

Social Distance and Safety Violations

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology degree in Information Technology

By

BURRI JANAKI RAM REDDY (37120016)

VAJRAPU ULLAS (37120080)



**DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **B.Janaki Ram Reddy (37120016)**, **V.Ullas (37120080)** who carried out the project entitled **"SOCIAL DISTANCE AND SAFETY VIOLATIONS"** under my supervision from December 2020 to March 2021.

Internal Guide
Mr. P. Saravanan M.E.,

Head of the Department
Dr. R SUBHASHINI M.E, Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **B.Janaki Ram Reddy(37120016)**, **V.Ullas(37120080)** hereby declare that the Project Report entitled "**Social Distance and Safety Violations**" done by me under the guidance of **Mr.P.Saravanan M.E.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE:

PLACE:

SIGNATURE OF THECANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D, Dean**, School of Computing, **Dr. R.Subhashini M.E., Ph.D. , Head**, Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mr. P.Saravanan M.E.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

Corona virus is type of virus which spreads through the air. It is also called as COVID-19. To stop the spread of the virus the government and the WHO step forward the two mandatory activities. The activities should be followed by the people to safeguard from the virus are wearing the face mask and maintaining the Social Distance between the people. The people should follow the minimum 6-feet distance to save us from the virus. The activities of the people are monitored by the government to check whether the people are following the Social Distance or not. The social distance can be checked by using the deep learning, OPENCV, etc and it is monitored by using the cc camera videos. To detect the persons from the image or video we use the Deep Learning techniques and find the distance between the people. If the distance between the people is less than the minimum distance are bounded with the red color and those who are following the safety rules are bounded in the green color. The proposed model will work on both the CPU and GPU to get the very high accuracy. If the GPU is used in the model the FPS is very high and it detects the people with high accurate.

TABLE OF CONTENTS

CHAPTER.NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	viii
	LIST OF ABBREVIATIONS	ix
1	INTRODUCTION	
	1.1 About Project	1
	1.2 OpenCV	2
	1.3 Deep Learning	2
	1.4 Proposed System	3
	1.5 About YOLO	4
	1.5.1 YOLO V3	4
2	AIM AND SCOPE OF PROJECT	
	2.1 Objective	5
	2.2 Literature Survey	5
3	MATERIALS AND METHODS	
	3.1 YOLO V3 Algorithm	9
	3.2 COCO Dataset	11
	3.3 Methodology	11
	3.3.1 Object Detection	11
	3.3.2 Object Tracking	13

	3.3.3 Distance Measurement	14
	3.4 Euclidean Distance	14
	3.5 Workflow	15
4	RESULT	
	4.1 Result	17
	4.2 Sending the Mails	19
5	CONCLUSION AND FUTURE WORK	
	5.1 Conclusion	20
	5.2 Future Work	21
	REFERENCES	22
	APPENDIX	
	A. SOURCE CODE	23
	B. SCREENSHOTS	29
	C. PUBLICATION WITH PLAGIARISM REPORT	32

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO
Fig.1.1	Social Distancing	1
Fig. 3.3.1	YOLO Architecture	12
Fig. 3.4.2	IOU Operation	13
Fig. 3.5	Workflow Diagram	16
Fig. 4.1	Output	17
Fig. 4.2	Violation Message	18

LIST OF ABBREVIATION

ABBREVIATION	EXPANSION
YOLO	You Only Look Once
GPU	Graphical Processing Unit
ROI	Region of Interest
DNN	Deep Neural Network
SSD	Single Shot Multi Box
IOT	Internet of Things
IOU	Intersection Over Union
CNN	Convolution Neural Networks
R-CNN	Region Based Convolution Neural Networks

CHAPTER 1

INTRODUCTION

1.1 ABOUT PROJECT:

The pandemic situation has taken over the world and has made the conditions worst, as of now there is no vaccination developed for the contagious disease and hence social distancing has emerged as one of the best methods to prevent the spread of COVID-19. As the name suggests, social distancing implies that people should physically distance themselves from one another. The cases have been escalating at a very fast rate all over the world and thus social distancing is important. To monitor social distancing at public places, this survey paper provides a pinpointing solution. In this pandemic period using CCTV and drones we can keep a track on human activities at public places and henceforth we can compute and summarize distances between people and monitor the social distancing violations across the city. To be safe from the communicable virus the government introduced the new method called 'Social Distancing'.

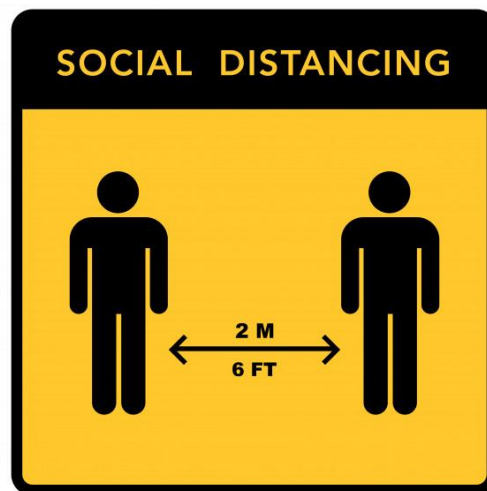


Fig-1.1 Social Distancing

Social Distance is to maintain the distance from the people. There are different technologies and the machines are introduced for detecting the social distance by

using the IOT devices and sensors. In the proposed system for detecting the social distance we use the deep learning technology. The algorithms used for the object detection are CNN, R-CNN, R-FCN, SSD, YOLO, etc. The object detection had playing the major role in the computer vision. The main goal of the object detection is to determine the objects are located in the given source. After the object detection there are different classes to classifying the each object. In the proposed system we use the YOLO algorithm to detect the objects in the give input.

1.2 ABOUT OPENCV

OPENCV Python is a library of Python bindings designed to solve computer vision problems. Python is a general purpose programming language started by Guido van Rossum that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability. Compared to languages like C/C++, Python is slower. That said, Python can be easily extended with C/C++, which allows us to write computationally intensive code in C/C++ and create Python wrappers that can be used as Python modules. This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it easier to code in Python than C/C++. OpenCV-Python makes use of Numpy, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCVarray structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib. OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language .

1.3 DEEP LEARNING

Deep learning is a branch of machine learning which is completely based on artificial neural networks, as neural network is going to mimic the human brain so deep learning is also a kind of mimic of human brain. In deep learning, we don't need to explicitly program everything. The concept of deep learning is not new. It has been around for a couple of years now. It's on hype nowadays because earlier

we did not have that much processing power and a lot of data. As in the last 20 years, the processing power increases exponentially, deep learning and machine learning came in the picture.

A CNN is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

1.4 PROPOSED SYSTEM

This Computer Vision based detection system would allow us to highlight in a given video the number of people who are violating the safe distance advised by World Health Organization. Apart from telling the overall status for example the amount of people following the guidelines or violating them, it highlights that individual on the screen so the authorities can take action. This can record the data at the place it has been deployed, for example a mall, and then the management and authorities can make decisions based on this data, this data could further be used in any other analytics project. This project is a proof of concept and without some hardware calibration it is not possible to achieve the exact distance, the answers will carry some small error which can be ignored. Violation of social distancing and other precautionary measures can be fatal, yet very common in almost all the public places, despite the gravity of the situation few have turned towards technology to tackle it. Currently there are no commercially used computer vision based systems to monitor and enforce social distancing. The problem is not difficult, nor are the pipelines used here are new to the computer vision community. However, due to it being a relatively young disease, not much work related to it could be published, the lack of literature makes it difficult to assume the problems currently faced by much of the computer vision scientists in solving the issue. This project aims at giving back to the community, in a form of a prototype that can be improved further by future researchers

1.5 ABOUT YOLO

You Only Look Once or YOLO is one of the popular algorithms in object detection used by the researchers around the globe. According to the researchers at facebook AI Research, the unified architecture of YOLO is extremely fast in manner. The base YOLO model processes images in real-time at 45 frames per second, while the smaller version of the network, Fast YOLO processes an astounding 155 frames per second while still achieving double the map of other real-time detectors. This algorithm outperforms the other detection methods, including DPM and R-CNN, when generalization from natural images to other domains like art works.

1.5.1 YOLO V3

YOLOv3 is the latest variant of a popular object detection algorithm YOLO – You Only Look Once. The published model recognizes 80 different objects in images and videos, but most importantly it is super fast and nearly as accurate as Single Shot Multi Box (SSD). First, it divides the image into a 13×13 grid of cells. The size of these 169 cells varies depending on the size of the input. For a 416×416 input size that we used in our experiments, the cell size was 32×32 . Each cell is then responsible for predicting a number of boxes in the image. For each bounding box, the network also predicts the confidence that the bounding box actually encloses an object, and the probability of the enclosed object being a particular class. Most of these bounding boxes are eliminated because their confidence is low or because they are enclosing the same object as another bounding box with very high confidence score. This technique is called non maximum suppression. Easy integration with an OpenCV application: If your application already uses OpenCV and you simply want to use YOLOv3, you don't have to worry about compiling and building the extra Dark net code. OpenCV CPU version is 9x faster: OpenCV's CPU implementation of the DNN module is astonishingly fast. For example, Dark net when used with OpenMP takes about 2 seconds on a CPU for inference on a single image. In contrast, OpenCV's implementation runs in a mere 0.22 seconds! Check out table below. Python support: Dark net is written in C, and it does not officially support Python.

CHAPTER 2

LITERATURE SURVEY

2.1 OBJECTIVE

Our ultimate goal is to compute the distance between two people present in an video stream and to check the criteria which we wanted to implement is implemented or not But challenge here is to send the mail to the multiple peoples at a time when the violations have been exceeded.

2.2 LITERATURE SURVEY

In this literature survey some of the related works of the person detection using the deep learning. Person detection for social distancing by using the ROI (Region of interest).In the paper the model is executed by suing the R-CNN algorithm which gives the medium accuracy for the detection of the people [1].

The camera view is also considered and the pedestrian detection is also done for the object detection with the low accuracy to detect the people [2].

A smart social distancing monitoring system is used to detect the distance between the people by using the My SD application and IOT devices. This model is high cost [3].

The visual social distancing problem says about the social distancing and gives the graphical representation of the person. It gives the parameters and the approaches should be taken to find the social distancing [4].

In the Real time face mask and the social distancing they had done by using the YOLO. It is considered for the face mask and the social distancing. It works only when the camera angle is convex lens [5].

Digital solution for the social distancing is done by using the YOLOv2 architecture. For the person detection they are using the real time pose estimation technology. For the pose estimation the MobileNetv1 and ResNet50 is used as backbone for the social distancing. And also their model uses the embedded devices and the web browser which leads to the high cost of the device [6].

The Novel Economical social distancing smart device is done by using the sensors like PIR sensor, grove, mini PIR, etc. The sensors used to detect the people movement in a particular range. For managing all those sensors a separate system should work. And the alerts system devices like Belt, Mobile alert, etc. But this paper model is ranges for high cost to manage and buy the sensors [7].

Social Distancing monitoring is done by using the mobile app. The algorithms used in the model are LSTM, MLP and BLE. The Bluetooth is used to find the social distancing between the mobile devices. In that model the accuracy is good but the while finding the measurement between the phones we can't get the exact alerts. A person may carry more than one phone and every time should carry with our phone. This makes the complexity to detect the social distancing. This model is using the SSD which takes the some extra time to detect the objects [8].

The social distancing is also done by using the IOT devices. But the sensors and IOT devices are high cost. We need to carry the devices wherever we go. The YOLO model is used but it only shows the violation of the people. The accuracy of the model is 90% [9].

Deep Neural Network is used to track the person and finds the social distance [10].

We can use the AI technology to find the social distance but takes larger time to compute the model. In 2017, Dr. S Syed Ameer Abbas and his co-authors proposed a system for human tracking and crowd management using raspberry pi and Open-CV. A cascade classifier was trained for head detection from the scene is trained using Haar features through OpenCV. The whole concept of their idea was to record the crowded scene using a camera and Raspberry pi3 that has a quad core ARMv8 central processing unit which processes the video frame by frame. The head count is measured and the crowd is managed by comparing the value with the threshold and if it surpasses the threshold the prevention can be done accordingly [11]

In 2018, Joel Joseph Joy and his co-authors proposed a system of traffic density identification which was based on image processing. The queue length and the traffic densities were recorded from the images taken from the camera. The video input was taken and fuzzy logic was applied to handle the concept of partial truth. The outcome of partial truth concept could range anywhere between completely true and completely false [12].

In 2020, Adrian Rosebrock published an article on social distancing detector which is based on OpenCV, Computer Vision and Deep Learning concept. The article throws a light on social distancing during the pandemic period and it focuses on social distance monitoring through CCTV cameras installed across streets. The camera records the distance between people in pixels and compares it with the standard measurement and thus behave as a social distancing detector. This social distance detector application logic resides in the file.py script and this file is responsible for looping over frame of a video stream and ensuring that people are maintaining a healthy distance from one another. It is compatible with both video files and webcam streams [13].

In 2019, Neel Bhave and his co-authors proposed a system which is a complete working model which comprised of Reinforcement model and Object detection

algorithms. In this they used YOLO (You Only Look Once) Real Time Object Detection which has less shortcomings, is much faster, provides accurate results and can be trained for more than 200 classes. Reinforcement learning is an area of machine learning which is responsible for providing the green phase timing according to the current state of traffic and learn from the actions taken [14].

CHAPTER-3

METHODOLOGY

3.1 YOLO V3 ALGORITHM

YOLO V3 is an improvement version of the former YOLO and YOLO 9000, and adopts some state-of-the-art architectures like Residual Network. The implementation which we use is its original, on C language. This implementation doesn't depend on other high level deep learning libraries like Tensorflow or Caffe and it could slightly improve its efficiency. Firstly we evaluate its pretrained model which is trained by the author. This model is trained on Pascal VOC + COCO. To get quantifiable evaluation results, we evaluate this model on the dataset Pascal VOC 07 Evaluation, for the class Person. Then, we train our model to figure out if we can improve its performance by training on different dataset. We trained by ourselves 2 models, one is trained on Pascal VOC 07 with one class Person and another is trained on COCO. The training is executed on the cluster of computation MYRIA in the Normandy calculate center CRIANN. The model trained on Pascal VOC 07 takes about 9000 batches and the model trained on COCO takes more than 16000 batches with the batch size of 32.

2 Nvidia P100 can train about 1500 batches per hour and 2 Nvidia K80 can train around 800 batches per hour. To quantify our models performance, we use the same evaluation dataset as the pre-trained model, Pascal VOC 07 Evaluation. Beside this quantifiable result, we also make the inference on a video of driver's perspective which is taken by us in the centre and urban of the city Rouen. Those results show two important consequences.

A rich dataset is critical for the performance of precision (map). The model trained on COCO dataset with 64115 images of person shows 5.6mAP improvement over the Pascal VOC dataset with 2000 images of persons.

The number of classes doesn't significantly influence the speed of inference. The one class model is only about 0.4FPS faster than the 80 classes model. The inference time mainly depends on the quantity of parameters in the neural network, but the suppression of classes has only impact on the number of parameters of the fully connected layer's, which is a small part of the whole

model.

Also, the most important parameter of the Algorithm, its Loss function is shown below. YOLO simultaneously learns about all the four parameters it predicts (discussed above).

Loss function for YOLO,

$$\begin{aligned} & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\ & + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 \\ & + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2 \\ & + \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \end{aligned}$$

So this was all about the YOLO Algorithm. We discussed all the aspects of Object detection along with the challenges we face in that domain. We then saw some of the algorithms that tried to solve some of these challenges but were failing in the most crucial one-Real time detection (speed in fps). We then studied the YOLO algorithm which outperforms all the other models in terms of the challenges faced, its fast-can work well in real-time object detection, follows a regression approach. Still improvements are being made in the algorithm. We currently have four generations of the YOLO Algorithm from v1 to v4, along with a slightly small version of it YOLO-tiny, it is specifically designed to achieve an incredibly high speed of 220fps.

3.2 COCO DATASET

Is a publicly available dataset having over 80 classes, the MobileNet SSD was trained on COCO dataset. Since we needed only 1 class from that dataset, that is 'person' class, thus all other classes were filtered out. The pre-trained MobileNet on COCO dataset was used instead of training from scratch, the weights and architecture were imported through OpenCV.

3.3 METHODOLOGY

In the proposed method 3 stages are included to find the social distance and safety violation. They are: Object Detection, Object Tracking and Distance Measurement

In this Social Distancing Detector, I'll be using the weights of the *YOLO v3* Object Detection Algorithm and the *COCO dataset* which are easily available online. Additionally, the main library being used will be the *OpenCV* along with the Deep Neural Network (*DNN*) module

The model consists of three modules. They are:

1. Object Detection
2. Object Tracking
3. Distance Measurement Between The Objects

3.3.1 OBJECT DETECTION

To detect the object in the given input video we use the YOLO Algorithm. The algorithm consists of the different classes with their specific details. The weights and the class are needed to detect the object. Each class has its own weights. For the object detection the input video is converted into the frames and each frame had a matrix to find the objects. The algorithm gives the much better performance on all the parameters and it runs with high fps in the real time usage. The algorithm aims to predict a class of an object and the bounding box of the specified object location. Each bounding box is described by using the four parameters:

Center of the bounding box (b_i , b_j),

width (w),

height (h)

and value 'c' is the class of an object. While detecting for the object detection the input video is converted into the frames and each frame had a matrix to find the objects. For the object detection the input video is converted into the frames and each frame had a matrix to find the objects. The algorithm gives the much better performance on all the parameters and it runs with high fps in the real time usage.

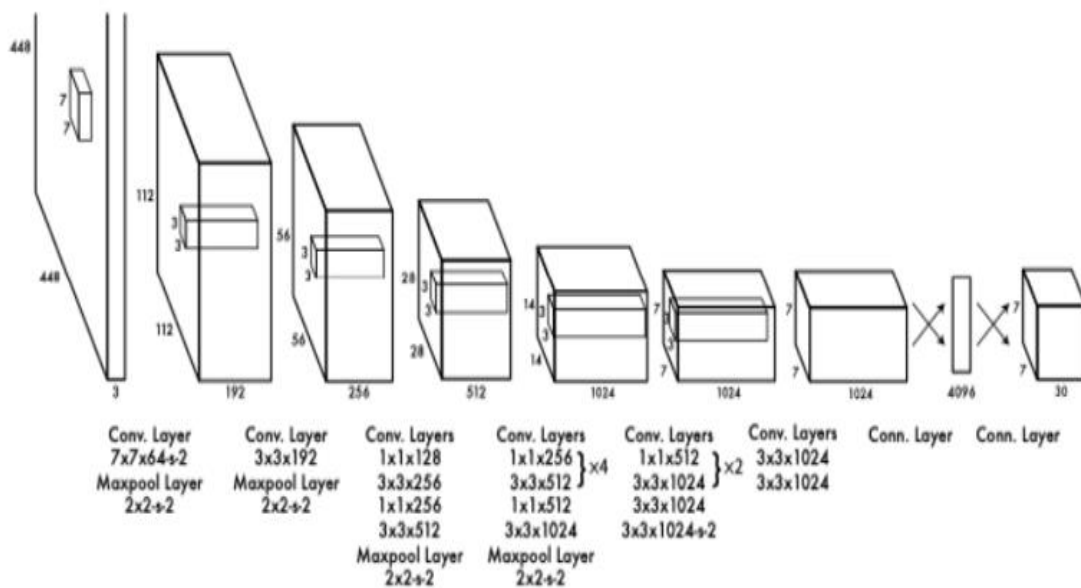


Fig-3.3.1 YOLO Architecture

Objects Non-max suppression problem raises. To solve the problem we use the IOU (Intersection over Union) with the one having the highest class probability among them in the input image.

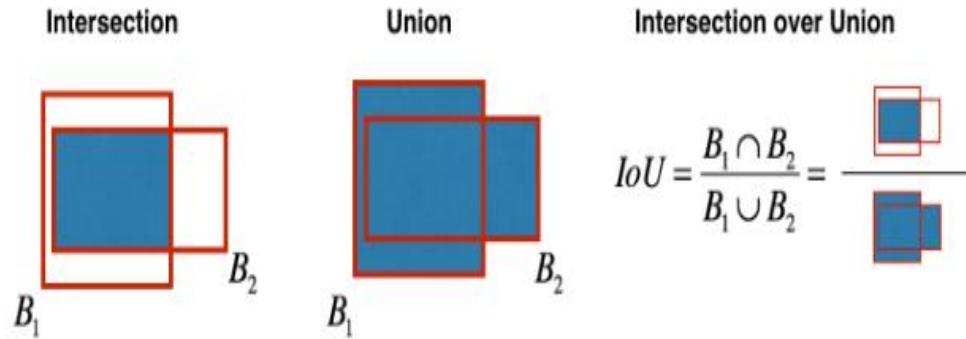


Fig-3.3.2 IOU operation

The IOU value calculates the bounding boxes with respect to the one having the highest class probability and then it rejects the remaining bounding values. And find the bounding box with highest class probability and given a single bounded box to the object.

3.3.2 OBJECT TRACKING

After the detection of the object it should be tracked. When the object is detected give the special Id to track the object. We have found the center of the object when it is detected. When the object moves to another new location find its centroid also. To measure the center point of the bounding box of the detected person or object is. Object tracking is a field within computer vision that involves tracking objects as they move across several frames.

We implemented object tracking algorithm using the openCV library.

This object tracking is called Centroid tracking as it relies on the Euclidean distance between two Centroid between subsequent frames in a video

$$M(x, y) = \left(\frac{X_{min} + X_{max}}{2}, \frac{Y_{min} + Y_{max}}{2} \right)$$

The above one is used to find the center point of the bounding box.

3.3.3 DISTANCE MEASUREMENT

To stop the spread of the communicable virus the persons should maintain the minimum 6 feet distance. After finding the center of the bounding box then find the distance between the other bounding boxes. To calculate the distance between the bounding boxes we use the Euclidean formula

$$D(M1, M2) = \sqrt{(X_{max} - X_{min})^2 + (Y_{max} - Y_{min})^2}$$

If the distance between the people is less than the 6 feet then the bounded box is in the red color and if the distance between the people is more than the 6 feet then the bounding box is in the green color. NMS (Non-maxima suppression) is also used to reduce overlapping bounding boxes to only a single bounding box, thus representing the true detection of the object. Having overlapping boxes is not exactly practical and ideal, especially if we need to count the number of objects in an image. Euclidean distance is then computed between all pairs of the returned centroids. Simply, a centroid is the center of a bounding box. Based on these pairwise distances, we check to see if any two people are less than/close to 'N' pixels apart.

Project Implementation

Apply object detection to detect all people (and only people) in a video stream or an CCTV footage from frame to frame.

Compute the pairwise distance between all detected people.

Based on these distances, check to see if any two people are less than N pixels apart.

And also check the violation if exceeded then mail will be sent to the near by authorities.

3.4. EUCLIDEAN DISTANCE

A threshold will be set and all those having distance less than the threshold will be highlighted by a red bounding box. The total number of violators will be displayed on the screen. The threshold or the minimum distance will be 1 meter as recommended by WHO. Although the best approach to get distance is to incorporate the vision part with depth sensors such as LiDAR, Radar, or Sonars, but since this is a purely software based project that option is out of bounds. Camera calibration is required

whenever this is being deployed, calibration will vary depending on the camera. For depth a pinhole camera model and similar triangle approach was used to estimate the depth of person from camera, doing so resulted in more accurate distance calculations between the persons.

3.5 WORKFLOW

As shown in the flow diagram the model processing goes on first the image input is taken as video/streaming video these video or stream is classified and then and pre-processed into frames and these frames are identified individually by one by one with help of the algorithms like yoloV3 which consists of the pre-trained classes which are used for the object detection such as people, animals, cars, vehicles, etc..These pre-trained classes which helps in object detection. And after classify the person and takes only the person class present in the objects of YOLO and finds the centroid of each frame and then calculate the distance between the points from frame to frame and finds the distance measurement and then displays the output on the screen which shows the number of people present on the screen and number of violation happening on the stream, also number of people following safety precautions i.e by maintaining a minimum distance of 2 meters(6 feet). And each of the following is displayed on the screen by a rectangular box denoted with color for the specify type such as the people following the safety rule are shown in green and the people violating or any abnormal violations are denoted in yellow and red color. The total workflow as shown in the below flow diagram step-by-step how does the process goes on from the input video stream to the converting into frames and then the bounding boxes are formed and final output is shown.

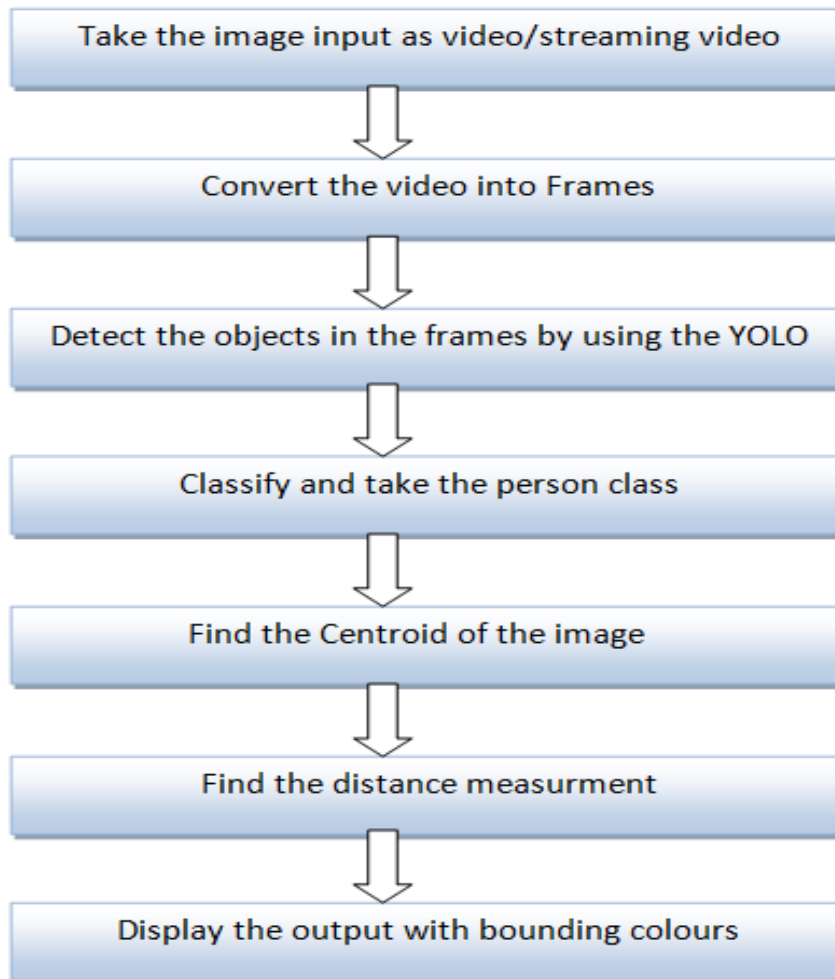


Fig-3.5 Workflow Diagram

CHAPTER 4

RESULT AND DISCUSSION

4.1 RESULT:

We tested our model using a video stream and images. Of which, we could see the proper detection of people according to the distance between a pair. The frames were also labelled as safe and unsafe accordingly. Also, the count of the violations made were counted and were constantly updating. While using the webcam, it is necessary to have people moving continuously else the detection goes incorrect. This could happen due to the detection method, wherein the entire frame is detected, and further, the distance calculation and comparison between the centroids takes place. The results obtained by the model are displayed in fig 4. The red color box indicates the people are not maintaining the social distance hence those boxes in red are considered as the serious violations.

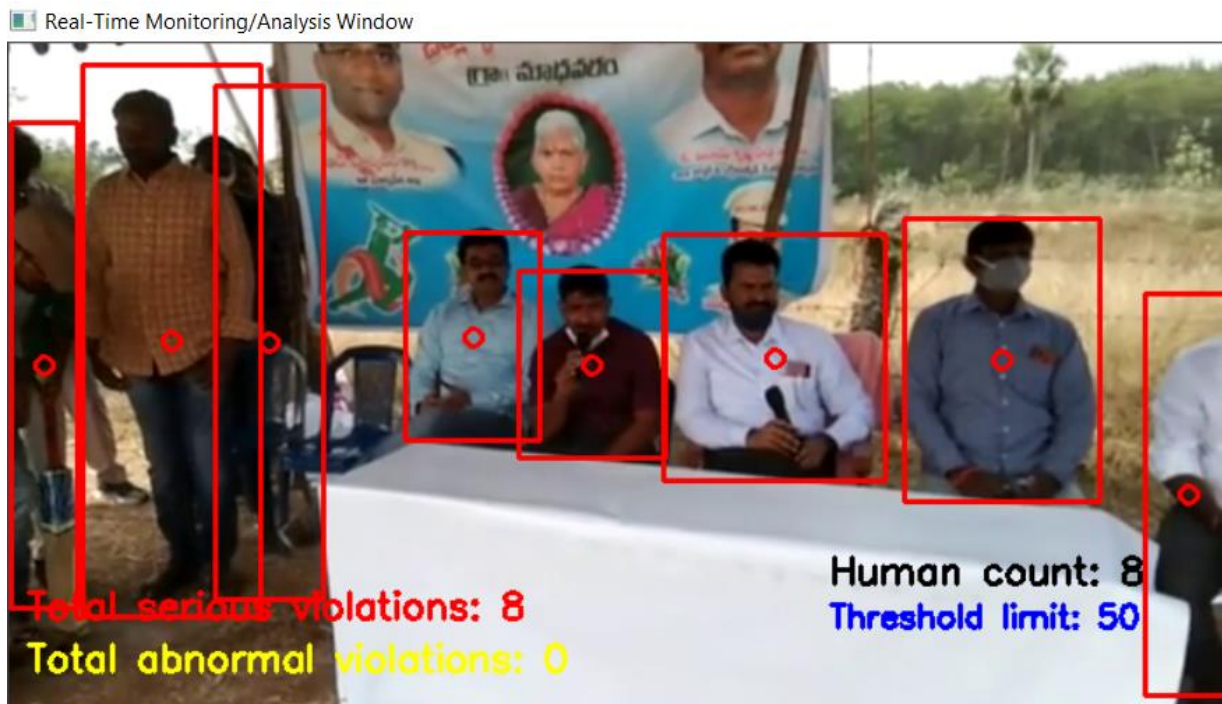


Fig-4.1 Output

4.2 SENDING MAIL

After detecting the people who are safe and non safe we just wanted to mail to some certain people who can look at the surrounding changes and that could help in attaining the safety measures by sharing through news, scrolling and few announcement or watching over the area. We had set mails such that if any violation occurs it would share the mails to the people, and we can share multiple mails at a time. That's we wanted to alert people through the authorities near by this could be one way of stopping the gathering of people.

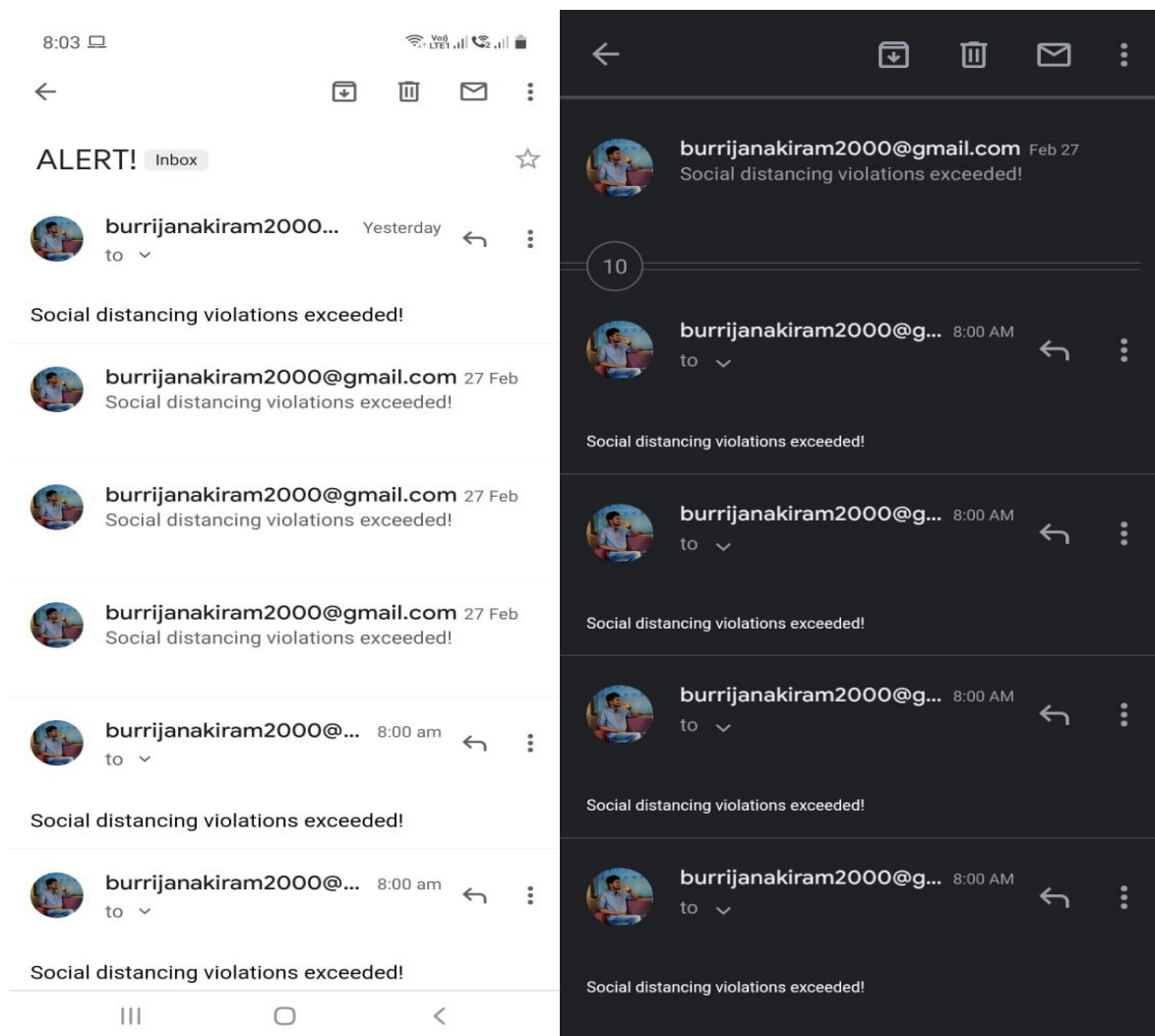


Fig-4.2 Violation Message

As shown in the above screenshots of the different mails where we had share the violations have been exceeded through a common mail and we had done SMTP and we have done it successfully. If the violations are exceed than the threshold limit then it will send a mail. We can also give the single and the multiple mails to send the alerts based on the threshold limit.

CHAPTER 5

SUMMARY AND CONCLUSIONS

5.1 CONCLUSION:

As we envision the world post COVID-19 pandemic the need of self-responsibility emerges irrefutably. The scenario would mostly focus on accepting and obeying the precautions and rules that WHO has imposed more precisely as responsibility of one will totally embark on themselves and not government. Social Distancing would undoubtedly be the most important factor as COVID 19 spreads through close contact with infected ones. In order to supervise large mobs, an effective solution is important and this survey paper focuses on that. Using installed CCTV and drones, authorities can keep a track of human activities and control large crowd to come together and prevent violating the law. As far as people are maintaining a safe distance they would be indicated with green light, and as the CCTV captures more and more crowd gathering, red light would pop-up and the allocated police of that area will be notified and the situation can come under control immediately. As controlling large mob is not an easy task, using this survey, conditions can be managed before situation goes out of control. Thus, implementing this idea can reduce the on ground efforts of the police and they can entirely focus on supervising conditions exclusively on those areas where conditions are unfavorable and thus, they can utilize time wisely and save energy for equitable situations.

To stop the spread of communicable virus we need to maintain the social distance with minimum 6 feet from person to person. We can also give the alert message to the people who are not following the social distance. In this model we can see the people who are safe are in the red bounded box, the persons who are near to the minimum distance are abnormal violation who are bounded in the yellow colour and the person who are bounded in the red color are violating the social distance. The accuracy, headcount, threshold limit, FPS and elapsed time is calculated based on the CPU and GPU. This program is a very simple and basic implementation of the Social Distancing Detector. This code can serve as a basis for more complex programs which can be developed from this.

5.2 FUTURE WORK:

Video can be calibrated to get bird's eye view for more accurate distance estimation between objects. We should improve our social distancing detector is to utilize a power camera calibration. By using the angle and the distance count in the people will make more accurate. Deployment of the model in the inbuilt camera or any other camera.

REFERENCES

1. Afiq Harith Ahamad, IEEE International Conference on Control System, Computing and Engineering (ICCSCE2020), 21-22 August 2020, Penang, Malaysia.
2. Yew Cheong Hou, International conference on Information technology and Multimedia (ICIMU), 2020.
3. Mohammed Ali, 8th International Conference on Information Technology and Multimedia (ICIMU), 2020.
4. Marco Cristani, Alessio Del Bue, IEEE (Accepted for publication), Volume 8, 2020, 126876.
5. Krisha Bhambani, Tanmay jain, IEEE (conference paper), 2020.
6. Arnab Ghorai, Sarch Gawde, International conference on Innovative Computing and Communication (ICICC 2020).
7. Rahul reddy Nadikattu, International Journal of Electrical Engineering and Technology (IJEET), Volume 11, Issue 4, June 2020, pp, 204-217, Article ID: IJEET_11_04_023.
8. Dr. Neelavathy pari S, Balaji Vasu, International journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol.9 Issue 09, September-2020.
9. Rucha Visal, Atharva theurkar, International research Journal of Engineering and Technology (IRJET), Volume:07 Issue:06, June 2020.
10. Mahdi rezaei, Mohsen Azarmi, Social Distancing Monitoring and Infection Risk Assessment in COVID-19 Pandemic.
11. Adrian Rosebrock, Opencv Social Distancing Detector, 2020.
12. World health organization. WHO Corona-virus Disease Dashboard (August 2020). Available at (<https://covid19.who.int/table>.)
13. C.Sun, D.Wang, "Person re-identification via distance metric learning with latent variables," IEEE Transactions on Image Processing, vol 26, no.1, pp, 23-33, 2016.

A. SOURCE CODE

run.py

```
from mylib import config, thread
from mylib.mailer import Mailer
from mylib.detection import detect_people
from imutils.video import VideoStream, FPS
from scipy.spatial import distance as dist
import numpy as np
import argparse, imutils, cv2, os, time, schedule

ap = argparse.ArgumentParser()
ap.add_argument("-i", "--input", type=str, default="",
help="path to (optional) input video file")

ap.add_argument("-o", "--output", type=str, default="",
help="path to (optional) output video file")

ap.add_argument("-d", "--display", type=int, default=1,
help="whether or not output frame should be displayed")

args = vars(ap.parse_args(["--input", "C://Users/burri/Pexels Videos
1721303.mp4", "--output", "my output.avi"]))

labelsPath = os.path.sep.join([config.MODEL_PATH, "coco.names"])
LABELS = open(labelsPath).read().strip().split("\n")
weightsPath = os.path.sep.join([config.MODEL_PATH, "yolov3.weights"])
configPath = os.path.sep.join([config.MODEL_PATH, "yolov3.cfg"])
net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)
if config.USE_GPU:
    print("")
    print("[INFO] Looking for GPU")
```

```

net.setPreferableBackend(cv2.dnn.DNN_BACKEND_CUDA)
net.setPreferableTarget(cv2.dnn.DNN_TARGET_CUDA)
ln = net.getLayerNames()
ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]
if not args.get("input", False):
    print("[INFO] Starting the live stream..")
    vs = cv2.VideoCapture(config.url)
    if config.Thread:
        cap = thread.ThreadingClass(config.url)
    time.sleep(2.0)
else:
    print("[INFO] Starting the video..")
    vs = cv2.VideoCapture(args["input"])
    if config.Thread:
        cap = thread.ThreadingClass(args["input"])
writer = None
fps = FPS().start()
while True:
    if config.Thread:
        frame = cap.read()
    else:
        (grabbed, frame) = vs.read()
        if not grabbed:
            break

    frame = imutils.resize(frame, width=700)
results = detect_people(frame, net, ln,
                        personIdx=LABELS.index("person"))
serious = set()
abnormal = set()

```

```

if len(results) >= 2:
    centroids = np.array([r[2] for r in results])
    D = dist.cdist(centroids, centroids, metric="euclidean")
    for i in range(0, D.shape[0]):
        for j in range(i + 1, D.shape[1]):
            if D[i, j] < config.MIN_DISTANCE:
                serious.add(i)
                serious.add(j)
            if (D[i, j] < config.MAX_DISTANCE) and not serious:
                abnormal.add(i)
                abnormal.add(j)
    for (i, (prob, bbox, centroid)) in enumerate(results):
        (startX, startY, endX, endY) = bbox
        (cX, cY) = centroid
        color = (0, 255, 0)
        if i in serious:
            color = (0, 0, 255)
        elif i in abnormal:
            color = (0, 255, 255) #orange = (0, 165, 255)
        cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)

        cv2.circle(frame, (cX, cY), 5, color, 2)
    Safe_Distance = "Safe distance: >{} px".format(config.MAX_DISTANCE)
    cv2.putText(frame, Safe_Distance, (470, frame.shape[0] - 25),
                cv2.FONT_HERSHEY_SIMPLEX, 0.60, (255, 0, 0), 2)
    Threshold = "Threshold limit: {}".format(config.Threshold)
    cv2.putText(frame, Threshold, (470, frame.shape[0] - 50), cv2.FONT_HERSHEY_SIMPLEX, 0.60,
                (255, 0, 0), 2)
    text = "Total serious violations: {}".format(len(serious))
    cv2.putText(frame, text, (10, frame.shape[0] - 55),
                cv2.FONT_HERSHEY_SIMPLEX, 0.70, (0, 0, 255), 2)

```

```

text1 = "Total abnormal violations: {}".format(len(abnormal))
    cv2.putText(frame, text1, (10, frame.shape[0] - 25),
cv2.FONT_HERSHEY_SIMPLEX, 0.70, (0, 255, 255), 2)
if len(serious) >= config.Threshold:
    cv2.putText(frame, "-ALERT: Violations over limit-", (10,
frame.shape[0] - 80),cv2.FONT_HERSHEY_COMPLEX, 0.60, (0, 0, 255), 2)
    if config.ALERT:
        print("")
        print('[INFO] Sending mail...')
        Mailer().send(config.MAIL)
        print('[INFO] Mail sent')
    if args["display"] > 0:
        cv2.imshow("Real-TimeMonitoring/Analysis Window", frame)
        key = cv2.waitKey(1) & 0xFF
        if key == ord("q"):
            break
    fps.update()
if args["output"] != "" and writer is None:
    fourcc = cv2.VideoWriter_fourcc(*"MJPG")
    writer = cv2.VideoWriter(args["output"], fourcc, 25,
(frame.shape[1], frame.shape[0]), True)
if writer is not None:
    writer.write(frame)
fps.stop()
print("[INFO] Elapsed time: {:.2f}".format(fps.elapsed()))
print("[INFO] Approx. FPS: {:.2f}".format(fps.fps()))
cv2.destroyAllWindows()
detection.py
from .config import NMS_THRESH, MIN_CONF, People_Counter
import numpy as np
import cv2

```

```

def detect_people(frame, net, ln, personIdx=0):
    (H, W) = frame.shape[:2]
    results = []
    blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416),
        swapRB=True, crop=False)
    net.setInput(blob)
    layerOutputs = net.forward(ln)
    boxes = []
    centroids = []
    confidences = []
    for output in layerOutputs:
        for detection in output:
            scores = detection[5:]
            classID = np.argmax(scores)
            confidence = scores[classID]
            if classID == personIdx and confidence > MIN_CONF:
                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))
                boxes.append([x, y, int(width), int(height)])
                centroids.append((centerX, centerY))
                confidences.append(float(confidence))
    idxs = cv2.dnn.NMSBoxes(boxes, confidences, MIN_CONF, NMS_THRESH)
    if People_Counter:
        human_count = "Human count: {}".format(len(idxs))
        cv2.putText(frame, human_count, (470, frame.shape[0] - 75),
            cv2.FONT_HERSHEY_SIMPLEX, 0.70, (0, 0, 0), 2)
    if len(idxs) > 0:
        for i in idxs.flatten():
            (x, y) = (boxes[i][0], boxes[i][1])

```

```
(w, h) = (boxes[i][2], boxes[i][3])  
r = (confidences[i], (x, y, x + w, y + h), centroids[i])  
results.append(r)  
return results
```

B. SCREENSHOTS

```
Anaconda Prompt (Anaconda3) - python run.py -i mylib/videos/garden.mp4

(base) C:\Users\vajrapu ullas>cd..

(base) C:\Users>cd..

(base) C:\>d:

(base) D:\>cd Social Distance2

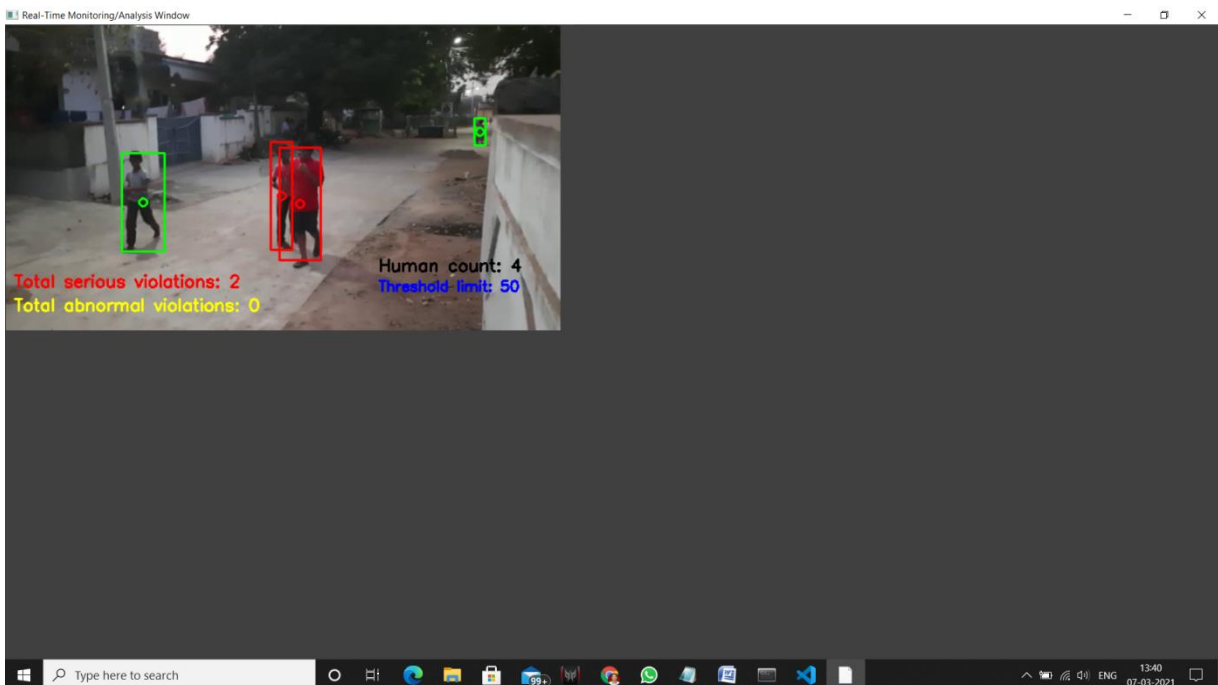
(base) D:\Social Distance2>cd Social-Distancing-Detection-in-Real-Time-main

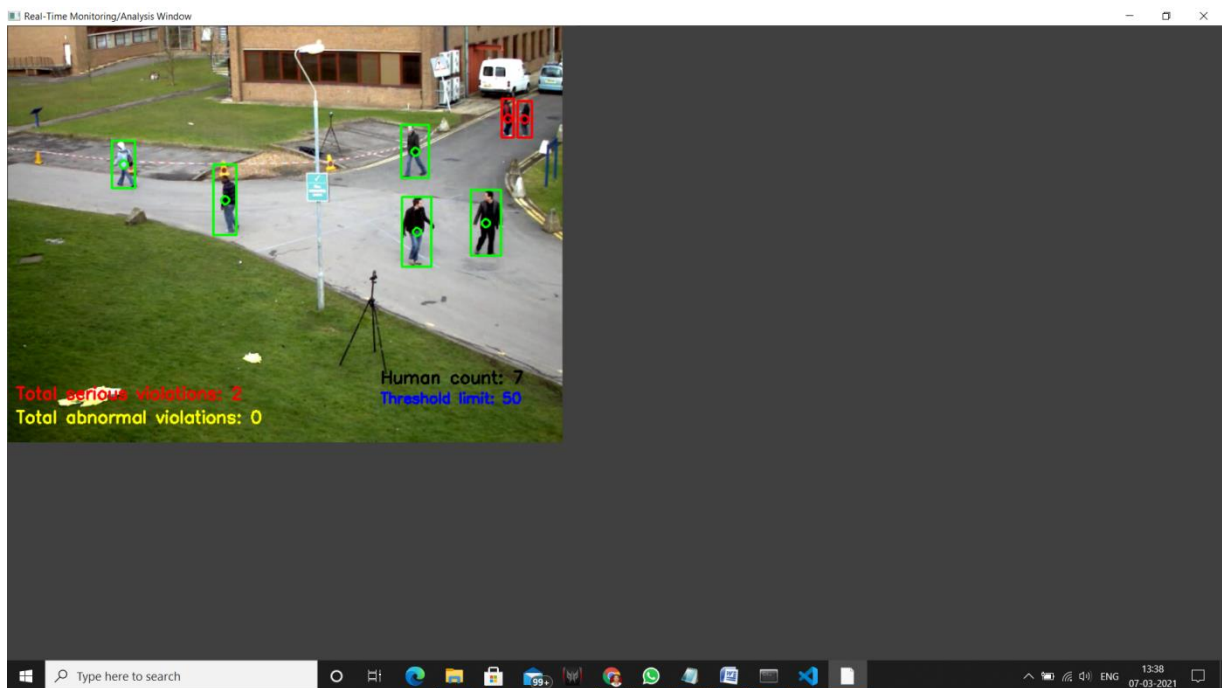
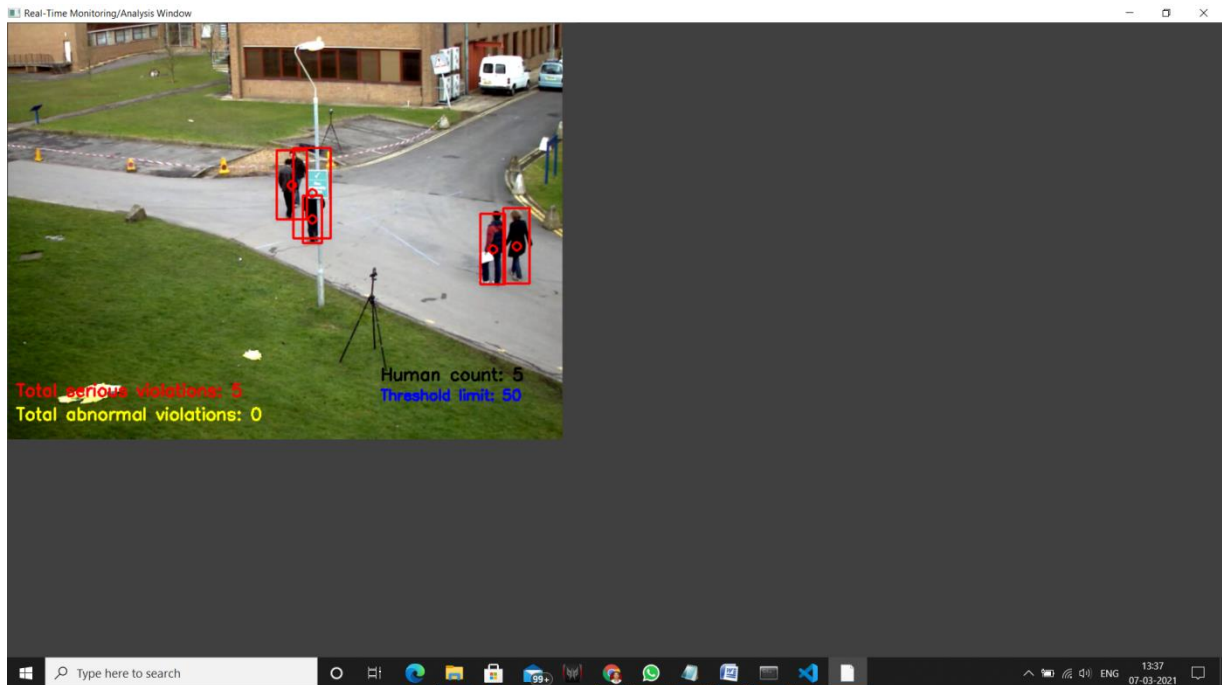
(base) D:\Social Distance2\Social-Distancing-Detection-in-Real-Time-main>python run.py -i mylib/videos/test.mp4

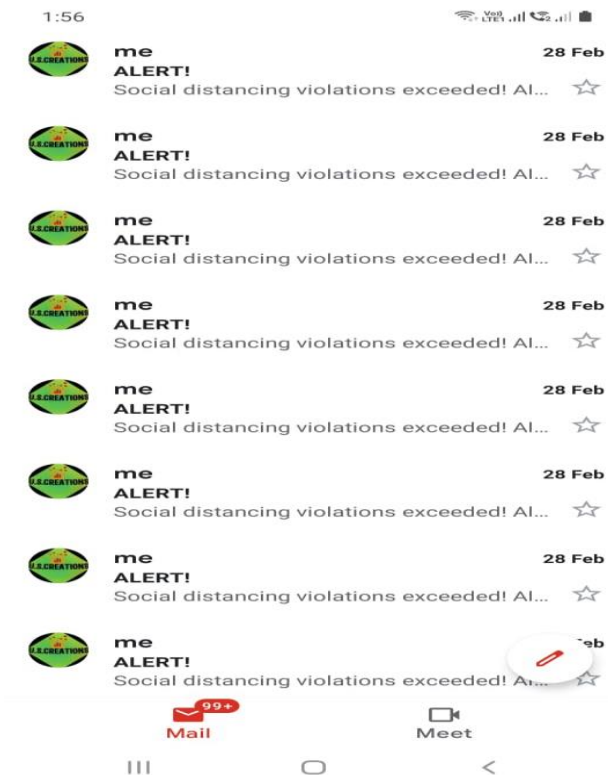
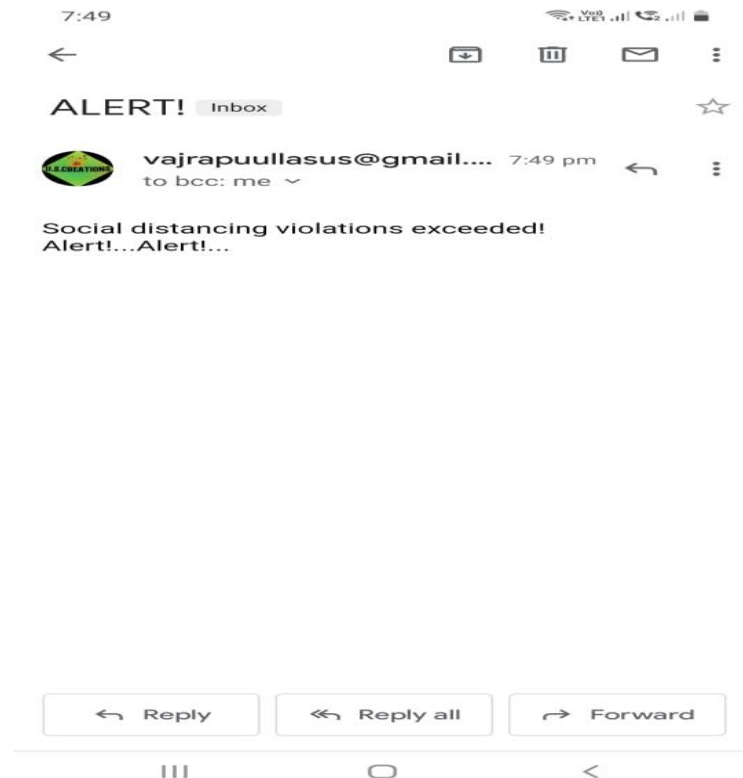
[INFO] Looking for GPU
[INFO] Starting the video..
=====
[INFO] Elapsed time: 0.00
[INFO] Approx. FPS: 0.00

(base) D:\Social Distance2\Social-Distancing-Detection-in-Real-Time-main>python run.py -i mylib/videos/garden.mp4

[INFO] Looking for GPU
[INFO] Starting the video..
[ WARN:0] global C:\Users\appveyor\AppData\Local\Temp\1\pip-req-build-oduouqig\opencv\modules\dnn\src\dnn.cpp (1442) cv:
:dnn::dnn_v20201117::Net::Impl::setUpNet DNN module was not built with CUDA backend; switching to CPU
```







C.PUBLICATION WITH PLAGIARISM REPORT

Social Distance Paper report 6.3.21

ORIGINALITY REPORT

6%

SIMILARITY INDEX

1%

INTERNET SOURCES

6%

PUBLICATIONS

3%

STUDENT PAPERS

PRIMARY SOURCES

1

"Missing Values Imputation using Feed Forward Neural Network", International Journal of Recent Technology and Engineering, 2019

Publication

2%

2

Afiq Harith Ahamad, Norliza Zaini, Mohd Fuad Abdul Latip. "Person Detection for Social Distancing and Safety Violation Alert based on Segmented ROI", 2020 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), 2020

Publication

1%

3

Joseph K. Paul, Tankala Yuvaraj, Karthikay Gundepudi. "Demonstrating Low-Cost Unmanned Aerial Vehicle for anti-Poaching", 2020 IEEE 17th India Council International Conference (INDICON), 2020

Publication

1%

4

Submitted to Korea National University of Transportation

Student Paper

1%