VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi -590018, Karnataka State, India

PROJECT REPORT

ON

"MONITORING COVID – 19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING"

Submitted in partial fulfillment of 8th Semester

BACHELOR OF ENGINEERING
IN
INFORMATION SCIENCE AND ENGINEERING

Submitted by:

AISHWARYA RAJU	**1SJ17IS001**
HARSHITH M	**1SJ17IS020**
HEMANTH M	**1SJ17IS022**
LEKHA DEVARAJ	**1SJ17IS030**

Under the guidance of:

Prof. Bhanumathi S
Assistant Professor
Dept. of ISE, SJCIT

SJC INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

For the academic year 2020-2021

|| Jai Sri Gurudev ||
Sri Adichunchanagiri Shikshana Trust

S.J.C INSTITUTE OF TECHNOLOGY

Information Science & Engineering Department
Chickballapur-562101

CERTIFICATE

This is to certify that the project work entitled "MONITORING COVID – 19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING" is a work being carried by **AISHWARYA RAJU** (1SJ17IS001), **HARSHITH M** (1SJ17IS020), **HEMANTH M** (1SJ17IS022) and **LEKHA DEVARAJ** (1SJ17IS030) in partial fulfilment for the award of Bachelor of Engineering in Information Science and Engineering in seventh semester of the Visvesvaraya Technological University, Belagavi during the year **2020-21**. It is certified that all corrections/suggestions indicated for the project have been incorporated in the report deposited in the department library. The project report has been approved as it satisfies the academic requirements in respect to Eighth Semester Project work prescribed for the Bachelor of Engineering degree.

Signature of the Guide	Signature of the HOD	Signature of the Principal
Bhanumathi S	Satheesh Chandra Reddy	Dr. G T Raju
Assistant Professor	Professor and HOD	Principal
Department of ISE, SJCIT	Department of ISE, SJCIT	SJCIT

PRINCIPAL
SJC Institute of Technology
CHICKBALLAPUR-562101.

External Viva:

Name of Examiners:	Signature with Date
1)	
2)	

DECLARATION

We **AISHWARYA RAJU, HARSHITH M, HEMANTH M, LEKHA DEVARAJ** Students of VIII semester B.E in Information Science and Engineering at S J C Institute of Technology, Chickballapur, hereby declare that this project work entitled **"MONITORING COVID-19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING"** has been carried out at by us under the supervision of our guide **Prof. BHANUMATHI S**, Assistant Professor, Dept. Of ISE, SJC Institute of Technology, Chickballapur and submitted in the partial fulfilment for the award of degree Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University, Belagavi during the academic year 2020-2021. We further declare that the report had not been submitted to any other university for the award of any other degree.

AISHWARYA RAJU (1SJ17IS001)
HARSHITH M (1SJ17IS020)
HEMANTH M (1SJ17IS022)
LEKHA DEVARAJ (1SJ17IS030)

ABSTRACT

With its devastating spread to more than 235 nations, the coronavirus disease 2019 (COVID-19) has caused a global crisis. The lack of active treatment drugs and immunity to COVID-19 makes the population more vulnerable. Because there are less vaccines available, social distancing is the only viable strategy for combating the epidemic. This paper provides a deep learning-based framework for automating the task of monitoring social separation using surveillance video. The suggested framework employs the YOLO v3 object detection model to distinguish peoples from the background, as well as the Deep Learning technique to track the recognised people using bounding boxes and issued IDs. The results of the YOLOv3 model are further compared with other models. Computed based three-dimensional feature space obtained by using the centroid coordinates and dimensions of the bounding box are also used. The violation index term is used to indicate the non-adoption of social distancing protocol.

ACKNOWLEDGEMENT

With Great pride we would like to convey our gratitude and appreciation to our alma-mater **"S.J.C Institute of Technology"** for giving us the required platform for doing the project on **"MONITORING COVID-19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING"** as per the V.T.U requirements, for the project.

We express my sincere thanks to **Dr. G T RAJU**, **Principal of SJCIT,** Chickballapur, for providing us with excellent infrastructure to complete this project.

We express wholehearted gratitude to **Prof. SATHEESH CHANDRA REDDY** who is the respectable **HOD of Information Science and Engineering Department**. We wish to acknowledge his help in making our task easy by providing us with his valuable help and encouragement.

It is our pleasure to thank our guide **Prof. BHANUMATHI S, Assistant Professor, Department of information Science and Engineering,** SJCIT for her guidance, encouragement and valuable suggestion from the beginning of the project work without which this project work would not have been accomplished. We are greatly indebted to her.

And last but not the least, We would be very pleased to express our heart full thanks to the teaching and non-teaching staff of the Department of Information Science and Engineering, SJCIT for their motivation and support.

We also thank all those who extended their support and co-operation while bringing out this project.

AISHWARYA RAJU (1SJ17IS001)
HARSHITH M (1SJ17IS020)
HEMANTH M (1SJ17IS022)
LEKHA DEVARAJ (1SJ17IS030)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

COVID-19 is a coronavirus-related disease that was first detected in late December 2020 in Wuhan, China. Corona Virus Disease (COVID-19) is an infectious disease caused by a coronavirus. Cold, cough, and fever are the most common symptoms of COVID-19 infection. They develop mild to moderate respiratory disease over time and recover without requiring specific treatment. There's a chance you're infected even if none of these symptoms appear. Older persons, as well as those with underlying medical conditions such as cardiovascular disease, diabetes, chronic respiratory disease, and cancer, are more likely to acquire a serious illness, which can be life-threatening. The COVID-19 virus is primarily transmitted through droplets of saliva or discharge from the nose when an infected person coughs or sneezes, hence respiratory etiquette is very important.

Knowing everything there is to know about the COVID-19 virus, the disease it produces, and how it spreads is the greatest strategy to avoid and slow down transmission. Hand-washing or using an alcohol-based rub frequently and not touching your face can help prevent the spread of infection. In the current situation, social distancing is touted as the most effective spread stopper, and all affected countries have agreed to implement it. This study aims to help support and prevent the coronavirus pandemic while minimising economic losses, as well as provide a method for detecting social distance among people gathered in public places.

The term "Social Distancing" refers to best practices in the direction of measures aimed at minimising or interrupting COVID-19 transmissions through a variety of techniques. Its goal is to reduce physical contact between potentially infected people and healthy people. According to WHO guidelines, persons should keep a space of at least 6 feet between them in order to practice social distancing. According to a recent study, social distancing is a critical containment mechanism that is necessary to avoid disease

spread since people with moderate or no symptoms may carry corona infection and infect others by chance.

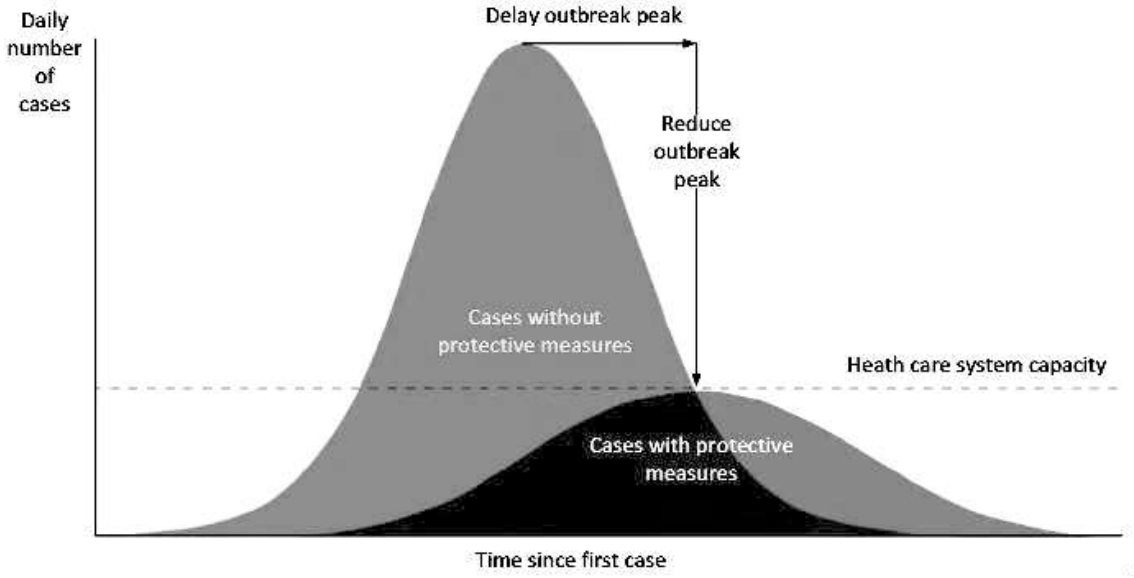

Figure 1.1.1: Effects of social distancing

The figure 1.1.1 indicates that proper social distancing is the best way to reduce the spread of infections. As can be observed in the figure, social distancing can reduce the peak number of infected cases to ensure that the number of patients does not exceed the public healthcare capacity. Moreover, social distancing also delays the outbreak peak so that there is more time to implement counter measures.

1.2 MOTIVATION

COVID-19 is a contagious disease which is caused by severe acute respiratory syndrome coronavirus. In December 2019, the first case of COVID-19 has been detected at Wuhan the capital of China's Hubei province. Now, this has been spread globally, resulting in the ongoing coronavirus pandemic.This pandemic has affected the countries socially and economically. Many companies are facing Recession and millions of people have lost their jobs across the globe. With this ongoing pandemic, many people are working from home and the public places such as Schools, Cinema halls, Parks are closed. Govt. Offices, Metro, Railways and other public transport system are being operated with minimum capacity maintaining Social Distancing incurring huge loss to government and people.

The absence of any active therapeutic drugs and the lack of immunity against COVID 19, increase the vulnerability of the population. Since there are shortage of vaccines, Social Distancing has been proven as an effective measure against the spread of the infectious COVID 19 disease.

However, individuals are not used to tracking the required 6 feet (2 meters) distance between themselves and their surroundings. Motivated by this notion, an active surveillance system capable of detecting distances between individuals and warning them can slow down the spread of the deadly disease. Furthermore, measuring social density in a Region Of Interest.

1.3 PROBLEM IDENTIFICATION AND DEFINITION

The Corona Virus is spreading around the world like wildfire. As there are no proved medications available for curing Corona Virus Disease, the only current preventive measure to curb the spread of disease is to practice “Social Distancing”. But the infrastructure of the public places is not enough or the general public is not used to keep an imaginary safety bubble around them. In practice, this means that avoiding close proximity to other people will aid in slowing the spread of infectious diseases.

Social distancing is one of the non-pharmaceutical infection control actions that can stop or slow down the spread of a highly contagious disease. The virus that causes COVID-19 is currently spreading easily from person-to-person. When a healthy person comes into contact with respiratory droplets from coughs or sneezes of an infected person, they are can catch the infection.

On the other hand, recording data and labelling individuals, who do not follow the measures will breach individual’s rights in free-societies. Hence an automatic warning System can help to maintain the required social distance of 6 feet or 2 meters between the individuals without breaching the privacy of individuals which does not record and store the data. COVID-19 may not be completely eliminated in the short term, but an automated system that can help monitoring and analysing social distancing measures can greatly benefit our society.

1.4 OBJECTIVES

The key goals of this work are as follows:

- To present a deep learning-based social distance monitoring framework using an overhead view perspective.
- To deploy pre-trained YOLOv3 for human detection and computing their bounding box centroid information. In addition, a transfer learning method is applied to enhance the performance of the model. The additional training is performed with overhead data set, and the newly trained layer is appended to the pre-trained model.
- In order to track the social distance between individuals, the Euclidean distance is used to approximate the distance between each pair of the centroid of the bounding box detected. In addition, a social distance violation threshold is specified using a pixel to distance estimation.
- Utilising a centroid tracking algorithm to keep track of the person who violates the social distance threshold.
- To assess the performance of pre-trained YOLOv3 by evaluating it on an overhead data set. The output of the detection framework is assessed with and without the transfer learning. Furthermore, the model performance is also compared with other deep learning models.

CHAPTER 2

LITERATURE REVIEW

After the rise of the COVID-19 pandemic since late December 2019, Social distancing is deemed to be an utmost reliable practice to prevent the contagious virus transmission and opted as standard practice on January 23, 2020. During one month, the number of cases rises exceptionally, with two thousand to four thousand new confirmed cases reported per day in the first week of February 2020. Later, there has been a sign of relief for the first time for five successive days up to March 23, 2020, with no new confirmed cases. This is because of the social distance practice initiated in China and, latterly, adopted by worldwide to control COVID-19. The study revealed that moderate stages of exercise could be allowed for evading a large outbreak.

So far, many countries have used technology-based solutions to overcome the pandemic loss. Several developed countries are employing GPS technology to monitor the movements of the infected and suspected individuals. The survey provides the different emerging technologies, including Wi-fi, Bluetooth, smartphones, and GPS, positioning, computer vision, and deep learning that can play a crucial role in several practical social distancing scenarios. Some researchers utilise drones and other surveillance cameras to detect crowd gatherings.

Convolutional neural networks with many layers have recently been shown to achieve excellent results on many high-level tasks such as image classification, object detection and more recently also semantic segmentation. Particularly for semantic segmentation, a two-stage procedure is often employed. Hereby, convolutional networks are trained to provide good local pixel-wise features for the second step being traditionally a more global graphical model.

2.1 Anastasios Doulamis, Nikolaos Doulamis, Klimis Ntalianis, and Stefanos Kollias[1] proposed an unsupervised video object (VO) segmentation and tracking algorithm based on an adaptable neural-network architecture. The proposed scheme comprises: 1) a VO tracking module and 2) an initial VO estimation module. Object

tracking is handled as a classification problem and implemented through an adaptive network classifier, which provides better results compared to conventional motion-based tracking algorithms. Network adaptation is accomplished through an efficient and cost effective weight updating algorithm, providing a minimum degradation of the previous network knowledge and taking into account the current content conditions. A retraining set is constructed and used for this purpose based on initial VO estimation results. Two different scenarios are investigated. The first concerns extraction of human entities in video conferencing applications, while the second exploits depth information to identify generic VOs in stereoscopic video sequences. Human face body detection based on Gaussian distributions is accomplished in the first scenario, while segmentation fusion is obtained using colour and depth information in the second scenario. A decision mechanism is also incorporated to detect time instances for weight updating. Experimental results and comparisons indicate the good performance of the proposed scheme even in sequences with complicated content (object bending, occlusion).

2.2 Bharath Hariharan, Pablo Arbel´aez, Ross Girshick, and Jitendra Malik[2] detects all instances of a category in an image and, for each instance, mark the pixels that belong to it. They call this task Simultaneous Detection and Segmentation (SDS). Unlike classical bounding box detection, SDS requires segmentation and not just a box. Unlike classical semantic segmentation, we require individual object instances. They build on recent work that uses convolutional neural networks to classify category independent region proposals (R-CNN), introducing a novel architecture tailored for SDS. They then use category-specific, topdown figure-ground predictions to refine our bottom-up proposals. They show a 7 point boost (16% relative) over our baselines on SDS, a 5 point boost (10% relative) over state-of-the-art on semantic segmentation, and state-of-the-art performance in object detection.

2.3 Camille Couprie, Clement Farabet, Laurent Najman and Yann LeCun[3] addresses multi-class segmentation of indoor scenes with RGB-D inputs. While this area of research has gained much attention recently, most works still rely on hand-crafted features. In contrast, they apply a multi-scale convolutional network to learn features directly from the images and the depth information.

They obtain state-of-the-art on the NYU-v2 depth dataset with an accuracy of 64.5%. They illustrate the labelling of indoor scenes in videos sequences that could be processed in real-time using appropriate hardware such as an FPGA.

2.4 C.P. Town and D. Sinclair[4] demonstrates an approach to content based image retrieval founded on the semantically meaningful labelling of images by high level visual categories. The image labelling is achieved by means of a set of trained neural network classifiers which map segmented image region descriptors onto semantic class membership terms. It is argued that the semantic terms give a good estimate of the salient features which are important for discrimination in image retrieval. Furthermore, it is shown that the choice of visual categories such as grass or sky which mirror high level human perception allows the implementation of intuitive and versatile query composition interfaces and a variety of image similarity metrics
for content based retrieval.

2.5 Christian Szegedy, Alexander Toshev and Dumitru Erhan[5] shown outstanding performance on image classification tasks. In this paper we go one step further and address the problem of object detection using DNNs, that is not only classifying but also precisely localising objects of various classes. They present a simple and yet powerful formulation of object detection as a regression problem to object bounding box masks. We define a multi-scale inference procedure which is able to produce high-resolution object detections at a low cost by a few network applications. State-of-the-art performance of the approach is shown on Pascal VOC.

2.6 Cl´ement Farabet, Camille Couprie, Laurent Najman, Yann Lecun[6] propose a method that uses a multi-scale convolutional network trained from raw pixels to extract dense feature vectors that encode regions of multiple sizes centred on each pixel. The method alleviates the need for engineered features, and produces a powerful representation that captures texture, shape and contextual information. They report results using multiple post-processing methods to produce the final labelling. Among those, they propose a technique to automatically retrieve, from a pool of segmentation components, an optimal set of components that best explain the scene; these components are arbitrary,

e.g. they can be taken from a segmentation tree, or from any family of over-segmentations. The system yields record accuracies on the Sift Flow Dataset (33 classes) and the Barcelona Dataset (170 classes) and near-record accuracy on Stanford Background Dataset (8 classes), while being an order of magnitude faster than competing approaches, producing a 320 × 240 image labelling in less than a second, including feature extraction.

2.7 Hongsheng Li, Rui Zhao, and Xiaogang Wang[7] present highly efficient algorithms for performing forward and backward propagation of Convolutional Neural Network (CNN) for pixel wise classification on images. For pixel wise classification tasks, such as image segmentation and object detection, surrounding image patches are fed into CNN for predicting the classes of centred pixels via forward propagation and for updating CNN parameters via backward propagation. However, forward and backward propagation was originally designed for whole-image classification. Directly applying it to pixel wise classification in a patch-by-patch scanning manner is extremely inefficient, because surrounding patches of pixels have large overlaps, which lead to a lot of redundant computation. The proposed algorithms eliminate all the redundant computation in convolution and pooling on images by introducing novel d-regularly sparse kernels. It generates exactly the same results as those by patch-by-patch scanning. Convolution and pooling operations with such kernels are able to continuously access memory and can run efficiently on GPUs. A fraction of patches of interest can be chosen from each training image for backward propagation by applying a mask to the error map at the last CNN layer. Its computation complexity is constant with respect to the number of patches sampled from the image. Experiments have shown that our proposed algorithms speed up commonly used patch-by-patch scanning over 1500 times in both forward and backward propagation. The speedup increases with the sizes of images and patches. Source code of GPU implementation is ready to be released to the public.

2.8 Jonathan Long, Evan Shelhamer and Trevor Darrell[8] show that convolutional networks themselves, trained end-to-end, pixels-to-pixels, exceed the state-of-the-art in semantic segmentation. Their key insight is to build “fully convolutional” networks that efficient inference and learning. They define and detail the space of fully convolutional

networks,explain their application to spatially dense prediction tasks, and draw connections to prior models. They adapt contemporary classification networks (AlexNet, the VGG net, and GoogLeNet) into fully convolutional networks and transfer their learned representations by fine-tuning to the segmentation task. They then define a novel architecture that combines semantic information from a deep, coarse layer with appearance information from a shallow, fine layer to produce accurate and detailed segmentations. Their fully convolutional network achieves state-of-the art segmentation of PASCAL VOC (20% relative improvement to 62.2% mean IU on 2012), NYUDv2, and SIFT Flow, while inference takes less than one fifth of a second for a typical image.

CHAPTER 3

SYSTEM REQUIREMENT SPECIFICATION

3.2 Functional Requirements

When input is given to the system, functions like calculations, manipulation and data processing is done.

- System must collect data from camera or stored file and support accurate and continuous real time data collection.
- System must analyze / process data.
- Store and monitor the data efficiently.
- System must be accessible at anytime and at anywhere.
- System must be field-configurable and easy to deploy.

Hardware Requirements

1. RAM - 4 GB
2. Hard Disk - 250 GB
3. Processor - Intel i3 or higher
4. System bus - 64-bit
5. HDD/SSD - 500GB
6. Input Devices - Mouse, Keyboard
7. Output Devices – Laptop

Software Requirements

1. Operating System - Windows 7 or higher
2. Tools – OpenCV
3. Platform - Pycharm
4. Coding Language - Python

3.2 Non Functional Requirements

- Security: It is the feature of the system which ensures that system must be protected from the unintentional or malignant harm.

- User-friendly: The graphical user interface (GUI) is user-friendly.
- Usability: Usability determines hoe easy it is to learn and use the system.
- Availability: It means hoe long the system is available for the users and for how long the system will be operational.
- Reliability: Reliability determines how often the system fails. The system is completely tested for robustness before the deployment. The module developed thus maintains data consistency.

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

4.1 General Architecture of YOLOv3

The architecture shown in Fig 6.1 is trained using an overhead data set. For that purpose, a transfer learning approach is adopted, that enhance the efficiency of the model. With transfer learning, the model is additionally trained without dropping the valuable information of the existing model. Further, the additional overhead data set trained layer is appended with the existing architecture. In this way, the model takes advantage of the pre-trained and newly trained information, and both detection results are further deliver better and faster detection results.

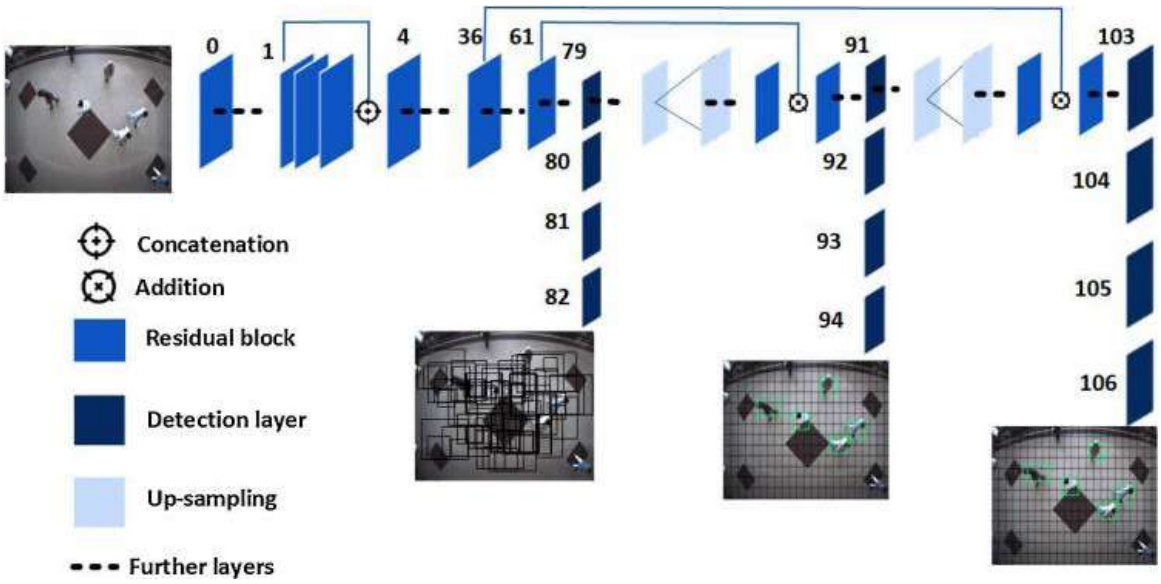

Figure 4.1.1 General Architecture of YOLO V3

The architecture shown in Figure 4.1.1 used a single-stage network for the entire input image to predict the bounding box and class probability of detected objects. For feature extraction, the architecture utilises convolution layers, and for class prediction, fully connected layers are used. During human identification, as seen in Figure 4.1.1, the input frame is divided into a region of S×S, also called grid cells. These cells are related to bounding box estimation and class probabilities. It predicts the probability of whether the centre of the person bounding box is in the grid cell or not.

The most current of the main versions is the third iteration of the approach, namely YOLOv3. In each version improvements have been made over the previous one. The initial version proposed the general architecture, after which the second variation improved accuracy significantly while making it faster. YOLOv3 refined the design further by using tricks, such as multi-scale prediction and bounding box prediction through the use of logistic regression. While the accuracy increased dramatically with this version, it traded off against speed which reduced from 45 to 30 frames per second.

YOLOv3 uses a variant of Dark-net, a framework to train neural networks, which originally has 53 layers. For the detection task another 53 layers are stacked onto it, accumulating to a total of a 106-layer fully convolutional architecture. This explains the reduction in speed in comparison with the second version, which only has 30 layers.

In the convolutional layers, kernels of shape 1x1 are applied on feature maps of three different sizes at three different places in the network. The algorithm makes predictions at three scales, given by down sampling the dimensions of the image by a stride of 32, 16, 8 respectively. Down sampling, the reduction in spatial resolution while keeping the same image representation, is done to reduce the size of the data. Every scale uses three anchor bounding boxes per layer. The three largest boxes for the first scale, three medium ones for the second scale and the three smallest for the last scale. This way each layer excels in detecting large, medium or small objects.

4.2 Camera Perspective

As the input video may be taken from an arbitrary perspective view, the first step is to transform perspective of view to a bird's-eye (top-down) view. As the input frames are monocular (taken from a single camera), the simplest transformation method involves selecting four points in the perspective view which define ROI where we want to monitor social distancing and mapping the m to the corners of a rectangle in the bird's-eye view.

Also, these points should form parallel lines in real world if seen from above (bird's eye view). This assumes that every person is standing on the same flat ground plane. This top view or bird eye view has the property that points are distributed

uniformly horizontally and vertically (scale for horizontal and vertical direction will be different). From this mapping, we can derive a transformation that can be applied to the entire perspective image. First four points will define ROI where we want to monitor social distancing. Next 3 points will define 180 cm (unit length) distance in horizontal and vertical direction and those should form parallel lines with ROI.

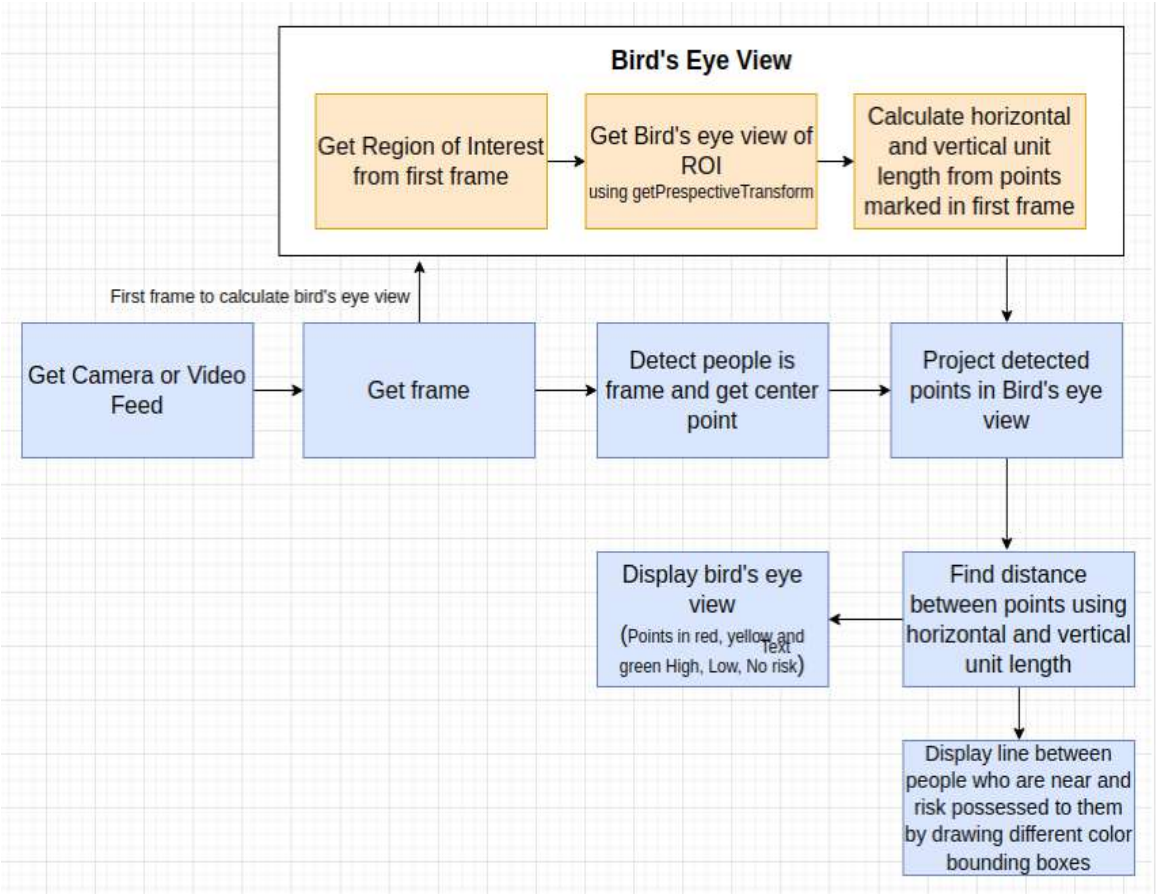

Figure 4.2.1 Bird's Eye View

4.3 Social Distancing Monitoring

Researchers use a frontal or side perspective for social distance monitoring. In this work, a deep learning-based social distance monitoring framework using an overhead perspective has been introduced. The flow diagram of the framework is shown in Figure 4.4.1 The recorded overhead data set are split into training and testing sets. A deep learning-based detection paradigm is used to detect individuals in sequences. There are a variety of object detection models available, such as R-CNN(Regional Convolution Neural Network), Fast R-CNN, Faster R-CNN and SSD(Single Shot MultiBox Defender) etc. Due to the best performance results for generic object detection, in this work,

YOLOv3 is used. The model used single-stage network architecture to estimate the bounding boxes and class probabilities.

After detection, the bounding box information, mainly centroid information, is used to compute each bounding box centroid distance. We used Euclidean distance and calculated the distance between each detected bounding box of peoples. Following computing centroid distance, a predefined threshold is used to check either the distance among any two bounding box centroids is less than the configured number of pixels or not. If two people are close to each other and the distance value violates the minimum social distance threshold. The bounding box information is stored in a violation set, as seen in Figure 4.4.1, and the colour of the bounding box is updated/changed to red. A centroid tracking algorithm is adopted for tracking so that it helps in tracking of those people who violate/breach the social distancing threshold. At the output, the model displays the information about the total number of social distancing violations along with detected people bounding boxes and centroids.

4.4 Flow Diagram

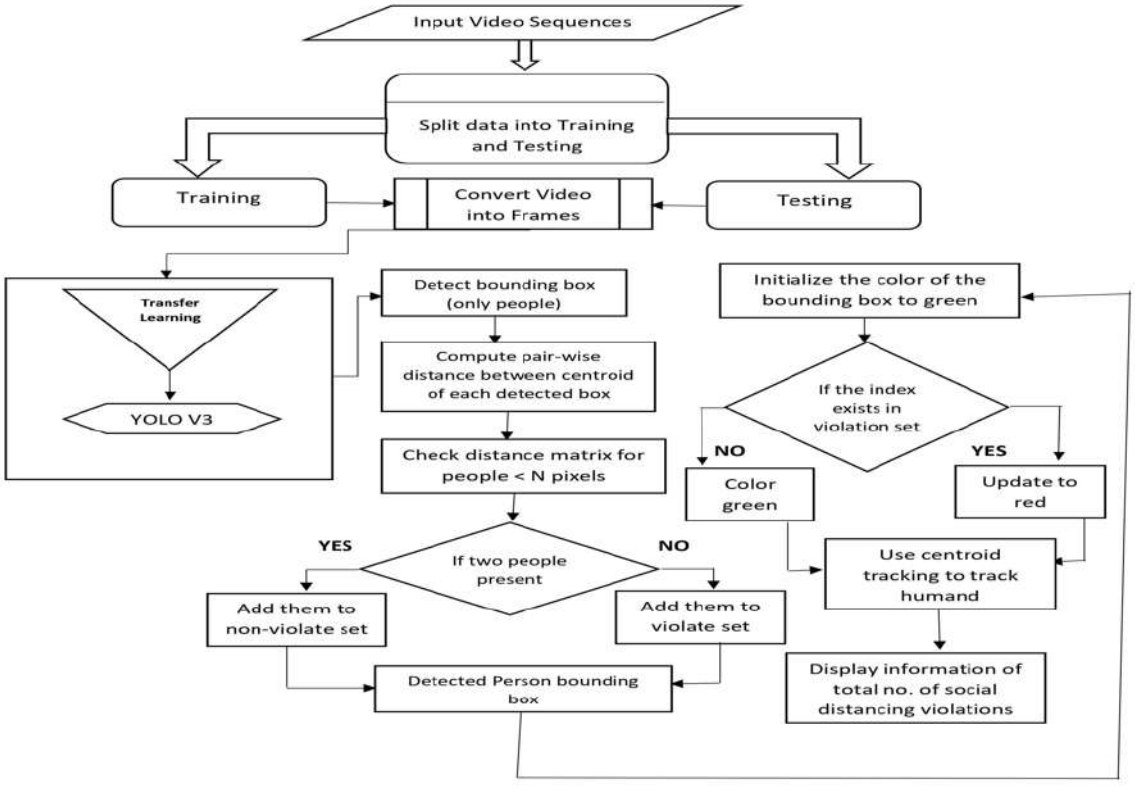

Figure 4.4.1 Flow Diagram of Social Distance Monitoring

The Figure 4.4.1 depicts the flow diagram of our model. At first input to the model is video sequences then the video sequences data set are split into training and testing sets. Then the video is converted into frames which is nothing but an image. And YOLO V3 algorithm is applied to the sequence of frames in order to identify the persons. The model used single-stage network architecture to estimate the bounding boxes and class probabilities.

After detection, the bounding box information, mainly centroid information, is used to compute each bounding box centroid distance. We used Euclidean distance and calculated the distance between each detected bounding box of peoples. Following computing centroid distance, a predefined threshold is used to check either the distance among any two bounding box centroids is less than the configured number of pixels or not. If two people are close to each other and the distance value violates the minimum social distance threshold then The bounding box information is stored in a violation set, if not the bounding box information is stored in non violation set. Then in order to easily identify between two sets the violated set is updated into Red colour and the non violation set is updated into green colour.

Finally a centroid tracking algorithm is adopted for tracking so that it helps in tracking of those people who violate/breach the social distancing threshold. At the output, the model displays the information about the total number of social distancing violations along with detected people bounding boxes and centroids.

In this work, YOLOv3 is used for human detection as it improves predictive accuracy, particularly for small-scale objects. The main advantage is that it has adjusted network structure for multi-scale object detection.

4.5 Distance Calculation

Now we have bounding box for each person in the frame. We need to estimate person location in frame. i.e. we can take bottom centre point of bounding box as person location in frame. Then we estimate (x, y) location in bird’s eye view by applying transformation to the bottom centre point of each person’s bounding box, resulting in the position in the bird’s eye view. Last step is to compute the bird’s eye view distance between every pair of people and scale the distances.

4.6 Working

Running the program will give you frame (first frame) where you need to draw ROI and distance scale. To get ROI and distance scale points from first frame Code to transform perspective to Bird's eye view (Top view) and to calculate horizontal and vertical 180 cm distance in Bird's eye view ROI and Scale points' selection for first frame. The second step to detect pedestrians and draw a bounding box around each pedestrian. To detect humans in video and get bounding box details. Now we have bounding box for each person in the frame. We need to estimate person location in frame. i.e, we can take bottom centre point of bounding box as person location in frame. Then we estimate (x, y) location in bird's eye view by applying transformation to the bottom centre point of each person's bounding box, resulting in the position in the bird's eye view. To calculate bottom centre point for all bounding boxes and projecting those points in Bird's eye view. Last step is to compute the bird's eye view distance between every pair of people (Point) and scale the distances by the scaling factor in horizontal and vertical direction estimated from calibration.

Lastly, we can draw Bird's Eye View for region of interest (ROI) and draw bounding boxes according to risk factor for humans in a frame and draw lines between boxes according to risk factor between two humans. Red, Yellow, Green points represent risk to human in Bird's eye view. Red: High Risk, Yellow: Low Risk and Green: No Risk. Red, Yellow lines between two humans in output tell they are violating social distancing rules.

CHAPTER 5

METHODOLOGY

Here we used an overhead view to provide an effective framework for social distance monitoring. The overhead perspective offers a better field of view and overcomes the issues of occlusion, thereby playing a key role in social distance monitoring to compute the distance between peoples. It might help overcome computation, communication load, energy consumption, human resource, and installation costs.

This work aims to present a deep learning-based social distance monitoring framework for the public campus environment from an overhead perspective. A deep learning model, i.e., YOLOv3 (You Only Look Once), is applied for human detection. The current model (pre-trained on frontal or normal view data sets) is initially tested on the overhead data set. Transfer learning is also used to improve the efficiency of the detection model. To the best of our knowledge, this work could be considered as the first effort to use an overhead view perspective to monitor social distance with transfer learning.

The detection model detects humans and gives bounding box information. After human detection, the Euclidean distance between each detected centroid pair is computed using the detected bounding box and its centroid information. A predefined minimum social distance violation threshold is specified using pixel to distance assumptions. To check, either the calculated distance comes under the violation set or not, the estimated information is matched with the violation threshold. The bounding box's colour is formerly initialised as green; if the bounding box comes under the violation set, its colour is updated to red. In addition, the centroid tracking algorithm is used to track a person who violated the social distancing threshold.

Our system focuses on how to identify the person on image/video stream whether the social distancing is maintained or not with the help of computer vision and deep learning algorithm by using OpenCV.

To control the spread of the virus, India has been following mainly three steps aggressive testing, contact tracking, social distancing. Out of these social distancing is arguably the most effective non-pharmaceutical way to control the spread of the virus.

Since it is very hard to maintain the safety bubble in public places an automatic warning system can help individuals to maintain the required distance of 6 meters or 2 feet. This might alert the individual if they cross their bubble.

Object detection and tracking are the preliminary requirements to determine if people are complying with social distance or not. Detection is the ability to detect the object in any given image where object classification identifies the class the image belongs to. Since there are various approaches for object detection, we are discussing object detection using deep learning.

Even this is a class of machine learning algorithm where it uses a multi-layer approach to extract high-level feature from the data that is being provided. It does not require any feature to be provided manually for classification instead it tries to transform data into an abstract representation. It is also influenced by artificial neural networks present in our brain most deep learning implement neural networks to achieve the results. All the deep learning models require a huge amount of computation power and a large volume of labelled data to learn from the features.

Some of the deep learning methods include RCNN, faster RCNN, YOLO.

Open CV- It is a huge source library for computer vision, machine learning, and image processing. By using OpenCV we can pre-process image videos to identify objects, faces, or even handwriting of a human. When it is integrated with libraries such as NumPy python is capable of processing the OpenCV array structure for analysis.

YOLO- You Only Look Once it focuses on the entire image as a whole and predicts the bounding boxes and then calculates the class probability to label the boxes. It is called YOLO unified as it unifies the object detection and classification model together as a single detection network.

During object detection to detect all people (and only people) in a video stream. Particularly in the detection phase, the use of a computationally more expensive object

tracker takes place that detects if any new objects have entered the frame. For all the detected objects, upgrading of object tracker with the new bounding box coordinates will take place. Similarly in the tracking phase, for all the detected objects an object tracker will be created to track the object moving around the frame. Therefore, the object tracker must be faster and more efficient than the detector. This process continues tracking until it reaches the N-th frame and the entire process then repeats. The next step is to pass a set of bounding boxes to the detected objects in the frame and quantification of their corresponding centroids will transpire by calculating the midpoint of a bounding box that measures the centroid of each bounding box for the detected person. The equation can be stated as

$$M(p,q)=((pmin + pmax)/2 , (qmin+qmax)/2)$$

The minimum and maximum values for the correspondent width pmin,pmax and height qmin ,qmax of the each bounding box are used to calculate the midpoint of the bounding box. Calculate the Euclidean distance between the latest centroids and subsisting centroids. Estimate the distance, m1(pmin , qmin) and m2(pmax , qmax) between each of the detected objects in the frame.The Euclidean distance equation can be stated as

$$ED(m1,m2)=(pmax - pmin)2+(qmax -qmin)2$$

The centroid tracker has been designed to equate centroids that deprecate their relevant Euclidean distances. Enrolling the latest objects eventuate if any latest object does not match with the current object. If any object has been found lost or left the frame, deregistering that particular object falls out.

Non-maxima suppression is the terminal step of the object detection algorithms and it is used to select the most convenient bounding box for the object. The objects in the frame can be of various proportions and forms, and to encapsulate each of these exquisitely, the object detection algorithms create numerous bounding boxes for each object but, for each object detected we must have a unique single bounding box. To choose the finest bounding box, from the numerous predicted bounding boxes, these object detection algorithms use non-max suppression. This approach is used to repress the unlikely bounding boxes and accepts the accurate ones.

It has the following features:

1. Detect humans in the frame.
2. Calculates the distance between every human who is detected in the frame.
3. Shows how many people are following the social distancing criteria.

The main purpose of this system is to process live or captured video footage for person detection and further processing for social distancing or safety violation. So, the process starts with reading the frames of a video feed one by one.

5.1 Steps to Build Social Distancing Detector

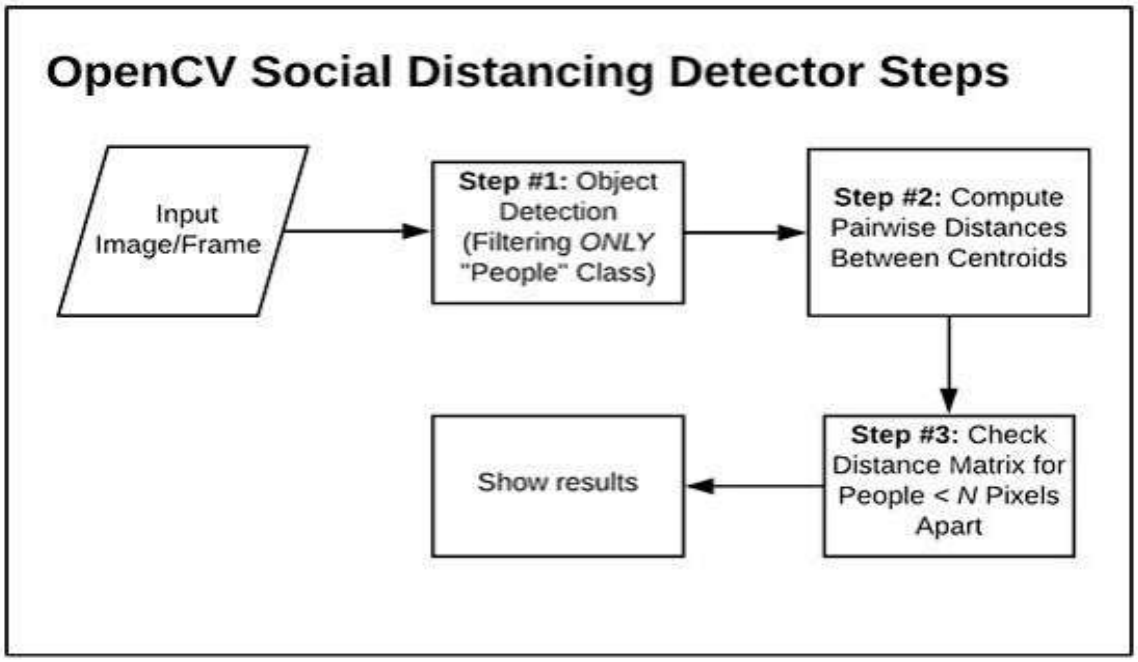

Figure 5.1.1 Steps to Build Detector

The steps to build a social distancing detector include:

1. Apply object detection to detect all people (and *only* people) in a video stream.
2. Compute the pairwise distances between all detected people.
3. Based on these distances, check to see if any two people are less than *N* pixels apart

5.2 Apply object detection to detect all people (and only people) in a video stream.

Detecting an object which is in motion, incorporates two stages:

- Object Detection
- Object Classification

When we apply object detection we are determining where in an image/frame an object is. Examples includes deep learning-based object detectors such as Faster R-CNNs, YOLO, and Single Shot Detectors (SSDs). An object tracker, on the other hand, will accept the input (x, y) coordinates of where an object is in an image and will:
Assign a unique ID to that particular object Track the object as it moves around a video stream, predicting the new object location in the next frame based on various attributes of the frame Combination of both object detection and object tracking Object trackers will combine the concept of object detection and object tracking into a single algorithm, typically divided into two phases:

- **Phase 1 — Detecting:**

During the detection phase we are running our computationally more expensive object tracker to detect if new objects have entered our view, and see if we can find objects that were “lost” during the tracking phase. For each detected object we create or update an object tracker with the new bounding box coordinates.

- **Phase 2 — Tracking:**

When we are not in the “detecting” phase we are in the “tracking” phase. For each of us detected objects, we create an object tracker to track the object as it moves around the frame. Our object tracker should be faster and more efficient than the object detector. We’ll continue tracking until we’ve reached the N-th frame and then re-run our object detector. The entire process then repeats.

5.3 Compute the pairwise distances between all detected people

STEP – 1

We accept a set of bounding boxes and compute their corresponding centroids. Calculate the center point of a bounding box to measure the center point,c(x,y).

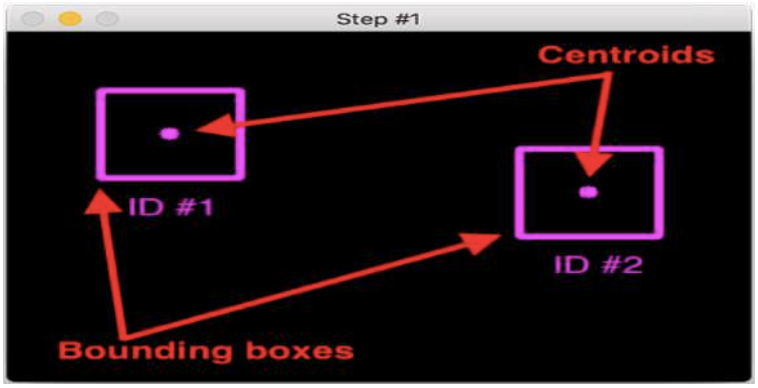

Figure 5.3.1 Bounding Box Calculation for Centroid

STEP – 2

We compute the Euclidean distance between any new centroids (yellow) and existing centroids (purple).

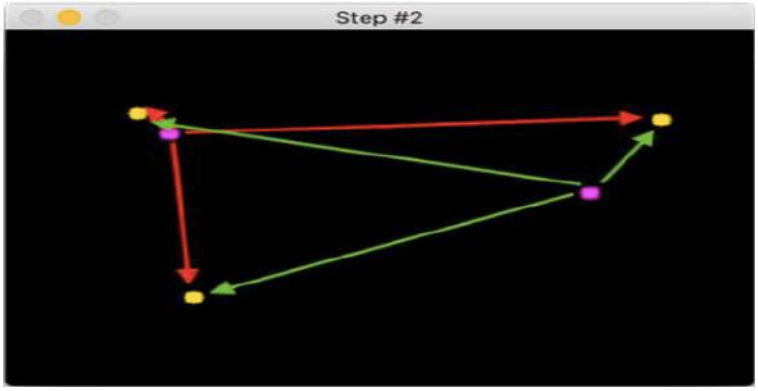

Figure 5.3.2 Computing Euclidean Distance

STEP – 3 Centroid tracker has chosen to associate centroids that minimise their respective Euclidean distances.

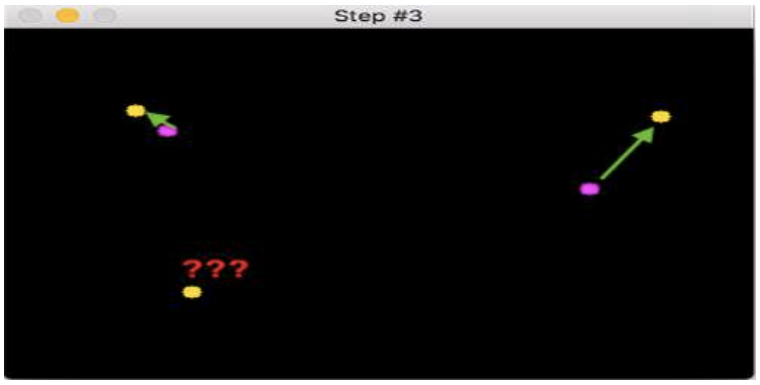

Figure 5.3.3 Associate Centroid

STEP – 4

Registering new objects: we have a new object that wasn't matched with an existing object, so it is registered as object ID #3.

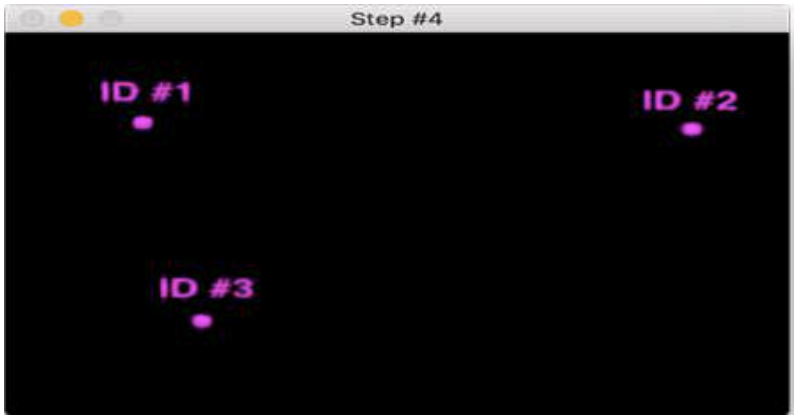

Figure 5.3.4 Register new Objects

STEP – 5

In the event that an object has been lost or has left the field of view, we can simply deregister the object.

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

The centre point of the bounding boxes is taken to determine between two different locations of the bounding boxes. After getting the centre points value, the algorithm will calculate if the distance is lower or higher than 300 pixels.

Figure shows the green bounding boxes for each possible detected person when the distance for each possible detected person bounding boxes is longer than the default value of social distance range. The experimental default pixels value shown in Figure is 300 pixels.

In contrast, when the safe social distance is violated, the bounding boxes of affected persons are changed to red indicating that the distance is below the minimum value for a safe distance. In this way, the difference between those who violate the safe distance and those who do not violate the minimum safe distance required can be seen.

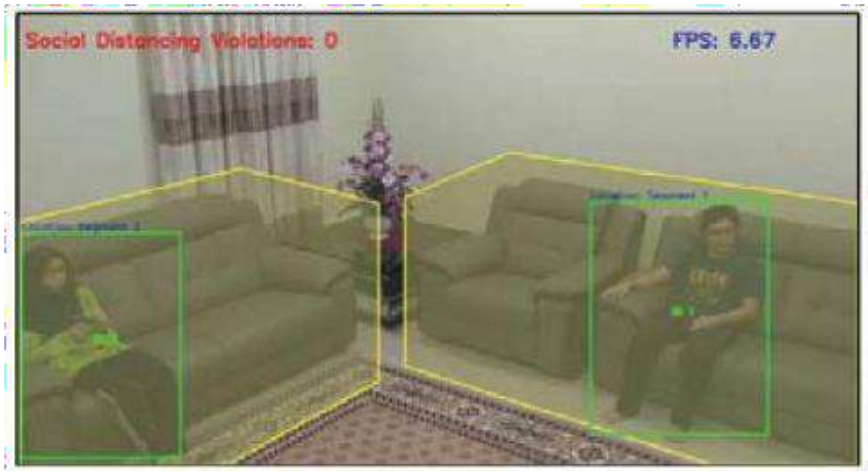

Figure 5.4.1 Green Bounding box

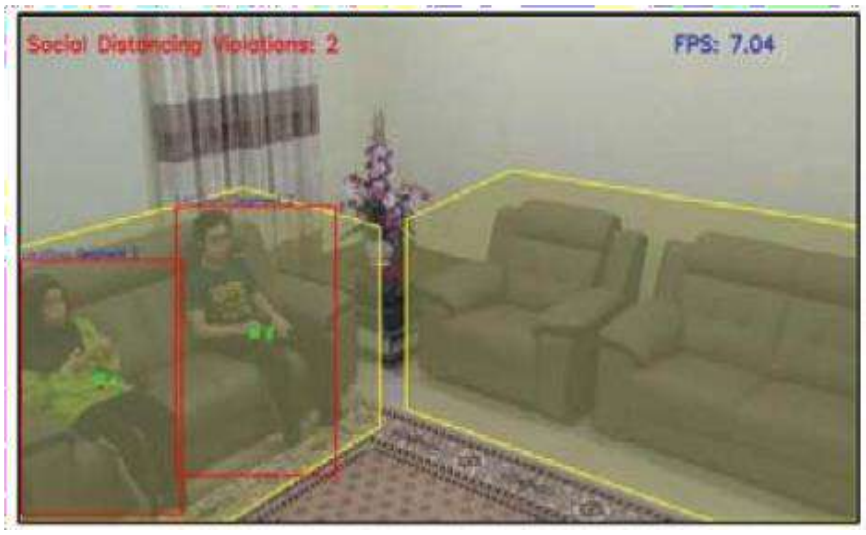

Figure 5.4.2 Red Bounding Box

CHAPTER 6
IMPLEMENTATION

Implementation is the realisation of an application, or execution of a plan, idea, model, design, specification, standard, algorithm or policy.

6.1 Implementation Using Python Language

Python can run equally on different platforms such as Windows, Linux, Unix etc. So, it is a portable language. It supports object oriented language and concepts of classes and objects come into existence. Graphical user interfaces can be developed using Python. It has a large library which provides rich set of module and functions for rapid application development.

6.2 Libraries and Packages

OpenCV

NumPy

SciPy

Pandas

Imutils

Argparse

Scikit-learn

6.3 Libraries and Packages Description

6.3.1 OpenCV

OpenCV is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel. The library is cross-platform and free for use under the open-source Apache 2 License. OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. All the OpenCV array 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. Earlier, there was only cv . Later, OpenCV came with both cv and cv2 . Now, present in the latest releases, there is only the cv2 module, and cv is a subclass inside cv2 . You need to call import cv2.cv as cv to access it.

6.3.2 NumPy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. NumPy stands for Numerical Python. NumPy is the fundamental package for scientific computing in Python. NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.

6.3.3 SciPy

SciPy is a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimisation, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering. SciPy in Python is an open-source library used for solving mathematical, scientific, engineering, and technical problems. It allows users to manipulate the data and visualise the data using a wide range of high-level Python commands. SciPy is built on the Python NumPy extension. SciPy is also pronounced as "Sigh Pi."

6.3.4 Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. Pandas is the most popular python library that is used for data analysis.

It provides highly optimized performance with back-end source code is purely written in C or Python. We can analyze data in pandas with Series. Pandas stands for “Python Data Analysis Library ”. According to the Wikipedia page on Pandas, “the name is derived from the term “panel data”, an econometrics term for multidimensional structured data sets.”

6.3.5 Imutils

Imutils are a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and both Python 2.7 and Python 3.

6.3.6 Argparse

Argparse is the Parser for command-line options, arguments and sub-commands. ... The argparse module makes it easy to write user-friendly command-line interfaces. The program defines what arguments it requires, and argparse will figure out how to parse those out of sys.argv. Using argparse means the doesn't need to go into the code and make changes to the script. Giving the user the ability to enter command line arguments provides flexibility.

6.3.7 Scikit-learn

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines. Scikit-learn is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy. Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.

6.4 PSEUDOCODE FOR IMAGE DETECTION

Step 1: Grab the dimensions of the frame and initialize the list of results

Step 2: Construct a blob from the input frame and then perform a forward pass of the bounding boxes

Step 3: Initialize our lists of detected bounding boxes, centroids, and confidences respectively

Step 4: Loop over each of the layer outputs

Step 4.1: Loop over each of the detections

Step 4.2: Extract the class ID and confidence (i.e., probability) of the current object detection

Step 4.3: Filter detections by (1) ensuring that the object detected was a person and (2) that the minimum confidence is meet

Step 4.4: Scale the bounding box coordinates back relative to the size of the image, keeping in mind that YOLO actually returns the center (x, y)-coordinates of the bounding box followed by the boxes' width and height.

Step 4.5: Use the center (x, y)-coordinates to derive the top and left corner of the bounding box

Step 4.6: Update our list of bounding box coordinates, centroids, and confidences.

Step 5: Apply non-maxima suppression to suppress weak, overlapping bounding boxes

Step 6: Ensure at least one detection exists

Step 6.1: Loop over the indexes we are keeping

Step 6.2: Extract the bounding box coordinates

Step 6.3: Update our results list to consist of the person prediction probability, bounding box coordinates, and the centroid.

Step 7: Return the list of results.

6.5 MODULES

The modules have been broadly classified into three parts –

1. Input Stream
2. Features Extraction
3. Calculation and Result

6.5.1 Input Stream with Pseudocode

The first step in this module is for the user to either select the live stream option or the stored video option.Based on this selection the live feed frames are read or the frames of the existing video are read. Once the frames are read, they are converted into a specific width and height as required.

Begin:

Step 1: User defined select option, either live streaming or stored video

Step 2: if (live)

Step 2.0: Enable camera to access

Step 2.1: Read live feed frames

Step 3: if (stored video)

Step 3.0: feed the path of input file

Step 3.1: Read frames

Step 4: Each frame, convert it to specified size, (w*h) @ 700

End

6.5.2 Features Extraction

This module can be further divided into two sub-modules

1. Loading Frames in YOLO training set

2. Creating list of Detections

6.5.2.1 Loading Frames in YOLO Training Set with Pseudo Code

Once the frames have been extracted and converted into desired width and height, the dimensions of the frames are initialised. A blob is then constructed from the input frame.The Frame is loaded into the YOLO Training set after which the classID and Minimum Confidence is calculated.

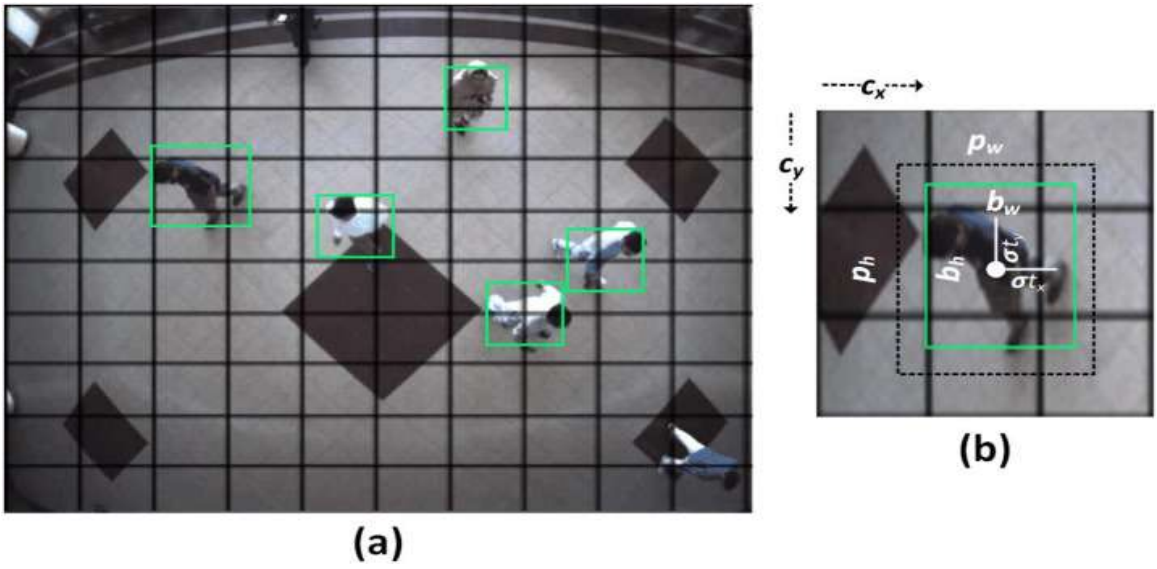

Figure 6.5.2.1.1 Detected Bounding Box

Begin

Step 1: Grab the dimensions of the frame and initialize it.

Step 2: Construct blob from input frame.

Step 2.0 : cv2.dnn.blobFromImage

Step 3: Load the frame into YOLO Training Set

Step 4: Extract class ID and confidence

Step 5: Compare the classID and MIN confidence

Step 5.0 : if classID=personIdx and confidence> MIN_CONF

6.5.2.2 Creating List of Detections with Pseudo Code

With the classID and Minimum confidence calculated, the next step is to separate the sub classes so that only the actual class of the person is extracted. A list of detections is created which will consist of the width and height and also the center_X and center_Y. Then a result list is created which will consist of the confidences, bounding box locations and the centroids. This list is then appended with the master list for the results.

Step 6: The sub class are filtered and the actual class of person is extracted

Step 7: Create a list of detection

Step 7.0: List of width and height

Step 7.1: List of center_X and center_Y

Step 8: Filter bounding boxes, Apply NMS

Step 9: Ensure at least one detection exists

Step 9.0: if len(idxs)>0:

Step 10: Create a result list

Step 10.0: confidence, bounding box locations, centroids

Step 11: Append to master list for results.

End

6.5.3 Calculation and Result

This module can be further divided into two sub-modules

1. Calculating Euclidean Distance

2. Creating list of violations

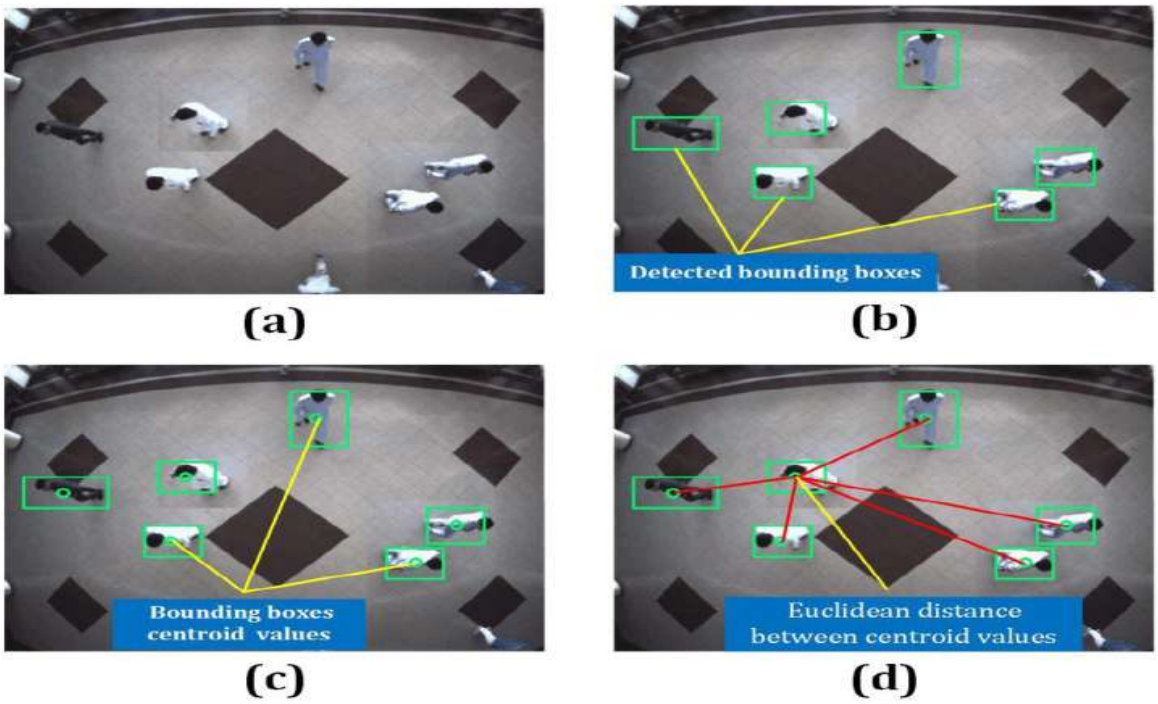

Figure 6.5.3.1 Distance between each bounding box and violation detection

6.5.3.1 Calculating Euclidean Distance with Pseudo Code

It must be ensured that at least two people are detected in the video frame after which the centroid is calculated. Once the centroids have been determined, the Euclidean Distance between these centroid are calculated. We then check whether or not the distance between two centroid pairs is less than MIN.

Begin

Step 1: Ensure at least two people are detected

Step 2: Extract the centroid from the results

Step 3: Compute the Euclidean Distance between all the pairs of the centroids

Step 3.0: dist.cdist(, , metric="euclidean")

Step 4: Loop over the matrix, compute the upper triangle method

Step 5: Check if the distance between any two centroid pairs are less than MIN

Step 5.0: if D[i,j]< config.MIN_DISTANCE

6.5.3.2 Creating list of violations with Pseudo Code

We now create a list of violations and a list of violation centroids. While displaying the image feed, if a person is violating the minimum distance, we mark in red. If a person is not violating the minimum distance, we mark in green. The output is saved in a video file.

Step 6: Create a list of violations

Step 7: Create a list of violation centroids

Step 8: Display the image feed

Step 8.0: Mark the identified violations in **RED**

Step 8.1: Mark the identified non-violations in **GREEN**

Step 8.2: Display the total number of violations

Step 9: Save the output stream in a video file.

End

6.6 Source Code

6.6.1 Social Distancing config.py

```
# base path to YOLO directory
MODEL_PATH = "yolo-coco"
# initialize minimum probability to filter weak detections along with
# the threshold when applying non-maxima suppression
MIN_CONF = 0.3
NMS_THRESH = 0.3
# boolean indicating if NVIDIA CUDA GPU should be used
USE_GPU = False
# define the minimum safe distance (in pixels) that two people can be from each other
MIN_DISTANCE = 40
```

6.6.2 Detection.py

```
# import the necessary packages

from .social_distancing_config import NMS_THRESH

from .social_distancing_config import MIN_CONF

import numpy as np

import cv2

def detect_people(frame, net, ln, personIdx=0):

# grab the dimensions of the frame and  initialize the list of

# results

(H, W) = frame.shape[:2]

results = []

# construct a blob from the input frame and then perform a forward

# pass of the YOLO object detector, giving us our bounding boxes

# and associated probabilities

blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416),

swapRB=True, crop=False)

net.setInput(blob)

layerOutputs = net.forward(ln)

# initialize our lists of detected bounding boxes, centroids, and confidences, respectively

boxes = []

centroids = []

confidences = []
```

```
# loop over each of the layer outputs for output in layerOutputs:

# loop over each of the detections for detection in output:

# extract the class ID and confidence (i.e., probability)

# of the current object detection

scores = detection[5:]

classID = np.argmax(scores)

confidence = scores[classID]

# filter detections by (1) ensuring that the object

# detected was a person and (2) that the minimum

# confidence is met

if classID == personIdx and confidence > MIN_CONF:

# scale the bounding box coordinates back relative to

# the size of the image, keeping in mind that YOLO

# actually returns the center (x, y)-coordinates of

# the bounding box followed by the boxes' width and

# height

box = detection[0:4] * np.array([W, H, W, H])

(centerX, centerY, width, height) = box.astype("int")

# use the center (x, y)-coordinates to derive the top

# and and left corner of the bounding box

x = int(centerX - (width / 2))

y = int(centerY - (height / 2))
```

```
# update our list of bounding box coordinates,

# centroids, and confidences

boxes.append([x, y, int(width), int(height)])

centroids.append((centerX, centerY))

confidences.append(float(confidence))

# apply non-maxima suppression to suppress weak, overlapping

# bounding boxes

idxs = cv2.dnn.NMSBoxes(boxes, confidences, MIN_CONF, NMS_THRESH)

# ensure at least one detection exists

if len(idxs) > 0:

# loop over the indexes we are keeping

for i in idxs.flatten():

# extract the bounding box coordinates

(x, y) = (boxes[i][0], boxes[i][1])

(w, h) = (boxes[i][2], boxes[i][3])

# update our results list to consist of the person

# prediction probability, bounding box coordinates,

# and the centroid

r = (confidences[i], (x, y, x + w, y + h), centroids[i])

results.append(r)

# return the list of results

return results
```

6.6.3 Social Distance Calculator.py

```
# import the necessary packages

from imagesearch import social_distancing_config as config

from imagesearch.detection import detect_people

from scipy.spatial import distance as dist

import numpy as np

import argparse

import imutils

import cv2

import os

# construct the argument parse and parse the arguments

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())

# load the COCO class labels our YOLO model was trained on

labelsPath = os.path.sep.join([config.MODEL_PATH, "coco.names"])

LABELS = open(labelsPath).read().strip().split("\n")
```

```
# derive the paths to the YOLO weights and model configuration

weightsPath = os.path.sep.join([config.MODEL_PATH, "yolov3.weights"])

configPath = os.path.sep.join([config.MODEL_PATH, "yolov3.cfg"])

# load our YOLO object detector trained on COCO dataset (80 classes)

print("[INFO] loading YOLO from disk...")

net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)

# check if we are going to use GPU

# if config.USE_GPU:

# set CUDA as the preferable backend and target

# print("[INFO] setting preferable backend and target to CUDA...")

# net.setPreferableBackend(cv2.dnn.DNN_BACKEND_CUDA)

# net.setPreferableTarget(cv2.dnn.DNN_TARGET_CUDA)

# determine only the *output* layer names that we need from YOLO

ln = net.getLayerNames()

ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]

# initialize the video stream and pointer to output video file

print("[INFO] accessing video stream...")

vs = cv2.VideoCapture(args["input"] if args["input"] else 0)

writer = None

# loop over the frames from the video stream

while True:

# read the next frame from the file
```

```
(grabbed, frame) = vs.read()

# if the frame was not grabbed, then we have reached the end

# of the stream

if not grabbed:

break

# resize the frame and then detect people (and only people) in it

frame = imutils.resize(frame, width=700)

results = detect_people(frame, net, ln,

personIdx=LABELS.index("person"))

# initialize the set of indexes that violate the minimum social

# distance

violate = set()

# ensure there are *at least* two people detections (required in

# order to compute our pairwise distance maps)

if len(results) >= 2:

# extract all centroids from the results and compute the

# Euclidean distances between all pairs of the centroids

centroids = np.array([r[2] for r in results])

D = dist.cdist(centroids, centroids, metric="euclidean")

# loop over the upper triangular of the distance matrix

for i in range(0, D.shape[0]):

for j in range(i + 1, D.shape[1]):
```

```
# check to see if the distance between any two

# centroid pairs is less than the configured number

# of pixels

if D[i, j] < config.MIN_DISTANCE:

# update our violation set with the indexes of

# the centroid pairs

violate.add(i)

violate.add(j)

# loop over the results

for (i, (prob, bbox, centroid)) in enumerate(results):

# extract the bounding box and centroid coordinates, then

# initialize the color of the annotation

(startX, startY, endX, endY) = bbox

(cX, cY) = centroid

color = (0, 255, 0)

# if the index pair exists within the violation set, then

# update the color

if i in violate:

color = (0, 0, 255)

# draw (1) a bounding box around the person and (2) the

# centroid coordinates of the person,

cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)
```

```
cv2.circle(frame, (cX, cY), 5, color, 1)

# draw the total number of social distancing violations on the

# output frame

text = "Distancing Violations: {}".format(len(violate))

cv2.putText(frame, text, (10, frame.shape[0] - 25),

cv2.FONT_HERSHEY_SIMPLEX, 0.85, (0, 0, 255), 3)

# check to see if the output frame should be displayed to our

# screen

if args["display"] > 0:

# show the output frame

cv2.imshow("Frame", frame)

key = cv2.waitKey(1) & 0xFF

# if the `q` key was pressed, break from the loop

if key == ord("q"):

break

# if an output video file path has been supplied and the video

# writer has not been initialized, do so now

if args["output"] != "" and writer is None:

# initialize our video writer

fourcc = cv2.VideoWriter_fourcc(*"MJPG")

writer = cv2.VideoWriter(args["output"], fourcc, 25,

(frame.shape[1], frame.shape[0]), True)
```

```
# if the video writer is not None, write the frame to the output

# video file

if writer is not None:

writer.write(frame)
```

CHAPTER 7

TESTING

7.1 TEST CASES

Table 7.1.1 People Detection Function Test Case

TEST CASE NUMBER	**TU01**
NAME OF TEST	PEOPLE DETECTION FUNCTION
INPUT	INPUT IMAGE
EXPECTED OUTPUT	ONLY HUMAN BEINGS PRESENT IN IMAGES MUST BE DETECTED
ACTUAL OUTPUT	AS EXPECTED
RESULT	SUCCESSFUL

Table 7.1.2 Making prediction by Measuring Distances

TEST CASE NUMBER	**TU02**
NAME OF TEST	MAKING PREDICTION MY MEASURING DISTANCES
INPUT	INPUT VIDEO STREAM
EXPECTED OUTPUT	THE PEOPLE IN THE VIDEO ARE DETECTED WHOSE DISTANCES ARE CALCULATED
ACTUAL OUTPUT	AS EXPECTED
RESULT	SUCCESSFUL

Table 7.1.3 Determining Bounding Boxes Test Case

TEST CASE NUMBER	**TU03**
NAME OF TEST	DETERMINING BOUNDING BOXES
INPUT	INPUT VIDEO STREAM
EXPECTED OUTPUT	THE BOUNDING BOX APPEARS FOR EACH DETECTED PERSON ALONG WITH THE CENTROIDS CALCULATED
ACTUAL OUTPUT	AS EXPECTED
RESULT	SUCCESSFUL

Table 7.1.4 Total Number of Violations Test Case

TEST CASE NUMBER	**TU04**
NAME OF TEST	TOTAL NUMBER OF VIOLATIONS
INPUT	INPUT VIDEO STREAM
EXPECTED OUTPUT	THE TOTAL NUMBER OF VIOLATIONS ARE DISPLAYED ON THE OUTPUT STREAM OF THE VIDEO
ACTUAL OUTPUT	AS EXPECTED
RESULT	SUCCESSFUL

Table 7.1.5 Final Output Frame Test Case

TEST CASE NUMBER	**TU05**
NAME OF TEST	FINAL OUTPUT FRAME
INPUT	INPUT VIDEO STREAM/LIVE STREAM
EXPECTED OUTPUT	THE OUTPUT STREAM IS SHOWN WITH CORRESPONDING BOUNDING BOXES FOR EACH HUMAN DETECTED ALONG WITH THE NO. OF VIOLATIONS
ACTUAL OUTPUT	AS EXPECTED
RESULT	SUCCESSFUL

CHAPTER 8

RESULT

8.1 Results of social distance monitoring using pre-trained model

In Figure 8.1.1, the testing results of the social distance framework using a pre-trained model has been visualised. The testing results are evaluated using different video sequences. The people in the video sequences are freely moving in the scenes; it can be seen from sample frames that the individual's visual appearance is not identical to the frontal or side view. The person's size is also varying at different locations, as shown in Figure 8.1.1. Since the model only considers human (person) class; therefore, only an object having an appearance like a human is detected by a pre-trained model.

The pre-trained model delivers good results and detects various size person bounding boxes, as shown with green rectangles in Figure 8.1.1(a)–(c). From sample frames of Figure 8.1.1, people are marked with green rectangles as they maintain a social distancing threshold. The model is also tested for multiple peoples, as depicted in Figure 8.1.1(g)–(i), multiple people are entering in the scene. In sample images, it can be seen that after person detection, the distance between each detected bounding box is measured to check whether the person in the scene violates the social distance or not.

In Figure 8.1.1(e) and (h), two people at the centre of the scene are marked with red bounding boxes as they violate or breaches the social distancing threshold. Some miss detections also occur that are manually labeled with a yellow cross in sample frames. From the sample frames, it can be seen that a person is effectively detected at several scene locations. However, in some cases, the person's appearance is changing; therefore, the model gives miss detections. The reason for miss detection maybe, as the pre-trained model is applied, and an individual's appearance from an overhead view is changing, which may be misleading for the model.

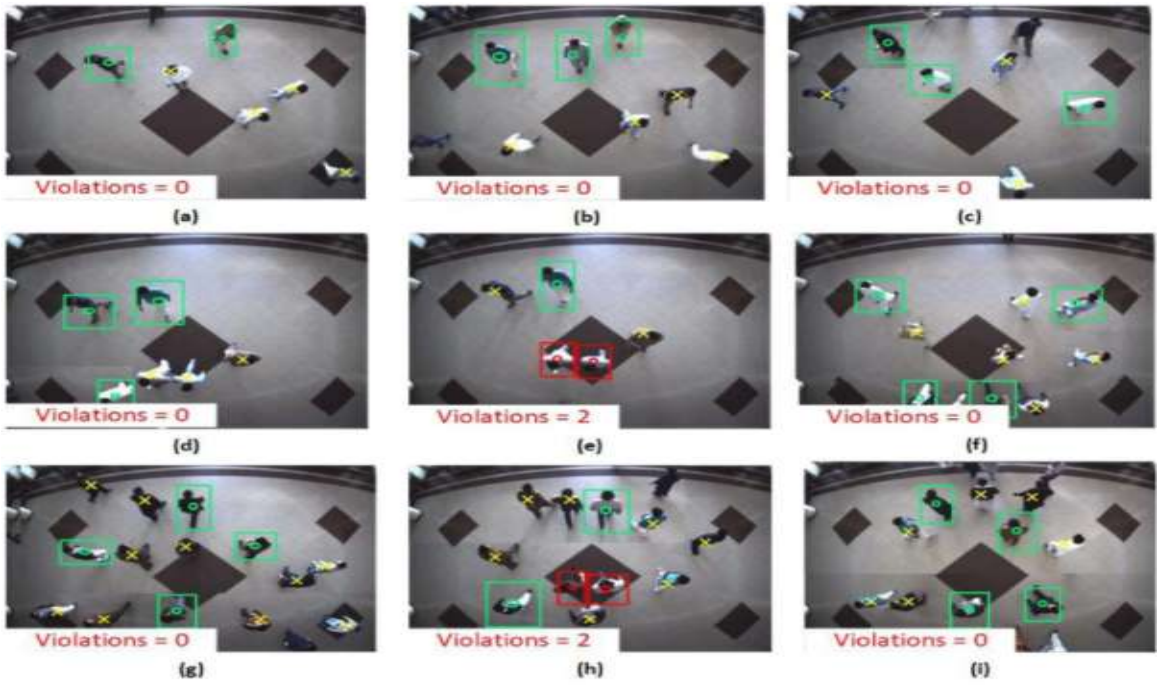

Figure 8.1.1 Results of social distance monitoring using pre-trained model

8.2 Results of social distance monitoring using transfer learning

The transfer learning methodology is applied to improve the accuracy of the detection model. The model is now tested for the same test video sequences, as discussed in the above sub-section. The experimental findings reveal that transfer learning significantly increases the detection results, as seen in Figure 8.2.1.

From the sample images, it can be visualised that the model detects the individuals at various scene locations. People with various characteristics are effectively-identified, and the social distance between people is also computed, as shown in the sample frames. In sample frames of Figure 8.2.1(a)–(c), there is no social distance violation found, since all people are marked with green rectangle boxes by the automated framework. While in the sample frame Figure 8.2.1(e), the violation is detected; however, the number of people present in the scene is small as compared to Figure 8.2.1(b), where all people are maintaining social distance, and therefore not a single violation is observed. In Figure 8.2.1(d)–(f), due to close interactions between people, violation is recorded by automated system.

The same behaviour can be found in Figure 8.2.1(g)–(i) where people are around dozen in both (g) and (h) and violation in (h) is three times as compared to (g). In Figure 8.2.1(d)–(f), multiple people are walking, and entering in the scene are detected and monitored. The framework effectively detected the breach of social distance between people and marked the bounding box as red rectangles if people are too close to each other.

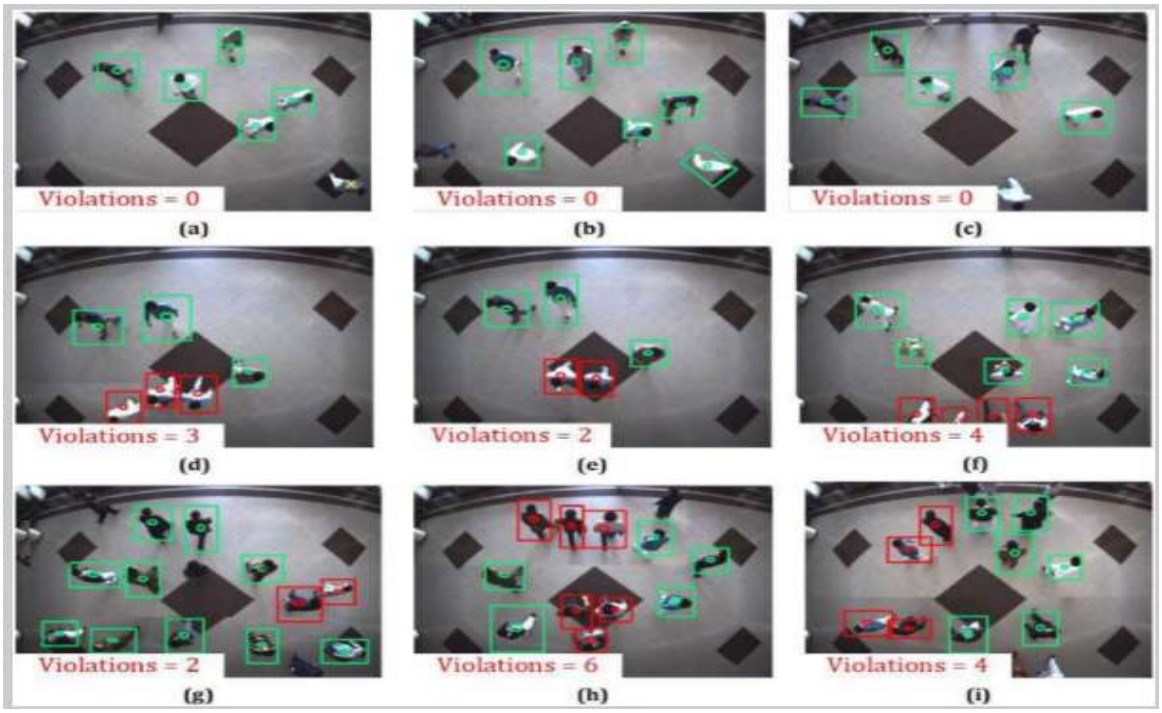

Figure 8.2.1 Results of social distance monitoring using transfer learning

8.3 Performance evaluation

Different quantitative metrics are used in this work to evaluate the performance of the framework for social distance monitoring using a deep learning model and an overhead perspective. To assess the efficiency of the detection model, Precision, Recall, and Accuracy is used. Furthermore, the findings are also compared with other deep learning models. For estimation of Precision, Recall and Accuracy, we used, tp true positive, fp, false positives, tn true negative and fn false-negative. The Accuracy Recall and Precision results are shown in Figure 8.3.1. It can be analysed that when the model is additionally trained for overhead view data set, the overall performance of the detection model is improved. The tracking accuracy is also given in Figure 8.3.2

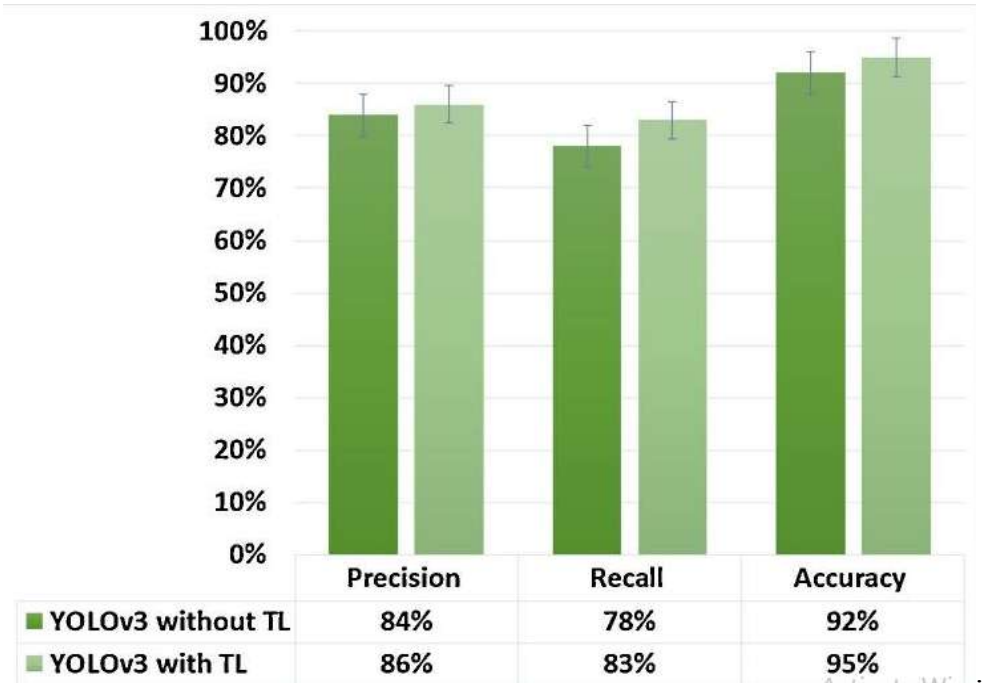

Figure 8.3.1 Precision Recall and Accuracy of model (YOLOv3) with and without transfer learning

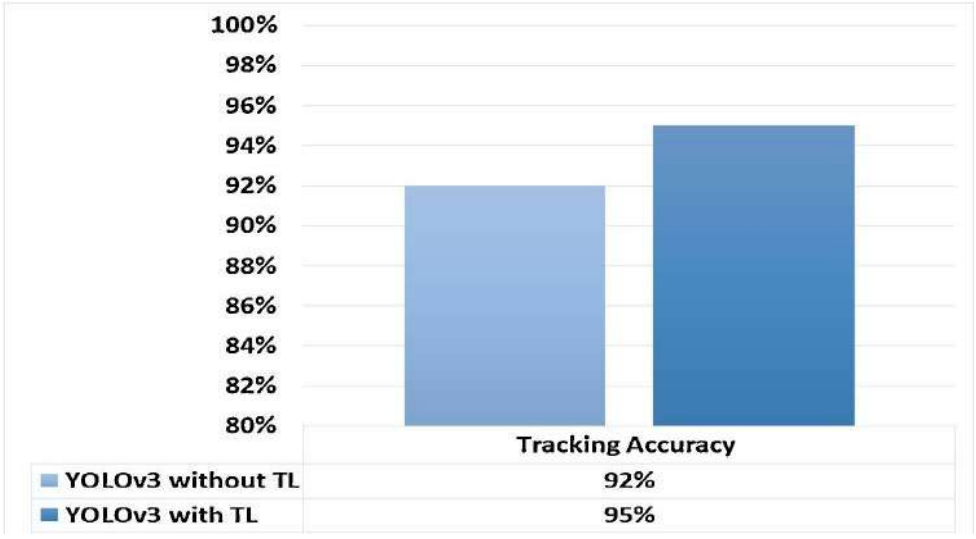

Figure 8.3.2 Tracking accuracy with pre-trained YOLOv3 detection model

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENT

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 unfavourable and thus, they can utilise time wisely and save energy for equitable situations.

The model proposes an efficient real-time deep learning-based framework to automate the process of monitoring the social distancing via object detection and tracking approaches, where each individual is identified in the real-time with the help of bounding boxes. Identifying the clusters or groups of people satisfying the closeness property computed with the help of Bird's eye view approach. The number of violations is confirmed by computing the number of groups formed and violation index term computed as the ratio of the number of people to the number of groups.

In this work, a deep learning-based social distance monitoring framework is presented using an overhead perspective. The pre-trained YOLOv3 paradigm is used for human detection. As a person's appearance, visibility, scale, size, shape, and pose vary significantly from an overhead view, the transfer learning method is adopted to improve the pre-trained model's performance. The model is trained on an overhead data set, and the newly trained layer is appended with the existing model. To the best of our knowledge, this work is the first attempt that utilised transfer learning for a deep learning-based detection paradigm, used for overhead perspective social distance monitoring.

The detection model gives bounding box information, containing centroid coordinates information. Using the Euclidean distance, the pairwise centroid distances between detected bounding boxes are measured. To check social distance violations between people, an approximation of physical distance to the pixel is used, and a threshold is defined. A violation threshold is used to check if the distance value violates the minimum social distance set or not. Furthermore, a centroid tracking algorithm is used for tracking peoples in the scene. Experimental results indicated that the framework efficiently identifies people walking too close and violates social distancing; also, the transfer learning methodology increases the detection model's overall efficiency and accuracy.

For a pre-trained model without transfer learning, the model achieves detection accuracy of 92% and 95% with transfer learning. The tracking accuracy of the model is 95%. The work may be improved in the future for different indoor and outdoor environments. Different detection and tracking algorithms might be used to help track the person or people who are violating or breaches the social distancing threshold.

BIBLIOGRAPHY

[1] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, “Behavior recognition ´ via sparse spatio-temporal features,” in 2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance. IEEE, 2005, pp. 65–72.

[2] M. Piccardi,“Background subtraction techniques: a review,” in 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583), vol. 4. IEEE, 2004, pp. 3099–3104.

[3] Y. Xu, J. Dong, B. Zhang, and D. Xu, “Background modeling methods in video analysis: A review and comparative evaluation,” CAAI Transactions on Intelligence Technology, vol. 1, no. 1, pp. 43–60, 2016.

[4] H. Tsutsui, J. Miura, and Y. Shirai, “Optical flowbased person tracking by multiple cameras,” in Conference Documentation International Conference on Multisensor Fusion and Integration for Intelligent Systems. MFI 2001 (Cat. No. 01TH8590). IEEE, 2001, pp. 91–96.

[5] A. Agarwal, S. Gupta, and D. K. Singh, “Review of optical flow technique for moving object detection,” in 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I). IEEE, 2016, pp. 409–413.

[6] S. A. Niyogi and E. H. Adelson, “Analyzing gait with spatiotemporal surfaces,” in Proceedings of 1994 IEEE Workshop on Motion of Nonrigid and Articulated Objects. IEEE, 1994, pp. 64–69.

[7] Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, “Object detection with deep learning: A review,” IEEE transactions on neural networks and learning systems, vol. 30, no. 11, pp. 3212–3232, 2019.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097– 1105.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in Advances in neural information processing systems, 2015, pp. 91–99.

[10] X. Chen and A. Gupta, "An implementation of faster rcnn with study for region sampling," arXiv preprint arXiv:1702.02138, 2017.

[11] https://blog.usejournal.com/social-distancing-aiusing-python-deep-learning-c26b20c9aa4c.

[12] N. S. Punn and S. Agarwal, "Crowd analysis for congestion control early warning system on foot over bridge," in 2019 Twelfth International Conference on Contemporary Computing (IC3). IEEE, 2019, pp. 1–6.

[13] Pias,"Object detection and distance measurement," https://github.com/ paul-pias/ Object-Detectionand-Distance-Measurement, 2020.

[14] A. Brunetti, D. Buongiorno, G. F. Trotta, and V. Bevilacqua, "Computer vision and deep learning techniques for pedestrian detection and tracking:A survey," Neurocomputing, vol. 300, pp. 17–33, 2018.

APPENDIX

APPENDIX A: Screenshots

Figure A.1 Sample Video

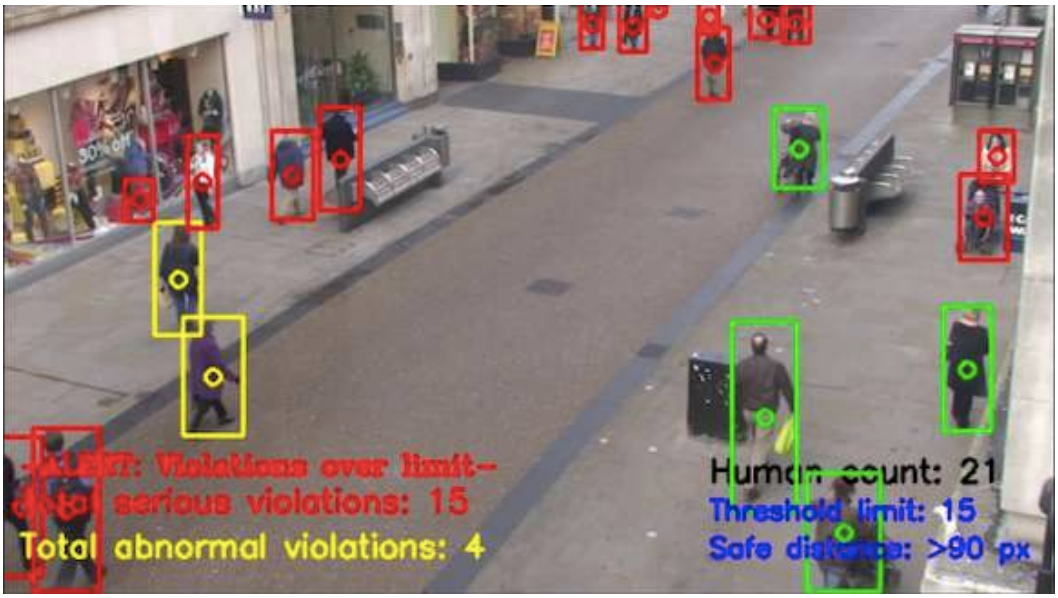

Figure A.2 Applying Social Distancing Algorithm

PAPER PRESENTATION DETAILS

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:03/Issue:07/July-2021 Impact Factor- 5.354 www.irjmets.com

MONITORING COVID-19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING

Aishwarya Raju*1, Harshith M*2, Hemanth M*3, Lekha Devaraj*4, Bhanumathi S*5

*1,2,3,4Visvesvaraya Technological University, Department Of Information Science And Engineering, SJC Institute Of Technology, Chickballapur, Karnataka, India.

*5Assistant Professor, Department Of Information Science And Engineering, SJC Institute Of Technology, Chickballapur, Karnataka, India.

ABSTRACT

Coronavirus, also acknowledged as COVID-19 with its devastating spread to more than 235 nations has caused a global crisis. The lack of active treatment, drugs, and immunity to COVID-19 makes the population more vulnerable. Though the vaccines are being rolled out, it does not ensure a complete shield to fight against the virus. On that account, social distancing can be considered as a viable strategy for combating the epidemic. This paper provides a deep learning-based framework for automating the task of monitoring social separation using surveillance video. Our project employs the concept of the YOLOv3 object detection model to discern human beings from the environment being observed. Deep Learning techniques can be used to keep track on the people that have been detected using bounding boxes and issued IDs.

Keywords: COVID-19, Crisis, Deep Learning, Bounding Boxes.

I. INTRODUCTION

Coronavirus was first detected in late December 2020 in Wuhan, China. The COVID-19 disease is a contagious disorder caused by SARS CoV2. The most common symptoms of this disease were found to be fever, cough, loss of smell and taste and cold. They develop mild to moderate respiratory disease over time and recover without requiring specific treatment. There's a chance you're infected even if none of these symptoms appear. Older persons, as well as those with underlying health issues such as cancer, cardiovascular disease, diabetes, chronic respiratory disease, and obesity, are more vulnerable to acquire a serious illness due to Covid-19, which can prove to5 be life-threatening.

The term "social distancing" refers to best practices in the direction of measures aimed at minimizing or interrupting COVID-19 transmissions through a variety of techniques. Its goal is to reduce physical contact between potentially infected people and healthy people. According to WHO guidelines, persons should keep a space of at least 6 feet between them to practice social distancing. As per a recent study, social distancing is a critical containment mechanism that is necessary to avoid disease spread since people with moderate or no symptoms may carry corona infection and infect others by chance.

Figure 1: Effects of Social Distancing

Figure 1 indicates that proper social distancing is the best way to diminish the spread of infections. As can be observed in the figure, social distancing can reduce the peak number of infected cases to ensure that the number of patients does not exceed the public healthcare capacity. Moreover, social distancing also delays the outbreak peak so that there is more time to implement counter measures.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:03/Issue:07/July-2021 Impact Factor- 5.354 www.irjmets.com

II. METHODOLOGY

After the rise of the coronavirus infections since late December 2019, Social distancing concept has been adopted as the most extreme solid practice to forestall infectious disease transmission and selected as standard practice in early January 2020. During one month, the quantity of cases rises incredibly, with 2,000 to 4,000 new affirmed cases detailed each day in the principal seven-day stretch of February 2020.To control the spread of the virus, India has been following mainly three steps aggressive testing, contact tracking, social distancing. Out of these options we can say that social distancing is arguably the most effective non-pharmaceutical way to try and decrease the spread of the virus.

Since it is very hard to maintain the safety bubble in public places an automatic warning system can help individuals to maintain the required distance of 6 meters or 2 feet. This might alert the individual if they cross their bubble. Our project is based on building a social distancing detector which uses the concept of OpenCV, Deep Learning and Computer Vision. In view of social distancing, our project focuses on social distance monitoring through CCTV cameras installed across streets. The camera is used to record the distance between people in pixels and compares it with the standard measurement and thus behaves as a social distancing detector. So, the social distancing detector's application logic is present in the file file.py. This file is the one that is responsible for creating loops on each of the frames of the video to ensure that the people detected are maintaining social distancing.

Detection and Classification

Object detection and tracking are the preliminary requirements to determine if people are complying with social distance or not. Detection is the ability to detect the object in any given image where object classification identifies the class the image belongs to. Since there are various approaches for object detection, we are discussing object detection using deep learning. Even this is a branch of machine learning algorithm where it uses a multi-layer approach to extract high-level feature from the data that is being provided. It does not require any feature to be provided manually for classification instead it tries to transform data into an abstract representation. It is also influenced by artificial neural networks present in our brain most deep learning implement neural networks to achieve the results. All the deep learning models require a huge amount of computation power and a large volume of labelled data to learn from the features. Deep Learning methods also include RCNN, faster RCNN, YOLO.

OpenCV is considered as a library for image processing, computer vision and machine learning. Preprocessing image videos to identify objects, handwriting or faces of a human is one of the many features of OpenCV. Libraries like NumPY can be combined to process the array structure of Open CV for the purpose of analysis. YOLO-You Only Look Once it focuses on the entire image as a whole and predicts the bounding boxes and then calculates the class probability to label the boxes. It is called YOLO unified as it unifies the detection of objects and the concept of the classification model together as a single detection network. Therefore, it can detect real-time objects very fast.

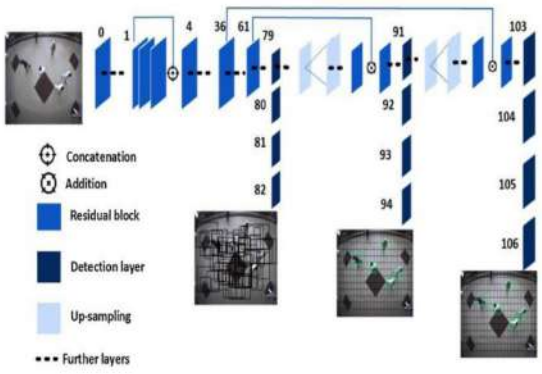

Figure 2: General Architecture of YOLOv3

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:03/Issue:07/July-2021 Impact Factor- 5.354 www.irjmets.com

box, from the numerous predicted bounding boxes, these object detection algorithms use non-max suppression. This approach is used to repress the unlikely bounding boxes and accepts the accurate ones. So, to find the distance from the camera that we are using for observation of a still object, the concept of triangular similarity is used. Triangle Similarity can be explained as the following: We must consider a still object that has a particular width, let us say w. This object must be kept at a certain distance, let us say d, from the camera. A picture of the object is taken. After this, width is measured in the form of pixels, let us say p. So we get the final focal length as:

$$f = (p \times d)/w$$

Figure 6: Focal Length Measurement

In this study, the centre point of the bounding boxes is taken to determine between two different locations of the bounding boxes. After getting the centre points value, the algorithm will calculate if the distance is lower or higher than 300 pixels. Green bounding boxes will be displayed if the social distancing norm is being followed. Figure 7 shows the green bounding boxes for each possible detected person when the distance for each bounding box for the corresponding person is longer than the default value of social distance range. The default of the minimum distance for social distance varies on the different video input as the camera perspective view is different. In contrast, when the safe social distance is violated, the bounding boxes of affected persons are changed to red indicating that the distance is below the minimum value for a safe distance. Similarly, the difference between those who violate the safe distance and those who do not can be seen. The results are in below Figure 8. Red bounding boxes indicate an alert or warning when there are social distance violations.

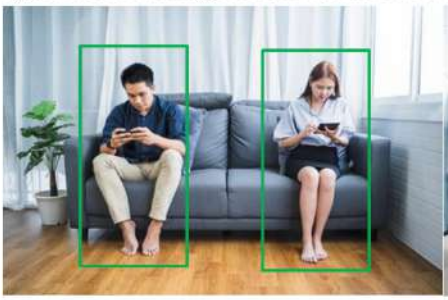

Figure 7: Green Bounding Box

Figure 8: Red Bounding Box

IV. RESULTS AND DISCUSSION

The result of the social distance structure is generated using a pre-trained model. The test result is a sacrifice and is appreciated in many videos. People in the video sequence move publicly in the picture. The model only considers the detectors assigned to the human category, so the pre-trained model only recognizes human-like objects. The pre-trained model provides sensitive results and detects people of different body types marked with inexperienced rectangles while observing social distance thresholds. Measure the gap between each identified bounding box to visualize whether people within the frame violate social distance. If two people in the middle of the screen are found to violate the social distance threshold, they will be marked as a red bounding box. The comparison between newly trained YOLOv3 and different deep learning models is shown in

the figure below. The degree of correct detection and error detection is represented by a variety of deep learning models. It can be seen from the results that transfer learning significantly improves the effect of empty-reading information recruitment.

The false detection rate of various deep learning models is incredibly low, ranging from 0.7% to 0.4% without proper training, which proves the effectiveness of deep learning. The training model is a pre-trained object recognition model that is tested on different sets of information. Although these models are trained with different front-end datasets, the results they show are very sensitive, with an accuracy rate of 90%.

Table 1. Comparison of YOLOv3 with other Deep Learning Techniques

S. no.	Model	Rate Of True Detection	Rate Of False Detection
1.	FAST-RCNN pre-trained model	90%	0.7%
2.	Faster-RCNN pre-trained model	92%	0.6%
3.	Mask-RCNN pre-trained model	92%	0.5%
4.	YOLOv3 pre-trained model	92%	0. 4%
5.	YOLOv3 trained overhead data set	95%	0.3%

V. CONCLUSION

The article proposes a productive ongoing profound learning-based system to mechanize the way toward checking the social distancing through object identification and following methodologies, where every individual is recognized in the continuous with the assistance of bounding boxes. The produced bounding boxes help in recognizing the bunches or gatherings of individuals fulfilling the closeness property figured with the assistance of a pairwise vectorized approach. The quantity of infringement is affirmed by processing the number of gatherings shaped and infringement file term figured as the proportion of the number of individuals to the number of gatherings. The broad preliminaries were directed with the famous best-in-class object discovery model YOLO v3, where YOLO v3 represented the productive execution with adjusted FPS and mAP score. Since this methodology is profoundly delicate to the spatial area of the camera, a similar methodology can be tweaked to all the more likely change with the relating field of view.

VI. REFERENCES

[1] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition ´ via sparse spatio-temporal features," in 2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance. IEEE, 2005, pp. 65–72.

[2] M. Piccardi,"Background subtraction techniques: a review," in 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583), vol. 4. IEEE, 2004, pp. 3099–3104.

[3] Y. Xu, J. Dong, B. Zhang, and D. Xu, "Background modeling methods in video analysis: A review and comparative evaluation," CAAI Transactions on Intelligence Technology, vol. 1, no. 1, pp. 43–60, 2016.

[4] H. Tsutsui, J. Miura, and Y. Shirai, "Optical flowbased person tracking by multiple cameras," in Conference Documentation International Conference on Multisensor Fusion and Integration for Intelligent Systems. MFI 2001 (Cat. No. 01TH8590). IEEE, 2001, pp. 91–96.

[5] A. Agarwal, S. Gupta, and D. K. Singh, "Review of optical flow technique for moving object detection," in 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I). IEEE, 2016, pp. 409–413.

[6] S. A. Niyogi and E. H. Adelson, "Analyzing gait with spatiotemporal surfaces," in Proceedings of 1994 IEEE Workshop on Motion of Nonrigid and Articulated Objects. IEEE, 1994, pp. 64–69.

e-ISSN: 2582-5208

International Research Journal Of Modernization In Engineering Technology And Science

Ref: IRJMETS/Certificate/Volume 3/Issue 7/336844

Issue Date: 11/07/2021

Certificate of Publication

*This is to certify that author "**Aishwarya Raju**" with paper ID "**IRJMETS336844**" has published a paper entitled "MONITORING COVID-19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING" in **International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 3, Issue 7, July 2021.***

Editor in Chief

IRJMETS

IRJMETS Impact Factor 5.354

We Wish For Your Better Future

www.irjmets.com

e-ISSN: 2582-5208

IRJMETS

International Research Journal Of Modernization In Engineering Technology And Science

Ref: IRJMETS/Certificate/Volume 3/Issue 7/336844

Issue Date: 11/07/2021

Certificate of Publication

*This is to certify that author "**Harshith M**" with paper ID "**IRJMETS336844**" has published a paper entitled "MONITORING COVID-19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING" in **International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 3, Issue 7, July 2021.***

IRJMETS Impact Factor 5.354

Editor in Chief

IRJMETS

We Wish For Your Better Future

www.irjmets.com

e-ISSN: 2582-5208

International Research Journal Of Modernization In Engineering Technology And Science

Ref: IRJMETS/Certificate/Volume 3/Issue 7/336844

Issue Date: 11/07/2021

Certificate of Publication

*This is to certify that author "**Hemanth M**" with paper ID "**IRJMETS336844**" has published a paper entitled "MONITORING COVID-19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING" in **International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 3, Issue 7, July 2021.***

IRJMETS Impact Factor 5.354

Editor in Chief

IRJMETS

We Wish For Your Better Future

www.irjmets.com

e-ISSN: 2582-5208

International Research Journal Of Modernization In Engineering Technology And Science

Ref: IRJMETS/Certificate/Volume 3/Issue 7/336844

Issue Date: 11/07/2021

Certificate of Publication

*This is to certify that author "**Lekha Devaraj**" with paper ID "**IRJMETS336844**" has published a paper entitled "MONITORING COVID-19 SOCIAL DISTANCING WITH PERSON DETECTION AND TRACKING USING IMAGE PROCESSING" in **International Research Journal Of Modernization In Engineering Technology And Science (IRJMETS), Volume 3, Issue 7, July 2021.***

Editor in Chief

IRJMETS

We Wish For Your Better Future

www.irjmets.com