S.T. Yau High School Science Award (Asia)
2020

Research Report

The Team

Registration Number: Comp-034

Name of team member: Ms. Sana Mohammed
School: The International School Bangalore (TISB)
City, Country: Bangalore, India

Title of Research Report

Combating COVID-19: Digital Wearable Solution for Social Distancing using Artificial Intelligence

Date

31 August 2020

[Please write title of research report here]

[Please write names of authors here]

Abstract

Background: Coronavirus has taken the world by surprise with more than 19.6 million cases and more than 750k deaths across the world (as of August 08, 2020). It has brought the whole world to its knees with countries borders closed, businesses shut, schools closed, many people losing the jobs and IMF predicting the recession could be worse than 2008 financial crisis.

In absence of Vaccination for Covid-19, the only tool that is available to governing authorities are social distancing and contact tracing. In response to this, several countries across the world are in race to build a digital solution focusing on contact tracing to defeat/slow down the spread of the virus. However, none of the solutions addresses helping people in implementing the recommended social distance of 4 to 6 feet.

Approach: In this report I have explored number of techniques to effectively calculate social distance between the two people. And shown how to effectively calculate the social distance by combining the hardware, machine learning and deep learning technologies. Using the combined approach of hardware and software solutions I propose a digital wearable device as one of the feasible solutions that reminds people whenever the recommended social distance of 4 to 6 feet is violated.

Results: A digital wearable device prototype is built using the hardware of Raspberry PI, BBC Microbit, PI Camera and how to employ Statistical Machine Learning and Deep Learning technologies to calculate the social distance.

Conclusion: Finally, it was shown that the wearable device works, which non-invasively reminds the people when recommended social distance is violated. The challenges, lessons learnt, cost constraints and next steps are discussed.

Keywords: Covid-19, Corona virus, social distancing, machine learning, deep learning, convolution neural networks, Raspberry PI, BBC Microbit, Triangulation

Video Demo Link: A short, 2-3 minutes video demo of this project can be accessed at the link below.

https://youtu.be/SOPq5EN4Egg

Acknowledgement

I sincerely thank you for the Google's open source community for making the facial recognition videos, short tutorials available online free of charge. Without the support of open source community, I would not have able to complete this report.

I also thank you to the owner of the Py Image Search website, Mr Adrian Rosebrock, for making the extensive tutorials on computer vision using Open CV and Deep Learning. Without the tutorials that are available on this website this project would not have come to end.

I also thank you to the owner of the Machine Learning Mastery website, Mr. Jason Brownlee, for making the tutorials and detailed explanation [16] of deep learning neural networks available to general public. These tutorials and step-by-step explanations paved the path to read the code and understand what was going on with the code.

I also thank you ZAS Academy for IOT and Robotics (http://zasacademy.org/) which was run by mother, Dr. Roohi Banu. I have been attending the IOT club with my classmates since 2018 and over the last 2 years I learnt about:

- **Micro-controllers**: BBC Microbit, Arduino, Raspberry PI,
- **Sensors**: Ultrasonic, IR, Light, Gas, Temperature, Pressure, IMU, GPS, Keypad, Transmitter/Receivers, Bluetooth, Relays, LCD/OLED Displays etc.
- **Robots**: Motor, Motor Drivers, BBC Microbit driver Robot Car, Arduino driven Robot car
- **Programming:** Python & Flask programming on AWS Clouds

Attending the IOT club provided be the firm foundations required to tackle this S.T. Yau High School Science Award (Asia) 2020 challenge. I sincerely thank you, ZAS Academy (Dr. Roohi Banu) for sharing their experience in guiding me to make right decision on components selection that make the final product compact and appealing. And also, for giving an idea to use BBC Microbit for wireless communication between Raspberry PI and wearable device.

Commitments on Academic Honesty and Integrity

We hereby declare that we

1. are fully committed to the principle of honesty, integrity and fair play throughout the competition.
2. actually perform the research work ourselves and thus truly understand the content of the work.
3. observe the common standard of academic integrity adopted by most journals and degree theses.
4. have declared all the assistance and contribution we have received from any personnel, agency, institution, etc. for the research work.
5. undertake to avoid getting in touch with assessment panel members in a way that may lead to direct or indirect conflict of interest.
6. undertake to avoid any interaction with assessment panel members that would undermine the neutrality of the panel member and fairness of the assessment process.
7. observe all rules and regulations of the competition.
8. agree that the decision of YHSA(Asia) is final in all matters related to the competition.

We understand and agree that failure to honour the above commitments may lead to disqualification from the competition and/or removal of reward, if applicable; that any unethical deeds, if found, will be disclosed to the school principal of team member(s) and relevant parties if deemed necessary; and that the decision of YHSA(Asia) is final and no appeal will be accepted.

(Signatures of full team below)

Sana Mohammed

X _____

Name of team member: Ms. Sana Mohammed

THE INTERNATIONAL SCHOOL BANGALORE

X Sherwin Kuri

Name of supervising teacher: Mr. Sherwin Kuri

Noted and endorsed by

(signature) Caparve

Name of school principal: Dr. Caroline Pascoe

Table of Contents

Table of Contents

1 Introduction

Coronavirus has taken the world by surprise with more than 23 million cases and more than 830k deaths across the world (as of August 31st, 2020). It has brought the whole world to its knees with countries borders closed, businesses shut, schools closed, many people losing the jobs and IMF predicting the recession could be worse than 2008 financial crisis.

The WHO, the United Nations and the world leaders have called for more solidarity and cooperation in fighting not only the medical aspects but the economic fallout.

In absence of Vaccination for Covid-19, the only tools that are available to governing authorities to contain or slow down the spread of Coronavirus are: (a) **social distancing** and (b) **contact tracing**.

Several countries across the world are in a race to build a digital solution focusing on **contact tracing** to defeat/slow down the spread of the virus. However, none of the solutions addresses helping people in implementing the recommended social distance of 4 to 6 feet.

In this report I propose a digital wearable solution that employs hardware, machine learning and deep learning technologies to calculate the social distance between the device wearer and the observer. The wearable device, non-invasively reminds the wearer whenever a recommended social distancing rule of 4 to 6 feet is violated.

The structure of the paper is as follows:

- In section 2, I present brief literature survey of existing techniques the people have employed in maintaining the social distance.
- In section 3, I present high level working principle of wearable device along with the required hardware and software components
- Section 4 describes hardware, software options along with different techniques to measure the distance between the camera and the human face with image as an input.
- In this section the ultrasonic sensors, infrared light sensors, triangulation techniques using single and dual camera are explained. To keep the design simple, compact and less bulky instead of using additional sensors to measure the distance I decided to build my own approximate model which uses statistical modelling approach to calibrate the camera and calculate the distance between camera and human face.
- Section 5 describes the statistical modelling implementation details and compares and contrasts the accuracy results of number of models such as Linear Regression, Random Forest Model, Polynomial Regression and Generalised Additive Models (GAM)

- Section 6 describes very briefly what are neural network model and introduces the convolution neural network model used for this social distancing project
- Section 7 discusses the implementation challenges and section 8 presents the conclusions and next steps.

2 Literature Survey

The literature survey showed number of people, organizations have come-up with innovative solutions to maintain the social distancing. The solutions for social distancing can be categorised into two groups: (a) Solutions from individuals and (b) Solutions from Organizations. These solutions to the data that are published in newspaper articles, magazines and social media are listed below.

Solutions from Individual Innovations

- A floppy/straw hat with diameter up to 1m
- A hat with soft toy style projections (3 to 4 feet away) on either side
- A hard hat with vertical pole to which a laser light attached which projects a beam of circular light showing the safe distance of 2m.
- A mask with soft moustache that extend up to 3 to 4 feet on either side

Solutions from Organizations

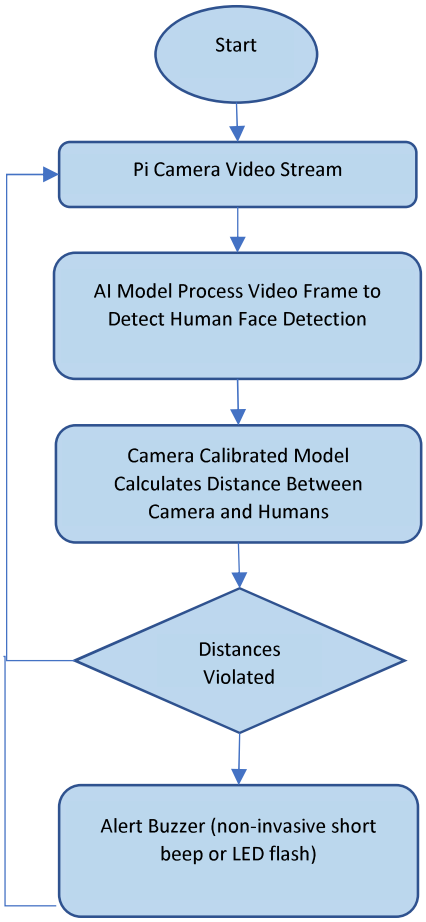
- A bracelet specifically programmed worn by all people returning to work in an organization. The bracelet alerts if the distance between two bracelets is less than 1.5m
- Other solutions in train/flights are rearrangement of alternate seats in reverse facing direction or leaving the middle seat completely empty.
- Posters and stickers at the busy places such as shop entrances, checkout areas to remind people of keeping the importance of recommend social distancing to prevent the spread of disease.
- LIDAR sensor continuously monitoring the distances at busy places like airport queues, train stations queues and displaying colour code of green/red depending on whether the distance rule is satisfied or not.
- A smart dome camera which are installed/attached to the roof. The camera has birds eye view of people in busier places and sends out messages over the speaker to remind people of importance of keeping the recommended distance apart.

Few of the above-mentioned solutions are not practical (masks/hats with projections), few other solutions work at specific places such as airports, super markets, stations etc. and few other solutions at specifically designed to work within the work places (bracelet). But there is no social distancing solution that works universally: crowded/non-crowded, outside/inside and homes/offices etc.

Addressing, the above issues I proposed a solution that combines wearable devices technology with Artificial Intelligence which works in all places (irrespective of whether it is busy/crowded places or few people). The solution architecture is shown below.

3 Working Principle and Hardware/Software Requirements

3.1 Working Principle



At a very high level the solution consists of the following:

(1) A camera is stitched to the women's waist belt or men's jacket/shirt breast pocket or hat.

(2) The camera is continuously shooting the video. The video frames are fed to Deep Learning model to detect the presence of human face in the video frame.

(3) If Human faces are present in the image, use the camera calibrated model to calculate the distance between camera and the human face based on the size (width and height) of the face in the image.

(4) If the distance is less than recommended social distance (which is approximately 6 feet or 2m), then a red LED light flashes or beeps a sound using buzzer.

(5) The LED lights strip stitched to women's fashion accessories like bangles or brooch or buzzer to men's shirt collar or breast pocket.

3.2 Implementation Requirements

The implementation requires the following:

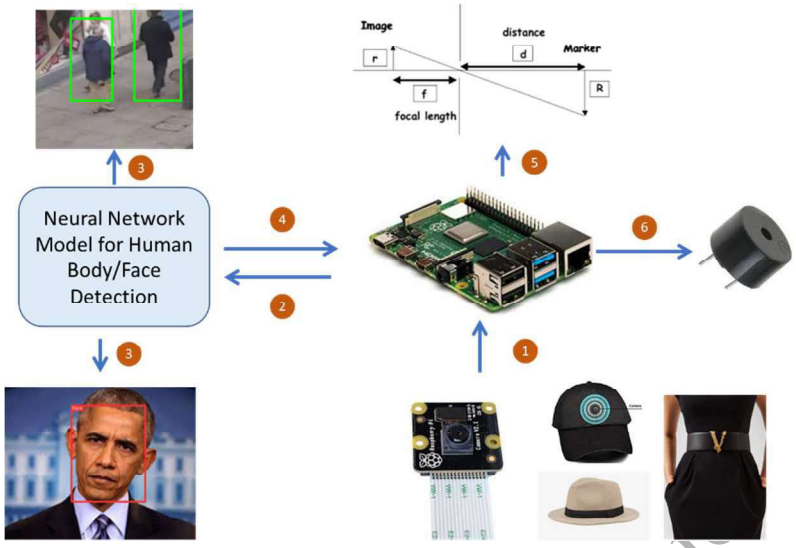
(1) Hardware: camera is required to shoot the video, an on-board computer to run the facial recognition model, LED light or buzzer to flash/beep when social distancing rules are violated.

(2) Software:

a. A deep learning model to find the human faces and programmatically flash the LED or beep a buzzer when social distancing rules are violated.

b. A model to find the distance between the camera and human faces detected in the captured video frame.

(3) Final Product Integration: Packaging all the software and hardware into something fashionable jewellery (brooch, bangle, hat or jacket).



3.3 Hardware

All the hardware required for this project is tabulated below along with its approximate cost.

Component	Cost (Indian Rupees)	Purpose
Raspberry PI 3B+	Rs. 3,500	Plays the role of a mini-computer for running the facial recognition and distance calculation model and also acts as a controller to flash the LED lights or beep.
PI Camera	Rs. 300	Camera for shooting the video
Batteries	Rs. 500	Power supply for the Raspberry PI
Few jumper cables, LED lights and a buzzer	Rs. 300	
Container	Rs. 100	A tight container for tightly packaging the hardware
Women's waist Belt, Brooch, Bangle or hat	Rs. 500	Wearable jewellery
BBC Microbit	Rs. 2500	For wireless communication between the hardware and wearable jewellery/device
Total	**Rs. 7000**	

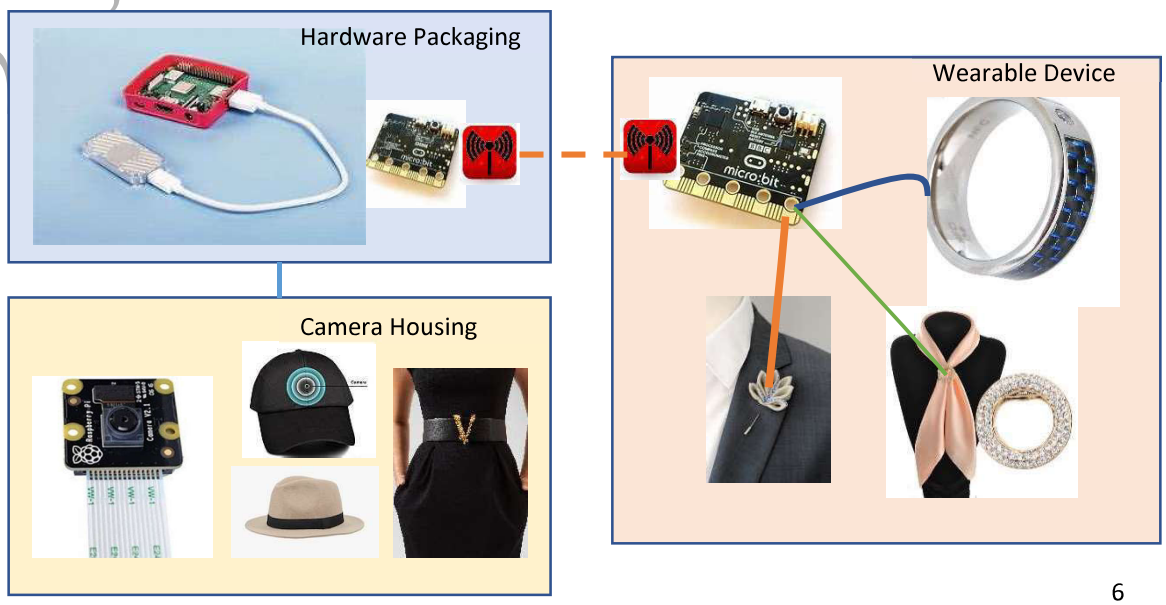
3.4 Software

Software Type	Programming Language	Comments
Distance Calculation Model	Python	The statistical machine learning model takes the width/height (in pixels) of the face in the image as input and predicts the distance from the camera.
Face Recognition Model	Python	I did not build this model, instead I used a pre-trained MobileNetV2 model which is available for free to explore from Google opensource community.
Data Visualizations	Python & R	Python & R languages are used for data exploration before building the statistical machine learning models.

3.5 Final Product

The overview of the finished final product is shown below. The final product has three parts: Hardware packaging, Camera Housing and Wearable Device.

- The hardware and camera boxing are physically connected with camera flat ribbon cable which is run behind the men/women clothing.
- The hardware and wearable devices are communicating wirelessly using the Radio Frequency Signals where one BBC Microbit is acting as signal transmitter and the other one as receiver.
- The raspberry PI controller when it detects violation of social distancing rule, it triggers the 1st BBC Microbit to send the alert signal the 2nd Microbit which flashes the LED lights sewed to wearable jewellery.



4 Structure of the Report

The rest of the paper is organised as follows:

Section 5 describes hardware and software techniques to measure the distance between the camera and the human face with image as an input.

In this section the ultrasonic sensors, infrared light sensors, triangulation techniques using single and dual camera are explained. To keep the design simple, compact and less bulky instead of using additional sensors to measure the distance I decided to build my own distance approximation model using the facial images shot by the PI camera at various distances between 30cm and 220cm.

Section 5 describes the statistical modelling implementation details; compares and contrasts the accuracy results of number of models such as Linear Regression, Random Forest Model, Polynomial Regression and Generalised Additive Models (GAM).

Section 6 describes very briefly what are neural network model and introduces the convolution neural network model used for facial recognition and hence for social distance calculations.

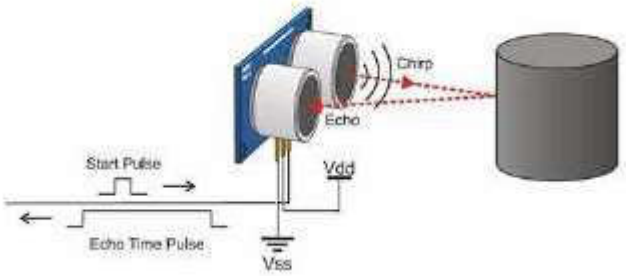
Section 7 discusses the implementation challenges and section 8 presents the conclusions and next steps.

5 Distance Measurement Techniques

There are number of options are available to measure the distance between the camera (the person holding/waring the device) and the other people. Some of the approaches are described below.

5.1 Ultrasonic Sensor

Ultrasonic is a cheap and reasonably accurate sensor to measure the distance between the sensor and target object. The ultrasonic sensor head emits the wave and receives [1] the wave reflected back from the target. Using the formula below the distance can be calculated.



Distance = 1/2 * T * S

In the above equation, T is the time between the emission and reception and S is the speed of sound wave which is 340m/s.

However, for the social distancing project this sensor is not suitable because it is meant for point-to-point distance calculation and doesn't cover the entire spectrum of surroundings.

5.2 LIDAR Sensor

LIDAR Sensor also works [2] on the same principle as simple ultrasonic sensor except that it emits the light instead of sound and its advantage is that it covers the 360 degrees surrounding. But the issue with this sensor is that it is bulky and need obstruction-free housing to facilitate the 360 degrees rotation of the light waves. The formula for distance calculation is: D = ½*Speed of Light * Time.

5.3 PI Camera for Distance Calculations

PI Camera is a tiny device with fixed focal length (camera specifications are below) and the distance between the camera and the target object can be calculated using various approaches described below.

5MP 1080p30 2592 x 1944 pixels
35mm focal length equiv.
focus 1 m to inf.
H 53 degrees x V 41 degrees
f/2.9

5.3.1 Single Camera: Triangle Similarity for Distance Calculation of a Known Object:

The following is the formula for distance calculation [3,4] between the camera and a known object using the triangle similarity approximation.

$$F = P * D / W$$

In the above equation,

F is focal length of the camera

D is the distance between the camera and a known object

W is the width or the known object

P is the measured pixel width of the object in an image.

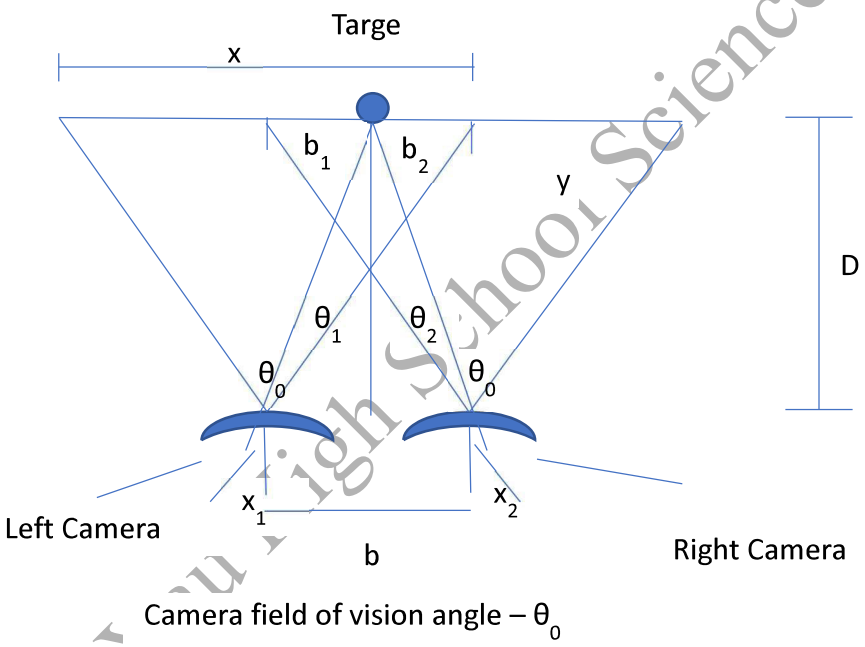
Once F is calculated or calibrated using the known measurements of the known object, the distance can be calculated as:

$$D = F * W / P$$

In the above equation, F is known (calibrated above), W is actual width of the known object, P is the pixel width calculated from the image, therefore, the distance D can be calculated by just plugging in the numbers. However, the key issue or the assumption with the triangle similarity is that it calculates the distance of only known object (because it requires W, the actual width of the reference object) and cannot be generalized.

5.3.2 Dual Camera: Triangular Similarity for Distance Calculations:

Using the dual camera (left & right) the distance [5] between the target and camera can be found using the formula below.



Camera field of vision angle – θ_0

In the above dual camera schematic diagram,

D = Distance between the camera centre and target

b = Distance between the centres of left and right camera

b = b1 + b2

x1 = image of the target T will be at a distance from left camera

x2 = image of the target will be at a distance from right camera

f = focal length of the camera

Applying the triangulation,

$$b1 / D = -x1 / f,$$

$$b2/D = x2 /f$$

$$b = D / f * (x2 – x1)$$

$$D = bf / (x2-x1)$$

The merits of the dual camera are that it no longer makes the assumption of knowing any known reference object.

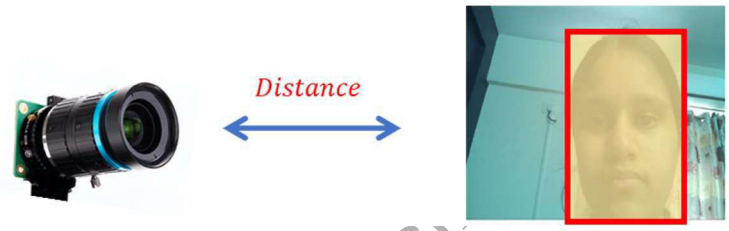
However, for the social distancing project, the implementation and final design of the solution makes bulky and also we do not need a very high accuracy for distance calculation, therefore, the use of dual camera is ruled out for social distancing project.

6 Statistical Modelling for Distance Measurements

In the previous section we looked at various hardware options to measure the distance and also concluded that the hardware devices make the wearable device quite bulky for social distancing project. Therefore, in this section I present a simple software solution for distance calculations.

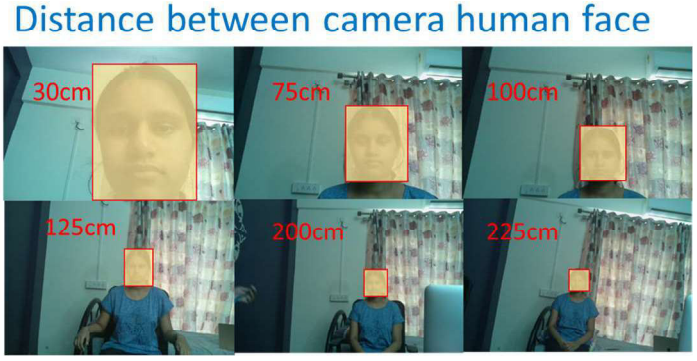
In the photograph we can observe that as the distance between the camera and object being is increased the size of the object reduces gradually. I leverage this **inverse relationship** between the distance and the size of the face bounding boxes and built a software model to calculate the social distance as described in the sections below.

$$Distance \propto \frac{1}{Size\ of\ Face\ Bounding\ Box}$$

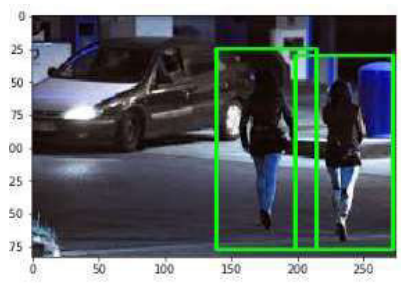


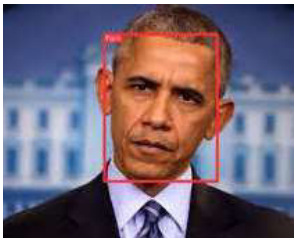
6.1.1 PI Camera Calibration for Distance Measurement

1. The PI camera that was used in this design is fixed focal length, therefore, the dimensions/size of the rectangle bounding box around the face is same if the distance between the camera and object/human face is kept constant.
2. Number of photographs are taken with human face as a focal point in the image, with varying distance between camera and human face from 30cm to 210cam.
3. To account for errors in measurement and to smoothen out the errors I have taken 20 images for each distance (from 30cm to 210 cm). Few sample images for distances 30cm, 40cm, 184cm and 220cm is shown below. From the images below, one can notice that as the object from the camera moves away its size shrinks gradually.



4. Each image is fed to the pre-trained convolution neural network [9,10], which detects the presence of facial or human body presence. On detecting the presence of human face or body, the model returns the bounding boxes (as shown below).





5. The bounding boxes returned by the deep learning model was used to calibrate the model.

6. For image I have taken in step 1 and using the bounding boxes returned by the deep learning model [11] in step 3, I have formulated the dataset as shown below.

```
# loop over the image paths
for imagePath in paths.list_images('./facial_images'):
    # load the image and resize it to (1) reduce detection time
    # and (2) improve detection accuracy
    #print(imagePath)
    image = cv2.imread(imagePath)
    image = imutils.resize(image, width=min(400, image.shape[1]))
    orig = image.copy()
    face_locations = face_recognition.face_locations(image)
    face_encodings = face_recognition.face_encodings(image, face_locations)
    for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
        print(imagePath, top,right,bottom, left, (bottom-top), (right-left))
```

```
./facial_images\fm.20200607_143103.dist_30.jpg 64 270 219 115 155 155
./facial_images\fm.20200607_143108.dist_30.jpg 98 270 253 115 155 155
./facial_images\fm.20200607_143114.dist_30.jpg 47 270 202 115 155 155
./facial_images\fm.20200607_143120.dist_30.jpg 64 270 219 115 155 155
```

Fig. Code snippet showing the face bounding box calculations

	FileName	top	right	left	bottom	w	h	area	Distance
0	./facial_images\fm.20200607_143103.dist_30.jpg	64	270	219	115	155	155	24025	30
1	./facial_images\fm.20200607_143108.dist_30.jpg	98	270	253	115	155	155	24025	30
2	./facial_images\fm.20200607_143114.dist_30.jpg	47	270	202	115	155	155	24025	30
3	./facial_images\fm.20200607_143120.dist_30.jpg	64	270	219	115	155	155	24025	30
4	./facial_images\fm.20200607_143126.dist_30.jpg	67	268	198	139	129	129	16641	30

Fig. Dataset in structured format

7. In the table above, the (top, right, left, bottom) represents the coordinates of the bounding boxes). From these dimensions, the width (w) and height (h) and the area of the bound box/rectangle are calculated.

8. Using width (w), height (h), area (a), I have formulated the following functional equation

$$F = f(w, h, a)$$

In the above equation,

F is the distance between the camera and target object (human face or body)

w is the width of the rectangle on the image

h is the height of the rectangle on the image

a is the area of the rectangle.

9. The functional form *f* that maps the w/h/a values to distance domain is modelled using the polynomial regression and Random Forest Decision based model.
10. The model output along with the model accuracy is presented below.
11. The below model results in section 6.6 show that the Polynomial regression is performing well as compared to the Random Forest Mode. Therefore, for the purpose of social distancing, the Polynomial Regression of degree 6 is used to calculate the distance between camera and the target object (human face/body).

6.2 Raw Data Visualizations

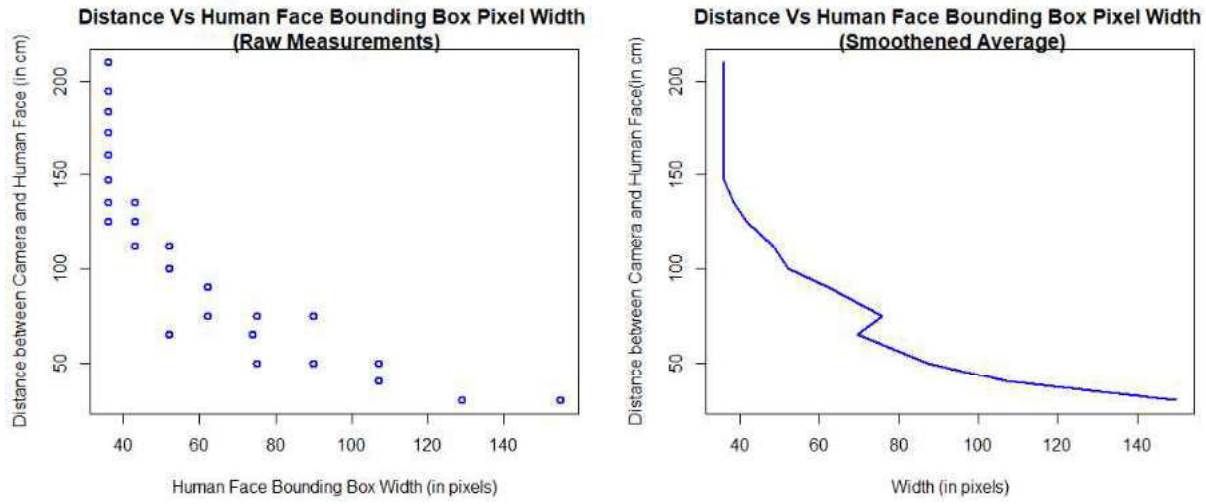
A snippet of raw data input data to the Polynomial and Random Forest models is shown below. The key inputs to the model are: width, height, are of the face rectangle bounding box in the image and the output is: distance (between camera and human face).

Raw Data

width	height	area	distance
155	155	24025	30
155	155	24025	30
155	155	24025	30
155	155	24025	30
129	129	16641	30
107	108	11556	40
107	108	11556	40
107	108	11556	40
107	108	11556	40
107	108	11556	40
90	90	8100	50
75	75	5625	50
107	107	11449	50

6.3 Raw Data Visualizations – Scatter Plots

The scatter plot of face bounding box width against the distance between the camera and the human face is shown below. The plot depicts as the distance between the camera and human increases, the size of the rectangle bounding box decreases exponentially. Since the valid social distance range varies from 125 to 200cm (depending on country/region regulations), I have taken more number of images in this range and hence you see more observations on the plot.



6.4 Modelling

This section presents the model results that transforms the *f(w,h,a)*, the human face bounding box width, height and area to distance *d* between the camera and human face.

To model the functional form *f(w,h,a)*, Linear Regression [6], Random Forest [8], Polynomial [7] and Generalised Additive Models and the results are presented below.

6.4.1 Linear Regression

Distance = w1*width + w2*height + w3*area + c

6.4.2 Random Forest

Distance = RandomForest(width, height ,area, number_of_trees =10)

6.4.3 Polynomial Fit using Natural Spline: Degree 2,4 and 6

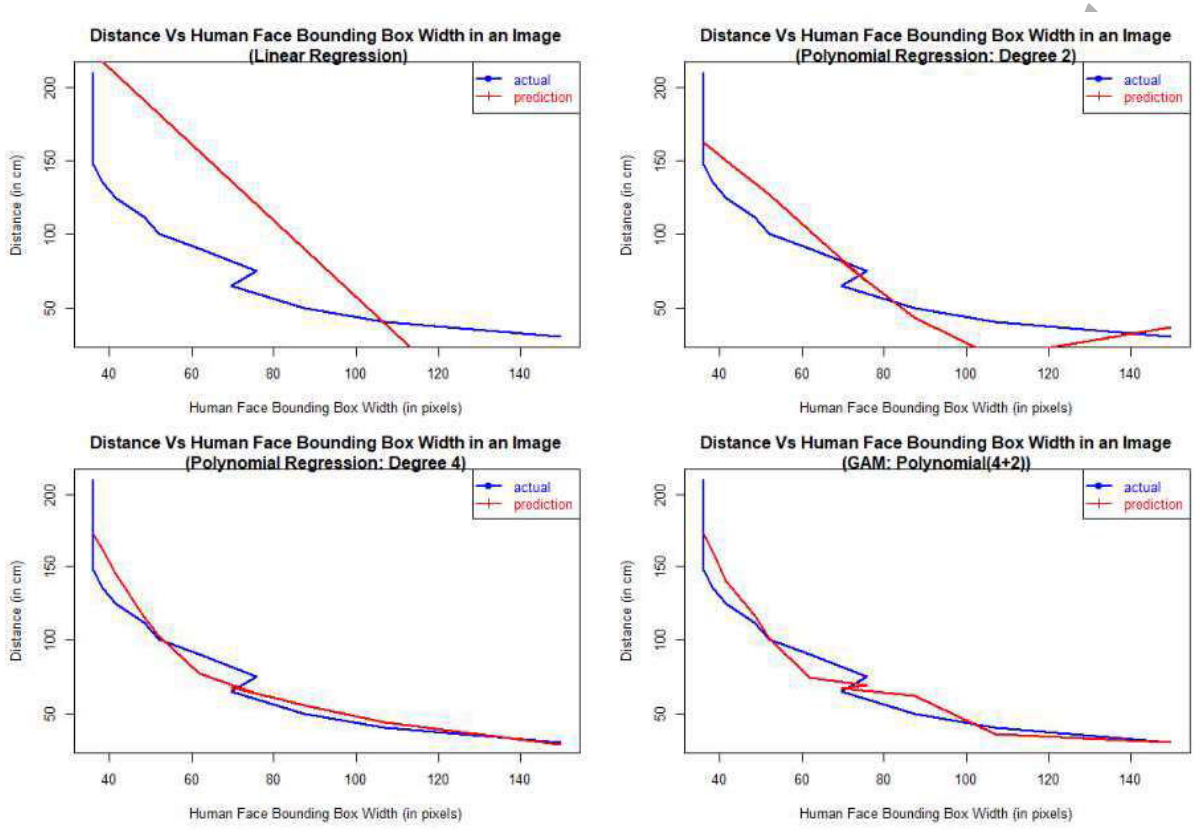
Distance = ns(width , height ,area, degrees =2)
Distance = ns(width , height ,area, degrees =4)
Distance = ns(width , height ,area, degrees =6)

6.4.4 Generalised Additive Model: Degree 2 + 4

Distance = GAM (ns(width , height ,area, degrees =2) +
ns(width , height ,area, degrees =4))

6.4.5 Results

The model fit results for the various models are shown below and the mean absolute errors are tabulated in the table below.



The MAE table shows, the Polynomial fit of degree 6 is giving the best performance, therefore, in this project I decided to use the polynomial model of order 6 to calculate the distance between the camera and human face detected by the deep learning model.

```
#Linear Regression
lm.fit <- lm(y_avg~x_avg$w+x_avg$h+x_avg$area,data=df_avg)

#Random Forest Model
rf.fit.1 <- randomForest(y~ ., data = df, maxnodes = 10, ntree = 1000)

#Polynomial Models
lm.fit.ns.2 <- lm(y_avg~ns(x_avg$w+x_avg$h+x_avg$area,2),data=df)
lm.fit.ns.4 <- lm(y_avg~ns(x_avg$w+x_avg$h+x_avg$area,4),data=df)
lm.fit.ns.6 <- lm(y_avg~ns(x_avg$w+x_avg$h+x_avg$area,6),data=df)

#GAM Model
gam.m3 = gam(y~ns(x_avg$w+x_avg$h+x_avg$area,4)+ns(x_avg$w+x_avg$h+x_avg$area,2)+x_avg$w+x_avg$h+x_avg$area,data=df_avg)
```

Fig. Code snippet of model fitting: Linear, Random Forest, Polynomial and GAM statistical models

Mean Square Error Comparison

MODEL	MSE
Linear Regression	350.3794
Random Forest	322.3804
Polymonial of Degree 2	271.1456
Polymonial of Degree 4	271.1456
Polymonial of Degree 6	242.8175
GAM degree 4+2+1	259.7174

Fig. Code snippet showing the MSE from all the statistical models

7 Deep Learning for Facial Recognition

In section 5 we looked at hardware choices for distance measurement and in the same section it was concluded that software model is the best option to keep the design simple and elegant.

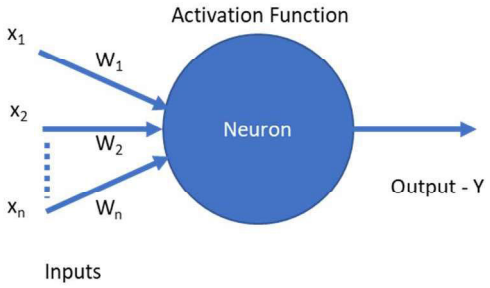
In section 6 we reviewed the methodology applied to calculate the social distance and concluded that polynomial regression of degree 6 is giving the best performance based on the input data of face bounding box on the image (width, height, area).

In this section, first Neural Networks are introduced briefly and then deep learning model (convolution neural network) for facial recognition is discussed in detail.

7.1 Neural Networks

Neural Networks are a set of algorithms [12, 13], modelled mimicking the human brain, that is designed to learn and recognise patterns. The patterns that are recognized are numerical in the form of vectors, into which all real-world data, whether it is images, audio, text or time series is mapped.

The basic block of the neural networks is Neuron (also called as Perceptron), which takes inputs, does computation, applies activation function and produces the output.

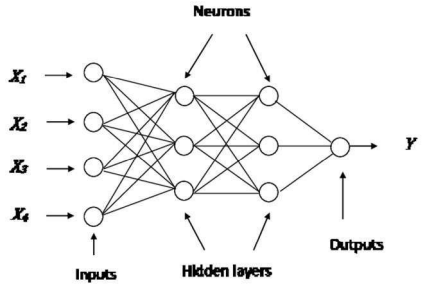


Output = ActivationFunction (X1*W1 + X2*W2 + … Xn*Wn)

$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{ threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{ threshold} \end{cases}$$

In the above equation, X represents the inputs, W represents the weights and Activation Function is some kind of non-linear function which acts like a barrier whether to pass or block the output based on some threshold value. There are different types of non-linear activation functions are available and the full discussion of these functions is beyond the scope of this report but interested reader can refer to the references[].

A typical Neural Network, as shown below, consists of series of layers. Each layer has number of nodes, which takes the inputs and multiplies with weights and produces 0 or non-zero output based on some threshold value calculated by a non-linear function (such as sigmoid, tanh etc.)



Every Neural Network will have an input layer, an output layer and 0 or more hidden layers. As the complexity of the patterns to be learnt increases, more and more hidden layers are added to the neural network. A simple neural network might have 1 to 3 hidden layers and some neural networks might have tens of hidden layers. In general, any neural network with large number of hidden layers are considered as Deep Learning Neural Network models.

7.2 Convolution Neural Network (CNN)

Convolution neural network [14] belongs to the class of deep learning model, most commonly used for image recognition. Yann Andre Lecun is the founding father of convolution neural networks and this network architecture is widely used for image recognition, image segmentation, image captioning and optical character extraction from the image.

A typical architecture of the CNN neural network model is presented below. The neural network takes the input image pixels in the form of vector of length MxN (image pixels width and height) and process through series of complex layers and the calculated output in the output layer is inferenced (facial recognition, object detection etc.) as shown in figure below.

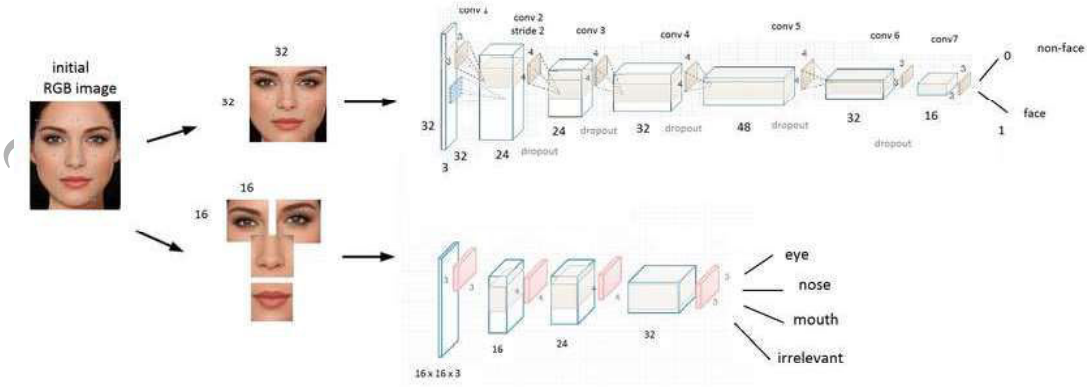


Fig.7.1. Facial Recognition Output

7.2.1 Facial Recognition for Social Distancing Project

There are number of implementations [15,16,17,18] are available in the literature for facial recognition and few of them are mentioned below.

Non-neural Network Implementation: HOG (Histogram Oriented Gradients)

Neural Network Implementation: YOLO, Face Net, VGGFace2, MobileNetV2 are different variations of CNN architectures described above.

For the social distancing model, I have used a pre-trained convolution neural network model provided by Google. The link to the source code and how to run the model can be found at the Google's GitHub website [19].

The python command to run the already pre-trained neural network model along with code snippet is provided below.

python3 detect.py --model ../all_models/mobilenet_ssd_v2_face_quant_postprocess_edgetpu.tflite

```
video_capture = cv2.VideoCapture(0)
while True:
    # Grab a single frame of video
    ret, frame = video_capture.read()

    # Convert the image from BGR color (which OpenCV uses) to RGB color (which face recognition uses)
    rgb_frame = frame[:, :, ::-1]

    # Find all the faces and face enqcodings in the frame of video
    face_locations = face_recognition.face_locations(rgb_frame)
    face_encodings = face_recognition.face_encodings(rgb_frame, face_locations)

    print('number of faces:', len(face_locations))
    # Loop through each face in this frame of video
    for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
        # See if the face is a match for the known face(s)
        wl = bottom-top
        hl = right-left
        distance = regressor.predict(np.array([wl,hl,wl*hl]).reshape(1,3))
        if (distance <= 200):
            led.on()
            GPIO.output(BUZZER,True)
        else:
            led.off()
            GPIO.output(BUZZER,False)

        cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)

        # Draw a label with a name below the face
        cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255), cv2.FILLED)
        font = cv2.FONT_HERSHEY_DUPLEX

        cv2.putText(frame, 'Distance = ' + str(distance) + ' cm', (left + 6, bottom - 6), font, 1.0, (255, 255, 255),

    # Display the resulting image
    cv2.imshow('Video', frame)
```

Capture video frame

Get face bounding boxes from Deep Learning moddel

From the face bounding box predict the distance using the regression model

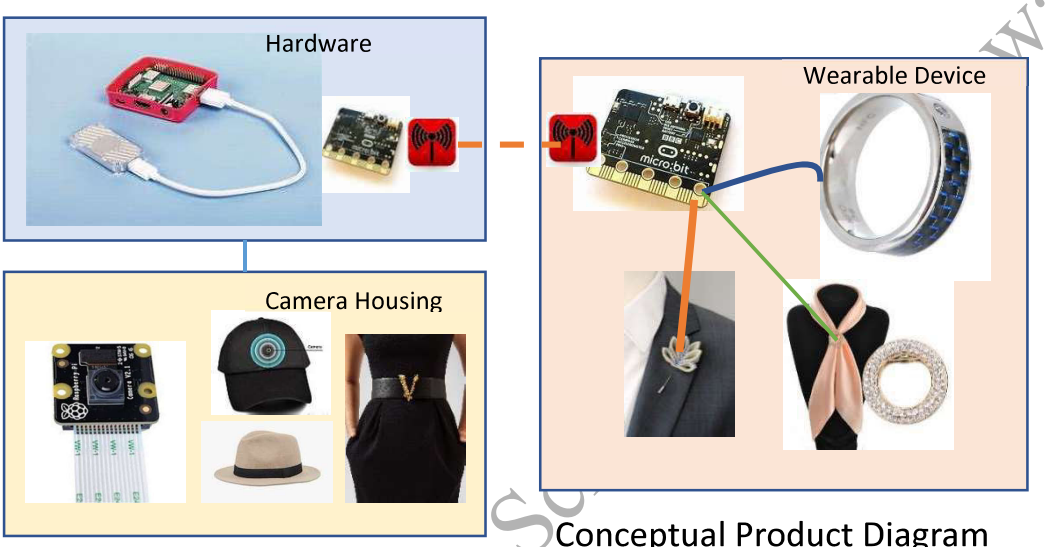
If **social distance** is violated set Raspberry PI signal to HIGH for LED/Buzzer alert

Fig. 7.4. High level code for social distancing violation

8 Finished Digital Wearable Product

The conceptual product diagram I imagined at the beginning of the project and final product is shown below. From the images below we can conclude that the final product is more or less similar to what I had imagined except that "it is not appealing and bit bulky", therefore, more work needs to be done to embellish the product design from aesthetics point of view.

8.1 Conceptual Product Image



Conceptual Product Diagram

8.2 Final Product Image

8.3 Video Demo Link

A short, 2-3 minutes video demo of this project can be accessed at the link below.

https://youtu.be/SOPq5EN4Egg

9 Results and Conclusions

Using Raspberry PI, PI Camera, Google Coral AI USB Accelerator I managed to put together a wearable digital solution that uses AI Deep Learning model for face recognition and experimented with Random Forest and Polynomial Regression Model for distance calculation (between camera and human face in an image).

The main conclusions from this project are: (a) the concept works very well and (b) a working prototype of wearable device is designed and (c) this wearable device can be used as a solution to alert/remind people whenever required social distancing rule is violated.

However, the journey has not been an easy ride. The challenges faced and the lessons learnt are discussed in the next section.

9.1 Issues with current solution

Even though the designed wearable devices work in principle, it has the following limitations:

- The PI Camera used in this project has a viewing angle of 150-160 degrees (i.e. the camera can detect only the faces that are within the viewing angle of 160) and outside of this range, the faces are undetected and the device would not alert the users
- The device would not alert if any social distance violations occur behind the person wearing the device. The only way to solve this issue is to introduce a 2nd camera facing behind or a camera with 360 degrees vision.
- The current design is bulky and it can fit on 15x8cm plastic box as shown in the section 8. Therefore, some work is needed to make the design compact and appealing.

10 Challenges

While executing this social distancing project I came across number of challenges during design, assembly and packaging phase. These challenges are described below.

10.1 Power Supply

Managing the power supply to hardware and wearable device was extremely challenging. The reason being the hardware (Raspberry Pi) requires 5V/2A power supply and the wearable device need at least 1.5v/500ma. Finding a light weight and portable power supply was challenging and the other challenge is finding a suitable power supply module that fit in product casing I have chosen for finished product.

10.2 Economics

After finishing the prototype and packing the final product, I found that the total manufacturing cost of this product is around Rs. 20,000/$275 (excluding the labour cost). This is too expensive and a lot more research needs to go on exploring the ways to bring the price down and embellish the design.

10.3 Manufacturing

While putting together the prototype I came across number of challenges where I needed variety of smaller parts, housing container in wide variety of configurations. Finding and sourcing these parts of was extremely challenging and some of the parts I couldn't find, therefore, I had to compromise on the final design. At that time, I wished that I could make those missing parts by myself and this led to the realization of the importance of 3D printing and the importance of computer aided design (CAD) manufacturing skills.

10.4 Packaging

This project was eye-opening for me in the sense: (a) having a crazy idea for making a product is one side of the coin and (b) the other side of the coin is its implementation and giving a final shape. Putting together the final product in compact design is even more challenging than the original idea.

The biggest lesson I learnt is no matter how good is the core idea, the process of manufacturing and compact packaging could make or break the project. The very first prototype I put together was of the size which fits in shoe box and then I spent countless number of hours in travel/surfing browser to find packaging parts that makes the design compact and pleasing to look at it.

10.5 Performance

When I initially began working on this prototype, I decided to use Raspberry PI 3B+ model which has 1GB RAM. Running a face recognition model on this Raspberry PI was giving me a throughput of 1 frame/image for every 30 seconds. That means the model was detecting the social distance violation

once in every 30 seconds. This is very slow and this kind of performance completely defeat the whole purpose of this project.

I initially thought the RAM size of 1GM was not enough, therefore, I decided to upgrade the "Raspberry PI 3B+ 1GM RAM" to "Raspberry PI 4B+ 2GB RAM". The new 4B+ model gave slightly better performance of 1 frame/image for every 20 seconds. Then I thought, if more RAM is added I could bring down the processing time so I upgraded the Raspberry PI 4B+ model from 2GB to 8GB RAM, didn't give the linear or exponential improvement I was hoping for.

When I further I researched, I found out that it is not the RAM but the CPU (no of computations/second) is the root cause for slower performance. This research leads me into finding a device called Google Coral AI USB Accelerator which is a small GPU (Graphical Processing Unit) for parallel computation.

The cost of this device is in USA is 75$ (Rs. 5600), but in India after import duties and tax the cost was Rs. 9300 (almost 80% more than the original price). This was a direct blow on my face but since I needed to boost the overall performance of the prototype I decided to buy the USB Accelerator and after integrating the Raspberry PI with the USB Accelerator, the performance of the prototype dramatically increased with 32 frames/images per second (which is almost 1000 times performance speed).

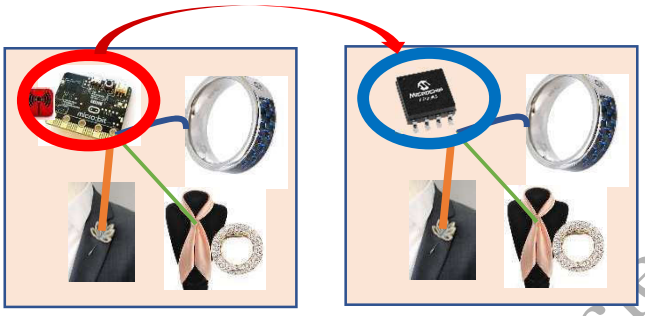
10.6 Lessons Learnt

This project taught me that innovation was the smallest part of the overall product design process. To bring any idea into a live/workable product one needs to keep an eye on design, materials availability, logistics, manufacturing, packaging, labour, time, and more importantly on financial side such as cost/tax/labour etc. Overall, it was exhilarating journey to see the entire design process: from Inception to Final Design.

11 Next Steps

11.1 Wearable Product

As a priority I certainly want to put some effort to embellish the finished design so it is attractive and fashionable to wear product. One of the ideas I have is integrating the BBC Microbit and Power Supply into a tiny chip called AT Tiny which is very small 5.4x5.35mm chip which can be ingrained into fabric or bangle reasonably easier to keep the design compact.



11.2 Overall Cost

I would like to explore the options to bring the overall cost of the product under Rs. 5000, some of the areas are:

- Explore Raspberry PI can be replaced with Arduino which will bring the cost down from Rs. 4200 to Rs 180
- Explore cheaper versions of the USB Accelerator or rewrite the model in c/c++ so the cost of accelerator, Rs 9200 can be excluded all together.

References

[1] Ptiik Universitas Brawijaya, May 30, 2013, Detection based on waves, https://siskomb.wordpress.com/2013/05/30/working-principle-ultrasonic-sensor-series/
[2] Bluesky International Ltd, @2020, How does LIDAR work: http://www.lidar-uk.com/how-lidar-works/
[3] Adrian Rosebrock on January 19, 2015, Find distance from camera to object/marker using Python and OpenCV: https://www.pyimagesearch.com/2015/01/19/find-distance-camera-objectmarker-using-python-opencv/
[4] Clayton Darwin, August 26, 2018, Distance (Angles Triangulation) - OpenCV and Python3 Tutorial - Targeting Part 5: https://www.youtube.com/watch?v=sW4CVI51jDY&t=736s
[5] Manaf A. Mahammed1, Amera I. Melhum, Faris A. Kochery, Object Distance Measurement by Stereo VISION, International Journal of Science and Applied Information Technology (IJSAIT), Vol.2, No.2, Pages: 05-08 (2013), http://www.warse.org/pdfs/2013/icctesp02.pdf
[6] Nhan Tran, Mar 20, 2019; Machine Learning: Polynomial Regression with Python, https://towardsdatascience.com/machine-learning-polynomial-regression-with-python-5328e4e8a386
[7] Michy Alice, April 18, 2017; Fitting Polynomial Regression in R, https://datascienceplus.com/fitting-polynomial-regression-r/
[8] Deepanshu Bhalla, Jun 2015, A COMPLETE GUIDE TO RANDOM FOREST IN R, https://www.listendata.com/2014/11/random-forest-with-r.html
[9] Adrian Rosebrock on June 18, 2018, Face recognition with OpenCV, Python, and deep learninghttps://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/
[10] AI Club @IIITB, Sep 10 2018, Semantic Segmentation using CNN's, https://medium.com/@aiclubiiitb/semantic-segmentation-using-cnns-357fbcfa1bc
[11] Adam Geitgey, MIT License, Face Recognition, https://github.com/ageitgey/face_recognition; https://pypi.org/project/face-recognition/
[12] Akshay L Chandra, August 22, 2018, Perceptron Learning Algorithm: A Graphical Explanation of Why It Works, https://towardsdatascience.com/perceptron-learning-algorithm-d5db0deab975
[13] Michael Nielsen / Dec 2019, CHAPTER 1Using neural nets to recognize handwritten digits http://neuralnetworksanddeeplearning.com/chap1.html
[14] Triantafyllidou, Danai & Tefas, Anastasios. (2016). Face detection based on deep convolutional neural networks exploiting incremental facial part learning. 3560-3565. 10.1109/ICPR.2016.7900186.
[15] Ahamed, Hafiz & Alam, Ishraq & Islam, Md. (2018). HOG-CNN Based Real Time Face Recognition. 10.1109/ICAEEE.2018.8642989., https://www.researchgate.net/publication/330684381_HOG-CNN_Based_Real_Time_Face_Recognition
[16] Jason Brownlee on May 31, 2019 in Deep Learning for Computer Vision, A Gentle Introduction to Deep Learning for Face Recognition, https://machinelearningmastery.com/introduction-to-deep-learning-for-face-recognition/
[17] Adrian Rosebrock on April 22, 2019, Getting started with Google Coral's TPU USB Accelerator, https://www.pyimagesearch.com/2019/04/22/getting-started-with-google-corals-tpu-usb-accelerator/
[18] Adrian Rosebrock on September 16, 2019, Install OpenCV 4 on Raspberry Pi 4 and Raspbian Buster, https://www.pyimagesearch.com/2019/09/16/install-opencv-4-on-raspberry-pi-4-and-raspbian-buster/
[19] Google Research, 2020, TensorFlow Deep Learning Models Repository, 2020, https://github.com/google-research?page=3