Project Report

on

Covid-19 Face Mask Detection and Social Distancing at Public Places

Submitted in the partial fulfillment of the requirements for
The Degree of
Bachelor of Technology
in
Electronics Engineering

By
Nallamala Pooja (1714110447)
Isha Mandawkar (1714110443)
Aditya Chakraborty (1714110414)

Under the guidance of
Prof.S.P.Tondare

Department of Electronics Engineering
Bharati Vidyapeeth
(Deemed to be University)
College of Engineering, Pune – 4110043
Academic Year 2021-2022

Bharati Vidyapeeth
(Deemed to be University)
College of Engineering,
Pune – 4110043

DEPARTMENT OF ELECTRONICS ENGINEERING

CERTIFICATE

Certified that the project report entitled**, "Covid-19 Face Mask Detection and Social Distancing at Public Places"** is a bonafied work done by **Nallamala Pooja**

(1714110447), Isha Mandawkar (1714110443), Aditya Chakraborty (1714110414) in partial fulfillment of the requirements for the award of degree of Bachelor of Engineering in Electronics Engineering

Date:

Prof. S.P.Tondare
Project Guide

Prof.Dr.Dhiraj Dhane
Project Co-ordinator

Prof.Dr.A.A.Shinde
Head Of Department

ACKNOWLEDGEMENT

We here take this opportunity to express our heartful gratitude to following personalities with whose guidance, advice and assistance we have been able to successfully complete our project on , **“Covid-19 Face Mask Detection and Social Distancing at Public Places”**.

We would like to thank **Prof. Dr. Vidula Sohoni, Principal**, Bharati Vidyapeeth (Deemed to be University) College Of Engineering Pune, **Prof. Dr. A.A. Shinde, Head of the Department**, **Electronics Engineering** for mentoring and motivating us to work on this project.

Prof. S.P.Tondare, Our project mentor has given her knowledge and her time to plan and execute this project. We would like to thank her for her full support without which we would not be able to do this project. We would like to thank her for her expertise and guidance in the Software as well as Hardware domain.

We would also like to express our gratitude to **Prof. Dhiraj Dhane, Project Co-ordinator** for his guidance.

Thanking You,

Nallamala Pooja (1714110447)

Isha Mandawkar (1714110443)

Aditya Chakraborty (1714110414)

LIST OF FIGURES & TABLES

List of Tables

List of Figures

INDEX

ABSTRACT

To propose a model which could classify in real-time if an individual is wearing a face mask or not wearing a face mask and maintaining social distancing or not. In this time, when COVID-19 is spreading rapidly, it is essential to maintain social distance and wearing a mask is compulsory and to avoid large public gatherings at one place to break the chain of corona infection. This whole system can be placed in public gatherings, traffic signals, on roads and at the entrance of schools and college gates.

This system uses a Raspberry Pi with an RPi camera for capturing live video. Here, we suggest Image processing for crowd detection and face mask detection because, the automation of data collection and analysis of crowd behavior is increasingly desirable in design of facilities and long-term management using image processing techniques with existing closed circuit television systems. Our method proposes to use Raspberry Pi board and RPi camera as a live capturing video process as an input and various image processing libraries of python like TensorFlow, Open CV and deep learning method to process that input. The SSDMNV2 approach uses single shot multibox detector as a face detector and MobilenetV2 architecture as a framework for the classifier, which is very lightweight and can be used in embedded devices to perform real-time mask detection.

The applications of the project consists of suspicious activity management,military management,safety monitoring,disaster management,public events management.

CHAPTER 1.

INTRODUCTION

1.1 INTRODUCTION

People's safety has become a major problem in many areas like malls, railway stations and streets during this pandemic situation where people gather in crowds. An efficient automated system to manage the crowd is essential. In a high-density crowd, because of inter-object occlusion, detection of humans in the crowd will be a challenge in computer vision. Wearing facemasks and maintaining social distance is rising due to the covid-19 virus all over the world. Before COVID-19, people used to wear masks to protect their health from air pollution. More than five million cases were infected by COVID-19 in less than 6 months across 188 countries. The virus spreads through close contact and in crowded and overcrowded areas.

We can tackle and predict new diseases by the help of new technologies such as artificial intelligence, IOT, big data and machine learning. In order to better understand infection rates might be decrease through our technique. People are forced by laws to wear face masks in public in many countries. These rules and laws were developed as an action to the exponential growth in cases and deaths in many areas. However, the process of monitoring large groups of people is becoming more difficult in public areas. So, we will create a automation process for detecting the faces. Social distancing is one of the pharmaceutical infection control actions that can stop or slow down the spread of a highly contagious disease. The virus that causes COVID-19 is spreading easily from person-to-person. We can tackle and predict new diseases with the help of new technologies such as artificial intelligence, IOT, big data and machine learning. Existing systems are incapable of detecting facial masks with drastically low latency. So, we will create an automation process for detection.

Here we introduce a facemask detection model that is based on computer vision and deep learning. The proposed model can be integrated with surveillance cameras to impede the COVID-19 transmission by allowing the detection of people who are 6 wearing masks not wearing face masks. The model is integration between deep learning and techniques with Open cv, Tensor flow and keras and image processing techniques are also used. We will achieve the highest accuracy and consume the least time in the process of training and detection. Image Processing is the process of using algorithms to manipulate and analyze images for the purpose of enhancing the quality or obtaining relevant information. Digital image processing is the use of a digital computer to process digital images through an algorithm. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and distortion during processing. . It will be used for object recognition, pattern recognition, locating duplicates, image search, etc. the most image process techniques are Image preprocessing, Image improvement, Image segmentation, feature extraction and Image classification.[2]

1.2 PURPOSE OF PROJECT

More than five million cases were infected by COVID-19 in less than 6 months across 188 countries. The virus spreads through close contact and in crowded and overcrowded areas. In this time, when Covid-19 is spreading rapidly, it is essential to maintain social distance, compulsory wearing a mask and to avoid large public gatherings at one place to break the chain of corona infection. But maintaining this is not easy. Many people, knowingly or unknowingly, gather on the streets without wearing a mask. Keeping an eye fixed on these activities isn't a simple job. The authorities need reliable technology to keep track of these activities. This project can help in monitoring the social distance and also it detects face mask. The system is accurate. Thus, it makes easier to deploy our model to embedded system like Raspberry Pi, etc This system uses a Raspberry Pi with an RPi camera for capturing live video. We believe that this approach will enlarge the safety of the individuals during the pandemic.

1.3 SPECIFICATIONS OF PROJECT

- We will be using Python for Image processing or other software because of its popularity as a scientific programming language and the free availability of many cutting-edge image processing tools in its ecosystem.
- We propose to use Raspberry Pi Camera for capturing real time images ,which would be integrated with our software for further processing. Python happens to be one of the most popular and desired language in the field of AI. Hence, its robustness and compatibility can go well with mulitple platforms, wherever this method is executed. The choice of this language also aims towards more reliability and easy access for the end user.
- There are some libraries for this specific purpose, each having their unique characteristics and ultimate adaptability. Some of them are:

1. Espeak, Numpy, Open CV.
2. Keras, TensorFlow
3. Deep Learning

- Raspberry Pi 4 Model B specifications:-

1. Processor: Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @
2. 1.5GHz
3. Memory: 1GB, 2GB or 4GB LPDDR4 (depending on model)
4. Connectivity: 2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth
5. 5.0, BLE Gigabit Ethernet 2 × USB 3.0 ports 2 × USB 2.0 ports.
6. GPIO: Standard 40-pin GPIO header (fully backwards-compatible with previous boards)
7. 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
8. Multimedia: H.265 (4Kp60 decode); H.264 (1080p60 decode, 1080p30 encode);
9. OpenGL ES, 3.0 graphics
10. 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)
11. Environment: Operating temperature 0–50ºC

- Raspberry Pi Camera:-

1. Image Sensor: Sony IMX 219 PQ CMOS image sensor in a fixed-focus module. Resolution: 8-megapixel
2. Still picture resolution:3280 x 2464
3. Max image transfer rate 1080p: 30fps (encode and decode) 720p: 60fps Connection to Raspberry Pi 15-pin ribbon cable, to the dedicated 15-pin MIPI Camera Serial Interface (CSI-2).
4. Image control functions Automatic exposure control Automatic white balance Automatic band filter Automatic 50/60 Hz luminance detection Automatic black level calibration
5. Temp range Operating: -20º to 60º Stable image: -20º to 60º

1.4 LITERATURE REVIEW

[1] Vinitha and velantina, "Covid-19 Facemask Detection with Deep Learning and Computer Vision" [1]

This study used integration between deep learning and machine learning techniques with Open CV, TensorFlow and keras. Deep learning is used for feature extractions and combined it with three machine learning algorithms. This helped to capture the images and differentiate the images of people who were wearing masks and people who were not wearing masks.

[2] Adrian Rosebrock, "Open CV Social Distancing Detector"[5]

This study used YOLO object detector to detect people in a video stream and determine the centroids for each detected person. By checking the pairwise distances were By checking the pairwise distances were <N pixels apart and indicating the pair of people violated social distancing rules.

[3] C.Jagadeeswari, M.Uday Theja, "Performance Evaluation of Intelligent Face Mask Detection System with various Deep learning Classifiers"[2]

The classification is done in 2 phases. First phase includes the facemask dataset is loaded into the system which are being trained with deep learning classifiers and the second phase loads the face mask classifier which detects the people with and without mask.

[4] Dr. Syed Ameer Abbas, "Human Monitoring and Crowd Control with Raspberry Pi and Open CV"[4]

A cascade classifier was educated for head detection from the scene using haar features. The aim of this study was to use a proejctor and a Raspberry Pi 3 with a quad core central Processing unit to film the crowded scene and process the footage frame by frame. The crowd is calculated and handled by adjusting the value to the threshold.

CHAPTER 2

SYSTEM CONFIGURATION

2.1 SOFTWARE AND HARDWARE TOOLS

HARDWARE TOOLS

[1] Raspberry Pi 4 Board

[2] Raspberry Pi Camera

[3] Power Adapter

[4] Wires

SOFTWARE TOOLS

[1] Required libraries in Python Numpy, Scipy, Opencv, Dlib, Keras, Tensorflow, Deep Learning.

2.2 RASPBERRY PI HARDWARE SPECIFICATIONS

- The raspberry pi board comprises a program memory (RAM), processor and graphics chip, CPU, GPU, Ethernet port, GPIO pins, Xbee socket, UART, power source connector. And various interfaces for other external devices. It also requires mass storage, for that we use an SD flash memory card. So that raspberry pi board will boot from this SD card similarly as a PC boots up into windows from its hard disk.

- Essential hardware specifications of raspberry pi board mainly include SD card containing Linux OS, US keyboard, monitor, power supply and video cable. Optional hardware specifications include USB mouse, powered USB hub, case, internet connection, the Model A or B: USB WiFi adaptor is used and internet connection to Model B is LAN cable.

- MEMORY:- The raspberry pi model Aboard is designed with 256MB of SDRAM and model B is designed with 51MB.Raspberry pi is a small size PC compare with other PCs. The normal PCs RAM memory is available in gigabytes. But in raspberry pi board, the RAM memory is available more than 256MB or 512MB.

- CPU (CENTRAL PROCESSING UNIT):-The Central processing unit is the brain of the raspberry pi board and that is responsible for carrying out the instructions of the computer through logical and mathematical operations. The raspberry pi uses ARM11 series processor, which has joined the ranks of the Samsung galaxy phone.

- GPU (GRAPHICS PROCESSING UNIT):-The GPU is a specialized chip in the raspberry pi board and that is designed to speed up the operation of image calculations. This board designed with a Broadcom video core IV and it supports OpenGL.

- ETHERNET PORT:-The Ethernet port of the raspberry pi is the main gateway for communicating with additional devices. The raspberry pi Ethernet port is used to plug your home router to access the internet.

- GPIO PINS:-

The general purpose input & output pins are used in the raspberry pi to associate with the other electronic boards. These pins can accept input & output commands based on programming raspberry pi. The raspberry pi affords digital GPIO pins. These pins are used to connect other electronic components. For example, you can connect it to the temperature sensor to transmit digital data.

- XBEE SOCKET:-

The XBee socket is used in raspberry pi board for the wireless communication purpose.

- POWER SOURCE CONNECTOR:- The power source cable is a small o switch, which is placed on side of the shield. The main purpose of the power source connector is to enable an external power source.

- UART:-The Universal Asynchronous Receiver/ Transmitter is a serial input & output port. That can be used to transfer the serial data in the form of text and it is useful for converting the debugging code.

- DISPLAY:-The connection options of the raspberry pi board are two types such as HDMI and Composite. Many LCD and HD TV monitors can be attached using an HDMI male cable and with a low-cost adaptor. The versions of HDMI are 1.3 and 1.4 are supported and 1.4 version cable is recommended. The O/Ps of the Raspberry Pi audio and video through HMDI, but does not support HDMI I/p. Older TVs can be connected using composite video. When using a composite video connection, audio is available from the 3.5mm jack socket and can be sent to your TV. To send audio to your TV, you need a cable which adjusts from

- 3.5mm to double RCA connectors.

2.3 RASPBERRY PI 4 BOARD

Fig 2.1:- Raspberry Pi 4 Board Model B (Courtesy:- Raspberry Pi.com) [5]

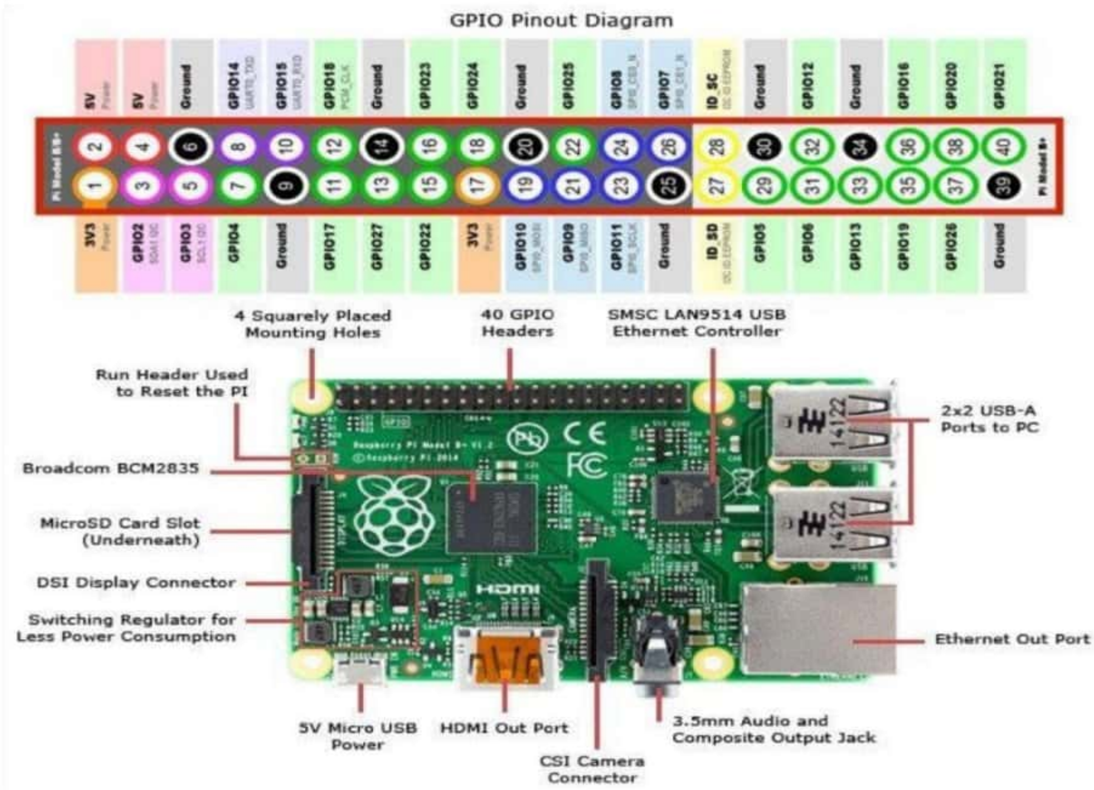

Fig 2.2:- Pin configuration and description of components of Raspberry Pi 4 Board Model B (Courtesy:- Raspberry Pi.com)[5]

The Raspberry Pi is a Broadcom BCM2835 SOC (system on chip board). It comes equipped with a 700 MHz, 512 MB of SDRAM and ARM1176JZF-S core CPU. The USB 2.0 port of the raspberry pi boards uses only external data connectivity options. The Ethernet in the raspberry pi is the main gateway to interconnect with other devices and the internet in model B. This draws its power from a micro USB adapter, with a minimum range of 2.5 watts(500 MA). The graphics, specialized chip is designed to speed up the manipulation of image calculations. This is in built with Broadcom video core IV cable, that is useful if you want to run a game and video through your raspberry pi.

2.3.1 FEATURES OF RASPBERRY PI 4 BOARD

- 512 MB SDRAM memory
- Broadcom BCM2835 SoC full high definition multimedia processor
- Dual Core Video Core IV Multimedia coprocessor
- Single 2.0 USB connector
- HDMI (rev 1.3 and 1.4) Composite RCA (PAL & NTSC) Video Out
- MM Jack, HDMI Audio Out

- MMC, SD, SDIO Card slot on board storage
- Linux Operating system
- Dimensions are 8.6cm*5.4cm*1.7cm On board 10/100 Ethernet RJ45 jack

2.4 RASPBERRY PI CAMERA

- High Definition camera module compatible with all Raspberry Pi models. Provides high sensitivity, low crosstalk and low noise image capture in an ultra small and lightweight design.

- The camera module connects to the Raspberry Pi board via the CSI connector designed specifically for interfacing to cameras. The CSI bus is capable of extremely high data rates, and it exclusively carries pixel data to the processor.

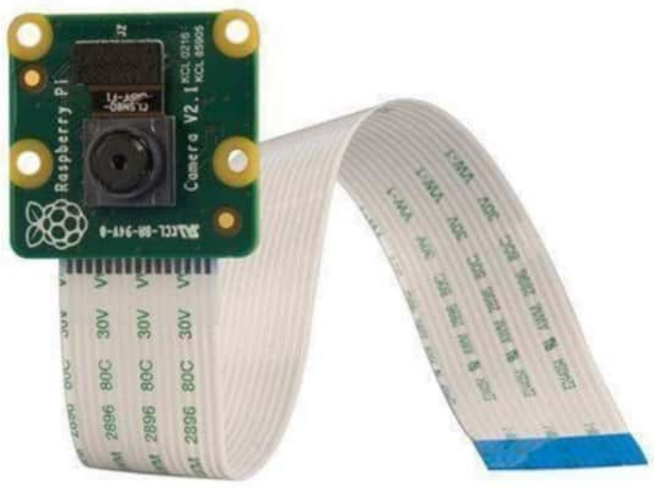

Fig 2.3:- Raspberry Pi Camera (Courtesy:-Raspberry Pi.com)[5]

2.5 IMAGE PROCESSING

- There are still significant problems in the planning, design and management of public facilities subject to dense pedestrian traffic. The automation of data collection and analysis of crowd behaviour is increasingly desirable in design of facilities and longterm site management using image processing techniques with existing closed-circuit television systems. We have investigated a number of techniques for crowd density estimation, movement estimation, incident detection and their relative merits using image processing. This paper presents techniques for background generation and calibration to enhance the previously-developed method of crowd density estimation using a reference image.

- An intensity region related to the average pixel intensity of each image in a sequence of crowd images is used to segment background pixels for generating a background image without pedestrians. The calibration approach, with which a previouslyestablished relationship between image parameters and crowd density at one site can be used to estimate crowd density at various sites, involves calibration of the crowd image as opposed to calibration of the camera. Both techniques may be used in other surveillance systems such as vehicle monitoring.

2.6 CONNECTION OF RASPBERRY PI BOARD

- Connect the camera to the RPi4 board using a ribbon cable and power the Raspberry Pi using an adaptor to get a live video output from the camera.
- You can also use a VNC. Just connect RPi and PC/Laptop with your network/hotspot and then mirror its screen using VNC. Now, connect a speaker to the audio port of RPI and power it. If you want any alert using bulb/buzzer, then connect the buzzer to GPIO pin 17 of Raspberry Pi.

2.7 BLOCK DIAGRAM

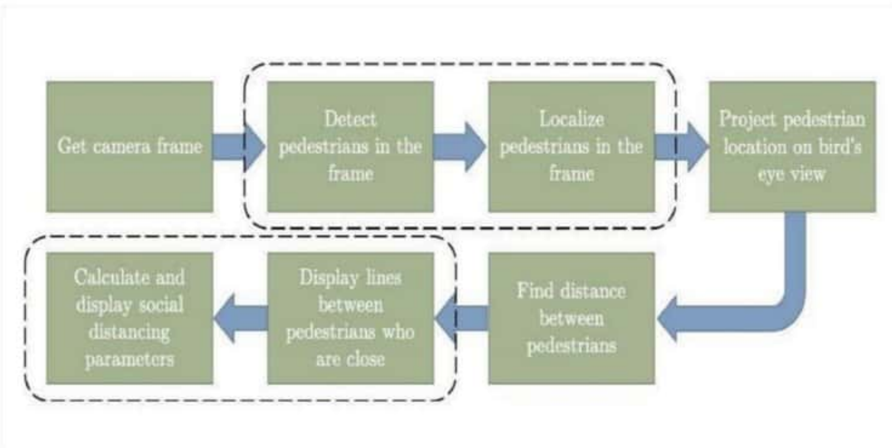

Fig 2.4:- Proposed diagram of Social Distancing

2.8 DATABASE

Data processing involves conversion of data. A technique adopted commonly in image data classification, which generates several images for a single source image at different orientations and zoom levels. It helps to substantially increase the data required for predictions and classifications by adjusting the conditions of data biasedness.

2.9 APPLICATION DIRECTORY

An application directory is a grouping of software code, help files and resources that together comprise a complete software package but are presented to the user as a single object. An application director oversees all activities related to application development. They coordinate tasks and supervise work phases related to the implementation of computer applications. Analyzing the performance of applications is also their responsibility.

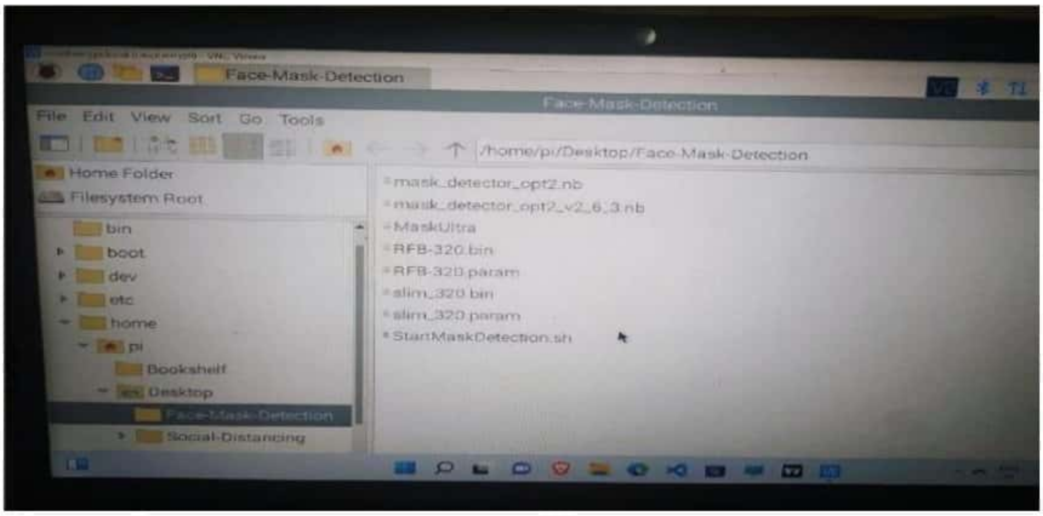

Fig 2.5:- Proposed diagram of Application Directory

2.10 WORKING:-

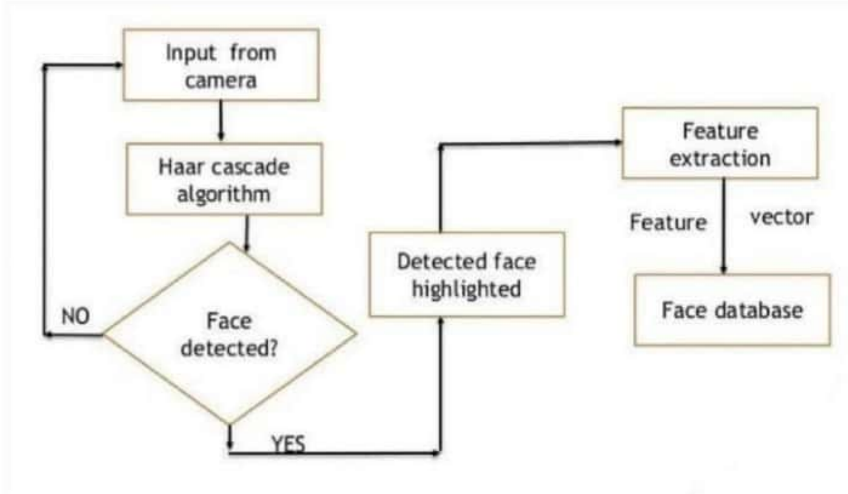

Fig 2.6:-Proposed Flowchart of Face Mask Detection

- First RPi camera captures the video in the area placed. The images are then converted into coordinates.
- In phase 1, we will train the model. Here we'll focus on loading the face mask detection dataset and training the model using TensorFlow and OpenCV on this dataset. These modules help to get the raw images data from huge open-source images data set.
- In phase 2, we will apply the face mask detector. Once the face mask detector is trained, we can then move on to loading the mask detector, performing face detection and then classifying each face as with mask or without mask.

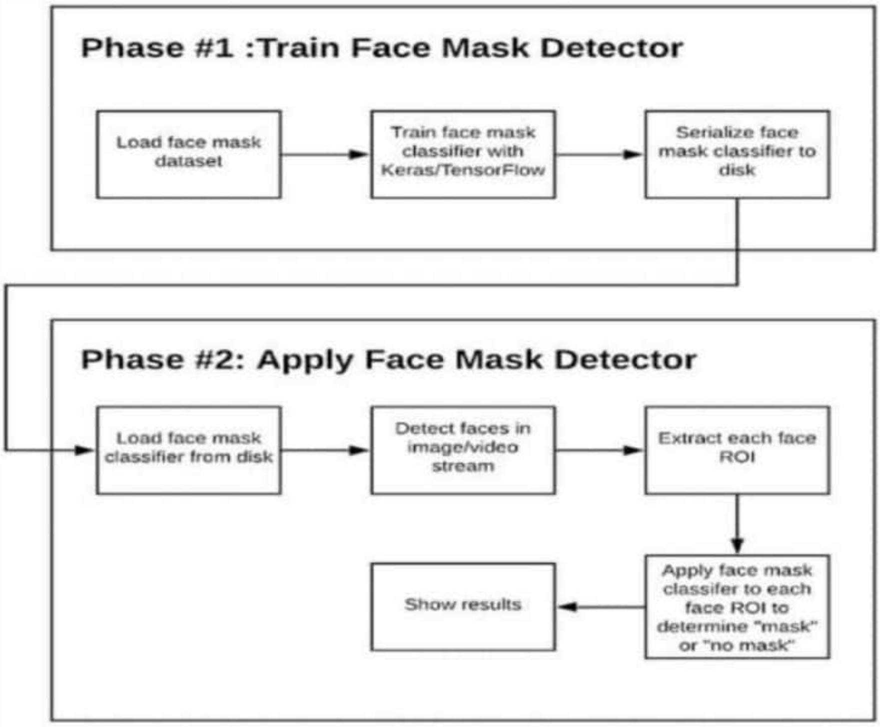

Fig 2.7:- Proposed Block diagram of Face Mask Detector

- Apply Object detection to detect all people in a video stream.
- Based on the distances, we check to see if any two people are less than N pixels apart.
- For the most accurate results, you should calibrate your camera through intrinsic/extrinsic parameters so that you can map pixels to measurable units.
- Open CV social distancing detector implementation will rely on pixel distances.

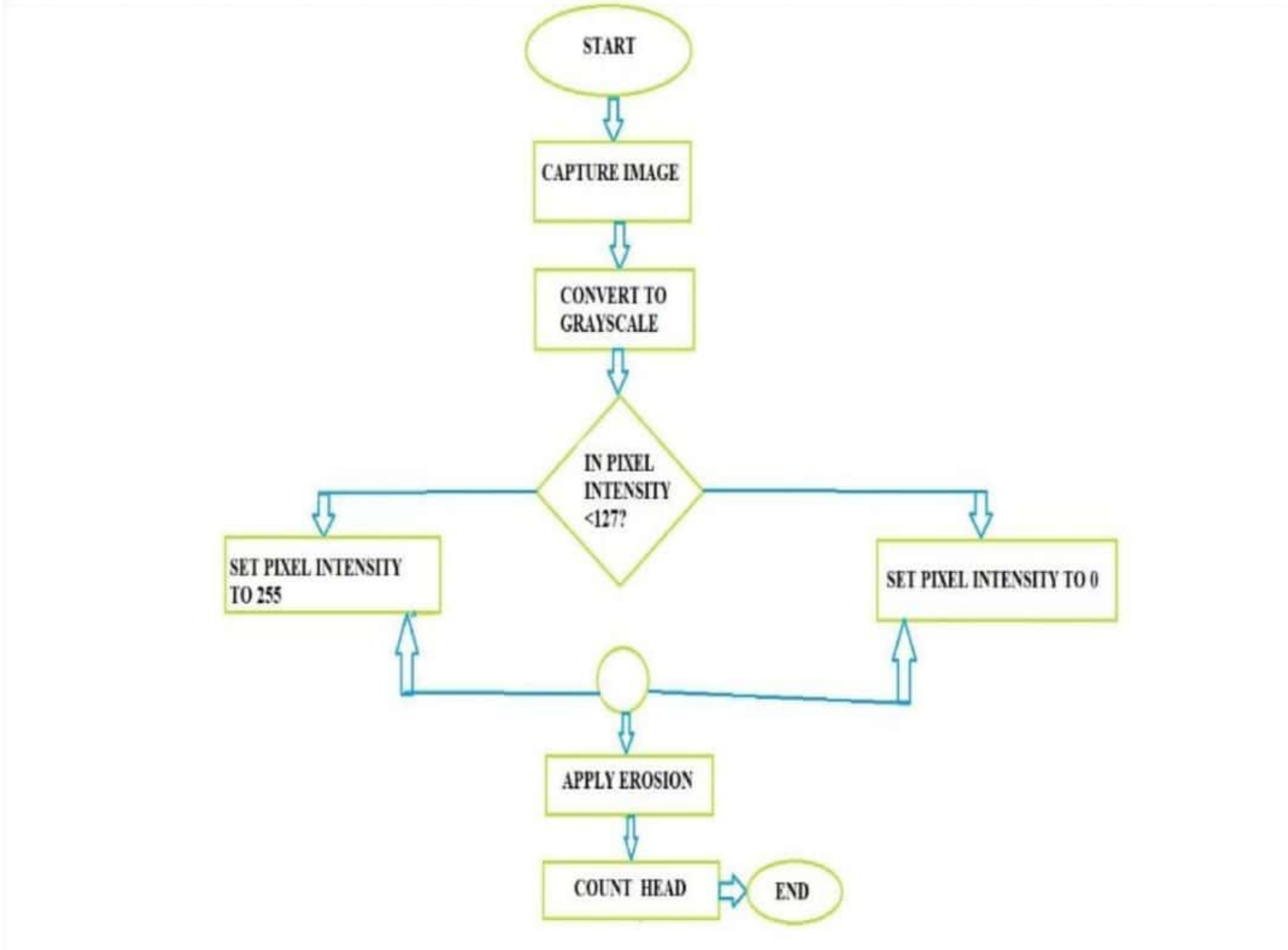

Fig 2.8:-Proposed Flowchart of Social Distancing

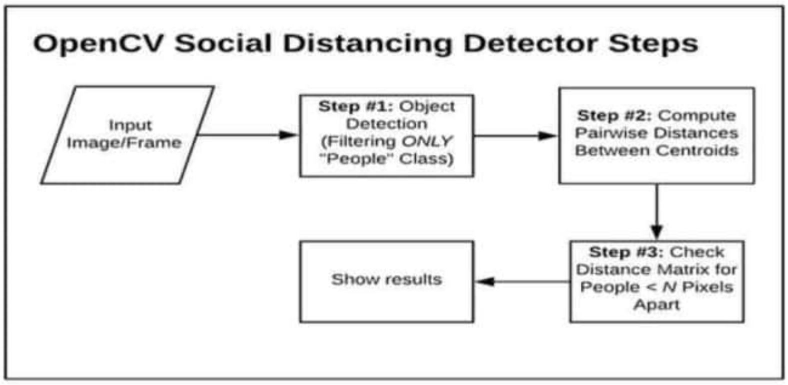

Fig 2.9:- Proposed diagram of Social Distancing Detection

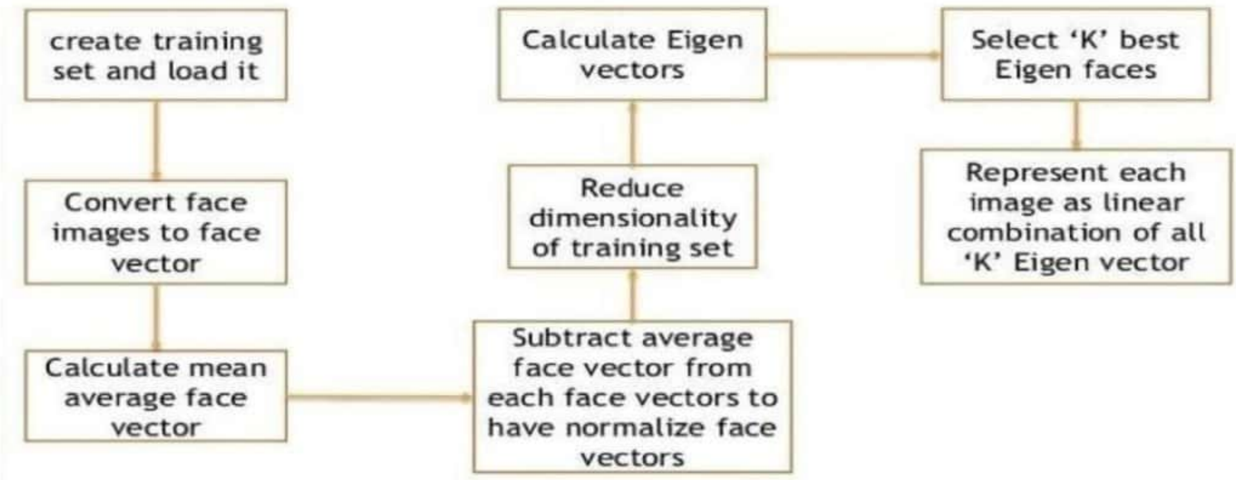

Fig 2.10 :- Proposed Block diagram of database using PCA algorithm for training dataset and to represent the eigen vector

DISTANCE MEASUREMENT :-

For each pedestrain, the position in the top-down view is estimated based on the bottom-center point of the bounding box. The distance between every pedestrain pair can be computed from the top-down view and the distances is scaled by the scaling factor estimated from camera view calibration. Given the position of two pedestrains in an image as (x1,y1) and (x2,y2) respectively, the distance between the two-

$$d = \sqrt{(x2 - x1)^2 + (y2 - y1)^2} \quad \text{…………………(Equation 1)}$$

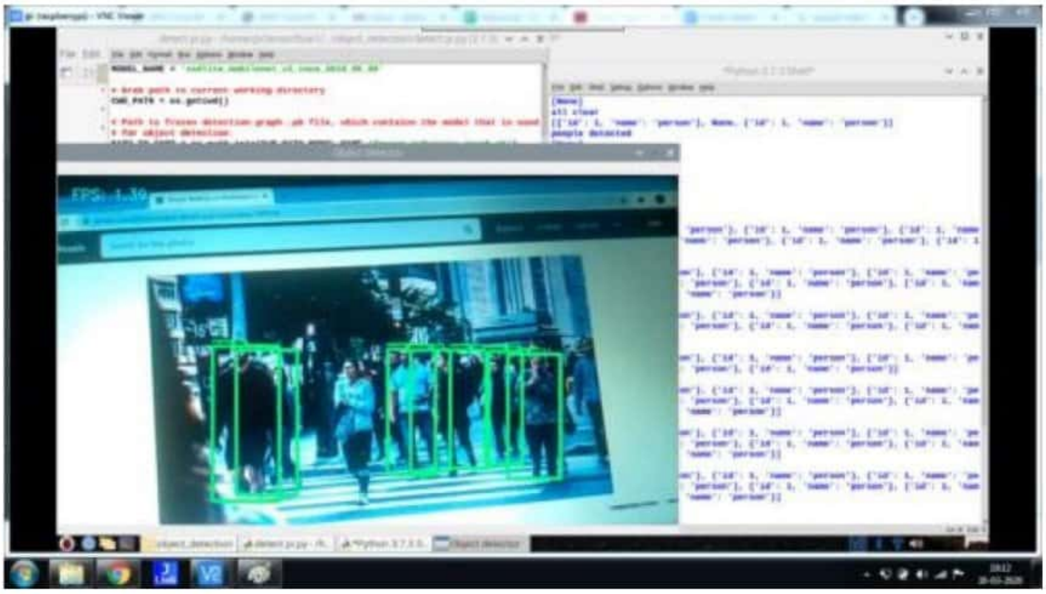

Fig 2.11:- Proposed Representation of an image detection window

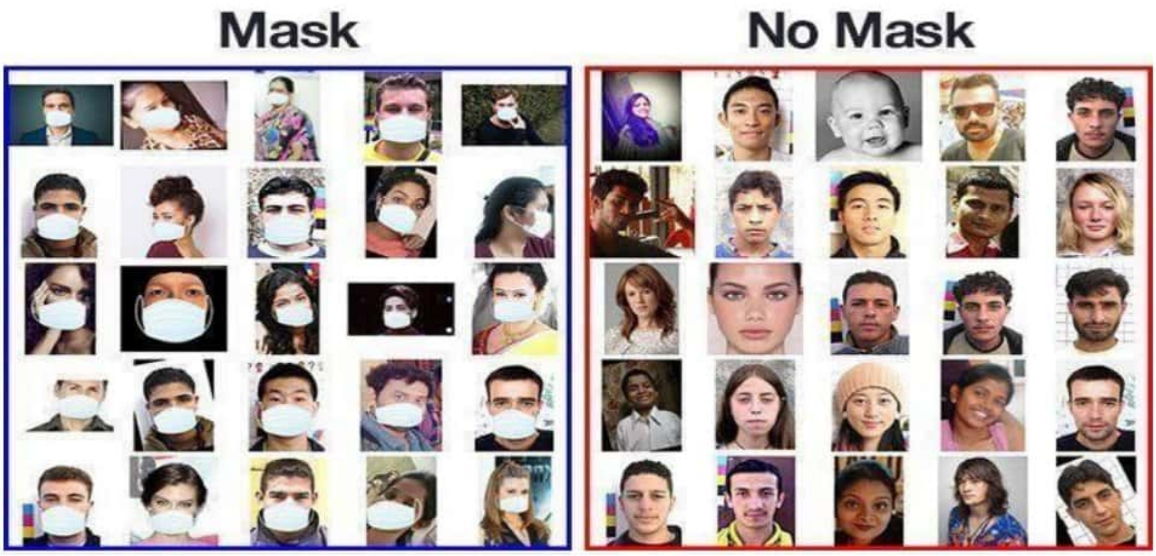

Fig 2.12 :- Representation of people wearing masks and not wearing masks

(Courtesy:- PyImage(pypi.org)) [8]

ADVANTAGES OF CROWD DETECTION CAMERA :-

- It can detect high traffic on roads.
- Identify any unauthorised human entry in a restricted area.
- Observe the number of people in hall/auditorium.
- Maintain and monitor the flow of the crowd in a protest demonstration

DISADVANTAGES OF CROWD DETECTION CAMERA :-

- It can't detect whether people are wearing masks or not. It can detected by using facial recognition system.
- It cannot detect in night without light.

APPLICATIONS OF CROWD DETECTION CAMERA :-

- Suspicious activity detection.
- Military management.
- Safety monitoring.
- Disaster management.
- Public events management

CHAPTER 3

RESULTS

The proposed work has been compared with existing research work in the same domain and it outfromed them. The prescribed model has also been evaluated on a test set. The model learns effectively on the training data. It learns fine details of images accurately that it ends up miss-classifying the predicted data. As a result, the model gets saturated while evaluating the validation set of data.

Fig-3.1:- Proposed image of a person without and with mask

Fig 3.2:- Proposed image of a person without and with mask

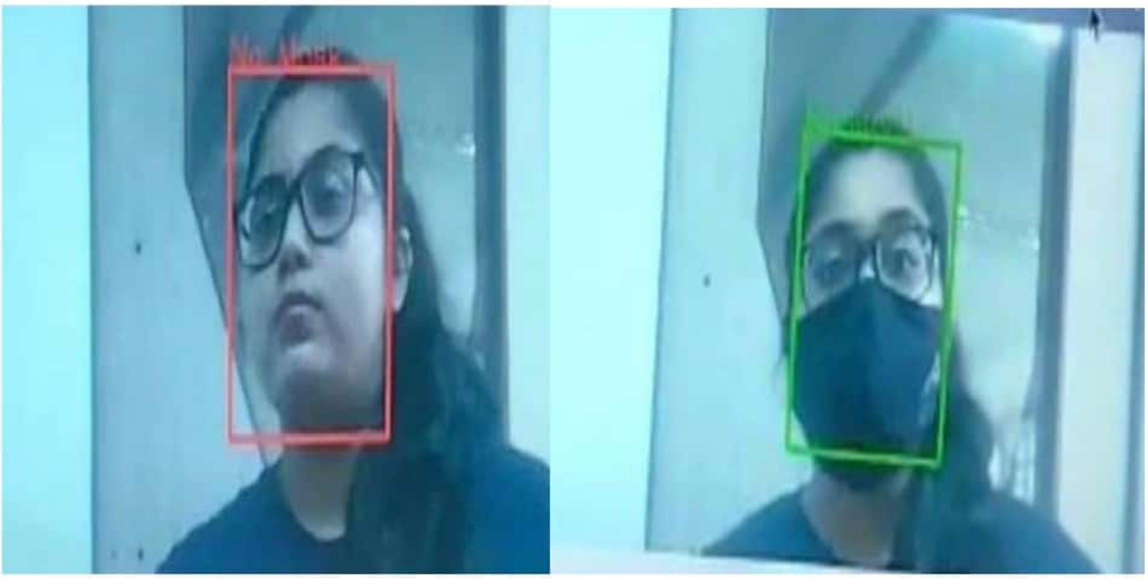

Fig 3.3:- Proposed image of a person without and with mask

Table 3.1:- Proposed Representation of the table consisting image accuracy

SR NO	IMAGES	ACCURACY
1.	Image 1 :- A person without and with mask (Fig 3.1)	13.31
2.	Image 2 :- A person without and with mask (Fig 3.2)	12.36
3.	Image 3:- A person without and with mask (Fig 3.3)	11.46
4.	Average accuracy of all three images	1.1

The below proposed captured images of the video shows the pedestrian walking on a public street.In this work, the video is fixed at a specified angle to the street. The perspective view of the video frame is transformed into a top down view for more accurate estimation. The social distancing detection in a video frame and results shown depicts the red points as whose distance with another pedestrian is below the acceptable threshold and the green points represent the pedestrian who keep a safe distance from other pedestrians.

Fig 3.4:-Images of Social Distancing at public street (Courtesy:-PyImagesearch.com)[9]

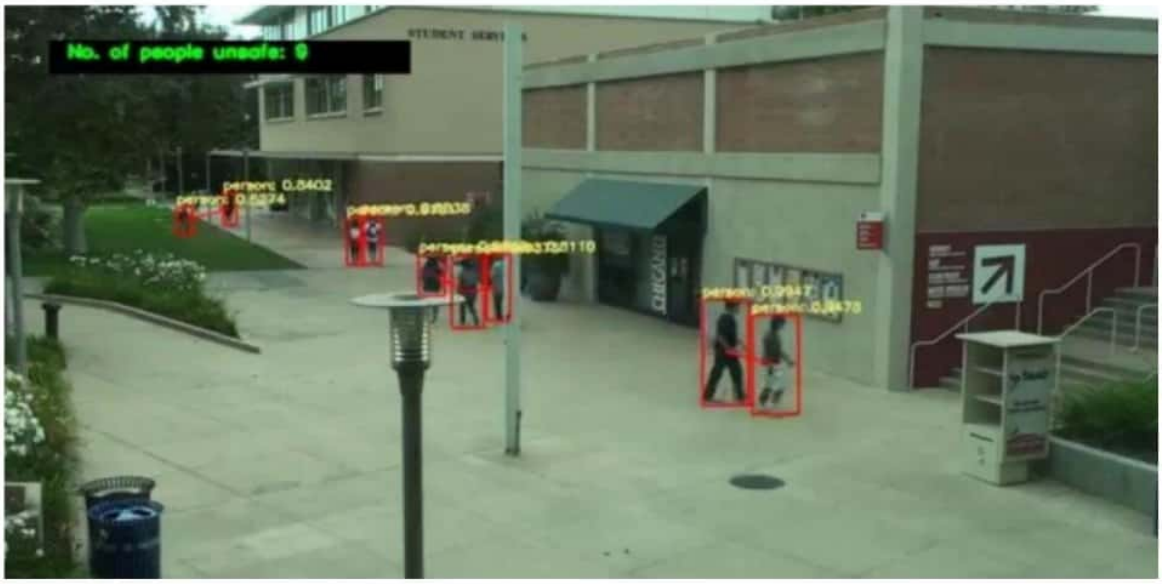

Fig 3.5:- Images of social distancing at public Street (Courtesy:PyImagesearch.com)[9]

CHAPTER 4.
CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The emerging trends and the availability of intelligent technologies make us develop new models that help to satisfy the needs of the emerging world. So, we have developed a Facemask detection and social distance monitoring which can possibly contribute to public healthcare. The model proposes an efficient real-time deep learningbased framework to automate the process of monitoring the social distancing and facemask detection via object detection and tracking approaches. The system is accurate. Thus, it makes it easier to deploy our model to embedded systems like Raspberry Pi, etc. There are many systems that can detect both facemask and social distancing in different procedures. We believe that this approach will enlarge the safety of the individuals during the pandemic. The technique deployed in this paper gives an accuracy score of 1.1.

4.2 FUTURE SCOPE

A cost efficient and reliable approach for facemask detection and social distance monitoring. In further we will be able to prevent stampede across the country. It can be used in the army to detect the enemy movements even in dark by some modification in the visuals of camera. It can be further modified into robot surveying for detection of booms and other things which can be watched from far distances. Future work of this study will be based on using much more efficient models that can predict masks worn improperly. The developed system faces difficulties in classifying faces that have covered their face with hands or improperly worn masks. A larger data set could be used as the larger data set helps with better training of the model. Datasets with a massive number of records are good for statistical analysis. The system can be further integrated to form an application that can be used to monitor public places.

REFERENCES

[1] Vinitha, Velantina,(2020), "Social distancing detection system with artificial intelligence using computer vision and deep learning",
International Research Journal of Engineering and Technology (IRJET) e-ISSN:2395-0056. Available from-:https://www.irjet.net/archives/V7/i8/IRJET-V7I8698.pdf

[2] C.Jagadeeswari, M.Uday Theja, "Performance evaluation of intelligent face mask detection system with various deep learning classifiers",
International Journal of Advanced Science and Technology, Vol.29, No.11s, pp.3074-30780, (2020). Available from-:https://www.ijert.org/review-on-literature-surveyofhumanrecognition-with-face-mask

[3] IEEE xplore, "Covid-19 Facemask detection using Tensorflow, Keras and Open CV". Available from:- https://ieeexplore.ieee.org/document/9342585

[4] Rucha visal, Atharva Theurkar, Bhairavi Shukla, "Monitoring social distancing for covid-19 using Open CV and deep learning", International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Available from:- https://www.irjet.net/archives/V7/i6/IRJET-V7I6422.pdf

[5] Raspberry Pi 4 Model B , Available from:- https://www.raspberrypi.com

[6] Pin configuration and description of components of Raspberry Pi Model 4 B ,Available from :- https://www.raspberrypi.com

[7] Raspberry Pi Camera ,Available from :- https://www.raspberrypi.com

[8] Representation of people wearing mask and not wearing mask, Available from :- https://www.pypi.org

[9] Images of Social Distancing at Public street,Available from:- https://www.pyimagesearch.com

APPENDIX -A

1 RASPBERRY Pi 4 MODEL B:-

1.1 FEATURES

- Quad core 64-bit ARM-Cortex A72 running at 1.5GHz
- 1, 2 and 4 Gigabyte LPDDR4 RAM options
- H.265 (HEVC) hardware decode (up to 4Kp60)
- H.264 hardware decode (up to 1080p60)
- VideoCore VI 3D Graphics
- Supports dual HDMI display output up to 4Kp60

1.2 INTERFACES

- 802.11 b/g/n/ac Wireless LAN
- Bluetooth 5.0 with BLE
- 1x SD Card
- 2x USB2 ports
- 2x USB3 ports
- 1x Gigabit Ethernet port (supports PoE with add-on PoE HAT)
- 1x Raspberry Pi camera port (2-lane MIPI CSI)
- 1x Raspberry Pi display port (2-lane MIPI DSI)

1.3 GPIO INTERFACE

The Pi4B makes 28 BCM2711 GPIOs available via a standard Raspberry Pi 40-pin header. This header is backwards compatible with all previous Raspberry Pi boards with a 40-way header.

1.4 GPIO ALTERNATE FUNCTIONS

Table 1.1 :- GPIO Alternate Functions

GPIO	Default Pull	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5
0	High	SDA0	SA5	PCLK	SPI3_CE0_N	TXD2	SDA6
1	High	SCL0	SA4	DE	SPI3_MISO	RXD2	SCL6
2	High	SDA1	SA3	LCD_VSYNC	SPI3_MOSI	CTS2	SDA3
3	High	SCL1	SA2	LCD_HSYNC	SPI3_SCLK	RTS2	SCL3
4	High	GPCLK0	SA1	DPI_D0	SPI4_CE0_N	TXD3	SDA3
5	High	GPCLK1	SA0	DPI_D1	SPI4_MISO	RXD3	SCL3
6	High	GPCLK2	SOE_N	DPI_D2	SPI4_MOSI	CTS3	SDA4
7	High	SPI0_CE1_N	SWE_N	DPI_D3	SPI4_SCLK	RTS3	SCL4
8	High	SPI0_CE0_N	SD0	DPI_D4	-	TXD4	SDA4
9	Low	SPI0_MISO	SD1	DPI_D5	-	RXD4	SCL4
10	Low	SPI0_MOSI	SD2	DPI_D6	-	CTS4	SDA5
11	Low	SPI0_SCLK	SD3	DPI_D7	-	RTS4	SCL5
12	Low	PWM0	SD4	DPI_D8	SPI5_CE0_N	TXD5	SDA5
13	Low	PWM1	SD5	DPI_D9	SPI5_MISO	RXD5	SCL5
14	Low	TXD0	SD6	DPI_D10	SPI5_MOSI	CTS5	TXD1
15	Low	RXD0	SD7	DPI_D11	SPI5_SCLK	RTS5	RXD1
16	Low	FL0	SD8	DPI_D12	CTS0	SPI1_CE2_N	CTS1
17	Low	FL1	SD9	DPI_D13	RTS0	SPI1_CE1_N	RTS1
18	Low	PCM_CLK	SD10	DPI_D14	SPI6_CE0_N	SPI1_CE0_N	PWM0
19	Low	PCM_FS	SD11	DPI_D15	SPI6_MISO	SPI1_MISO	PWM1
20	Low	PCM_DIN	SD12	DPI_D16	SPI6_MOSI	SPI1_MOSI	GPCLK0
21	Low	PCM_DOUT	SD13	DPI_D17	SPI6_SCLK	SPI1_SCLK	GPCLK1
22	Low	SD0_CLK	SD14	DPI_D18	SD1_CLK	ARM_TRST	SDA6
23	Low	SD0_CMD	SD15	DPI_D19	SD1_CMD	ARM_RTCK	SCL6
24	Low	SD0_DAT0	SD16	DPI_D20	SD1_DAT0	ARM_TDO	SPI3_CE1_N
25	Low	SD0_DAT1	SD17	DPI_D21	SD1_DAT1	ARM_TCK	SPI4_CE1_N
26	Low	SD0_DAT2	TE0	DPI_D22	SD1_DAT2	ARM_TDI	SPI5_CE1_N
27	Low	SD0_DAT3	TE1	DPI_D23	SD1_DAT3	ARM_TMS	SPI6_CE1_N

2. RASPBERRY PI CAMERA:-

Hardware Specifications

Table 2.1:- Hardware specifications of Raspberry Pi Camera

	Camera Module v1	Camera Module v2	HQ Camera
Net price	$25	$25	$50
Size	Around 25 × 24 × 9 mm		38 x 38 x 18.4mm (excluding lens)
Weight	3g	3g	
Still resolution	5 Megapixels	8 Megapixels	12.3 Megapixels
Video modes	1080p30, 720p60 and 640 × 480p60/90	1080p30, 720p60 and 640 × 480p60/90	1080p30, 720p60 and 640 × 480p60/90
Linux integration	V4L2 driver available	V4L2 driver available	V4L2 driver available
C programming API	OpenMAX IL and others available	OpenMAX IL and others available	
Sensor	OmniVision OV5647	Sony IMX219	Sony IMX477
Sensor resolution	2592 × 1944 pixels	3280 × 2464 pixels	4056 x 3040 pixels
Sensor image area	3.76 × 2.74 mm	3.68 x 2.76 mm (4.6 mm diagonal)	6.287mm x 4.712 mm (7.9mm diagonal)
Pixel size	1.4 µm × 1.4 µm	1.12 µm x 1.12 µm	1.55 µm x 1.55 µm
Optical size	1/4"	1/4"	
Full-frame SLR lens equivalent	35 mm		
S/N ratio	36 dB		
Dynamic range	67 dB @ 8x gain		
Sensitivity	680 mV/lux-sec		

Dark current	16 mV/sec @ 60 C		
Well capacity	4.3 Ke-		
Depth of field	approx. 1 m to infinity	adjustable with supplied tool	N/A
Focal length	3.60 mm +/- 0.01	3.04 mm	Depends on lens
Horizontal field of view	53.50 +/- 0.13 degrees	62.2 degrees	Depends on lens
Vertical field of view	41.41 +/- 0.11 degrees	48.8 degrees	Depends on lens
Focal ratio (F-Stop)	2.9	2.0	Depends on lens

Software features

Table 2.2:- Software features of Raspberry Pi Camera

Libcamera stack

Picture formats	JPEG, JPEG + DNG (raw), BMP, PNG, YUV420, RGB888
Video formats	raw h.264 (accelerated), MJPEG
Post-processing	User-definable image effects, customisable DRC and HDR, motion detection, OpenCV integration, TensorFlowLite integration
Exposure modes	normal, short, long, fixed fps, customisable
Metering modes	centre-weighted, average, spot, customisable
Automatic white balance modes	off, auto, incandescent, tungsten, fluorescent, indoor, daylight, cloudy, customisable
Triggers	Keypress, UNIX signal, timeout
Extra modes	timelapse, circular buffer, motion detection, segmented video, many features through flexible post-processing

Legacy stack

Picture formats	JPEG (accelerated), JPEG + RAW, GIF, BMP, PNG, YUV420, RGB888
Video formats	raw h.264 (accelerated)
Effects	negative, solarise, posterize, whiteboard, blackboard, sketch, denoise, emboss, oilpaint, hatch, gpen, pastel, watercolour, film, blur, saturation
Exposure modes	auto, night, nightpreview, backlight, spotlight, sports, snow, beach, verylong, fixedfps, antishake, fireworks
Metering modes	average, spot, backlit, matrix
Automatic white balance modes	off, auto, sun, cloud, shade, tungsten, fluorescent, incandescent, flash, horizon
Triggers	Keypress, UNIX signal, timeout
Extra modes	demo, burst/timelapse, circular buffer, video with motion vectors, segmented video, live preview on 3D models

APPENDIX-B

Source Code

1. Face Mask Detection

```
#include<iostream>

#include<string>

#include<vector>

#include"opencv2/core.hpp"

#include"opencv2/imgcodecs.hpp"

#include"opencv2/imgproc.hpp"

#include<opencv2/highgui.hpp>

usingnamespacestd;

int main(intargc,char**argv)

{

    floatf;

    floatFPS[16];

    inti,Fcnt=0;

    cv::Matframe;

    intclassify_w=128;

    intclassify_h=128;

    floatscale_factor=1.f/256;
    intFaceImgSz=classify_w*classify_h;

    //Maskdetection(secondphase,whenthefacesarelocated)
```

```
MobileConfigMconfig;

std::shared_ptr<PaddlePredictor>Mpredictor;

//sometiming

chrono::steady_clock::time_pointTbegin,Tend;

for(i=0;i<16;i++)FPS[i]=0.0;

//loadSSDfacedetectionmodelandgetpredictor

UltraFaceultraface("ABC-230.bin","ABC-890.param",640,640,2,0.7);//configmodel input

//loadmaskdetectionmodel

Mconfig.set_model_from_file("mask_detector_opt2.nb");
Mpredictor=CreatePaddlePredictor<MobileConfig>(Mconfig);
std::cout<<"Loadclassificationmodelsucceed."<<std::endl;

//GetInputTensor

std::unique_ptr<Tensor>input_tensor1(std::move(Mpredictor->GetInput(0)));

input_tensor1->Resize({1,3,classify_h,classify_w});

//GetOutputTensor

std::unique_ptr<constTensor>output_tensor1(std::move(Mpredictor->GetOutput(0)));

cv::VideoCapturecap(0);

if(!cap.isOpened()){
cerr<<"ERROR:Unabletoopenthecamera"<<endl;

return0;

}

cout<<"Startgrabbing,pressESConLivewindowtoterminate"<<endl;

while(1){

cap>>frame;
```

```
if(frame.empty()){
 cerr<<"ERROR:Unabletograbfromthecamera"<<endl;
break;
}
Tbegin=chrono::steady_clock::now();
ncnn::Matinmat=ncnn::Mat::from_pixels(frame.data,ncnn::Mat::PIXEL_BGR2RGB,
frame.cols,frame.rows);
//getthefaces
std::vector<FaceInfo>face_info;
ultraface.detect(inmat,face_info);
auto*input_data=input_tensor1->mutable_data<float>();
for(longunsignedinti=0;i<face_info.size();i++){
autoface=face_info[i];
//enlarge10%
floatw=(face.x2-face.x1)/20.0;
floath=(face.y2-face.y1)/20.0;
cv::Pointpt1(std::max(face.x1-w,float(0.0)),std::max(face.y1-h,float(0.0)));
 cv::Point
pt2(std::min(face.x2+w,float(frame.cols)),std::min(face.y2+h,float(frame.rows)));
//RecClip is completely inside the frame
cv::Rect  RecClip(pt1,pt2);
 cv::Mat  resized_img;
 cv::Mat  imgf;
if(RecClip.width>0&&RecClip.height>0){
```

```
//roihassizeRecClip
cv::Matroi=frame(RecClip);
//resized_imghassize128x128(uchar)
cv::resize(roi,resized_img,cv::Size(classify_w,classify_h),0.f,0.f,
cv::INTER_CUBIC);
//imgf has size128x128(floatinrange0.0-+1.0)
resized_img.convertTo(imgf,CV_32FC3,scale_factor);
//input tensor has size128x128(floatinrange-0.5-+0.5)
//fill tensor with mean and scale and trans layout: nhwc->nchw,neon
speedup
//offset_nchw(n,c,h,w)=n*CHW+c*HW+h*W+w
//offset_nhwc(n,c,h,w)=n*HWC+h*WC+w*C+c
constfloat*dimg=reinterpret_cast<constfloat*>(imgf.data);
float*dout_c0=input_data;
float*dout_c1=input_data+FaceImgSz;
float*dout_c2=input_data+FaceImgSz*2;
for(inti=0;i<FaceImgSz;i++){
    *(dout_c0++)=(*(dimg++)-0.5);
    *(dout_c1++)=(*(dimg++)-0.5);
    *(dout_c2++)=(*(dimg++)-0.5);
}
//ClassificationModelRun Mpredictor->Run();
auto*outptr=output_tensor1->data<float>();
floatprob=outptr[1];
```

```
//DrawDetectionandClassificationResults
boolflag_mask=prob>0.5f;
 cv::Scalarroi_color;
if(flag_mask)roi_color=cv::Scalar(0,255,0);
 else
 roi_color=cv::Scalar(0,0,255);
//Drawroiobject
cv::rectangle(frame,RecClip,roi_color,2);
}
}
Tend=chrono::steady_clock::now();
//calculateframerate
f=chrono::duration_cast<chrono::milliseconds>(Tend-Tbegin).count();
if(f>0.0)FPS[((Fcnt++)&0x0F)]=1000.0/f;
for(f=0.0,i=0;i<16;i++){f+=FPS[i];}
cv::putText(frame,
cv::format("FPS %0.2f",f/16),
cv::Point(10,20),cv::FONT_HERSHEY_SIMPLEX,0.6,
cv::Scalar(0,0,255));
//showoutput
cv::imshow("RPi64OS-1,95GHz-2MbRAM",frame);
charesc=cv::waitKey(5);
if(esc==27)break;
 }
```

```
cout<<"Closingthecamera"<<endl;
cv::destroyAllWindows(); cout<<"Bye!"<<endl;
return0;
}
```

2. Social Distancing

```
Import numpy as np
Import imutils
Import time
Import cv2
Import os
Import math

From itertools importchain
From constants import*

LABELS = open(YOLOV3_LABELS_PATH).read().strip().split('\n')

np.random.seed(42)
COLORS=np.random.randint(0,255,size=(len(LABELS),3),dtype='uint8')

print('LoadingYOLOfromdisk...')

neural_net=cv2.dnn.readNetFromDarknet(YOLOV3_CFG_PATH,YOLOV3_WEIGHTS_PATH)
layer_names=neural_net.getLayerNames()

vs=cv2.VideoCapture(VIDEO_PATH)
writer=None
(W,H)=(None,None)
try:
        if(imutils.is_cv2()):
              prop=cv2.cv.CV_CAP_PROP_FRAME_COUNT
        else:
              prop=cv2.CAP_PROP_FRAME_COUNT
        total=int(vs.get(prop))
        print('Totalframesdetectedare:',total)
  exceptExceptionase:
        print(e)
        total=-1

whileTrue:
               (grabbed,frame)=vs.read()

ifnotgrabbed:
     break
```

```
if W is None or H is None:
        H,W = (frame.shape[0],frame.shape[1])

blob=cv2.dnn.blobFromImage(frame,1/255.0,(416,416),swapRB=True,crop=False)
neural_net.setInput(blob)

start_time = time.time()
layer_outputs = neural_net.forward(layer_names)
end_time = time.time()

boxes=[]
confidences=[]
classIDs=[]
lines=[]
box_centers=[]

for output in layer_outputs:
       for detection in output

scores = detection[5:]
classID = np.argmax(scores)
confidence = scores[classID]

if confidence > 0.5 and classID == 0:
        box = detection[0:4]*np.array([W,H,W,H])
        (centerX,centerY,width,height)=box.astype('int')

              x=int(centerX-(width/2))
              y=int(centerY-(height/2))

              box_centers = [centerX,centerY]

              boxes.append([x,y,int(width),int(height)])
              confidences.append(float(confidence))
              classIDs.append(classID)

idxs = cv2.dnn.NMSBoxes(boxes,confidences,0.5,0.3)

iflen(idxs)>0:
      unsafe=[]
       count=0

       for I in idxs.flatten():

              (x,y)  =  (boxes[i][0],boxes[i][1])
              (w,h) = (boxes[i][2],boxes[i][3])
             centeriX = boxes[i][0]+(boxes[i][2]//2)
```

```
            centeriY = boxes[i][1]+(boxes[i][3]//2)

color = [int(c)forcinCOLORS[classIDs[i]]]
text = '{}:{:.4f}'.format(LABELS[classIDs[i]],confidences[i])

idxs_copy=list(idxs.flatten())
idxs_copy.remove(i)

for j in np.array(idxs_copy):
        centerjX = boxes[j][0]+(boxes[j][2]//2)
        centerjY = boxes[j][1]+(boxes[j][3]//2)

        distance=math.sqrt(math.pow(centerjX-centeriX,2)+math.pow(centerjY-centeriY,2))

        if distance <= SAFE_DISTANCE:
cv2.line(frame,(boxes[i][0]+(boxes[i][2]//2),boxes[i][1]+(boxes[i][3]//2)),(boxes[j][0]+(boxes[j][2]//2),
boxes[j][1]+(boxes[j][3]//2)),(0,0,255),2)unsafe.append([centerjX,centerjY])
unsafe.append([centeriX,centeriY])

       if centeriX in chain(*unsafe)andcenteriYinchain(*unsafe):
                count+=1
                cv2.rectangle(frame,(x,y),(x+w,y+h),(0,0,255),2)
       else:
                 cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),2)
         cv2.putText(frame,text,(x,y-5),cv2.FONT_HERSHEY_SIMPLEX,0.5,color,2)
         cv2.rectangle(frame,(50,50),(450,90),(0,0,0),-1)
        cv2.putText(frame, 'No. of people unsafe: {}'.format(count),(70,70),
cv2.FONT_HERSHEY_SIMPLEX,0.7,(0,255,0),3)

    if writer is None:
           fourcc = cv2.VideoWriter_fourcc(*'MJPG')
          writer = cv2.VideoWriter(OUTPUT_PATH,fourcc,30,(frame.shape[1],frame.shape[0]),True)

     if total > 0:
           elap = (end_time-start_time)
          print('Singleframetook{:.4f}seconds'.format(elap))
           print('Estimatedtotaltimetofinish:{:.4f}seconds'.format(elap*total)

       writer.write(frame)

print('Cleaningup...')
writer.release()
vs.release()
```

BIODATA

1. Nallamala Pooja

- Email ID: npooja1999@gmail.com
- DOB: October 8,1999
- Resident: Yedapally,Telangana
- Mobile No.: 9121460788
- Bharati Vidyapeeth (Deemed to be University) College of Engineering, Pune

  B.Tech Electronics Engineering, 2017-2022
- 12th – Narayana Junior College,Hyderabad
- 10th – Vijaya Mary High School, Bodhan

2. Isha Mandawkar

- Email ID: ishamandawkar2017@gmail.com
- DOB: August 13, 1999
- Resident: Ghatkopar,Mumbai
- Mobile No.: 9082100959
- Bharati Vidyapeeth (Deemed to be University) College of Engineering, Pune

  B.Tech Electronics Engineering, 2018-2022
- 12th - Khar Education Society Junior College Of Science,Mumbai
- 10th – Fatima High School, Mumbai

3. Aditya Chakraborty

- Email ID: aditya.hdl1112@yahoo.com
- DOB: December 11, 1999
- Resident: Varanasi,Uttar Pradesh
- Mobile No.: 914084646767
- Bharati Vidyapeeth (Deemed to be University) College of Engineering, Pune

  B.Tech Electronics Engineering, 2018-2022
- 12th – Little Flower House,Kakarmatta,Varanasi
- 10th – St John's DLW School