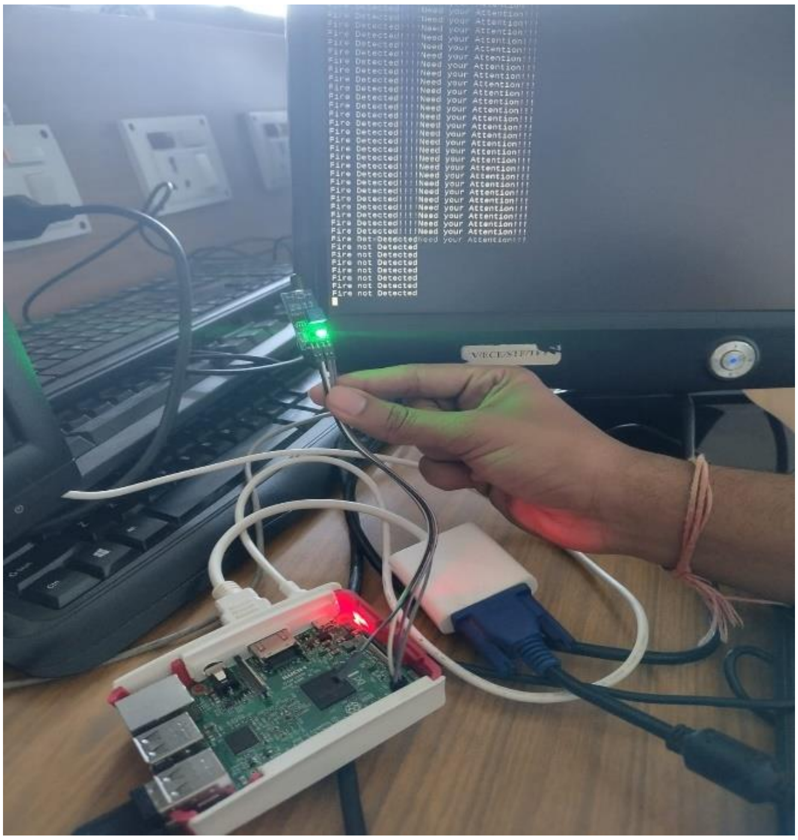

Figure 7.1: No Detection of fire

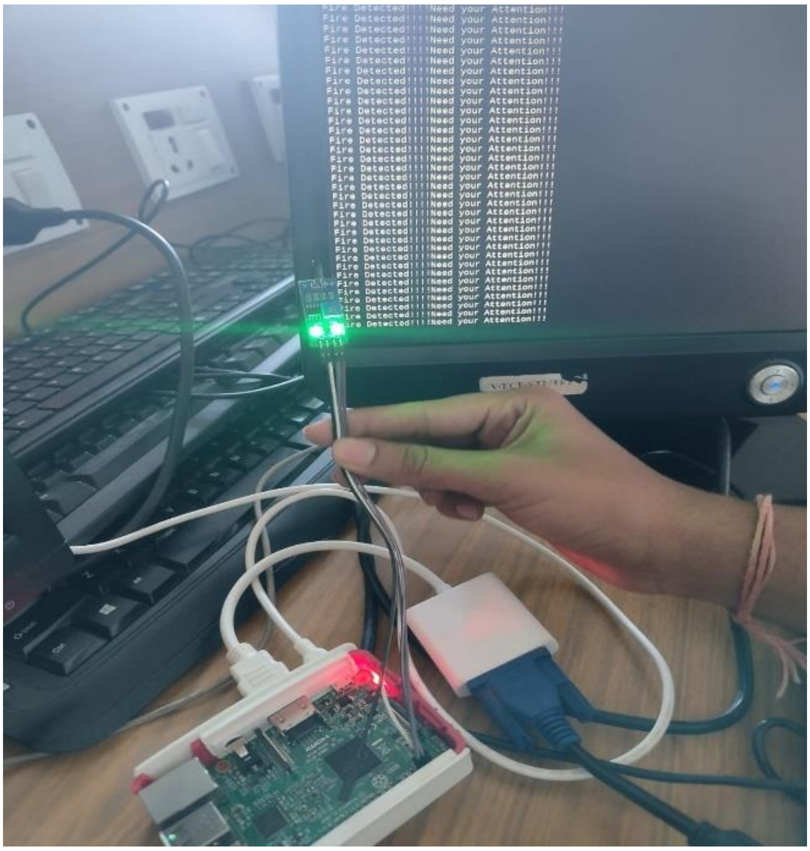

Figure 7.2: Detection of fire

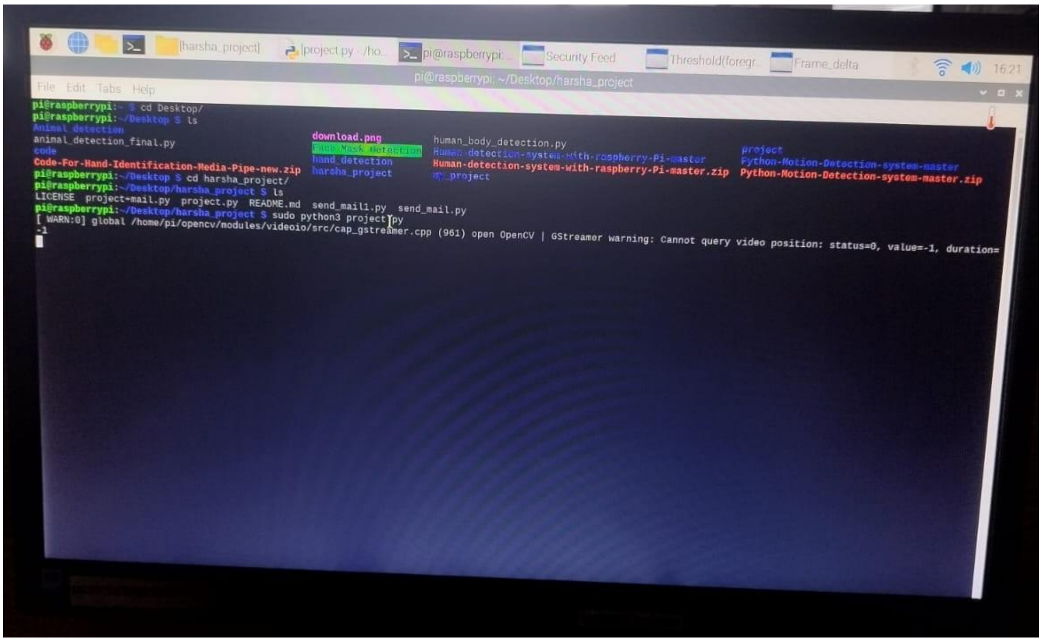

Figure 7.3: Display of Result

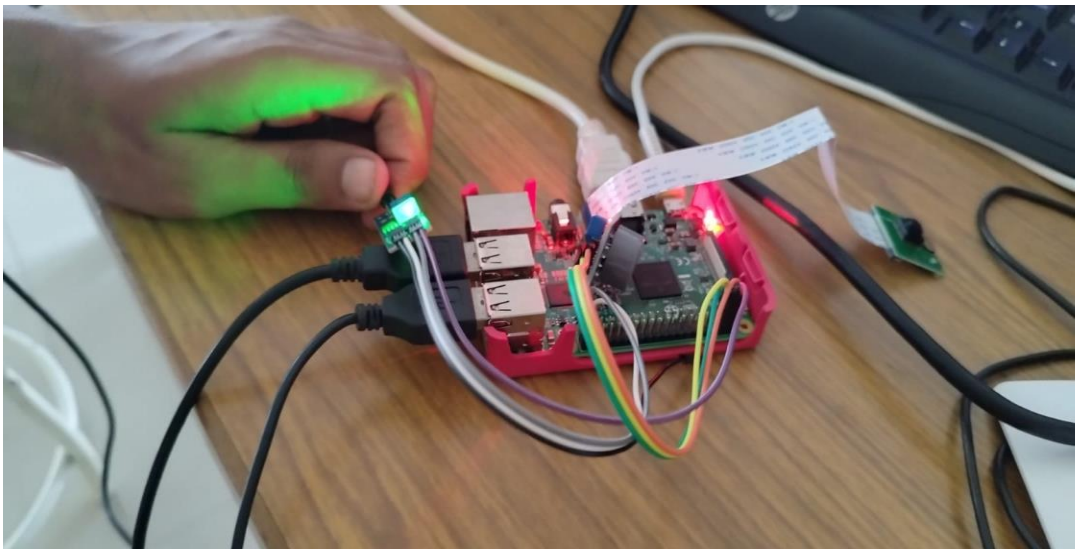

Figure 7.4: Connection of all Sensors

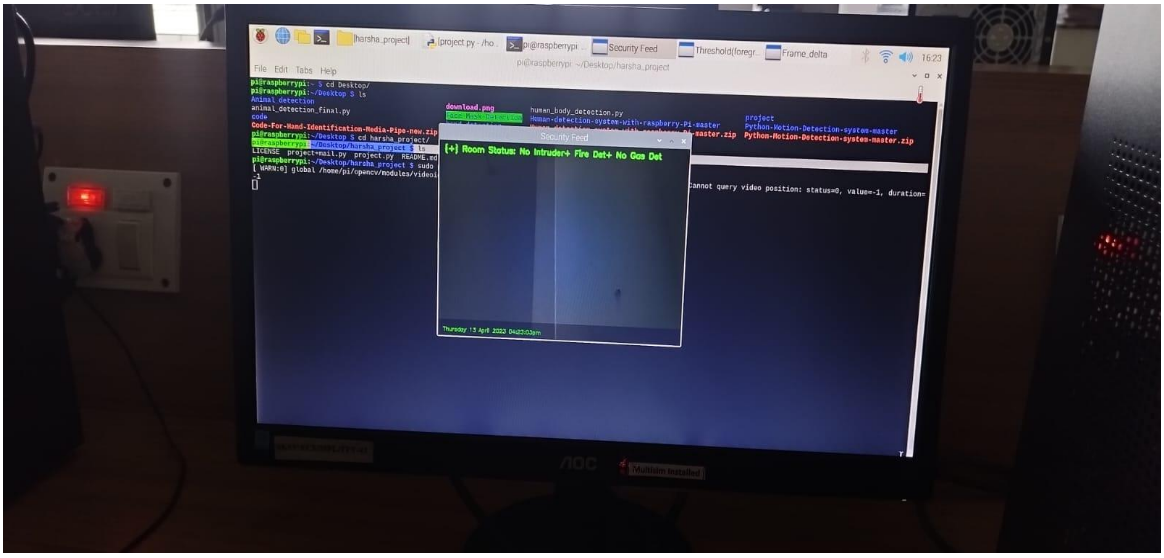

Figure 7.5: No Intruder, No Fire, No Gas

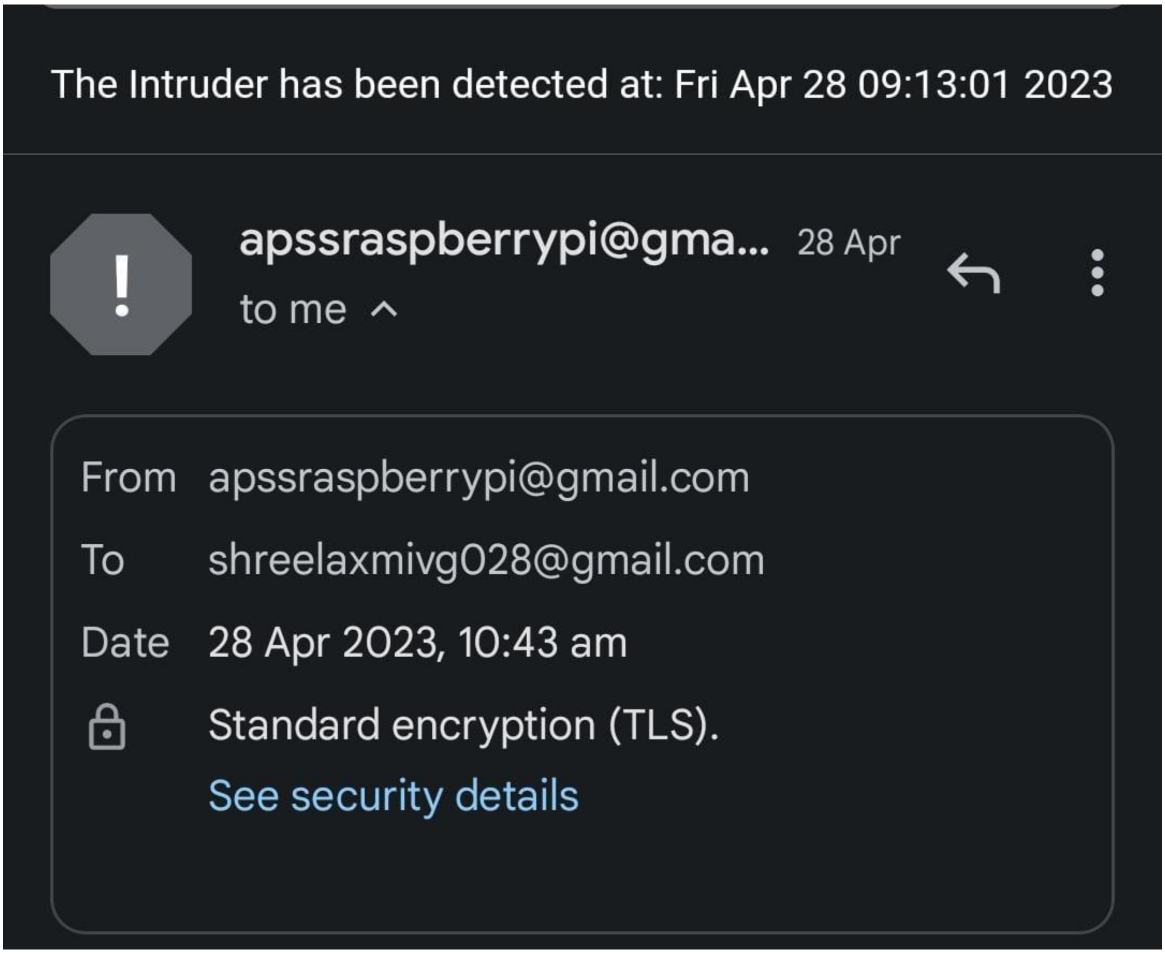

Figure 7.6: Email Alert

CONCLUSION

The traditional home security system is based on the cloud server, all data collected from user's home by sensor would be uploaded to cloud and processed in it. What's more, the users cannot change anything about processing the data. To deal with these problems, in this project, the Raspberry Pi is used as the server to replace the cloud server. In the designed home security system, all the sensors, including PIR sensor, temperature sensor and humidity sensor will be connected to Pi directly and Pi will be worked as server to process all the data collected by sensors. When Pi detects something wrong, it can send a warning email to user directly. Team members finished the prototype of this equipment in Raspberry Pi 3. Then, team members translated the code from Raspberry Pi 3 to Raspberry Pi 0 in order to reduce cost. Then the software and hardware part were integrated. Finally, according to the circuit size team members designed the container. After plugging in and working for one week, we proved that this equipment is successful for using.

REFERENCES

[1] Akash V. Bhatkule, Ulhas B. Shinde, Shrinivas R. Zanwar, ‘Home Based Security System using Raspberry Pi and GSM’, International Journal of Innovative Research in Computer and Communication Engineering, Vol. 4,2016.

[2] S. Snigdha, K. Hari Babu,’IoT based Security System using Raspberry Pi and Mail Server’, International Journal of Innovative Research in Computer and Communication Engineering, Vol.8,2019.

[3] Dhanshri Mali, R.T. Patil, Nagaraj Dharwadkar, Chaitanya R. Devale, Omprakash Tembhurne.’Real-Time Smart Surveillance using Raspberry Pi’, Rajarambapu Institute, Rajaramnagar,24 February 19.

[4] DeepakS.Kumbhar, Shubhangi M. Taur, H.C. Chaudri, Shubhangi S. Bhatambrekar,’IoT Based Home Security System using Raspberry Pi’, IJRAR, Vol. 6, January 2019.

Appendix

Code

```
#// Don't forget to hit SUBSCRIBE, LIKE, COMMENT, and LEARN! its good to learn :)

#imports
import smtplib
import cv2
import time
import datetime
import imutils
import RPi.GPIO as GPIO
from time import sleep
import sys
import Adafruit_DHT

#Email Variables
SMTP_SERVER = 'smtp.gmail.com' #Email Server (don't change!)
SMTP_PORT = 587 #Server Port (don't change!)
GMAIL_USERNAME = 'apssraspberrypi@gmail.com' #change this to match your gmail
account
GMAIL_PASSWORD = 'wdxnewxujlmvusox'

#change tccccccchis to match your gmail app-password
sendTo = 'souravsinghj68@gmail.com'

class Emailer:
    def sendmail(self, recipient, subject, content):

        #Create Headers
        headers = ["From: " + GMAIL_USERNAME, "Subject: " + subject, "To: " +
          recipient,
              "MIME-Version: 1.0", "Content-Type: text/html"]
```

```
        headers = "\r\n".join(headers)

        #Connect to Gmail Server
        session = smtplib.SMTP(SMTP_SERVER, SMTP_PORT)
        session.ehlo()
        session.starttls()
        session.ehlo()

        #Login to Gmail
        session.login(GMAIL_USERNAME, GMAIL_PASSWORD)

        #Send Email & Exit
        session.sendmail(GMAIL_USERNAME, recipient, headers + "\r\n\r\n" + content)
        session.quit

sender = Emailer()

def motion_detection():
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(21,GPIO.IN) #Gas sensor
    GPIO.setup(2,GPIO.IN) #Fire sensor
    gpio = 3 #DHT11 sensor
    GPIO.setup(4,GPIO.IN) #PIR sensor
    video_capture = cv2.VideoCapture(0) # value (0) selects the devices default camera
    time.sleep(2)

    first_frame = None # instinate the first fame

    while True:
        frame = video_capture.read()[1] # gives 2 outputs retval,frame - [1] selects frame
        text = 'No Intruder'
```

```
greyscale_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY) # make each frame greyscale wich is needed for threshold

gaussian_frame = cv2.GaussianBlur(greyscale_frame, (21,21),0)
# uses a kernal of size(21,21) // has to be odd number to to ensure there is a valid integer in the centre

# and we need to specify the standerd devation in x and y direction wich is the (0) if only x(sigma x) is specified

# then y(sigma y) is taken as same as x. sigma = standerd deveation(mathmetics term)

blur_frame = cv2.blur(gaussian_frame, (5,5)) # uses a kernal of size(5,5)(width,height) wich goes over 5x5 pixel area left to right

    # does a calculation and the pixel located in the centre of the kernal will become
# a new value(the sum of the kernal after the calculations) and then it moves to the right one and has a new centre pixel
# and does it all over again..untill the image is done, obv this can cause the edges to not be changed, but is very minute

greyscale_image = blur_frame
# greyscale image with blur etc wich is the final image ready to be used for threshold and motion detecion

if first_frame is None:
    first_frame = greyscale_image
    # first frame is set for background subtraction(BS) using absdiff and then using threshold to get the foreground mask
    # foreground mask (black background anything that wasnt in image in first frame but is in newframe over the threshold will
    # be a white pixel(white) foreground image is black with new object being white...there is your motion detection
```

else:

pass

frame = imutils.resize(frame, width=500)

frame_delta = cv2.absdiff(first_frame, greyscale_image)

calculates the absolute diffrence between each element/pixel between the two images, first_frame - greyscale (on each element)

edit the ** thresh ** depending on the light/dark in room, change the 100(anything pixel value over 100 will become 255(white))

thresh = cv2.threshold(frame_delta, 100, 255, cv2.THRESH_BINARY)[1]

threshold gives two outputs retval,threshold image. using [1] on the end i am selecting the threshold image that is produced

dilate_image = cv2.dilate(thresh, None, iterations=2)

dilate = dilate,grow,expand - the effect on a binary image(black background and white foregorund) is to enlarge/expand the white

pixels in the foreground wich are white(255), element=Mat() = default 3x3 kernal matrix and iterartions=2 means it

will do it twice

cnt = cv2.findContours(dilate_image.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)[0]

contours gives 3 diffrent ouputs image, contours and hierarchy, so using [1] on end means contours = [1](cnt)

cv2.CHAIN_APPROX_SIMPLE saves memory by removing all redundent points and comppressing the contour, if you have a rectangle

with 4 straight lines you dont need to plot each point along the line, you only need to plot the corners of the rectangle

and then join the lines, eg instead of having say 750 points, you have 4 points.... look at the memory you save!

for c in cnt:

if cv2.contourArea(c) > 800: # if contour area is less then 800 non-zero(not-black) pixels(white) (x, y, w, h) = cv2.boundingRect(c) # x,y are the top left of the contour and w,h are the width and hieght

cv2.rectangle(frame, (x,y), (x+w, y+h), (0, 255, 0), 2) # (0, 255, 0) = color R,G,B = lime / 2 = thickness(i think?)(YES IM RITE!)

image used for rectangle is frame so that its on the secruity feed image and not the binary/threshold/foreground image

as its already used the threshold/(binary image) to find the contours this image/frame is what image it will be drawed on

```
        text = 'Intruder'
        emailSubject = "Intruder Detected!"
        emailContent = "The Intruder has been detected at: " + time.ctime()
        sender.sendmail(sendTo, emailSubject, emailContent)
        print("Email Sent")
        break
        sleep(2)
        # text that appears when there is motion in video feed
      else:
        pass

    ''' now draw text and timestamp on security feed '''
    font = cv2.FONT_HERSHEY_SIMPLEX
    flag=0
    if (GPIO.input(21)):
      text += ' No Fire Det'
    else:
       text += ' Fire Det'

    if (GPIO.input(2)):
       text += ' No Gas Det'
    else:
       text += ' Gas Det'
    sensor = Adafruit_DHT.DHT11
```

```
humidity, temperature = Adafruit_DHT.read_retry(sensor,gpio)
        if humidity is not None and temperature is not None:
            print('Temp={0:0.1f}*C  Humidity={1:0.1f}%'.format(temperature, humidity))
        else:
            print('Failed to get reading. Try again!')



        cv2.putText(frame, '{+} Room Status: %s' % (text),
            (10,20), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)
        # frame is the image on wich the text will go. 0.5 is size of font, (0,0,255) is R, G, B color of font, 2 on end is LINE THICKNESS! OK :)

        cv2.putText(frame, datetime.datetime.now().strftime('%A %d %B %Y %I:%M:%S%p'),
            (10, frame.shape[0] - 10), cv2.FONT_HERSHEY_SIMPLEX , 0.35, (0, 255, 0),1)
# frame.shape[0] = hieght, frame.shape[1] = width,
        # using datetime to get date/time stamp, for font positions using frame.shape() wich returns a tuple of (rows,columns,channels)
        # going 10 accross in rows/width so need columns with frame.shape()[0] we are selecting columns so how ever many pixel height
        # the image is - 10 so oppisite end at bottom instead of being at the top like the other text

        cv2.imshow('Security Feed', frame)
        cv2.imshow('Threshold (foreground mask)', dilate_image)
        cv2.imshow('Frame_delta', frame_delta)

        key = cv2.waitKey(1) & 0xFF # (1) = time delay in seconds before execution, and 0xFF takes the last 8 bit to check value or sumin
        if key == ord('q'):
            cv2.destroyAllWindows()
            break
if __name__=='__main__':
    motion_detection ()
```