A

PROJECT REPORT

ON

THE PROJECT ENTITLED

Security System Based
On Artificial Intelligence

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND TELE-COMMUNICATION ENGINEERING

BY

Akash R. Biradar	**Exam No: B150433004**
Vishwajeet D. Jagtap	**Exam No: B150433011**
Soham M. Wadhonkar	**Exam No: B150433045**

Under the Guidance of

Prof. Y. R. Bachkar

Sinhgad Institutes

Submitted to

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING

STES'S SINHGAD ACADEMY OF ENGINEERING, PUNE-411048

2021-2022

CERTIFICATE

This is to certify that the project report entitled

SECURITY SYSTEM BASED ON ARTIFICIAL INTELLIGENCE

Submitted By

Akash R. Biradar	Exam No: B150433004
Vishwajeet D. Jagtap	Exam No: B150433011
Soham M. Wadhonkar	Exam No: B150433045

Is a bonafide work carried out by them under the supervision by Prof. Y. R. Bachkar and it is approved for the partial fulfilment of the requirement of **Savitribai Phule Pune University** for the Project in the Final Year of Electronics and Telecommunication Engineering.

This project report has not been earlier submitted to any other institute or University for the award of any degree or diploma.

Prof. Y. R. Bachkar	**Dr. K.M. Gaikwad**	**Dr. Kishor P. Patil**
Project Guide	**H.O.D and Vice Principal**	**Principal, SAOE, Pune**

Place: Pune

Date: **Examiner**

ACKNOWLEDGEMENT

We would like to thank our Project Guide Mr. Y. R. Bachkar sir for their valuable guidance and support towards completion of project. We would also like to extend gratitude to the Head Of Department Prof. K.M. Gaikwad sir and Principle Dr. K. P. Patil sir for providing us with all facility that was required.

Project Group Members:

Akash R. Biradar **Exam No: B150433004**

Vishwajeet D. Jagtap **Exam No: B150433011**

Soham M. Wadhonkar **Exam No: B150433045**

Abstract

Security system has established its importance and benefits numerous times by providing immediate monitoring of the house. This is because of the increasing home theft and burglary incidents that create an awareness among most of the house owners. CCTV-based security systems are not real-time because the alert comes to the owner after the incident occurred unless they are at home during the incident. To overcome this problem, many researchers are developing cost-effective custom based security systems, which are affordable for everyone. Most of these systems use a Passive Infrared (PIR) motion sensor for motion detection. Although affordable, such a system still has many limitations.

For example, false alarms triggered due to an abnormal condition such as rapid heating from sunlight exposure. In this work, the objective is to design a Security system based on artificial intelligence which is a machine learning based security platform that automatically monitors and detects people in a scene, and then alerts the user in real time by sending an image and video of victim through their email or app.

Table of Contents

Acknowledgement 3
Abstract 4
Table of Contents 5
List of Figures 7
List of Tables 8

Chapter 1 Introduction 9

1.1 Motivation 10
1.2 Problem Statement 10
1.3 Objectives of the proposed work 11

Chapter 2 Literature Review 12
2.1 Introduction 13
2.2 Literature Survey 14
2.3 Reference Papers 15

Chapter 3 System overview 16
(Design, Theory, Flowchart and Application)
3.1 Introduction 17
3.2 Block Diagram 17
3.3 Explanation of Block Diagram 18
3.4 Software Requirements 19
3.4.1 Machine Learning Frameworks 19
3.4.2 Required Python Libraries 23
3.5 Hardware Requirements 25
3.5.1 Raspberry Pi 4 model b 25
3.5.2 Raspberry Pi Camera model 26
3.6 Flowchart 28
3.7 Code 31
3.7.1 Sending E-mail Notification 31
3.7.2 Sending Telegram Notification 33
3.8 Applications 35

3.9 Features 36

Chapter 4 Results 37

4.1 Connections 38

4.2 Results 39

4.2.1 Live Streaming 39

4.2.2 E-Mail Notification 40

4.2.3 Call Notification 41

4.2.4 Telegram Notification 42

Chapter 5 Conclusion 43

5.1 Conclusion 44

5.2 Future Scope 44

5.3 References 45

5.4 Total Expenditure 46

List of Figures

Figure		Page no.
	Chapter 3	
3.2.1	Block Diagram	17
3.4	SSD architecture 3.4(a).	21
	Depth-wise separable convolution 3.4(b).	22
3.6	Flowchart 3.6(a)	28
	Flowchart 3.6(b)	30
	Chapter 4	
4.1	Final setup	38
4.2	live-stream 4.2.1(a).	39
	Object detection 4.2.1(b)	39
	E- mail Notification 4.2.2	40
	Call Notification 4.2.3	41
	Telegram approach 4.2.4	42

List of Tables

Table		Page
2.2	Literature Survey	14
3.4.1	Available models in the Tensorflow-api	20
5.4.	Total Expenditure	46

CHAPTER – 1

INTRODUCTION

1.1 Motivation:

Home break-ins, porch pirates stealing deliveries, trespassers, break-ins of shops such as jwellery shops. Installation of surveillance cameras equipped with motion sensors are a common solution to this problem. However, surveillance cameras only detect motion and collect data - meaning that the user must go through frames in the video footage to discern any meaningful information. As a result, this visual information is typically inspected after an unfortunate incident, such as a package being stolen or robbery.

To improve this situation, we propose AI a machine vision security system to automatically collect and analyse relevant images to identify critical and actionable information and send it to the user, in real-time.

1.2 Problem Statement:

There are roughly 2.5 million burglary a year, 66% of those being home break-ins, commercial shops, government property. Police solve only 13% due to lack of evidence. CCTV camera footages are usually analysed hours after event has happened, this delays investigation and further damage. Studies shows, a burglar usually checks on your house few times before robbing to ensure a perfect time to rob. A CCTV camera has no means to alert you when a stranger shows up at your doorstep.

However, there are some security systems available in which either users can manually monitor there property manually or the most common and traditional features of the home security system are motion detection, live monitoring, and alert notification. Systems relying only on a Passive Infrared (PIR) sensor to accommodate for motion detection have unreliable detection rate because it could trigger a false alarm due to abnormal conditions such as pet intrusion or rapid heating [2] e.g. from sunlight exposure. False alarms can have significant impacts such as in security systems that trigger calls to the police [3] or other emergency agencies.

In order to overcome these limitations. this Report presents an implementation of AI-based intruder detection system prototyped on a Raspberry pi 4B+.

1.3 **Objective of the proposed of work:**

The objective is to design a Security System Based On Artificial Intelligence which is a machine learning based security platform that automatically monitors and detects people in a scene, and then alerts the user in real time by sending an image and video of victim through their email or app and alert them by making call to the user number.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction:

Security has always been a major issue everywhere around the globe and the importance of security cannot be denied in today's society because of the increasing crime rate.AI security system is an intelligent vision-based security service designed to automatically detect humans in a visual scene and notify the property owner in real-time. This is to ensure any and all unauthorized personnel on the private property are identified as soon as possible to provide a record of intrusions. The functionality is provided using a combination of deep learning and edge computing where the camera on-board the edge device continuously monitors an area. When the real-time analytics on the edge device detects one or more people in the scene, an image is immediately captured and the user is notified via email. Soon afterwards, a video clip of the scene with the person(s) is recorded and also sent to the user via email. The image and video are stored in the internal memory of device for on-demand user access.

2.2 Literature Survey:

Sr no.	Paper Name	Author	Description
1	Automated security system with surveillance	P. Vigneswari, V. Indhu, R. Narmatha, A Sathinisha	A PIR sensor and a camera were installed respectively to detect the presence of an intruder and capture his/her picture. The owner will be alerted through Short Message Service (SMS) using the GSM technology. At the heart of the system was an Atmega644p microcontroller, which receives and processes signals from the PIR sensor and decides whether it is necessary to send a notification message with the captured image over SMS.
2	Home monitoring and security system	S.Suresh, J.Bhavya, S.Sakshi, K. Varun, and G. Debarshi	PIR sensor and a temperature and humidity sensor are connected to an Arduino Uno microcontroller. The system intends to apply changes in both motion and temperature in a monitored room to improve the accuracy of the intrusion detection by reducing false detections based on line of sight that can be cut by any entity and not necessarily an intruder. If the temperature is above a set threshold and a change in motion is detected, an SMS message will then be sent to the owner's mobile phone via GSM.
3	Low cost multi-level home security system for developing countries.	H. U. Zaman, T.E.Tabassum, T. Islam, and N. Mohammad	An Arduino-based, low-cost, and multi-level home security system was proposed by Zaman for developing countries. Their system consists of two parts, namely internal and external parts to make the system more effective. The internal part is controlled by an Arduino Mega microcontroller with GSM shield to detect intruder(s) and notify the owner through SMS or phone call. Each room was installed with PIR sensors and LED lights attached to the ceiling. When a PIR sensor is triggered, a signal is sent to the microcontroller, LED lights blink, respective lights of the affected room get turned on, an installed alarm starts to buzz for three seconds with intervals to alert the surrounding, and the GSM module sends an SMS notification to the owner.

2.3 Reference Papers:

1) P. Vigneswari, V. Indhu, R. Narmatha, A. Sathinisha, and J. Subashini, "Automated security system usingsurveillance**," International journal of current engineering and technology, vol. 5, no. 2, 882-884, (2015).**

2) S. Suresh, J. Bhavya, S. Sakshi, K. Varun, and G. Debarshi, "Home monitoring and security system," in **2016 International Conference on ICT in Business Industry & Government (ICTBIG, 2016), 1-5.**

3) H. U. Zaman, T. E. Tabassum, T. Islam, and N. Mohammad, "Low cost multi-level home security systemfor developing countries," in *2017* **International Conference on Intelligent Computing and Control Systems (ICICCS, 2017), 549-554.**

4) S. Sruthy and S. N. George, "WiFi enabled home security surveillance system using Raspberry Pi and IoTmodule," in 2017 **IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES, 2017), 1-6.**

5) R. R. Ragade, "Embedded home surveillance system with pyroelectric infrared sensor using GSM," in 2017 **1st International Conference on Intelligent Systems and Information Management (ICISIM, 2017), 321-** 324.

6) S. Prasad, P. Mahalakshmi, A. J. C. Sunder, and R. Swathi, "Smart surveillance monitoring system usingRaspberry Pi and PIR sensor," in **International Journal of Computer Science and Information Technologies, vol. 5, no. 6, 7107-7109, (2014).**

7) B. Kaur, P. K. Pateriya, and M. K. Rai, "An illustration of making a home automation system usingraspberry Pi and PIR sensor," in **2018 International Conference on Intelligent Circuits and Systems (ICICS), 2018, pp. 439-444.**

CHAPTER 3

SYSTEM OVERVIEW

3.1 Introduction:

The **Security System Based On Artificial Intelligence is a machine learning based security** platform that automatically monitors and detects people in a scene, and then alerts the user in real time by sending an image and video to their email or telegram account. The system is enabled through a combination of edge computing and deep learning infrastructure. The edge platform used here is the Raspberry pi 4b+.

3.2 Block Diagram:

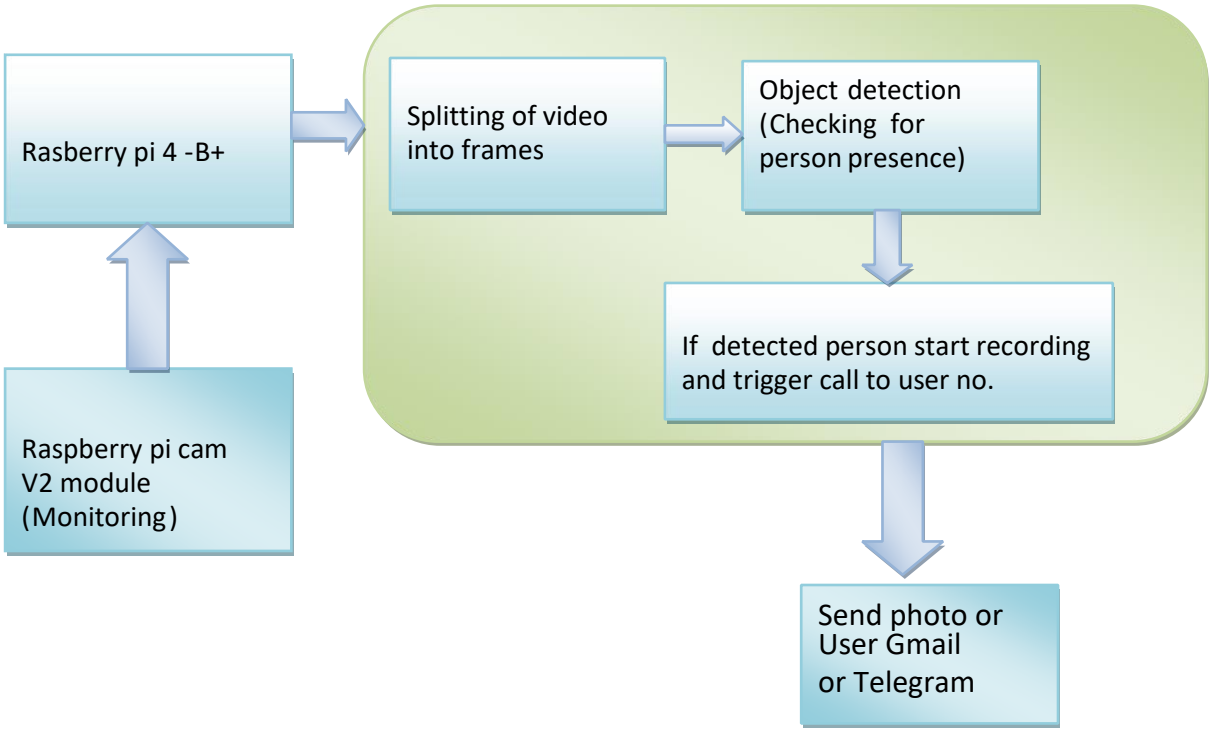


Fig 3.2(a) BLOCK DIAGRAM

3.3 Explanation of Block Diagram:

1) Monitoring:

First step is capturing the streaming data this could be coming from an rtsp stream from a file or from a usb or csi camera. In our case input video is coming from Raspberry pi camera v2 module. The Pi camera v2 video module connects directly to the Camera Serial Interface (CSI) port in the Raspberry pi 4b+ using a 15-pin ribbon cable. The pattern noise and smearing from an earlier version have been fixed, which reduces faulty image. There are automatic control functions such as luminance detection, white balance and exposure control.

2) Analysis:

Given the goal of detecting a human in a scene, the initial plan was to use a hardware sensor on a Raspberry Pi 4 Model and use a hardware sensor to detect motion in that scene. Which would enable real time processing, a significant advantage for security. But after realizing limitation of sensor we decided to use object detection model. With the help of machine learning model it can detect or classify between objects ,animals, person present in its feed.

The input video stream is hosted on local server using flask for live-stream purpose. Locally connected device can access the stream. The input video is classified using Tensor-flow Object Detection api. It is configured to detect only human neglecting all other classes. Once it detect person class in video feed. Start writing frame and create video of span of 2 or 3 minutes, start capturing photos and keep process repeating until intruder present in the scene. Also trigger a call from device to user phone number with message saying intruder detected check ur device.

3) Execution:

Once the recording is done. The file manager program checks for new generated file and executes mail function or telegram api depend on user choice. Once the mail function is called the video or image is attached to mail and sent to the user with date and timestamp using SMTP library or if user chooses telegram option then telegram api is called and telegram automated bot sends the video or image to the user telegram account. User can start or stop advance security feature by sending command to the bot.

4) Live-stream:

Flask was used as a server to develop the platform for video streaming. The streaming of the video was performed by encoding the frames to a JPEG format and uploading continuously to the server. User connected to local network can access the stream.

3.4 Software Requirements:

a) OpenCV:

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now.

3.4.1 Machine Learning Frameworks:

a) Tensor-Flow:

Tensor-Flow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. It is pre-requisite for working of object detection api.

b) **Tensor-flow Object Detection api:**

The Tensor-Flow object detection API is the framework for creating a deep learning network that solves object detection problems. There are already pre-trained models in their framework which they refer to as Model Zoo. This includes a collection of pre-trained models trained on the COCO dataset, the KITTI dataset, and the Open Images Dataset. These models can be used for inference if we are interested in categories only in this dataset

Available models in api:

Model Name	**Speed**	**COCO mAP**	**Outputs**
ssd_mobilenet_v1_c ɔco	**fast**	**21**	**Boxes**
ssd_inception_v2_C)CO	fast	24	Boxes
rfcn_resnet101_coco	medium	30	Boxes
faster_rcnn_resnet1 1_coco	medium	32	Boxes
faster_rcnn_inception_resnet_v2_atrous_coco	Slow	37	Boxes

c) Ssd_mobilenet_v1_coco:

SSD architecture:

The SSD architecture is a single convolution network that learns to predict bounding box locations and classify these locations in one pass. Hence, SSD can be trained end-to-end. The SSD network consists of base architecture (Mobile-Net in this case) followed by several convolution layers as shown in fig 3.4(a):

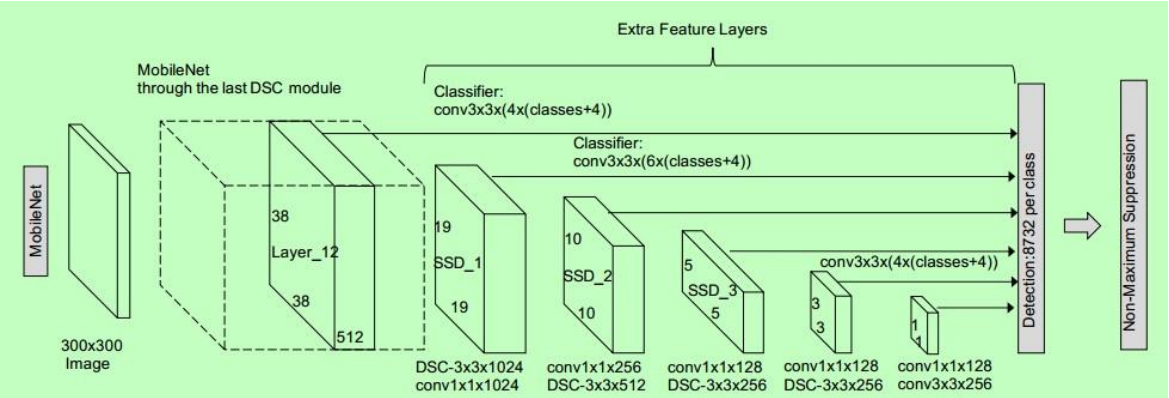


Figure 6. Mobile-Det: SSD-based detection with MobileNet as backbone, modified based on [14]

Fig 3.4(a)

SSD operates on feature maps to detect the location of bounding boxes. Remember – a feature map is of the size Df * Df * M. For each feature map location, k bounding boxes are predicted. Each bounding box carries with it the following information:

i) 4 corner bounding box **offset** locations (cx, cy, w,h)

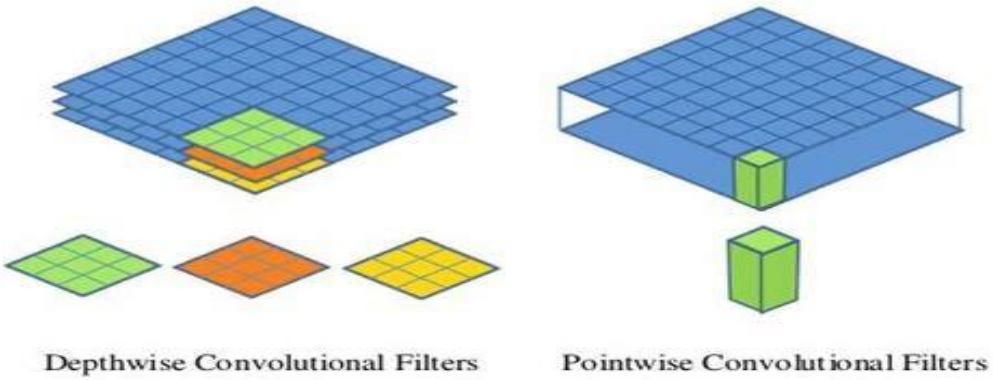
ii) C class probabilities (c1, c2, …cp)

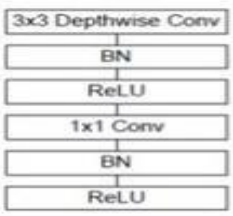
SSD **does not** predict the shape of the box, rather just where the box is. The k bounding boxes each have a predetermined shape. The shapes are set prior to actual training. For example, in the figure above, there are 4 boxes, meaning k=4.

Mobile-Net:

The Mobile-Net model is based on depth-wise separable convolutions which are a form of factorized convolutions. These factorize a standard convolution into a depth-wise convolution and a 1 × 1 convolution called a point-wise convolution .For Mobile Nets, the depth-wise convolution applies a single filter to each input channel. The point-wise convolution then applies a 1 × 1 convolution to combine the outputs of the depth-wise convolution. A standard convolution both filters and combines inputs into a new set of outputs in one step. The depth wise separable convolution splits this into two layers – a separate layer for filtering and a separate layer for combining. This factorization has the effect of drastically reducing computation and model size.

Fig 3.5(b) shows The Mobile-Net model based on depth-wise separable convolutions.





Depthwise Separable Convolution

Fig,3.4b)

Output Result:

Below is the example image tested on ssd_mobilenet_v1_coco (Mobile Net-SSD trained on the COCO dataset):

3.4.2 Required python libraries:

I. Flask:

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. Flask provides native support for streaming responses through the use of generator functions. A generator is a special function that can be interrupted and resumed. The method that works well with the streaming feature of Flask is to stream a sequence of independent JPEG pictures. This is called Motion JPEG, and is used by many IP security cameras.

II. **Twilio:**

Twilio is a web application programming interface (API) that software developers can use to add communications such as phone calling, messaging, video and two-factor authentication into their Python applications. Twilio's API abstracts the telecommunications pieces so as a developer you can simply use your favorite programming languages and frameworks in your application. A voice API is a tool for software developers to make and receive phone calls with a simple, easy to understand API. Behind the scenes, a voice API bridges the Public Switched Telephone Network (PSTN) and applications connected to the internet. By using a voice API, software developers can program voice calling into their applications without specialized telecommunications knowledge and hardware. Using a voice API enables developers to build call logic that extends to users on any device, over any network, anywhere in the world. It also allows developers to add full VoIP functionality to apps to make and receive calls over the internet alone, without interfacing with the PSTN.

III. **SMTP:**

Simple Mail Transfer Protocol (SMTP) is used as a protocol to handle the email transfer using Python. It is used to route emails between email servers. It is an application layer protocol which allows to users to send mail to another. The receiver retrieves email using the protocols POP(Post Office Protocol) and IMAP(Internet Message Access Protocol)**.** When the server listens for the TCP connection from a client, it initiates a connection on port 587.Python provides a **smtplib** module, which defines an the SMTP client session object used to send emails to an internet machine.

3.5 Hardware Requirements:

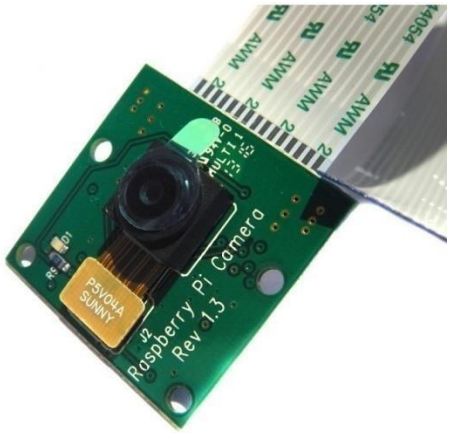
3.5.1 Raspberry pi 4 model b:

- Raspberry Pi 4 Model B is the latest product in the popular Raspberry Pi range of computers. It offers ground-breaking increases in processor speed, multimedia performance, memory, and connectivity compared to the prior generation Raspberry Pi 3 Model B+, while retaining backwards compatibility and similar power consumption. For the end user, Raspberry Pi 4 Model B provides desktop performance comparable to entry-level x86 PC systems.
- This product's key features include a high-performance 64-bit quad-core processor, dual-display support at resolutions up to 4K via a pair of micro HDMI ports, hardware video decode at up to 4Kp60, up to 4GB of RAM, dual-band 2.4/5.0 GHz wireless LAN, Bluetooth 5.0, Gigabit Ethernet, USB 3.0, and PoE capability (via a separate PoE HAT add-on).
- The dual-band wireless LAN and Bluetooth have modular compliance certification, allowing the board to be designed into end products with significantly reduced compliance testing, improving both cost and time to market.

Specifications:

- Processor: Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- Memory: 1GB, 2GB or 4GB LPDDR4 (depending on model)
- Connectivity: 2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth 5.0, BLE Gigabit Ethernet 2 × USB 3.0 ports 2 × USB 2.0 ports.
- GPIO: Standard 40-pin GPIO header (fully backwards-compatible with previous boards)
- Video & sound: 2 × micro HDMI ports (up to 4Kp60 supported) 2-lane MIPI DSI display port 2-lane MIPI CSI camera port 4-pole stereo audio and composite video port Multimedia: H.265 (4Kp60 decode); H.264 (1080p60 decode, 1080p30 encode); OpenGL ES, 3.0 graphics SD card support: Micro SD card slot for loading operating system and data storage Input power: 5V DC via USB-C connector (minimum 3A1) 5V DC via GPIO header (minimum 3A1) Power over Ethernet (PoE)–enabled (requires separate PoE HAT) Environment: Operating temperature 0–50ºC

3.5.2 Raspberry Pi Camera Module V2:



- The 5MP Raspberry Pi Camera Module with Cable v1.3 equips flexible cable for attaching with Raspberry Pi 4 Model B. The 5MP camera module is perfect for small Raspberry Pi projects
- The high-definition 5MP camera delivers outstanding photos but can also shoot video, ideal for drones or a CCTV project. The lightweight camera module allows for it to be used in more practical roles.

- This Raspberry Pi Camera Module is a custom designed add-on for Raspberry Pi. It attaches to Raspberry Pi by way of one of the two small sockets on the board upper surface. This interface uses the dedicated CSI interface, which was designed especially for interfacing to cameras. The CSI bus is capable of extremely high data rates, and it exclusively carries pixel data.

- The board itself is tiny, at around 25mm x 23mm x 8mm. It also weighs just over 3g, making it perfect for mobile or other applications where size and weight are important. It connects to Raspberry Pi by way of a short flexible ribbon cable. The camera connects to the BCM2835 processor on the Pi via the CSI bus, a higher bandwidth link which carries pixel data from the camera back to the processor. This bus travels along the ribbon cable that attaches the camera board to the Pi.
- The sensor itself 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 1080p30, 720p60 and 640x480p60/90 video.

Specifications:

- Supported Video Formats: 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 video.
- Fully Compatible with Raspberry Pi 4 Model B.
- Small and lightweight camera module.
- Plug-n-Play camera for Raspberry Pi 3 Model B.
- Very light, ~ 3g
- Wider image, capable of 2592x1944 stills, 1080p30 video and so forth.
- CSI(Camera Serial Interface)

3.6 Flowchart:

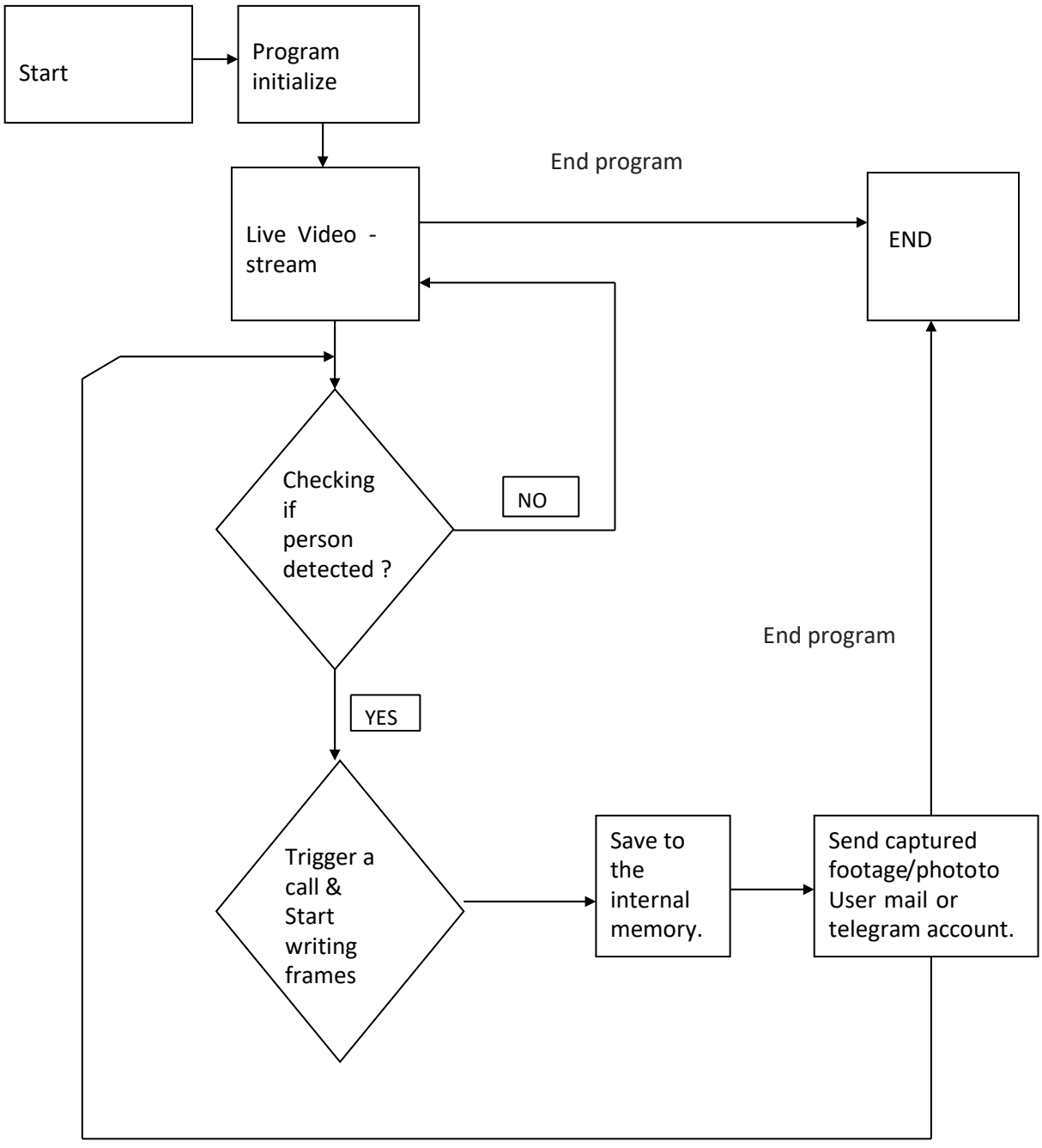


Fig 3.6(a)

i) Figure 3.6(a) shows the design flow of the developed prototype for the security system. A few major components in this system include Object model with Open-CV implementation, live video streaming, and email/telegram notification, call alert. Flask was used as a server to develop the platform for video streaming. The streaming of the video was performed by encoding the frames to a JPEG format and uploading continuously to the server. The stream is available for local device.

ii)Next, Open-CV was implemented on the camera to process the image before streaming. The tensor-flow model descriptor approach is more suitable for a pedestrian based system, which has larger frame views. Now check for presence of human.

iii)Once a person is detected then the system will call twilio api function inside program. Twilio is cloud based calling service. Where user can purchase a virtual no. Call will be generated with message programmable voice saying "intruder detected check u r device". Call generated by system when it detects person through Twilio api it is achieved. Whenever it detects presence of human it will trigger a call to user device to alert the user. Using the Twilio REST API, we can make outgoing calls to phones, SIP-enabled endpoints, and Twilio Client connections. Twilio's Voice API makes it easy to make, retrieve, control and monitor calls. Using this REST API, we can make outgoing calls, modify calls in progress, and query metadata about calls.

iv)System will also start writing frames and create video or capture image. If a person is detected, then the system will capture the current frame and send to the user through email. In addition, to prevent multiple uploads and spamming the intended email, the difference between the previous upload time and the current upload time was set to more than 15 seconds.

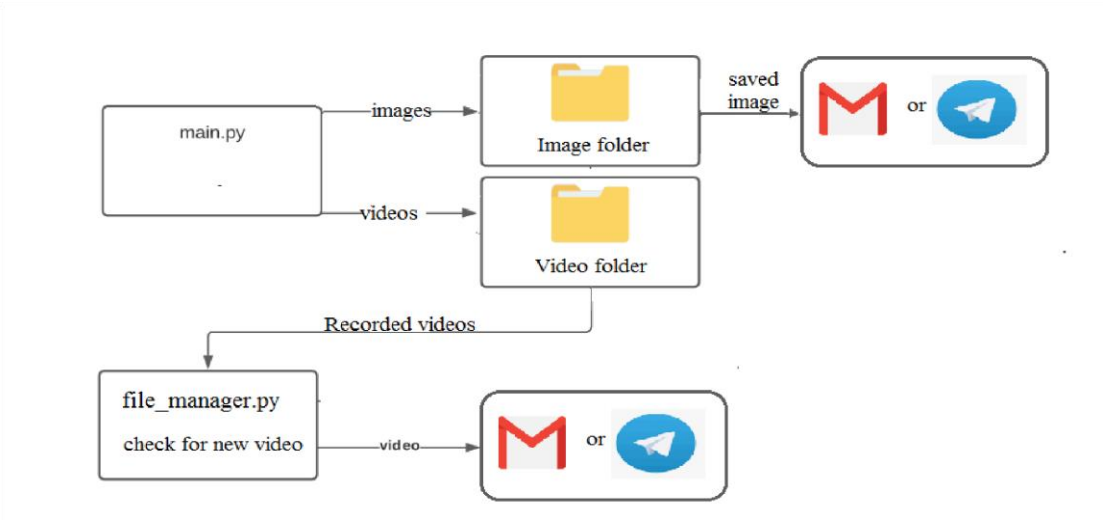


Fig 3.6(b)

v)Fig 3.6(b) shows the workflow of execution once recording of video or capturing of photo is done. Once presence of human is detected capture a image save inside folder once image is successfully sent delete the image. Simultaneously start recording videos.

vi)Once the video recording is done file manager checks for new video. If file manager detects new video stored inside video folder then mail function is triggered. The Multipurpose Internet Mail Extension (MIME) was used to accomplish the process of sending images through email. In this case, the captured image was attached and a simple Mail Transfer Protocol (SMTP) was used to send email from system to the Gmail server. If the photo or video is successfully sent delete the video from directory to avoid the memory problem.

vii)For user who wants to opt telegram service. Tele-pot bot library used to send the video or captured image to the user account. When program is initialized it listens for user command if user sends command /start. It activate advance security system feature. If user send command /stop to the bot then stop advance security feature only run live-stream. Once recording is done. Recorded video or captured image is sent to the user account. Telepot bot api gets called using request library and video is sent to user telegram account or mail account depends upon user choice. By using telepot lib a telegram bot can be automated. It provides various functions whether it is sending video, sending photo or sending message all these things can be automated.

viii)Continuously check for presence of human and repeat the process if system detects the presence of human.

3.7 Code:

3.7.1 Sending E-mail Notification:

```
import cv2
import sys
from mail import sendEmail
from flask import Flask, render_template, Response
from camera import video_camera
from flask_basicauth import BasicAuth
import time
import threading
import file_man
import RPi.GPIO as GPIO
from time import sleep
import pycalle

email_update_interval = 30 # sends an email only once in this time interval

# App Globals (do not edit)
app = Flask(__name__)
app.config['BASIC_AUTH_USERNAME'] = 'Vishwajeet'
app.config['BASIC_AUTH_PASSWORD'] = 'vj1999'
app.config['BASIC_AUTH_FORCE'] = True

basic_auth = BasicAuth(app)
last_epoch = 0
t1=threading.Thread(target=file_man.file,args=())
t1.daemon=True
t1.start()

```

```
def check_for_objects():
    global last_epoch
    GPIO.setmode(GPIO.BCM)
    buzzer=24
    GPIO.setup(buzzer,GPIO.OUT)
    counter=1
    cll_counter=0
    while True:
        try:
            frame,found_obj = video_camera.get_object()
            if found_obj and (time.time() - last_epoch) > email_update_interval:
                last_epoch = time.time()
                counter=counter+1
                cll_counter=cll_counter+1
                print ("Sending email...")
                sendEmail(frame)
                print ("done!")
                video_camera.record(counter,10)

                pycalle.x.cll()
                GPIO.output(buzzer,GPIO.HIGH)
            elif (time.time() - last_epoch) > 30:
                    GPIO.output(buzzer,GPIO.LOW)
            else:
                new_scene=False
                counter=0


        except:
            print ("Error sending email: ", sys.exc_info()[0])


@app.route('/')
# @basic_auth.required
def index():
    return render_template('index.html')

def gen(camera):
    while True:
        frame = camera.get_frame()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')

```

```
@app.route('/video_feed')
def video_feed():
    return Response(gen(video_camera),
                    mimetype='multipart/x-mixed-replace; boundary=frame')

if __name__ == '__main__':
    try:
        t = threading.Thread(target=check_for_objects, args=())
        t.daemon = True
        t.start()
        app.run(host='0.0.0.0',debug=False)
    except KeyboardInterrupt:
        buzzer=23
        GPIO.output(buzzer,GPIO.LOW)
        sys.exit()
```

3.7.2 Sending Telegram Notification:

```
from unicodedata import name
from flask import Flask,render_template,Response
from s import tl
from pycalle import x
import cv2
from flask_basicauth import BasicAuth
import time
import numpy as np
import sys
import os
# import tele22
import requests
import threading
import file_man
from camera import video_camera
from datetime import datetime
font = cv2.FONT_HERSHEY_DUPLEX
app=Flask(__name__)

basic_auth = BasicAuth(app)

email_update_interval = 30
t1=threading.Thread(target=file_man.file, args=())
t1.daemon=True
t1.start()
```

```python
# stop_threads = False

last_epoch = 0
def check_for_objects():
    unknown_counter=0
    global last_epoch

    now = datetime.now()
    video_file_count = 1

    tl.msg1()
    while True:
        try:

        # global stop_threads
            frame, found_obj = video_camera.get_object()
            if found_obj and  tl.flag :
                if (time.time() - last_epoch) > email_update_interval:
                    unknown_counter =unknown_counter+1
                    video_file_count += 1
                    now = datetime.now()
                    unknowns_name = "unknown" + str(unknown_counter)+".jpg"
                    last_epoch = time.time()
                    print ("Sending email...")
                    cv2.imwrite(rf"/home/pi/telegram_project/flask/unknown/{unknowns_name}",frame)
                    with open(rf"/home/pi/telegram_project/flask/unknown/{unknowns_name}","rb") as fobj:
                        files1 = {'photo': fobj }
                        resp=requests.post(f"https://api.telegram.org/bot5017602806:AAFHY5Ij7ytBTQ0Rv8LM
                        HxXDWLJ5T1cdEBM/sendPhoto?chat_id=-639570889&caption=security_alert+
                        {str(now.strftime('%Y-%m-%d %I-%M'))}",files=files1)
                    print(resp.status_code)
                    if resp.status_code==200:
                        os.remove(rf"/home/pi/telegram_project/flask/unknown/{unknowns_name}")
                    print ("done!")

                    if tl.cll_flag==True:
                        x.cll()
                    video_camera.record(video_file_count,15)
        except:
            print("Error sending message ", sys.exc_info()[0])

        # elif tele22.y==False:
        #     break

def gen(camera):
    while True:
        frame = camera.get_frame()
        # mail.sendEmail(frame)
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')
@app.route("/")
def index():
    return render_template("index.html")
@app.route("/video")
def video():
    return Response(gen(video_camera),mimetype='multipart/x-mixed-replace; boundary=frame')

if __name__=="__main__":

    t = threading.Thread(target=check_for_objects, args=())
    t.daemon = True
    t.start()
    app.run(host='0.0.0.0',debug=False)
    if tl.flag==False:
        sys.exit()
```

3.8 Applications:

- **Home security system:**
  Home security system has established its importance and benefits numerous times by providing immediate monitoring of the house. CCTV based security systems are not real-time because the alert comes to the owner after the incident occurred unless they are at home during the incident. The traditional approach of security system can be replaced by AI security system

- **Restricted Areas:**
  There are some places such as government properties or military areas where no one can enter without permission.In such places this system can be implemented ensuring that no one allowed if intruder presence detected security authorities get notified with footage.

- **Private Sector:**
  Private sector does not hold back when it comes to installing the cameras in the work environment mainly to avert unethical practices or the department where confidential files are kept.

- **Banking Sectors:**
  Banking sectors such as locker rooms. Where only authorized person are allowed.

- **Retails Shops:**
  There are chances of break-ins in shops such as jewellery shops, mobile shops etc and the chances of break-ins mostly at night. In such cases intruder get detected instantly and owner get notified with footage instantly

3.9 Features:

- Smart hub that alerts you of suspect behavior.
- Smart camera connected to monitor efficiently your house.
- Smart camera that sends you snapshots of instance once the presence of human is detected.
- Once it detects the situation the video is also recorded and that video with attachment is also sent along with timestamp.
- Remotely start an call to the user when intruders have been spotted in the scene.
- Feature of live-stream is also added devices connected over local network has access toward stream. Support for remote live-stream can be further added when connected with router along with port forwarding.
- Also support for telegram is also available user can choose either telegram or mail system depend on user convenience.

CHAPTER 4

RESULTS

4.1 Connections:

Prototype of the developed system using Raspberry Pi 4 model B and Pi Camera Video Module. Fig 4.1 shows connections for final system.

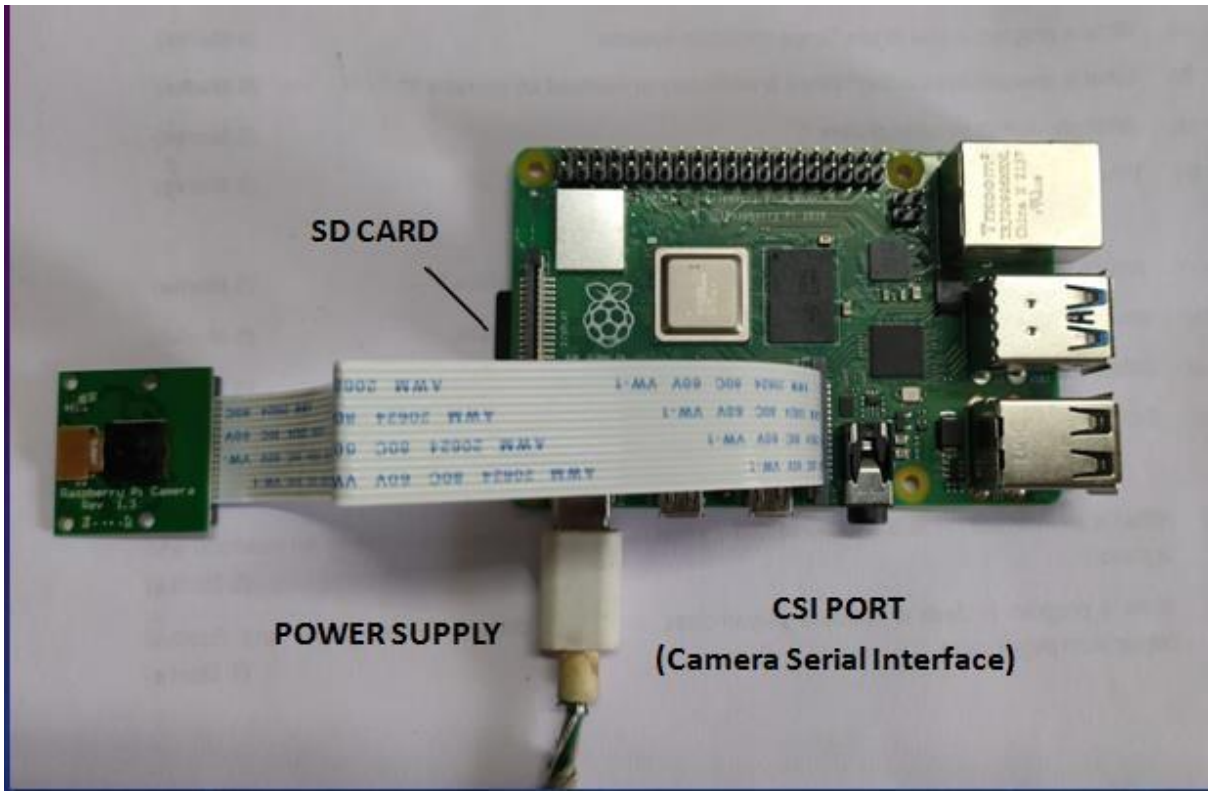


Fig4.1 Final Setup

4.2 Results:

4.2.1 Live Streaming:

With the help of flask a live-stream from camera can be displayed on html page and it is hosted on local server. Fig 4.1(a) shows the stream hosted on flask server.

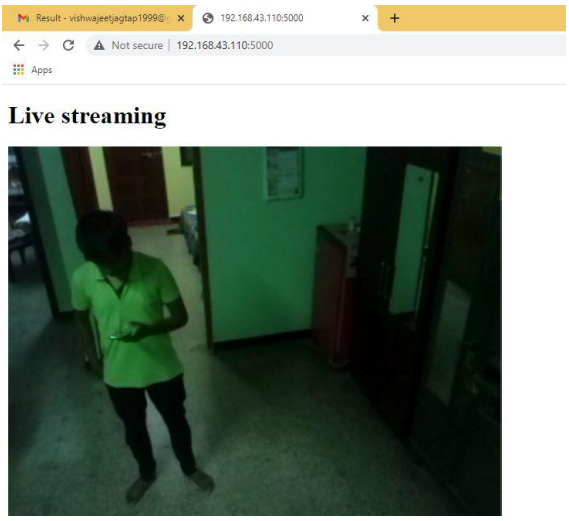


Fig 4.2.1(a)

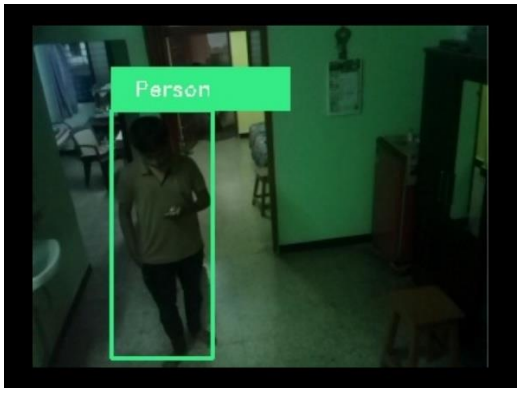


Fig 4.2.1(b)

The Figure 4.1(b) shows the captured frames which are again processed with tensor-flow object detection api using pre-trained ssd_mobilenet_v1_coco model. Successfully tested model and it is capable for detecting various classes but instead it is set for detecting only human class neglecting all other classes.

4.2.2 E-Mail Notification:

Figure 4.1(c) shows the email notification received from the system once person is detected. Mail feature is working as expected. Along with image instance of video is also sent which contains positive classification.

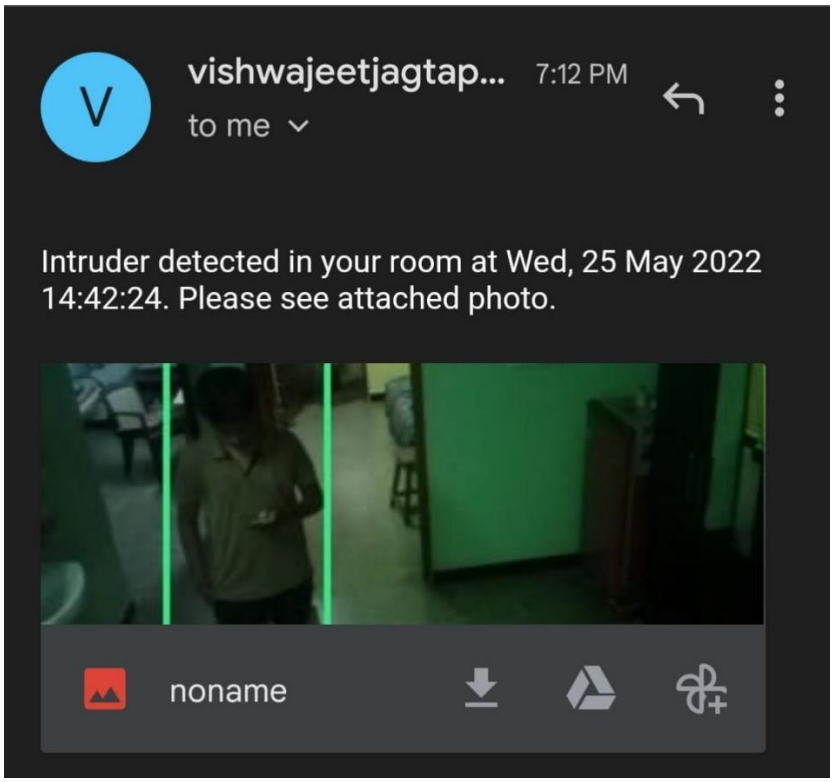


Fig 4.2.2

4.2.3 Call Notification:

When the presence of human is detected call is triggered with the help of twilio api. Fig 4.1(d) shows the call generated by system once the presence of human is detected.

Fig 4.2.3

4.2.4 Telegram Notification:

Same system can be implemented with the help of telegram bot. Telegram provides bot which can be automated with the help of telepot lib. Which can give user access to turn on or turn off the advance security system feature with the help of bot commands. The bot can also added on personal group in the group more members can be added so that simultaneously all the users which are members of the group can be notified at the same time.

Fig 4.2.4(a)shows system approach with telegram connectivity.

Fig 4.2.4(a)

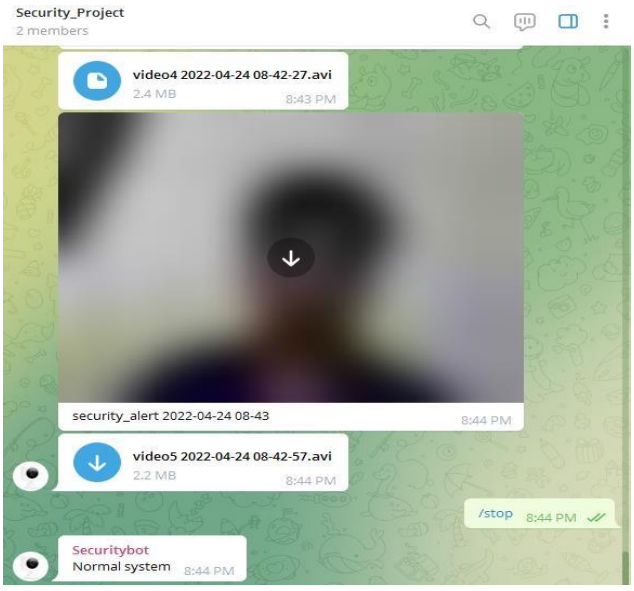


Fig 4.2.4(b)

Fig shows the video and photo received with the help of telegram which is another approach of this project.

CHAPTER 5

CONCLUSION

5.1 Conclusion:

- This proposed system aims at enhancing the safety of property and security from intruders/burglars with the help camera integrated with ML model.

- The footage can be processed with Raspberry pi 4B and with the help of machine learning algorithm it can automatically monitors and detects people in a scene, and then alerts the user in real time by sending an image and video of victim through their email or Telegram app and also alert user by making phone call.

5.2 Future Scope:

- Implement Facial Recognition algorithm to the system. Using the facial images that the owner provides, the system will be able to recognize those faces and it can differentiate between a familiar faces and unfamiliar faces. During a burglary, the system easily recognizes the burglar because of his unfamiliar face and automatically triggers the alarm.

- Integrate Deep Learning algorithm to the system. The facial characteristics of a person will differ from time to time. Thus, it will be troublesome for the user to keep updating the latest facial images of the familiar faces. Using Deep Learning algorithm, a collection of data from the owner's smart-phone such as captured images and videos is used to train the system automatically to recognize familiar faces. This will further improve the rate of detection and accuracy of the system to make it more robust.

- The second improvement that could be made is to increase the model's performance on person identification. For example, the model could be updated with a white-list, and then the Security system would only alert if someone was present who was not on the white-list. Another area would be to update the model to recognize people delivering packages or mail.

5.3 References:

1. Zijun Zhang, "Improved Adam Optimizer for Deep Neural Networks", *IEEE/ACM 26th* International Symposium on Quality of Service, *vol. 201*.

2. Yao Ying, Jianlin Su and Peng Shan, "Rectified Exponential Units for Convolution Neural Network", *International Journal of IEEE Access*, vol. 7, pp. 101633-101640, 2019.

3. A Sensor-based home security system using PIR sensor on Raspberry Pi 3 AIP Conference Proceedings **2173**, 020013 (2019)

4. https://towardsdatascience.com/video-streaming-in-web-browsers-with-opencv-flask-93a38846fe00

5. https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/training.html

6. https://docs.nvidia.com/metropolis/deepstream/dev-guide/

7. The Ultimate Guides For Making Computer Vision Using Python: Conda Install Opencv-Python Jesus Sanos

8. Open-Cv documetation https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html

9. https://telepot.readthedocs.io/en/latest/

5.4 Total Expenditure:

SR. NO.	Item	Quantity	Price (In RS)
1	RASPBERRY PI – 4	1	8000.00
2	RASPBERRY PI CAMERA 5MP	1	350.00
	TOTAL	**2**	**8350.00**
		Transport charges	60.00
		CGST	757.00
		SGST	757.00
		TOTAL	**9924.00**