**PROJECT DOCUMENTATION**

on

**Team Members:**

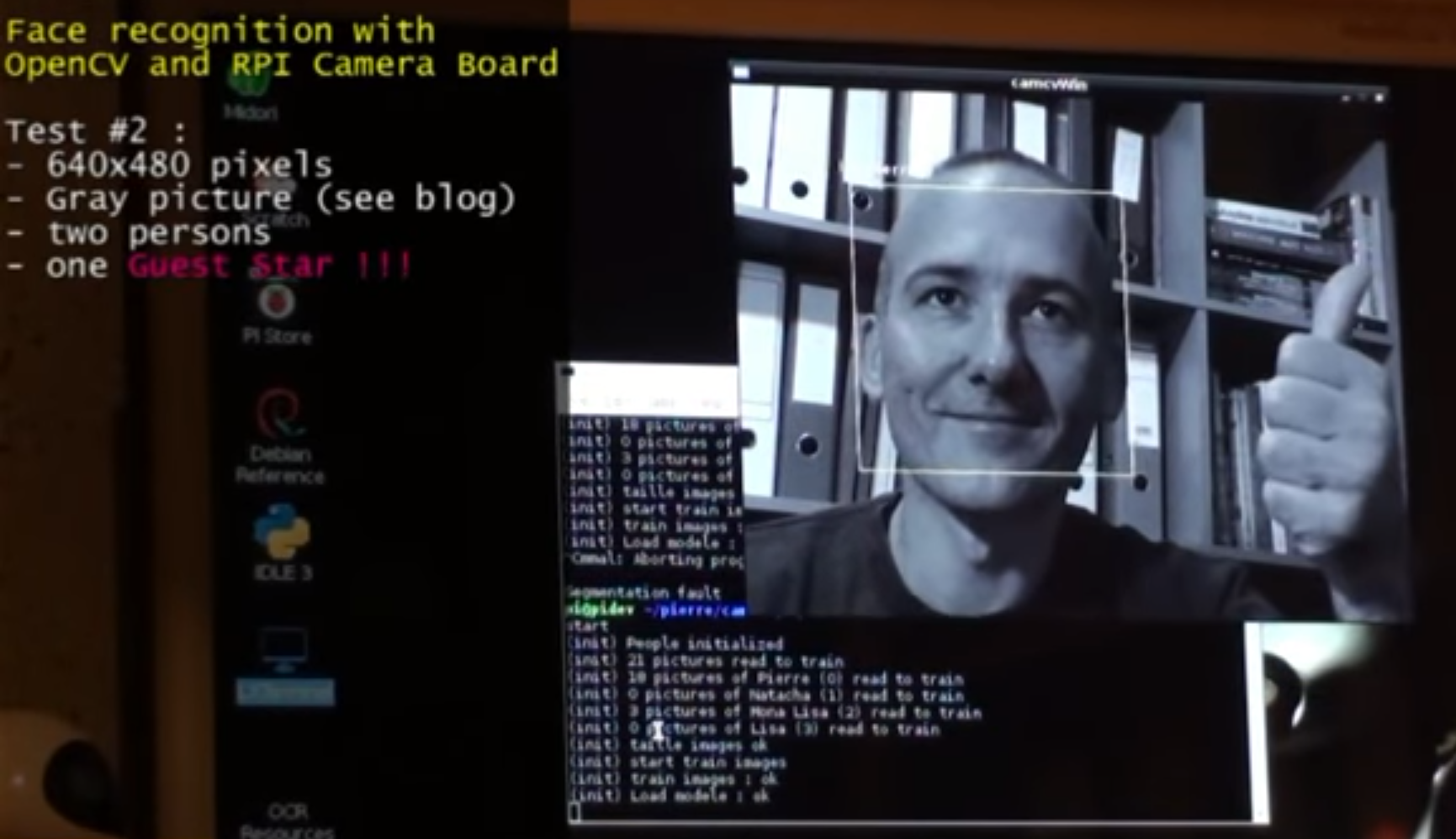
*Vanka Hari Janardhan (Team Lead)*

*S Vinay*

*Thota Manikanta Naga Hanuman*

*DP Narendra Sai*

**Intelligent Access Control Systems for Safety Critical Areas in Industries**



**stems for Safety Critical Areas in Industries**

**TABLE OF CONTENTS**

**S No Title Page No**

1. Abstract of Project 2

with Problem Statement

1. Project Working Process 3 – 5

With block diagram

(3) Components with their 6 – 11

Specifications and Principles

(4) Installations 12-15

(5) Coding Part 16 - 21

**ABSTRACT**

**Problem Statement**: Intelligent Access Control Systems for Safety Critical Areas in Industries

In some industries it is necessary for the workers to wear safety helmets and shoes while working. So, to check weather workers are taking safety precautions or not we are proposing this system.

We can train our classifier to identify helmet and safety shoes with Clarify API. There will be video streaming near the entry of the industries where we can first detect the face of a person and if any person is present then we can capture the image of that moment and send it to Clarifai API to detect whether the person is wearing helmet or shoe.

 If the person is wearing shoe and helmet, we can give him access by opening the door. If he is not wearing then we can restrict his access by not opening the door. We can even warn him through voice commands to take the safety precautions.

**Project Highlights**:

* Interfacing Camera and audio device with Raspberry Pi.
* Face detection using Open Cv.
* Visual recognition using Clarifai API.
* Text to speech conversion using Raspberry pi.

**Hardware**:

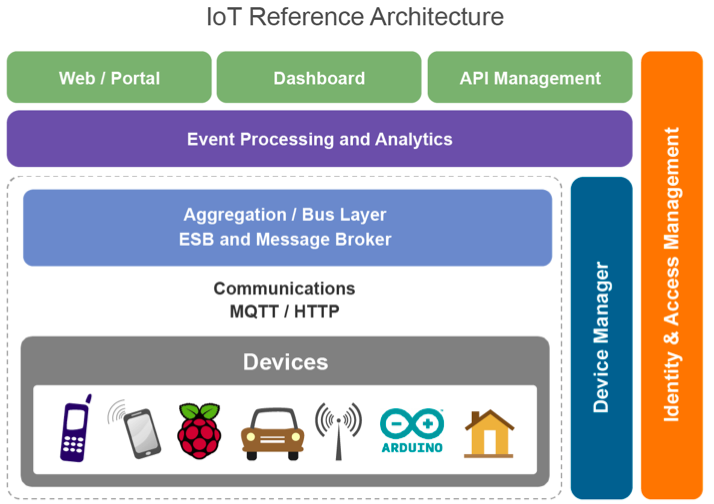
* Raspberry Pi with 16GB Micro SD Card
* Speakers
* Camera
* Servo Motor

**Softwares**:

* Python IDE
* IBM Visual Recognition Engine

**PROJECT WORKING PROCESS**

Let us see the architecture of IOT.



The following steps need to be performed in this project:

* 1. Connect the Raspberry Pi to a system using VNC Viewer
  2. Install Raspbian Operating System on the Raspberry Pi 3 Model B using a Micro SD Card
  3. Install Python IDLE version 2 or 3 in the Terminal
  4. Install OpenCV Software, Clarifai API and Python Text to Speech module on the device using Terminal
  5. Connect USB Camera to the Raspberry Pi and install the camera modules in the device
  6. Classify the images
     + Workers with only Helmet
     + Workers with only Boots

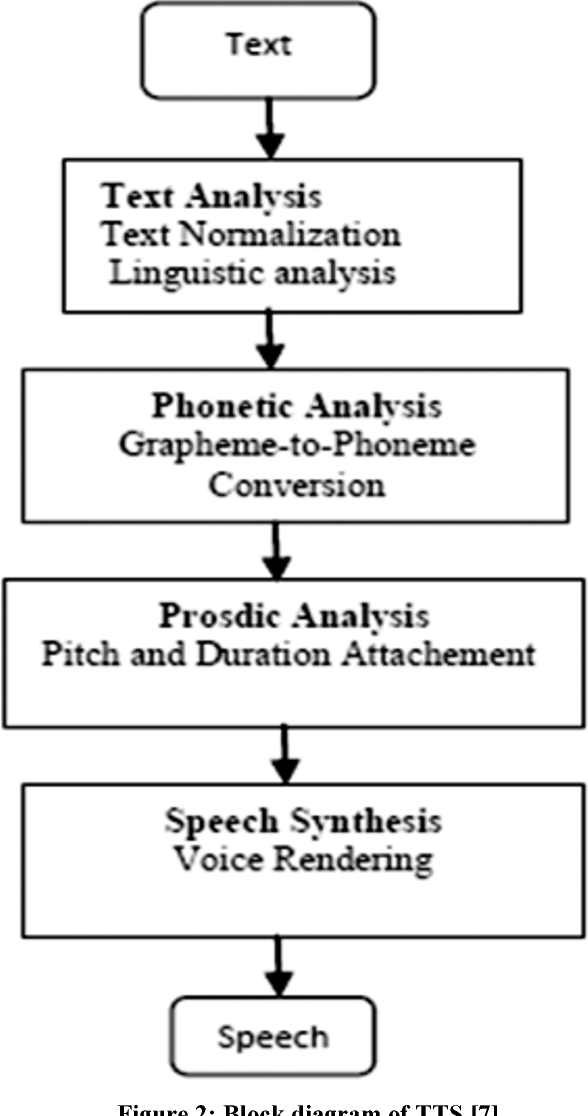
Using the Clarifai API and then train the model.

Verify that JSON code is generated for above activity.

* 1. Coding Part
     + Write code for the Camera to start video recording and clicking pictures and saving them in specified directory.
     + If the camera is not able to read the person’s image, then an audio must be played via speaker asking them to stand in position for face detection.
     + These images are verified with those trained models in Clarifai API and an accuracy score is generated.
     + Based on the accuracy score, the program will decide whether the worker is to be allowed access.
     + It will convert text to speech and give output via a speaker, i.e. if a person is found with both helmet and boots then he is given access. Else, he will receive a warning from the Speaker saying that access is restricted.
     + A Servo Motor must begin rotating when access is allowed. Else, it will not rotate.
     + You can check the JSON code generated in Python IDLE using the online JSON Viewer for more clarity on the image and its accuracy.



**Text To Speech Conversion**



**COMPONENTS USED IN THIS PROJECT**

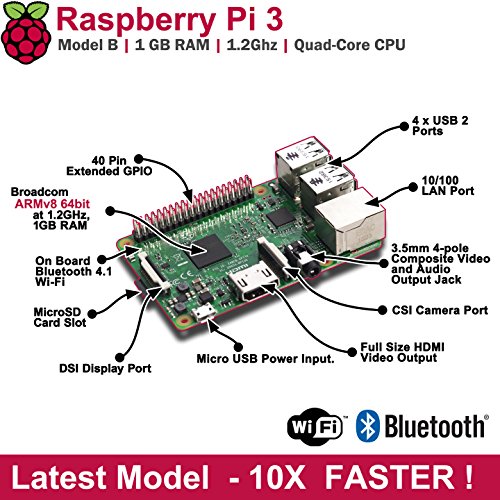
The components used in this project are classified into two types:

1. Hardware Components
2. Software Components

Let us see the **Hardware Components** .

1. **Raspberry Pi 3 Model B**

**Raspberry Pi®** is an **ARM** based credit card sized **SBC**(Single Board Computer) created by [Raspberry Pi Foundation](http://www.raspberrypi.org/). Raspberry Pi runs Debian based **GNU/Linux** operating system [Raspbian](https://www.raspberrypi.org/downloads/raspbian/) and ports of many other OSes exist for this SBC.

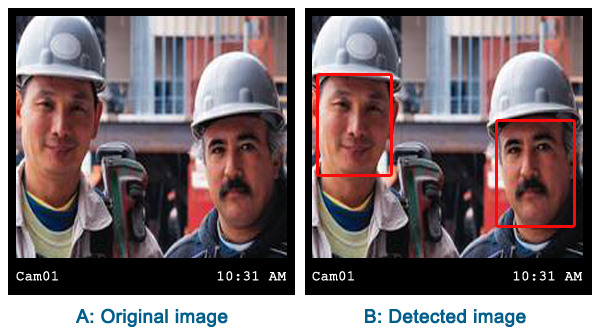




**SoC:** Broadcom BCM2837  
**CPU:** 4× ARM Cortex-A53, 1.2GHz  
**GPU:** Broadcom VideoCore IV  
**RAM:** 1GB LPDDR2 (900 MHz)  
**Networking:** 10/100 Ethernet, 2.4GHz 802.11n wireless  
**Bluetooth:** Bluetooth 4.1 Classic, Bluetooth Low Energy  
**Storage:** microSD  
**GPIO:** 40-pin header, populated  
**Ports:** HDMI, 3.5mm analogue audio-video jack, 4× USB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)

1. **USB Camera**





* HD 720p video calling and HD video recording, 2.4 GHz Intel Core2 Duo, 2 GB RAM,200 MB hard drive space
* Video capture: Up to 1280 x 720 pixels, Logitech Fluid Crystal
* Crisp 3 MP photos Technology, Hi-Speed USB 2.0
* Compatible with: Windows 10 or later, Windows 8, Windows 7, Works in USB Video Device Class (UVC) mode with supported video-calling clients: macOS 10.10 or later, Chrome OS, Android v 5.0 or above

1. **Micro SD Card**

Capacity: 16/32/64/128GB \*ANY

Capture more photos

Share more of your favourite photos and videos with your friends and family

Organize and transfer your photos, videos and music with included SanDisk Media Manager software

Record more video on the go, even in full HD

Play more of your favourite music

1. **Speakers**



Achieve impressive sound from this JBL Link voice-activated speaker. Bluetooth technology ensures solid connectivity to your smartphone to play your favourite music, and the built-in Google Assistant provides hands-free operation on a variety of tasks. This JBL Link voice-activated speaker works with both iOS and Android devices for optimal convenience.

1. **Servo Motor**



A **servo motor** is a closed-loop system that uses position feedback to control its motion and final position. In industrial type **servo motors** the position feedback sensor is usually a high precision encoder, while in the smaller RC or hobby **servos** the position sensor is usually a simple potentiometer.

Now let us see the **software components** we need.

**(1)** **NOOBS (New Out of Box Software)**

**An easy Operating System installer for the Raspberry Pi**

NOOBS is designed to make it easy to select and install operating systems for the Raspberry Pi without having to worry about manually imaging your SD card.

**About**

On first boot NOOBS will repartition your SD card and allow you to select which OSes you want to install from a list. This OS list is automatically generated from both locally available OSes (i.e. those contained in the /os directory on disk) or those available from our remote repository (network connection required).

Only the latest version of each OS will ever be displayed meaning that you can be sure that you have installed the most up-to-date release of your selected OS.

On any subsequent boot you can then press the SHIFT key to enter the NOOBS interface and easily reinstall your choice of OSes.

The NOOBS interface provides the following functionality:

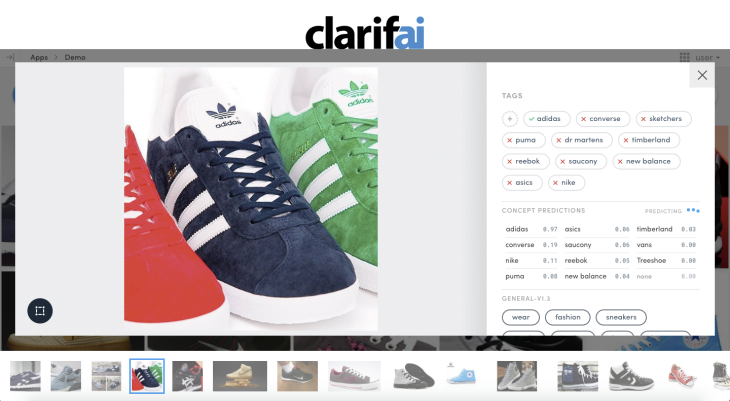
* **Install**: Installs the selected OSes onto your SD card. Changing this selection erases all OSes currently installed.
* **Edit Config**: Opens a text editor allowing the cmdline and config for the selected installed OS to be edited.
* **Online Help**: [Networking Required] Open a browser that displays the Raspberry Pi Help page ( <http://www.raspberrypi.org/help/> ), allowing people to quickly access help and troubleshooting.
* **Exit**: Quits NOOBS and reboots the Pi into the OS boot menu.
* **Language Selection**: Allows you to select the language to be displayed.
* **Keyboard Layout Selection**: Allows you to select the keyboard layout to be used.
* **Display Mode Selection**: By default, NOOBS will output over HDMI at your display's preferred resolution, even if no HDMI display is connected. If you do not see any output on your HDMI display or are using the composite output, press 1, 2, 3 or 4 on your keyboard to select HDMI preferred mode, HDMI safe mode, composite PAL mode or composite NTSC mode respectively.

**(2)** **Python IDLE**

* IDLE is integrated development environment (IDE) for editing and running Python 2.7.3
* The IDLE GUI (graphical user interface) is automatically installed with the Python interpreter. IDLE was designed specifically for use with Python.
* IDLE has a number of features to help you develop your Python programs including powerful syntax highlighting.

**(3) Clarifai API**

Clarifai Inc. is an artificial intelligence (AI) company that specializes in computer vision and uses machine learning and deep neural networks to identify and analyze images and videos. The company offers its solution via API, mobile SDK, and on-premise solutions. Clarifai is headquartered in New York City with two satellite offices in San Francisco and Washington D.C. Clarifai API produces an accuracy score for each image and classifies them under the appropriate class.

**INSTALLATIONS**

**First, we have to install Raspbian OS in the Micro SD Card present in the Raspberry Pi 3 Model B device.**

## Step 1: Download the Required Software and Files

## Step 2: Get the SD Card and the Card Reader

## Step 3: Check the Drive in Which the SD Card Is Mounted

## Step 4: Format the SD Card

## Step 5: Write the OS on the SD Card

## Step 6: Eject the SD Card

**Download VNC Server and VNC Viewer and Install them**

Steps to perform to connect Raspberry Pi to VNC Viewer

1. connect ethernet of Raspberry Pi to ethernet of laptop

2. connect micro USB of raspberry pi to USB of laptop

3. go to network and sharing centre

4. go to details in Wi-Fi

5. go to properties and then sharing and click on allow option and then select ethernet

6. ipv4 address will be present

7. use the following IP address

- 192.168.137. \*

- check subnet mask values

- in CMD

- ping 192.168.137.255

- arp -a

->copy non 255 address in VNC viewer

username - pi

password - raspberry

if not connecting

* Then in CMD **ping raspberrypi.mshome.net**

take that IP address and paste in VNC viewer and set up the OS

preferences -> raspberry pi configuration -> check whether VNC is enabled

**Next, type the following commands in the terminal to install camera modules and change to enable camera in configurations.**

sudo apt-get install fswebcam

fswebcam -S 20 123.jpg

**Next, we have to install OpenCV software.** Follow below commands

**sudo apt-get update && sudo apt-get upgrade**

**sudo apt-get install build-essential cmake pkg-config**

**sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev**

**$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev**

**$ sudo apt-get install libxvidcore-dev libx264-dev**

**$ sudo apt-get install libgtk2.0-dev libgtk-3-dev**

**$ sudo apt-get install libatlas-base-dev gfortran**

**$ sudo apt-get install python2.7-dev python3-dev**

**$ cd ~**

**$ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.3.0.zip**

**$ unzip opencv.zip**

**$ wget -O opencv\_contrib.zip https://github.com/Itseez/opencv\_contrib/archive/3.3.0.zip**

**$ unzip opencv\_contrib.zip**

**$ wget https://bootstrap.pypa.io/get-pip.py**

**$ sudo python3 get-pip.py**

**$ pip3 install numpy**

**$ cd ~/opencv-3.3.0/**

**$ mkdir build**

**$ cd build**

**$ cmake -D CMAKE\_BUILD\_TYPE=RELEASE \**

**-D CMAKE\_INSTALL\_PREFIX=/usr/local \**

**-D INSTALL\_PYTHON\_EXAMPLES=ON \**

**-D OPENCV\_EXTRA\_MODULES\_PATH=~/opencv\_contrib-3.3.0/modules \**

**-D BUILD\_EXAMPLES=ON ..**

**make -j4**

**$ sudo make install**

**$ sudo ldconfig**

**Checking whether installed or not --- using below information**

Open up a new terminal,

>>> import cv2

>>> cv2.\_\_version\_\_

'3.3.0'

>>>

**Next, we have to install Clarifai API and Python Text to Speech on the Raspberry Pi**

* sudo pip install clarifai
* sudo pip install pyttsx
* sudo apt-get install espeak

**Connections for Raspberry Pi to Servo Motor**

**Servo Motor Raspberry Pi**

Brown Ground (Pin No 6)

Yellow GPIO 7 (Pin No 26)

Red 3V3 (Pin No 1)

**CODING PART**

import numpy as np

import cv2

import time

from datetime import datetime

import pyttsx

from clarifai.rest import ClarifaiApp

from clarifai.rest import Image as ClImage

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(7, GPIO.OUT)

p = GPIO.PWM(7, 50)

p.start(7.5)

app1= ClarifaiApp(api\_key='6cd50e41c156483cbfde2771293ffe10')

#app1= ClarifaiApp(api\_key='a75a5d1ddf924167a62f700e800b31be')

model1 = app1.models.get('apparel')

face\_cascade = cv2.CascadeClassifier('haar-face.xml')

def voice():

engine = pyttsx.init()

engine.say(Voice)

engine.runAndWait()

time.sleep(2)

cap = cv2.VideoCapture(0)

print 'camera is initialized'

while True:

Voice='please stand in the position...we are capturing your image'

voice()

time.sleep(5)

ret, img = cap.read()

if ret:

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(gray, 1.3, 5)

for (x,y,w,h) in faces:

cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)

roi\_gray = gray[y:y+h, x:x+w]

roi\_color = img[y:y+h, x:x+w]

picname = datetime.now().strftime("%y-%m-%d-%H-%M")

picname = picname+'.jpg'

cv2.imwrite(picname,img)

print "Saving Photo"

pic='/home/pi/Downloads/10.jpg'

print pic

pic1='/home/pi/Downloads/'+picname

print pic1

image = ClImage(file\_obj=open(pic1, 'rb'))

response=model1.predict([image])

data1 = response['outputs'][0]['data']['concepts']

print data1

for row in data1:

if row['name'] == 'helmet':

if row['value']>= 1.590712e-08:

x=1

else:

print 'please wear the helmet'

time.sleep(2)

t=model1.predict([image])

data2 = t['outputs'][0]['data']['concepts']

print data2

d="Men"+"'"+"s"+" "+"Sandals"

e="Men"+"'"+"s"+" "+"Boots"

for row in data2:

if row['name'] ==d:

if row['value']>0.00001:

y=1

elif row['name'] ==e:

if row['name']>0.00001:

y=1

else:

z=0

if(x==1 and y==1):

print "you can enter inside"

p.ChangeDutyCycle(12.5) #180°

time.sleep(2)

p.ChangeDutyCycle(7.5)

Voice = 'you can enter inside'

voice()

else:

print "your entry is restricted"

Voice = 'please wear the shoes and helmet to enter'

voice()

time.sleep(10)

cv2.imshow('img',img)

time.sleep(0.1)

k = cv2.waitKey(30) & 0xff

if k == 27:

break

cap.release()

cv2.destroyAllWindows()