PHYSICAL DISTANCE MEASUREMENT USING RASPBERRY PI AND OPEN CV

A project report submitted

in partial fulfillment of requirement for the award of degree

BACHELOR OF TECHNOLOGY

in

ELECTRONICS & COMMUNICATION ENGINEERING

by

P.ACHALA	**(17K41A0422)**
P.PARDHAVI	**(17K41A0451)**
B.DHANALAXMI	**(17K41A0406)**
M.SRAVAN KUMAR	**(17K41A0446)**

Under the guidance of

Mr. S. Srinivas

Assistant Professor, Department of ECE.

SR Engineering College

Innovation . Creativity . Entrepreneurship

SR ENGINEERING COLLEGE

Ananthasagar, Warangal.

CERTIFICATE

This is to certify that this project entitled **"PHYSICAL DISTANCE MEASUREMENT USING RASPBERRY PI AND OPEN CV"** is the bonafied work carried out by **P.ACHALA**, **P.PARDHAVI, B. DHANALAXMI and M. SRAVAN KUMAR** as a major project in the partial fulfillment of the requirement for the award of degree **BACHELOR OF TECHNOLOGY** in **ELECTRONICS & COMMUNICATION ENGINEERING** during the academic year 2020-2021 under our guidance and Supervision.

Mr. S. Srinivas
Asst. Prof. & Guide,
S R Engineering College,
Ananthasagar, Warangal.

Dr. Sandip Bhattacharya
Assoc. Prof. & HOD (ECE),
S R Engineering College,
Ananthasagar, Warangal.

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved Principal, **Dr. V. MAHESH,** for his continuous support and guidance to complete this project in the institute.

We owe an enormous debt of gratitude to **Dr. Sandip Bhattacharya, Associate Professor** & **HOD of ECE Department** and as well as our project guide **Mr. S. Srinivas, Asst.Prof.** for guiding us from the beginning through the end of the project with their intellectual advices and insightful suggestions. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

We express our thanks to project co-ordinators **Mr. S. Srinivas, Asst. Prof., Dr.T.Laxman Raj. Asst. Prof.** for their encouragement and support.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

ABSTRACT

Social distancing is a method used to control the spread of contagious diseases.As the name suggests social-distancing is an effective way to slow down the transmission of infectious virus (such as corona virus). People are advised to minimize their contact with each other to minimize the risk of the disease being transmitted through direct contact. Maintaining a safe distance is a challenge for many places like factories, banks, buses or railway stations, etc.So here we are going to build a Social Distancing Detector system using Open CV and Raspberry Pi.Using computer vision technology based on Open CV and YOLO -based deep learning, we are able to estimate the social distance of people in video streams.We'll be using the weights of the YOLO v3 Object Detection Algorithm with the Deep Neural Network module.Raspberry Pi is always a good choice for Image processing projects as it has more memory and speed than other controllers.

LIST OF FIGURES

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

In late 2019, a pneumonia outbreak was reported in Wuhan which is the capital of Hubei province, China. [1]A month later, the transmission with regards to the said pneumonia outbreak reached the Philippines last January 22, 2020, when a married couple from China travelled to the Philippines. The couple was the first two confirmed cases in the Philippines and the local transmission started and reached more than 600 cases before the government executed a nationwide Enhanced Community Quarantine (ECQ). [2]The order was mandated by the president last March 16, 2020, as a measure in decreasing the rate of local transmission. The pathogen that was responsible for the disease outbreak was later discovered as beta coronavirus which was confirmed through the process of unbiased next-generation sequencing. The coronavirus was named SARS-CoV-2 which demonstrated symptoms of fever and respiratory ailments such as dry cough, shortness of breath, and pressure in the chest which was observed in the patients of the pneumonia outbreak. [3]The disease was later recognized as a pandemic which was named COVID-19. The mode of transmission of the virus was first hypothesized as zoonotic due to earlier findings that link the virus to the Huanan seafood wholesale market but the possibility of droplet borne and airborne transmission was further investigated as modes of transmission. [1]The viral infection SARS-CoV-2 is spread by direct contact between persons or indirectly by making contact with surfaces that the infected person has touched, or a droplet dispersed by the infected person on the surface. Air was also considered a route of transmission because droplets expelled from the infected person are small that could travel freely up to 10 meters from the source while carrying the virus which is more affected by air current than gravitation. [4]Physical distancing, also known as social distancing, was one of the measures implemented by most countries to intervene with the transmission of the virus. Physical distancing refers to a non-therapeutic measure that reduces the frequency of physical contact between persons to reduce the risk of spreading the virus. [5]Physical distancing measures that are followed constantly will lead to a significant impact on the interruption of the transmission of COVID-19. Furthermore, a projection of one-time physical distancing until 2022 is necessary to mitigate seasonal outbreaks of the virus. Intermittent physical distancing is also predicted to be necessary to prevent further outbreak of COVID-19 during the post-pandemic period. [6]The existence of the COVID-19 pandemic brought different study fields to develop a solution to minimize the spread of the virus. As physical distancing prevents the transmission rate to exceed critical healthcare capacities, different technologies related to physical distancing are under discussion which includes technologies such as machine learning and computer vision. [5]

Computer Vision is an emerging technology that lets the computer analyze an image or videos with multiple frames like a human eye; [7]With the aid of artificial intelligence (AI), computers could provide real-time inferences on video frames such as identification and classification of a single or multiple objects. The regional approach [8]has high accuracy in object detection but also requires high computational capability on devices. Whereas, the unified approach is a more suitable approach on devices with limited computational capabilities which are done by reduction of the complexity of the algorithm with only a single step. [5]You Only Look Once (YOLO), an example of the unified approach, could make real-time inferences without compromising the accuracy. [9]The drawback of the first YOLO algorithm model is it's lacking in spatial ability; thus, the algorithm struggles on detecting small objects flocked together in a single frame. [10]YOLOv3 [11]and YOLOv4 [12]are a few of the improved versions of the algorithm which conduct object detection at higher speeds and higher accuracy. Other studies integrate features to the YOLOv3 algorithm such as semantic segmentation to improve accuracy in detecting smaller objects by a generation of multiple prediction grids at different sizes. [13]YOLOv3 was also utilized for pedestrian detection which was aided by object tracking algorithms such as the Deep sort algorithm to efficiently track objects moving in and out of the frame by predicting the next position of the object on the next frame. Upon comparing YOLOv3 with other state-of-the-art models such as R-CNN and SSD, YOLOv3 displayed balanced performance in terms of mean average precision (mAP) and frames per second (FPS) score. [14]YOLOv4 framework was released with increased optimal speed and accuracy of object detection are the major improvements in terms of speed and accuracy as compared from the previous object detection frameworks. [12]Other versions of YOLO architecture such as the Mini-YOLOv3 [15]and YOLO Nano [16]are made for embedded systems application. Jetson Nano developer kit is a small computer built for Artificial Intelligence applications such as running neural networks for applications such as object detection, image classification, and segmentation. [17]Recent studies applied Jetson Nano in real-time detection systems such as apple detection in apple orchards which had the best price/fps ratio as compared to Intel Movidius NCS, Intel NCS 2, and Jetson AGX Xavier. The Jetson Nano performs up to 8 fps using the YOLO algorithm which is still acceptable for real-time applications for a low-end small computer. [18]

The challenges in creating a real-time detection program are improving the frames per second with regards to processing the video without decreasing the accuracy of the object detection process. [14]Furthermore, bigger models such as YOLOv3 and YOLOv4 require devices that have a higher computational capacity to attain real-time speeds; however, smaller models of the algorithm that are suited for embedded systems could be utilized but may render a less accurate detection of the algorithm. In light of the COVID-19 pandemic, numerous opportunities arise in maximizing computer vision through various applications such as symptom detection by pose estimation, public place monitoring, and physical distance monitoring. However, to improve the efficiency of computer vision in enforcing physical distancing measures, there is a need to reduce the complexity of computer vision programs while improving the program’s accuracy. Further advancements in computer vision will aid in monitoring techniques by the integration of computer vision technology to existing surveillance systems. [5]

The objectives of the study are:

(1) To utilize the raspberry pi developer kit that will run the physical distancing detection program,

(2) To create a custom dataset specified for person detection and train YOLOv4 under the dataset with the darknet-53 framework,

(3) To develop a physical distancing detection program using YOLOv4 algorithm as an object detection algorithm together with DeepSORT object tracking algorithm and

(4) To evaluate the performance of the physical distancing detection program in terms of speed, accuracy, and reliability.

Every day, researchers learn and discover new things about COVID-19. Currently, there is no vaccine available for COVID-19. However, there are some clinical trials that are being conducted to evaluate potential therapeutics for COVID-19. Furthermore, public health measures are critical to slow the spread of the virus [5, 13, 17]. People should respect preventive actions like:

- Social distance, keeping enough space between each other
- if sick, stay at home
- wear mask, cover mouth and noise, don't touch the face
- hygiene and sanitary practices like washing hands frequently and cleaning surfaces and objects

1.2 Motivation:

Coronavirus has now become the talk of the town, Like most people in the world right now are suffering badly and everyday thousands of people are dying because of COVID-19, We were genuinely concerned about them. So, Instead of sitting idle and let negative thoughts grow day by day, we decided to do what we do best.

1.3 Objective of Project:

The concept of social distancing is very important to prevent infection. By standing further away from others, avoiding crowds and not touching. Regrettably, people around the globe are not conscious enough of the seriousness of COVID-19. They are ignoring social distancing protocol, thus helping to further spread of the virus.

1.4 Feasibility Study:

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time.

There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility
- Economical Feasibility

1.4.1 Technical Feasibility:

The technical issue usually raised during the feasibility stage of the investigation
includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

The software and hard requirements for the development of this project are not many and are already available in-house or are available as free as open source. The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing a fast feedback to the users irrespective of the number of users using the system.

1.4.2 Operational Feasibility:

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following:

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

1.4.3 Economic Feasibility:

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

In this paper, we propose a solution to urge people to respect the social distancing protocol. It’s a smart surveillance system that generates vocal alerts in the case of disrespecting social distance. We make a test prototype based on the Raspberry pi. And we perform object detection model based on deep learning algorithms to detect persons and calculate distances between them. The proposed prototype can also detect groups of persons not respecting social distance.

Our proposed prototype can be implemented in many places that require the respect of social distance. For instance, a classroom where students should keep distance, administrations, university corridors and common spaces, and others.

The proposed system does not aim to provide a cure for COVID-19 and contact tracing on people. Furthermore, testing of the proposed system will be implemented in a public area setting such as outdoor streets where the area is lightly populated. For the camera setup, a high angle-shot will be used to detect many objects in an image.

The rest of the document is organized as follow. In chapter. 2, we present some related works regarding highly to object detection algorithms in literature. In chapter. 3, we cover the methodology used including hardware specifications and object detection algorithms. Then, gives the detection process and describes the proposed prototype. After that, in chapter. 5, we give the obtained results. Finally, we make conclusions and we introduce some of the future works.

CHAPTER 2

LITERATURE SURVEY

2.1 RELATED WORKS

The spread of COVID-19 virus and the ensuing large scale lockdowns across the globe has given rise to an alarming situation. The resumption of production in manufacturing setups across all sectors is a key prerequisite for kickstarting economic activity of a nation. While there is an urgent need to resume operations at these plants, the safety of the workforce operating these plants cannot be compromised. Accordingly, processes are being put in place to educate the workforce regarding new safety regulations at the workplace which helps reduce the risk of virus transmission. However, to help the workforce transition into a post COVID world, there was a need for us to build solutions that help monitor and alert individuals once a safety violation occurs. All plants have CCTV installation, with at least a few hundred cameras as part of their security system setup. It is however not practical to monitor all these feeds concurrently due to the manual nature of the task. We have built a system that takes in these feeds and analyzes frames using deep learning models to detect whether violations have occurred or not. Once detected, a real time voice alert is triggered in the area of the violation. This feedback helps reduce the violations and thus contributes to the overall safety at the plant. In addition, these alerts are stored in a central repository that helps the management analyze the trends and take suitable actions to curb the violations. Given the context of COVID-19, we focused on building features that help reduce the risk of virus transmission.

Research indicated that maintaining social distance between co-workers as well as wearing face masks was effective means of reducing this risk. We hence built solutions that could monitor these actions through video feeds. World Health Organization (WHO) has recommended that a social distance of at least 2m be maintained between individuals. While the requirement is simple, monitoring this aspect through video feeds that provide a perspective view makes it difficult to ascertain the exact distance on ground. WHO has also recommended that personnel are encouraged to wear face masks to avoid the risk of virus entering the body through the nasal / oral cavity. During the lockdown, it was encouraged by the Indian government for the people to come up with mask substitutes as most countries saw a scarcity of required PPE. Hence, the face masks worn are not of standard type and come in different colors, shapes and sizes. The lack of such diversified data for training purposes makes mask detection a challenging task. The approach taken to overcome this problem, the model and other details are discussed. Under the current Covid-19 lockdown time such a system is definitely important to prevent the spread in many use cases.

Recently many intelligent surveillance systems have been developed; each has its own particularity. Depending on the need, surveillance system could focus more on a point than another. These are some of the similar implementations we get inspired from.

In [1], authors proposed an efficient CNN model able to detect relevant objects in real-time video surveillance applications. So they exploit a pre-trained model on a large dataset. After that, they fine-tune the network and apply a transfer learning in several others datasets. Testing the resulting model on the Penn-Fudan dataset with a GPU they get a speed of 53 FPS and accuracy of 95%.

In [3], we found the use of a new methodology called multi-object detection and tracking (MODT) using an optimal Kalman filtering technique to track moving objects. Tested on video clips, the accuracy of MODT framework is equal to 76.23% for detection and 86.78 for tracking. However it requires powerful graphics processing units.

In addition, another application of video surveillance concerns traffic flow estimation. Researchers in [4] employed the state-of-the-art Faster R-CNN two-stage detector together with SORT tracker to solve their issue. As a result, their solution was able to count vehicles and driving directions with less than 10% mean average percentage error.

Object detection task is high computation-intensive and energy-consuming, designed to work on advanced GPU architectures. Adapting a deep neural network used for object detection purpose for embedded devices is a challenging task. However, we found many interesting works proving that is possible to perform average quality object detection on embedded devices, such as [6, 8–11, 15].

Moreover, wide ranges of application use object detection algorithms, in many domains. For example, autonomous driving [14] where detecting objects in the road is very important. Other examples include surveillance, robotics and smart cities. Finally in our case, we use object detection pretrained models for the surveillance of social distancing in the era of COVID-19.

Particularly, our proposed solution implement a CNN based model in an embedded device; Raspberry Pi. Our main goal is to detect persons that ignore social distance by generating vocal alerts in real-time. And the use of Raspberry pi gives us the ability to put the prototype anywhere we want. It shouldn't necessary be fixed or connected to a computer.

2.2 SARS-COV2 and COVID-19

Corona viruses are pathogens that resemble the shape of a crown specifically a round particle with spikes that resemble a solar corona. The origin of the virus belongs to the *Coronaviridiae* a family that is known to induce mild cases of respiratory diseases in humans. [1]Severe Acute respiratory syndrome (SARS) and the Middle East respiratory syndrome (MERS) are included in the corona virus family which garnered a high mortality rate in 2003 and 2012, respectively. SARS-CoV-2, which caused the COVID-19 pandemic is closely related to SARS-CoV in terms of genetics while MERS-CoV also falls under the group but is less related. [19]According to earlier findings on COVID-19, the nature of transmission was due to the Zoonotic transmission which means that the virus was transmitted from animal to human, which was believed to be amplified by the wet market in Wuhan, Province of China last December 2019[20]; however, certain cases of people who acquired the disease didn't expose themselves from the wet market or didn't have any contact with any people who had visited the wet market. Due to the increase of those certain cases, person-to-person transmission is suggested as a mode of transmission of the virus. Furthermore, aside from the modes of transmission mentioned, most respiratory viruses are transmitted through droplets either directly or indirectly through objects that could be carriers of the virus. Transmission among asymptomatic carriers or transmission during the asymptomatic phase of the virus is also possible. [1]High risks of infection are still present until herd immunity is established through vaccination, this caused most government agencies to adopt public health measures such as isolation, quarantine, and physical distancing.

2.2.1 COVID-19 transmission dynamic in the post-pandemic period:

Recent studies show that the resurgence of COVID-19 cases is possible even after the major outbreak of the COVID-19 pandemic. Seasonal outbreaks are still possible especially for countries in the temperate zone during winter seasons. One-time or intermittent physical distancing should be applied until 2022 to reduce the risk of seasonal outbreaks; However, extended critical healthcare capacities, establishing herd immunity, and effective treatments are still a few of the key factors of the success of physical distancing. Physical distancing must be extended for months to eliminate the possibility of a resurgence of COVID-19. [6]Upon evaluation of the impact of one-time physical distancing efforts with the said key factors in play, with or without seasonal outbreak reinforcement, pandemic peak size was reduced; however, in all cases resurgence of the virus is imminent after releasing physical distancing measures. As for the assessment of the effects of intermittent physical distancing, critical care capacity could not be exceeding with intermittent physical distancing measures; Furthermore, it delays the reaching of peak demand of care by three weeks from the start of physical distancing given that there is no seasonal reinforcement of the virus.

On the other hand, when the transmission is seasonally forced, physical distancing during the summer season could be less frequent given that the infection rate R_0 is constantly at its maximum during wintertime. Time lengths between physical distancing measures decreases due to established herd immunity impedes the resurgence of infection. [6]If immunity to SARS-CoV-2 is delayed as similar to the related coronaviruses, seasonal outbreaks are imminent to occur in the coming years. Prolonged one–time physical distancing might exhaust health care resources but intermittent physical distancing might be successful in maintaining the threshold for critical care capacity; however, surveillance methods are essential to produce appropriate timing in implementing intermittent physical distancing methods. In the absence of medical interventions, the extension of measures such as surveillance and physical distancing might be needed to maintain until 2022 which could cause strain on the society and the economy. [6]

2.2.2. Physical Distancing against COVID-19

Physical Distancing which is also referred to as “social distancing” in public health implies keeping distance between two or more persons outside their homes. Practicing physical distancing requires people to keep a distance of a minimum of 6 feet from each other. [22]Could 2 meters/ 6 feet be enough to reduce the risk of transmission? According to a recent study, there should be at least 1.83 meters of physical distancing to prevent SARS-CoV-2 contraction. [23]Apart from the distance, factors such as wind speed could affect the travel distance of a saliva droplet for up to 6 meters with a significant decrease in size. Indoor environments are more exposed to the risk of virus transmission where small infected droplets may cover up to 10 meters.

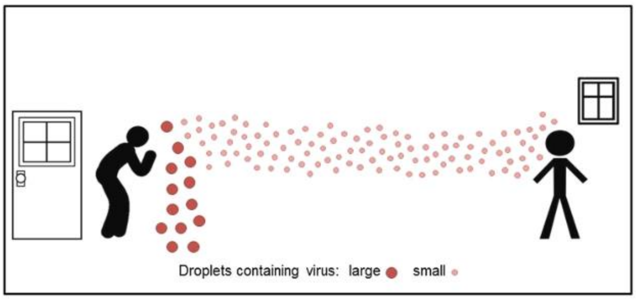


Figure 1: Large droplets close to the source (droplet transmission) and smaller droplets traveling tens of meters (aerosol transmission).

Based on recent studies, SARS-CoV-2 was proven to have higher aerosol and surface stability as compared to its predecessor, SARS-CoV-1. The virus tends to persist in the aerosol droplet form with a high possibility of infection. 1 to 2 meters of distance is safe given that people are wearing protective face masks; otherwise, people are still at risk. [24]WHO provided reviews that point out pieces of evidence of airborne transmission route, specifically via aerosol are present in indoor environments; viruses could be transmitted at significant distances which may result in faster transmission rates. [4]

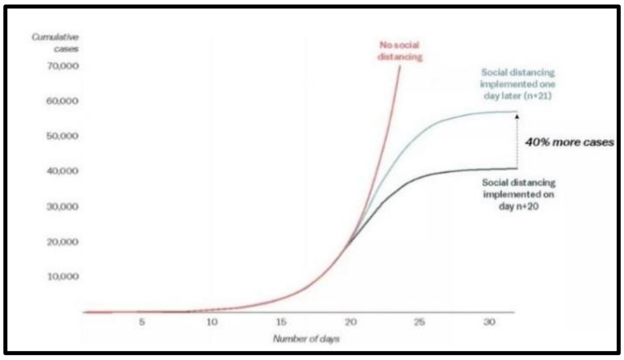


Figure 2: Impact of one-day delay of implementing physical distancing measures and no physical distancing measures. [5]

Figure2 indicates that without protective measures such as physical distancing, a significant increase in the number of daily cases is expected as compared; Furthermore, the threshold of the health care capacity is exceeded by the daily cases if protective measures were not observed while with protective measures, health care capacity is barely reached. Figure2.3 states that delaying of implementation of physical distancing may cause 40 % of more cases upon flattening the curve while no physical distancing measures would not flatten the curve.

2.3 Survey on the emerging technologies on practicing physical distancing

COVID-19 pandemic forced most governments to implement strict measures of social distancing such as restricting travels, controlling border entries, closing of public spaces, and giving public warnings about keeping 1.5 to 2 meters of distance with each other. However, such aggressive measures are challenging to implement. Technology might be a key factor in facilitating social distancing measures; with various developments such as wireless technology which could implement alert systems on people when social distancing measures are violated, or Artificial Intelligence could enforce social distancing automatically with supervision from authorities. [5]Remote surveying public spaces from behind are much safer for the authorities than stationing persons on public spaces.

2.3.1 Machine Learning

One of the challenges with easing restrictions would be on transport authorities; big adjustments are expected with the tasks of keeping traveling services running specifically trains and securing the health of the passenger. Juggling such tasks will eventually lead to the key factor which is the prediction and measurement of passenger movement densities. Real-time monitoring and machine learning are some of the fields that could enable solutions to the problem[25]Machine learning could develop applications that allow users to monitor distances between people effectively, with the aid of machine learning and computer vision, even contact tracing for infected people is possible. Machine learning is also a key player in predicting population density, with the application of real-time crowd monitoring. Predictions are set concerning the history of people's movements which is applied in network traffic predictions. [5]Other technologies that may implement social distancing are wireless technologies include such as Wi-Fi, Bluetooth, and RFID which require mobile devices or tags. However, not all part of the public has access to third party devices such as mobile devices and tags.

2.3.2 Computer Vision

Computer vision is a field that paves the way for creating analysis and modifications, and extraction of information within images and videos. One of the applications of computer vision is intelligent video surveillance in monitoring populated areas. Real-time monitoring with computer vision might require higher computational power and faster computing speeds of about 30 to 40 milliseconds. [26]OpenCV, which is an open-source computer vision library offers the tools necessary for applications such as face detection, pedestrian detection, feature matching, and tracking. One of the advantages of the software is that it includes a GPU module that performs much faster than CPU when performing tasks in image processing. The GPU module could be utilized without experience in GPU programming. Computer vision could be utilized in public place monitoring with scenarios of social distancing violation despite imposed restrictions. Smart cameras could be deployed that would not only monitor people but also detect and recognize social distancing violators. [5]

2.3.3 Neural Networks

Primarily, the first artificial neural network (ANN) was created by Dr. Robert Hecht-Nielsen, and he defines it as "...a computing system made up of several simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs.". The neuron is the basic unit of computation in a neural network. The input will be received from some other nodes or to some external sources and will compute an output. For every input, there is an associated weight (w) that is assigned, and it has relative importance as it is the basis for other inputs. The node will apply a function to the weighted sum of its inputs.[27]

In a neural network architecture, it consists of input nodes (input layer), hidden nodes (hidden layer), and output nodes (output layer). In the input nodes, no computation was done as they just only pass the information to the next layer. In the hidden nodes, intermediate processing or computation occurs. The hidden nodes transfer weights from the input layer to the following layer that could be another hidden layer or output layer. Output nodes use an activation function that maps to the desired output format. Primarily, the activation function of a node defines the possible output of the node given input or sets of inputs. Lastly, the parameters of a neural network can be modified for the given input to the network will produce a favored output and the learning rule algorithm is responsible for it.[27]

2.3.4 Development on Computer Vision Algorithm on Object Detection

Object detection is one of the major fields of computer vision that aids in pose estimation, vehicle detection, and surveillance. Object detection tries to distinguish objects of interest in an image by drawing a box around the interest. Standard convolutional neural networks (CNN) might not be applicable due to variations on the length of the output layer. Regions of interest may be drawn in an image but these objects may vary in aspect ratio, thus a huge number of regions may not be classified and will require much higher computing power. R-CNN and YOLO were introduced as faster object detection algorithms. [10]

2.3.5 Convolutional Neural Network

There are other types of neural networks. The study will only focus on utilizing convolutional neural network (CNN) because the object detector model is a CNN based. The convolutional neural network has a wide range of application including image recognition, image classification, object detection, face recognition, etc. Primarily, in deep learning CNN models, each input image will pass through a series of convolutional layers with filters, pooling, fully connected layers, and apply Softmax function, which is an activation function, that can classify an object having a probabilistic value between 0 and 1. In a convolution layer, it extracts features from an input image. The convolution can preserve the relationship between pixels by learning image features using small squares of input data. Because of different filters, the convolution of an image can perform a different operation such as edge detection, blur, sharpen, etc. Striding in a convolutional neural network is the number of pixels shifts over the input matrix. Padding in a convolutional neural network is a technique if the filter does not fit perfectly from the input image and it can be two options where the picture will be pad with zeros to that the image will fit or drop the part of the image where the filter did not fit. For non-linear operation, the most commonly used is the Rectified Linear Unit (ReLU). The primary purpose ofReLU is to introduce non-linearity to the convolutional neural network because it is necessary for the convolutional neural network to learn non-negative linear values with a given real-world data that contains negative values. The pooling layer in a convolutional neural net will reduce the number of parameters when the images are too large and spatial pooling reduces the dimensionality of each map but retains important information. Spatial pooling can be Max pooling, average pooling, or sum pooling. After spatial pooling, the feature map matrix will be converted as a vector and feed it into a fully connected layer. The combination of a fully connected layer will create a model and lastly, an activation function will be used to classify the outputs. [28]Figure 2.4 shows an example of a network with many convolutional layers.

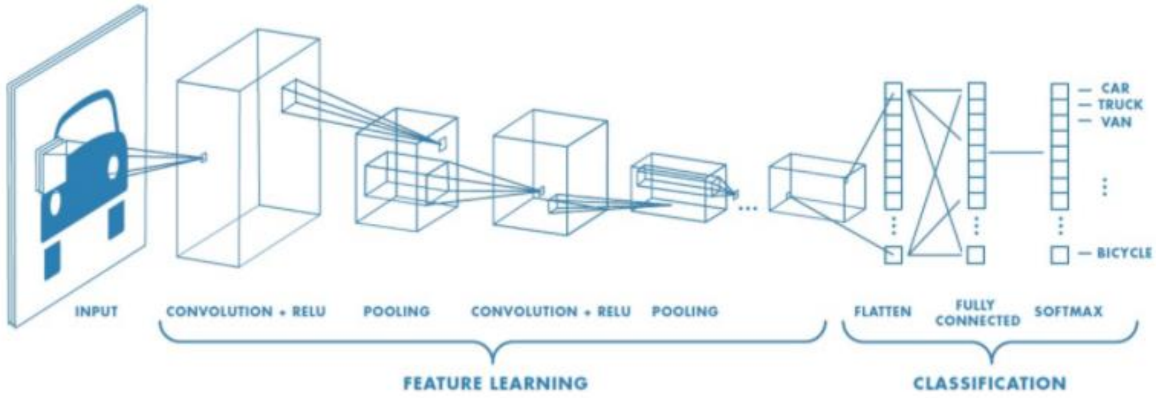


Figure 3: Example of Neural Network with Convolutional Layers. [29]

2.3.5.1 Darknet

The convolutional neural network architecture of the object detector model that will be used in the study is a darknet framework based. The darknet framework is an open-source neural network framework that is written in C and CUDA. [30]

2.3.5.2 Region-based convolutional neural network

R-CNN algorithm selects 2000 regions which are referred to as regional proposals and apply selective search in the said regions. Images are segmented into numerous candidate regions, then algorithm groups similar regions into larger ones repetitively until generated regions are distinguished. [8]

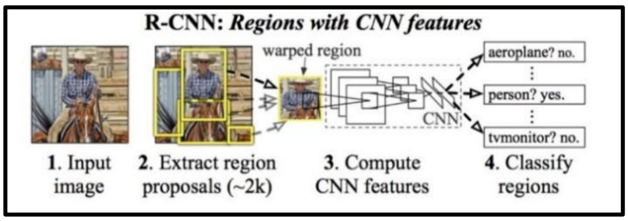


Figure 4: R-CNN Algorithm.

Extracted regional proposals will then enter to a CNN which extracts feature which will be scored by a Support Vector Machine (SVM) as a classifier of the object within the region proposal. [31]Challenges offered by R-CNN are the amount of time it takes to train a neural network to classify 2000 region proposal for a single image, R-CNN is not plausible for real-time as it takes 47 seconds to test a single image, and the selective search is fixed, therefore no learning is present in that algorithm; hence bad region proposal could not be distinguished. [10]

2.3.5.3 Fast and faster R-CNN

Fast R-CNN was later introduced and solved some of the disadvantages of R-CNN. Instead of feeding the CNN with 2000 regional proposals, the whole image is fed to the CNN to create a convolutional feature map. From the map, regions of proposals are identified and are warped into squares, reshaping the squares will require the RoI pooling layer which is part of the RoI feature vector. The softmax layer is used to offset the values for the bounding box and classify the proposed region. [31]

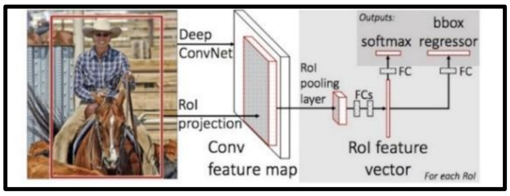


Figure 5: Fast R-CNN Algorithm.

Fast R-CNN is better in terms of R-CNN in terms of the time of training and testing. The common denominator between R-CNN and Fast R-CNN is the selective search which slows down the network and degrades its performance. [10] Faster R-CNN eliminates the use of selective search algorithms but instead, the network is trained to learn regional proposals. The image serves as an input to the neural network which generates a convolutional feature map for identification of region proposals. Region proposals are reshaped with the Region of Interest (RoI) pooling layer and classify the image within the proposed region, while offset values are predicted for the bounding boxes. [32]Faster R-CNN is significantly faster than Fast R-CNN and R-CNN alone which makes it applicable for real-time monitoring.

2.3.5.4 YOLO – "You Only Look Once" Algorithm

Another object detection algorithm which is known as YOLO which stands for "You Only Look Once" utilizes only a single neural network for predictions of bounding boxes and classifiers instead of relying on region proposals. [10]Parts of the image with a high probability of having an object are detected then are classified.

YOLO has a significant edge over the R-CNN algorithm due to its unified neural network and could maintain real-time speeds without the tradeoff of low accuracy. [9]The explanation of the YOLO algorithm is as follows: First, an image is divided into S × S grids where each grid is predicted by bounding boxes and classified.

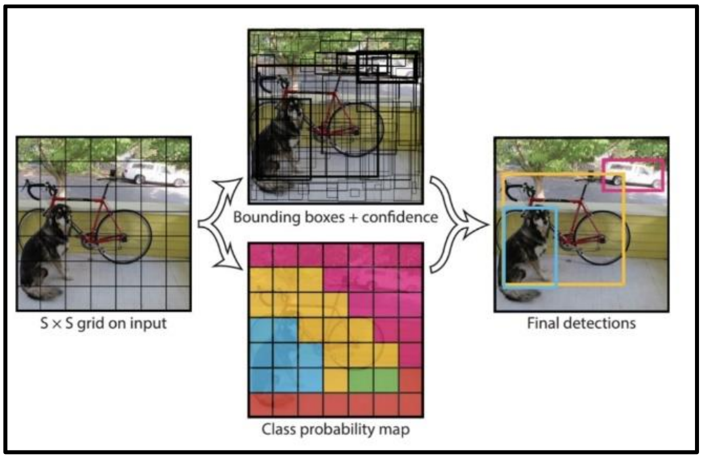


Figure 6: “You Only Look Once,” YOLO algorithm.

2.3.5.5 Bounding Boxes

The height and the width of these bounding boxes could be predicted in any object detection algorithm but are not practically applied because the prediction of these parameters might lead to instability of the gradient during the training of the network. As an alternative, YOLO predicts offsets or technically known as log-space transforms to define bounding boxes initially. The pre-defined boxes are referred to as anchors. Anchors are computed on the dataset through k-means clustering. Prediction of width and height of the bounding box is done through offsets generated by cluster centroids. The center of the bounding box is then predicted using a sigmoid function. [33]

$$b_x = \sigma(t_x) + c_x \quad (2.1)$$

$$b_y = \sigma(t_y) + c_y \quad (2.2)$$

$$b_w = p_w e^{t_w} \quad (2.3)$$

$$b_h = p_h e^{t_h} \quad (2.4)$$

Equations 1 and 2 computes the coordinates of the center of the bounding box with the outputs of t values from the network. The values c_x and c_y corresponds to the top and left coordinates of the bounding box. Equations 3 and 4 computes the width and height of the bounding box respectively where the p values correspond to the dimensions of the anchor box.

2.4 YOLOv2 algorithm

YOLOv2 an improved version of the YOLO algorithm which works on different scales and runs on different object sizes. The extraction of boxes, features, and target classification is unified in YOLOv2. YOLOv2 is applicable for pedestrian applications with the right training dataset and could distinguish the person from objects using the IOU of the prepared truth and the person or the object. YOLOv2 utilizes end–to–end training techniques instead of traditional step by step training which is similar to YOLO. YOLOv2 based Pedestrian Detection has the modified YOLO algorithm in terms of parameters that are more suited for pedestrians. [34]

2.5 YOLOv3 algorithm

The YOLOv3 algorithm is more accurate than its predecessor architectures such as the YOLO base version architecture and the YOLOv2.[35]The algorithm is comprised of 53 convolutional neural networks drawn from darknet-53 architecture. [33]The model could be modified easily to compensate for speed or accuracy by rescaling the model without the process of retraining [36]Recent studies used the YOLOv3 algorithm and tweaked the model by replacing the last layer which consists of pre-trained weights and untrained weights. The modified network could detect pedestrians in a real-time setting with approximately 29 frames per second (fps) on high resolution. [35]

It is seen in Figure 2.10 that the YOLOv3 is subpar with different state-of-the-art object detection algorithms in terms of mean average precision or mAP with a score of 51.5, but the algorithm leads in terms of speed with a score of 22. [11]Past studies also designed lightweight versions of the YOLOv3 to incorporate embedded systems such as microcontrollers. The Mini-YOLOv3 is a modified algorithm from YOLOv3 with reduced training parameters and floating-point operators. [15]

2.6 YOLOv4 algorithm

Research shows that accurate modern neural networks do not operate in real-time and it primarily requires a large number of GPUs used for training having with large mini-batch size. Therefore, Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao created a CNN that can operate in real-time on a conventional GPU and they recognize it as YOLOv4. The training for CNN only requires one conventional GPU. Primarily, YOLOv4 can run twice faster than EfficientDet, AP improves by 10%, and FPS improves by 12%. Furthermore, YOLOv4 has a great advantage in terms of the operating speed of the neural network in production systems and optimization for parallel computation. YOLOv4 uses the CSPDarknet53 neural network model as the backbone of the object detector. An addition of SPP block was used over CSPDarknet53 to increase the receptive field, separates the important context features, and also causes no reduction in terms of network operation speed. The SPP block is a part of the neck of the object detector. For the method of parameter aggregation, YOLOv4 uses PANet and it serves as the neck of the object detector. Lastly, the YOLOv3 head was used as the architecture for YOLOv4. [12] Meanwhile, some of the recent application of YOLOv4 algorithm was used to evaluate the sound source localization result of a sound imaging instrument [37]and for real-time vehicle detection to optimize the anchor box predictions by using k-means clustering. [38]

2.7 DeepSORT Object Tracking Algorithm

The advancement of object detection in the field of computer vision resulted in the development of an object tracking algorithm. Multiple Hypothesis Tracking (MHT) and Joint Probabilistic Data Association Filter (JPDAF) are some of the traditional object tracking algorithms that require a generation of hypothesis on a frame by frame basis and generate better performances but require computational capability due to the complexity of the algorithm. SORT which stands for Simple online and real-time tracking is a simple framework that utilizes Kalman filtering and the Hungarian method to provide inferences in a frame by frame method. [39]

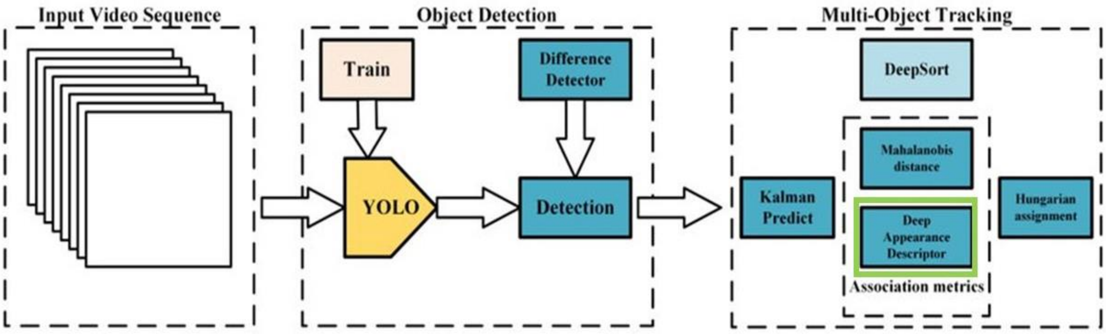


Figure 7: DeepSORT Framework.

The Kalman filter predicts the position of the object detected at the next frame through the movement of the object throughout the previous frames. The Kalman filter provides localization on the object detected to compensate for occlusions. The prediction Kalman filter is based on the assumption of the constant velocity of an object. The Hungarian algorithm, on the other hand, provides identification on the object by placing a unique identifier on the detected object in the determination of whether the object is present in the previous frame. [41] The state of the object detected in the frame is comprised of 8 variables which are*(u, v, a, h, u', v', a', h')* where *u* and *v* correspond to the center of the bounding boxes object detected while a and h is the aspect ratio and height respectively of the object. The other four components are their velocity components assumed through the constant velocity model of the object. [42] Note that the object detection and object tracking are two different algorithms working independently which consequently poses a problem in associating newly detected objects into the object tracker. Hungarian Method comes into purpose as an efficient algorithm in associating new data on the object tracker by assigning ID on the newly detected object. The distance metric is also incorporated in the algorithm as appearance descriptors. The Mahalanobis distance which is suited for dealing with distributions. The purpose of the Mahalabonis distance is to take into consideration the uncertainty of the Kalman Filter. [42]Mahalanobis distance metric provides short-term predictions on where the object could be located at the next frame based on the motion of the object; on the other hand, another distance metric is used for the association which is referred to as the cosine distance metric. The cosine distance metric is capable of long-term prediction of locations specifically in recovering the features of the object due to long-term occlusions. [39]

CHAPTER 3

METHODOLOGY

Chapter 3 discusses the evaluation of the physical distancing detection program. The chapter includes the process flow illustrating the processes conducted by the us in building the program and the interfacing of the hardware used in the physical distancing detection program. The process with regards to training the dataset used to train the program will also be tackled as well as the test setup for the investigation. Lastly, the statistical tools used in evaluating the reliability and accuracy of the program will be explained in this chapter.

3.1 SYSTEM ANALYSIS

3.1.1 Process Model

A software process model is a simplified representation of a software process. Each model represents a process from a specific perspective. These generic models are abstractions of the process that can be used to explain different approaches to the software development. They can be adapted and extended to create more specific processes.

3.1.2 Waterfall Model

The Waterfall Model is a linear sequential flow in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of software implementation. This means that any phase in the development process begins only if the previous phase is complete.

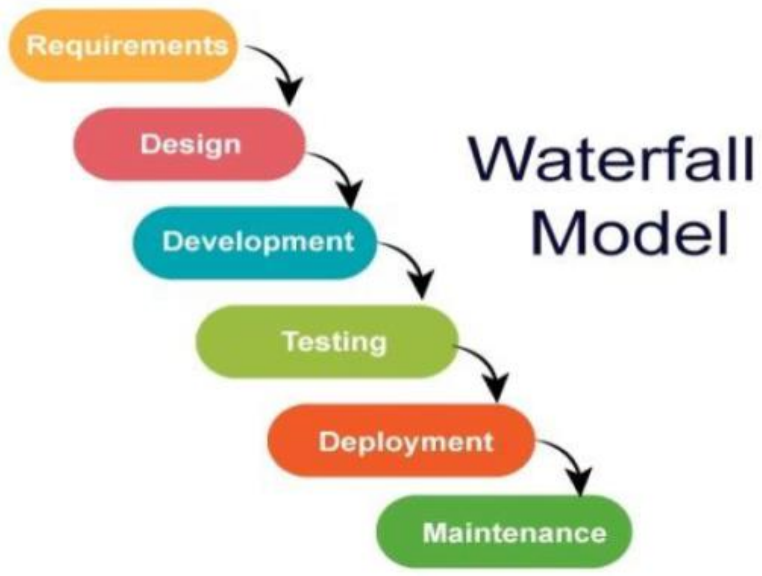


Fig. 8: Waterfall Model

The waterfall approach is the earliest approach and most widely known that was used for software development.

- Easy to explain to the users.
- Structures approach.
- Stages and activities are well defined.
- Helps to plan and schedule the project.
- Verification at each stage ensures early detection of errors/misunderstanding.
- Each phase has specific deliverables.

3.1.3 Software Development Life Cycle

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software. The SDLC aims to produce high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

- SDLC is the acronym of Software Development Life Cycle.
- It is also called as Software Development Process.

- SDLC is a framework defining tasks performed at each step in the software development process.
- ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

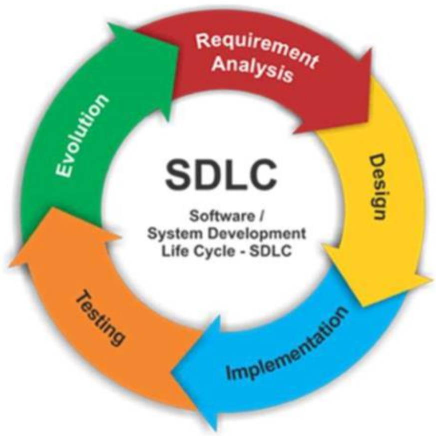


Fig. 9: Software Development Life Cycle

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process. The following figure is a graphical representation of the various stages of a typical SDLC.

A typical Software Development Life Cycle consists of the following stages:

Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas. Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification. This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product. A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 4: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle. Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high-level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing). Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

3.1.4 Non-Functional Requirements

- Security
- Performance
- Accuracy
- User interface

Security

Security and confidentiality are the top most concerns of the client. The proposed system should provide the following.

Performance

- Requests should be processed within no time.
- Users should be authenticated for accessing the requested data.

User Interface

A menu interface has been provided to the client to be user friendly.

Accuracy

The quality or state of being correct or precise.

3.1.5 Software Requirements

- ✓ Operating System **:** Raspbian OS
- ✓ Technology **:** Python
- ✓ Frameworks **:** YOLO, OpenCV
- ✓ Software **:** Graphical Windows Interface with Python Interpreter

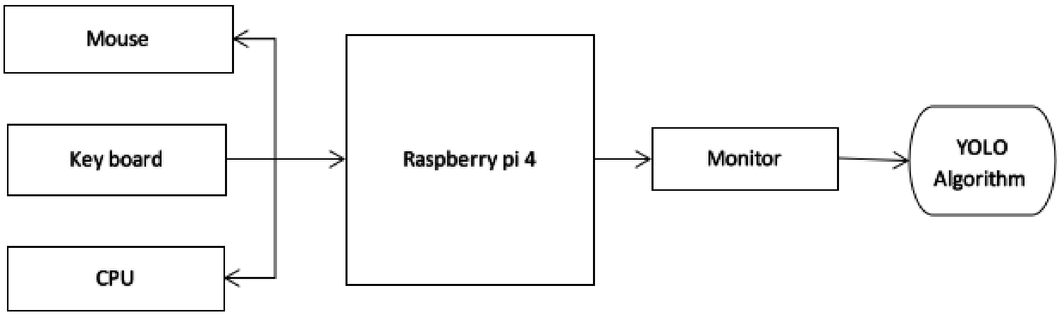
3.1.6 Hardware Requirements

- ✓ Processor **:** Intel Based Core Systems >= i3
- ✓ Hard Disk **:** 40 GB
- ✓ RAM **:** 4 GB

3.2 SYSTEM DESIGN

In this section we cover hardware materials used in the development of the prototype, including Raspberry Pi and Camera Pi. Also, we present deep learning model used for object detection task: YOLO and SSD models. Then, we give techniques used to calculate the social distance.

3.2.1 Block Diagram:



3.2.2 Conceptual framework:

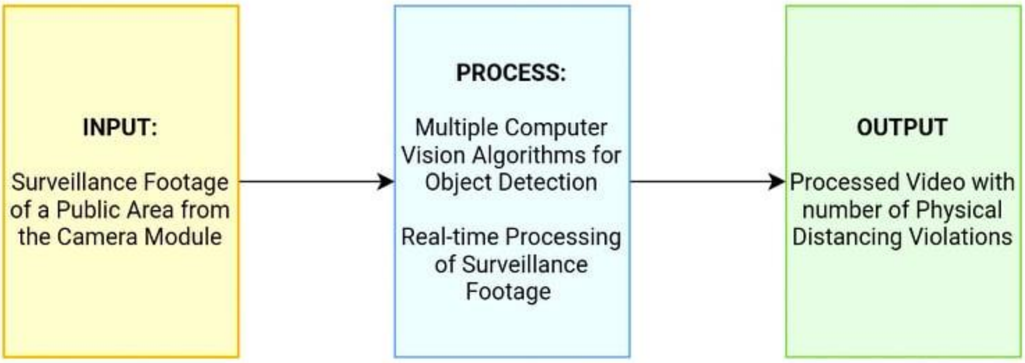


Figure 10: Conceptual Framework of the Study.

As demonstrated in Figure 3.1, the input for the study would be the real-time surveillance footage which is fed to the Raspberry pi Developer Kit by the camera module. After receiving the input, the Raspberry pi will run the physical distancing program that will process the footage through the stated algorithms. YOLOv4 is a detection algorithm that is responsible for detecting objects in a single frame and identifying what objects are seen on the frame. YOLOv4 is capable of drawing bounding boxes on the footage and pinpoint the centroid of the object detected based on the coordinates of the bounding boxes. Since we are dealing with video footage that has a continuous number of frames, an object tracking algorithm such as the DeepSORT algorithm is necessary. DeepSort is a deep learning algorithm that could track an object's next position based on the relative position of the object on the previous frames. The camera's perspective view would be transformed in the program to become in bird's eye view perspective, from the bird's eye view perspective, the distances between centroids detected in the frame could be calculated easily. The algorithms utilized and the framework of the program will be further explained in the next sections of the chapter. The output of the program is a processed video processed in real-time that could detect the number of physical distancing violations present in a single area from the surveillance footage.

3.3.3 Process flow

Before the creation of the program, the algorithms must be first trained under the custom dataset. The scope of the study is not limited to the creation of the program but extends to the training of the algorithms using available and custom datasets. Datasets are essential in using deep learning algorithms because datasets are a major factor in determining the accuracy and reliability of the algorithm. The researchers plan on training the algorithm under different datasets and determine which dataset is optimized for the physical distancing detection program. The procedure for the whole study is illustrated in Figure 3.2.

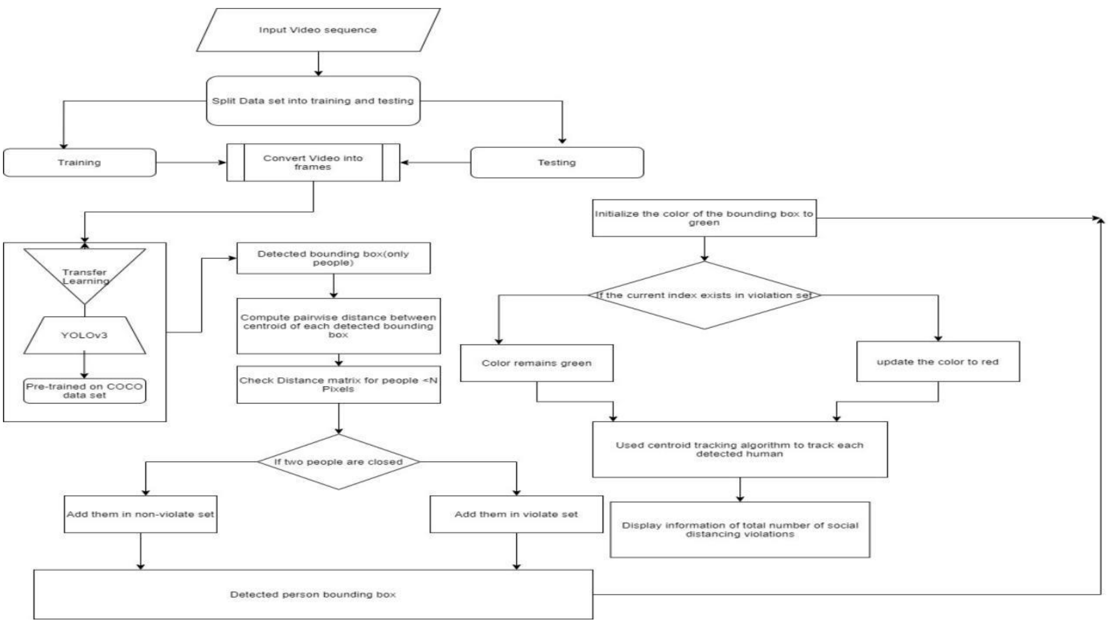


Figure 11: Process Flow chart.

The researchers will gather data from surveillance footage with persons and will provide annotations on the persons detected on all the frames gathered on the surveillance footage. The custom dataset will remodel the object detection algorithm, specifically the YOLOv4, which would be optimized for the application of a physical distancing program. Since the program is only concerned with persons at risk of infection due to breach of physical distancing protocols, the model would be optimized for the detection of persons in the frames exclusively. Upon training the algorithm with the custom dataset, it would generate a weights file which is essential in machine learning algorithms that could influence the output of the model given an input. The trained YOLOv4 will be evaluated through a confusion matrix to determine the performance of the algorithm before including the algorithm of the program, the target accuracy for the YOLOv4 algorithm is at least 80% to perform well with other algorithms. YOLOv4 algorithm together with pre-defined algorithms such as DeepSort for object tracking and perform distance measurement between centroids. After building the Physical Distancing Detection Program, the researchers will evaluate the performance of the program through a confusion matrix by determination of the number of true positives, true negatives, false positives, and false negatives to compute for the accuracy, precision, misclassification rate, and other statistics related to the performance of the program. Note that the training process would be done on a computer while the program will be deployed into the Raspberry Pi Developer Kit through transfer learning.

3.3 Raspberry Pi

The Raspberry Pi is a series of tiny and cheap single board computers developed in the United Kingdom by the Raspberry Pi Foundation [18]. It’s a credit card sized that can use a computer monitor or TV, a standard keyboard and mouse. Particularly, the Raspberry Pi series provides a set of GPIO pins that allows the control of various electronic devices such as sensors. It now is widely used even in research projects, such as weather monitoring, robotics, Smart Home Automation projects and others. Since 2012, 4 generations of the Raspberry Pi have been developed meeting all needs. We give a brief timeline of all version of the Raspberry Pi in Fig. 3.3.

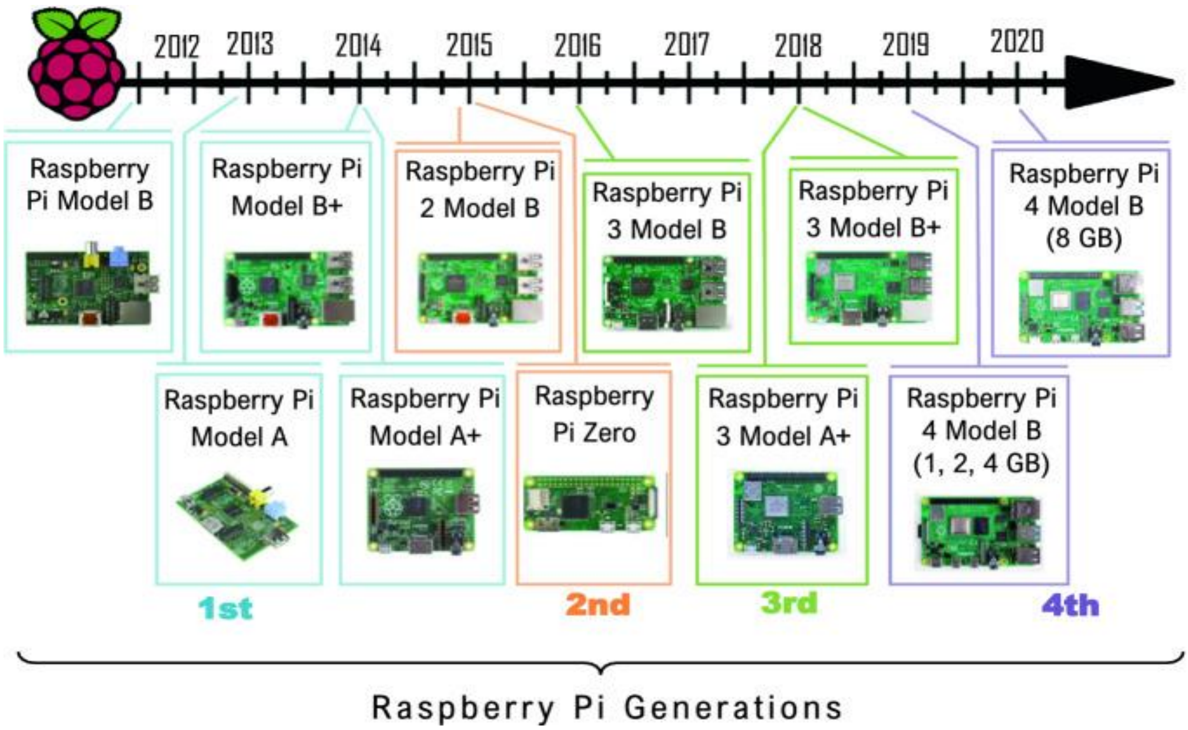


Fig 12 : The Rapsberry Pi generations timeline

In our proposed prototype, we use the 3rd generation Raspberry Pi 3 Model B because of its availability during quarantine. Technical specifications of the Raspberry Pi 3 Model B are presented in the following points:

- Quad Core 1.2 GHz Broadcom BCM2837 64bit CPU
- 1 GB RAM
- Bluetooth 4.1 and Wi-Fi
- 40 Pin extended GPIO
- 4 × USB 2.0 Ports
- 10/100 LAN Port
- 3.5 mm 4-pole Composite Video and Audio Output Jack
- CSI Camera Port
- Full size HDMI Output
- Micro USB Power Input 2.5A
- DSI Display Port

- MicroSD Card Slot

The official operating system for all the Raspberry Pi versions is provided by the same Foundation and called Raspberry Pi OS (previously Raspbian). We use the latest version available, it is the Raspberry Pi OS based on Debian buster released on May 2020 with a kernel version 4.19. Then to perform the object detection model, we used Python3.7, TensorFlow v2, OpenCV, Numpy and others useful libraries.

3.4. Raspbian OS

Raspberry pi OS earlier known to be Raspbian - Raspberry Pi. It has all the software we need for any basic task on a computer. We also get LibreOffice as an office suite that is used as a web browser, emailing service, and other tools to help in with programming for everyone. It is very widely used in every domain and IoT industy, that also give tips for bug reports and fixing.

Raspberry Pi OS is highly optimized for the Raspberry Pi line of compact single-board computers with ARM CPUs. It runs on every Raspberry Pi except the Pico microcontroller. Raspberry Pi OS uses a modified LXDE as its desktop environment with the Openbox stacking window manager, along with a unique theme. The distribution is shipped with a copy of the algebra program Wolfram Mathematica[4] and a version of Minecraft called Minecraft: Pi Edition, as well as a lightweight version of the Chromium web browser.

Raspberry Pi OS looks similar to many common desktops, such as macOS and Microsoft Windows. The menu bar is positioned at the top and contains an application menu and shortcuts to Terminal, Chromium, and File Manager. On the right is a Bluetooth menu, a Wi-Fi menu, volume control, and a digital clock.

3.5. Object Detection Pretrained Models

Object detection is a hottest topic in the computer vision field. An object detection model is able to detect multiple objects within an image, with bounding boxes. Particularly, in our case we are interested only on the person class. To do so, we test one pretrained object detection model:YOLOv3.

3.6. YOLOv3.

You Only Look One is an object detection approach proposed in [12]. It is based on a deep Convolutional Neural Network architecture. The main advantage of YOLO is the high performance and its high speed. Also, it is an open source real-time detection system [19] dedicated to work on GPU. For instance, using a NVIDIA TITAN X Pascal GPU YOLO detection model processes images at 30 FPS (Frame/Second) and has a mAP of 57.9% on COCO test-dev, which is better than real-time. Many versions of the YOLO had been developed regarding highly to the ability of detection a wide variety of objects correctly and rapidly. Latest improved version is YOLOv3. Using miniaturized embedded devices, the conventional YOLOv3 algorithm runs slowly.

In our application, as we are using the Raspberry Pi, we test the YOLOv3-tiny network. The tiny version of YOLOv3 is dedicated to work on limited hardware resource and can basically satisfied real-time requirements.

3.7. Social Distance

After using the object detection model, we have the detected persons' coordinates. These coordinates are used to display bounding boxes exactly where the person is. In our case, we use these coordinates to compute firstly the midpoint of the box then the distance between each two persons. To do so, we adopt the Euclidean distance give in the formula below (Eq. 1).

$$distance\,(A,\,B) \;=\; \sqrt{\sum\nolimits_{i=1}^{n} \left(A_i \;-\; B_i\right)^2} \tag{1}$$

For a given box with coordinates: P1(x_{center}, y_{center}), where P1 is the detected person 1 and (x_{center}, y_{center}) are the coordinates of the center of the box. When detecting for example two persons we have: P1(x_{center}, y_{center}) and P2(x_{center}, y_{center}). Now we apply the Euclidean distance formula to get distance between them, as you can see in Fig. 3.5.

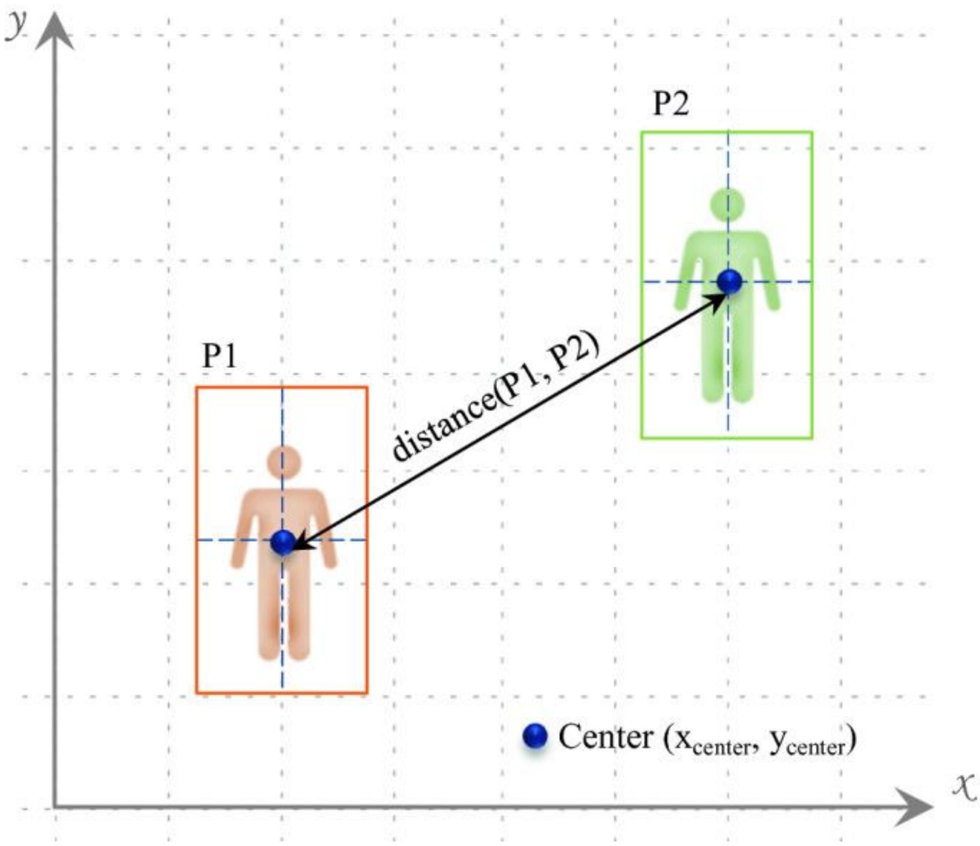


Fig 13: Description of distance computation between two persons

3.8. Proposed Method

The proposed 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 the Open CV.

Approach

1. Detect humans in the frame with yolov3.

2. Calculates the distance between every human who is detected in the frame.

3. Shows how many people are at High, Low and Not at risk.

CameraPerspectiveTransformationorCameraCalibration: 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 them 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. 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.

Figure 14: ROI and Scale points selection

Above image shows how we can select Region of Interest (ROI) and this is one time step. We draw 8 points on first frame using mouse click event. 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. In above image we can se point 5 and point 6 defines 180 cm in real life in horizontal direction and point 5 and point 7 defines 180 cm in real life in vertical direction.As we can see ROI formed by first 4 points has different length in horizontal and vertical direction, so number of pixels in 180 cm for horizontal and vertical direction will be different in rectangle (bird’s eye view) formed after transformation. So from point 5, 6, 7 we are calculating scale factor in horizontal and vertical direction of the bird’s eye view, e.g. how many pixels correspond to 180 cm in real life. Detection The second step to detect pedestrians and draw a bounding box around each pedestrian.To clean up the output bounding boxes, we apply minimal post-processing such as non-max suppression (NMS) and various rule-based heuristics, soas to minimize the risk of over fitting. 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 center 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 center point of each person's bounding box, resulting in their 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 by the scaling factor in horizontal and vertical direction estimated from calibration.

Working

Running the program will give you frame (firstframe)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 center 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 center point of each person's bounding box, resulting in their position in the bird's eye view. To calculate bottom center 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.

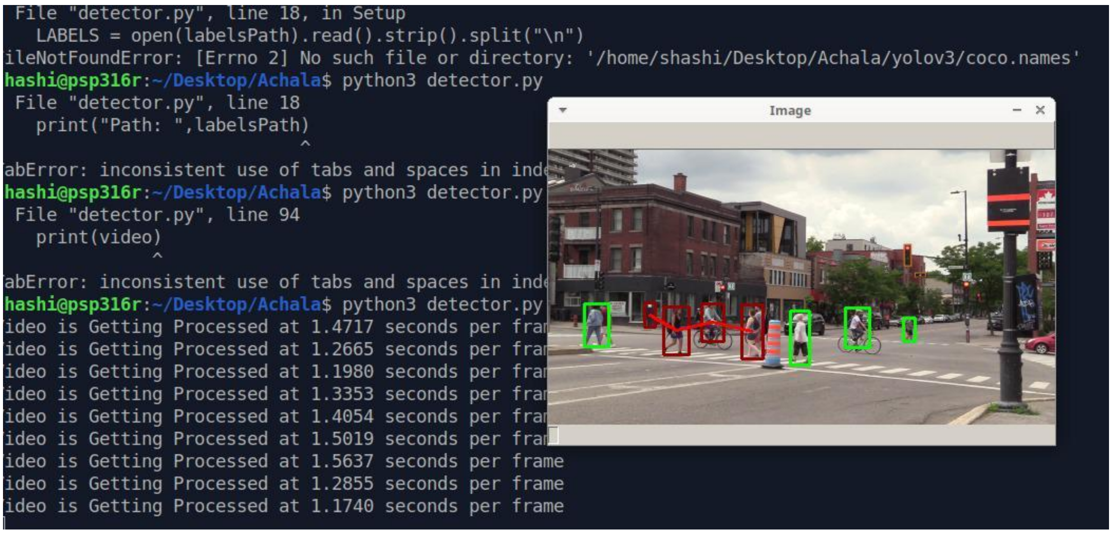


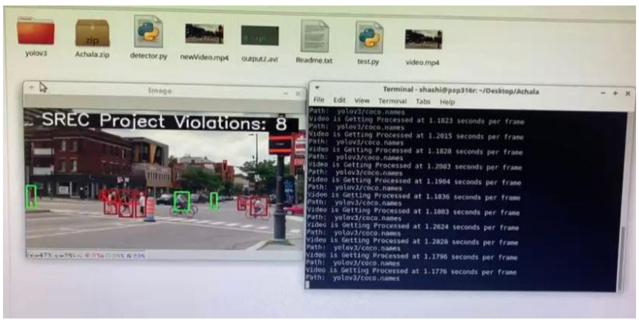
Fig. 15: Social distance detector detecting the distance in a video steam the number of people under low risk, high risk and safe.

CHAPTER 4

RESULTS AND DISCUSSION

4.1. Result Analysis:

In Fig 3.7. the testing results of the social distance framework using a pre-trained model has been visualized. 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. 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. From sample frames of figure, 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, 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, two people at the center 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.



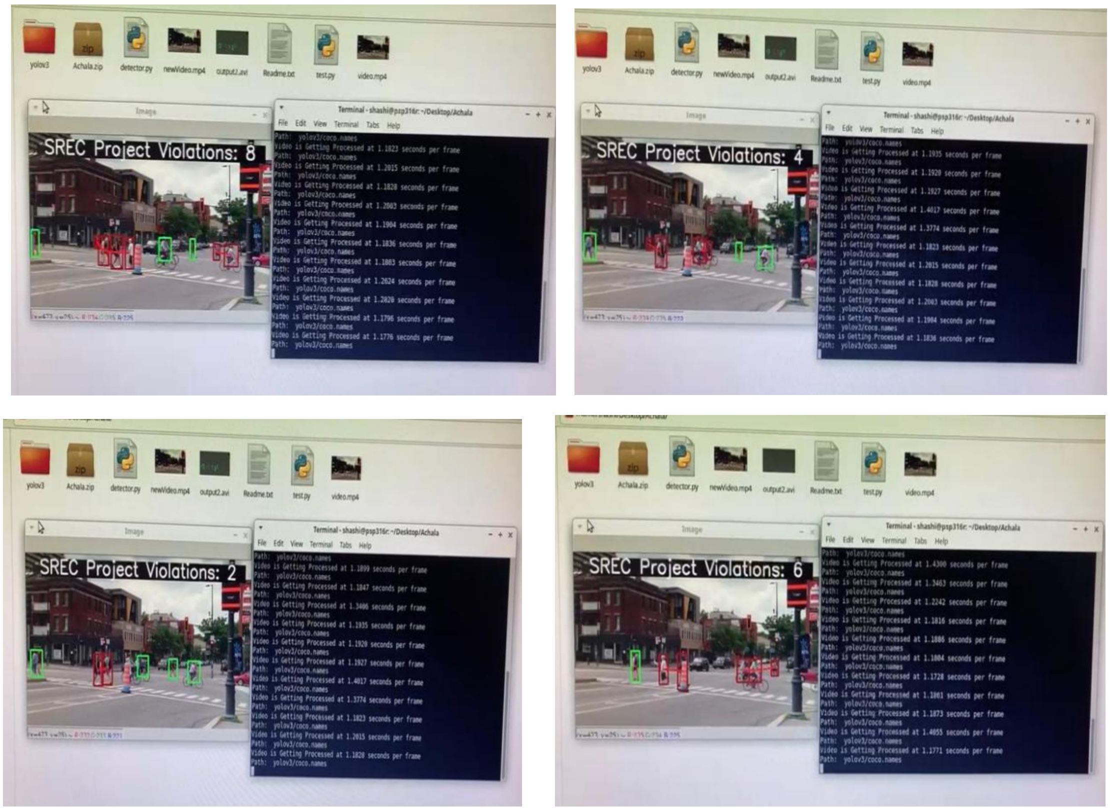


Figure 16: Results of social distance monitoring using YOLO algorithm

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

To conclude, in this paper we proposed a smart surveillance prototype based on the Raspberry Pi and Advanced AI algorithms for social distancing monitoring during coronavirus disease (COVID19). Our main goal is to warn people using vocal alerts in real-time while they are ignoring social distance protocols. To do so, we make a comparative study of some pretrained object detection models. As a result the SSD-MobileNet is doing a satisfying job. So the final prototype implemented the SSD model to detect persons on real-time, then we calculate the distance between each two detected persons. If two persons are too close, not respecting social distance a vocal alert is generated. Also if a group of persons are not respecting the mentioned distance a big vocal alert is generated. The most challenging part in this project is to fit the CNN based model to the Raspberry Pi which is limited in term of computing resources. At this time, the result covers our needs. However we are looking forward to improve the object detection model in order to increase the accuracy and speed of the system. In addition, it is recommended to use the latest version of the Raspberry Pi 4 which has a much more powerful processing power than the Raspberry Pi 3. So we suppose that it would improve the system quality and accuracy.

5.2 FUTURE SCOPE

As perspective, we are looking forward to improve our deep learning model taking on consideration 3D situations. For more innovation, other details could be added. For example, facemask detection systems to verify either people are wearing their mask or not. Even more, a thermal camera connected to the raspberry will check people with common symptoms, and many others additions we're working on to prevent spread of the COVID-19.

REFERENCES

[1] S. D. Chowdhury and A. M. Oommen, “Epidemiology of COVID-19,” pp. 3–7, 2020.

[2] E. M. Edrada *et al.*, “Erratum: First COVID-19 infections in the Philippines: A case report (Trop Med Health (2020) 48 (21) DOI: 10.1186/s41182-020-00203-0),” *Trop. Med. Health*, vol. 48, no. 1, 2020, doi: 10.1186/s41182-020-00218-7.

[3] M. S. Nadeem *et al.*, “Origin, potential therapeutic targets and treatment for coronavirus disease (COVID-19),” *Pathogens*, vol. 9, no. 4, pp. 1–13, 2020, doi: 10.3390/pathogens9040307.

[4] L. Morawska and J. Cao, “Airborne transmission of SARS-CoV-2: The world should face the reality,” *Environ. Int.*, vol. 139, no. April, p. 105730, 2020, doi: 10.1016/j.envint.2020.105730.

[5] C. T. Nguyen *et al.*, “Enabling and Emerging Technologies for Social Distancing: A Comprehensive Survey,” pp. 1–42, 2020, [Online]. Available: http://arxiv.org/abs/2005.02816.

[6] S. M. Kissler, C. Tedijanto, E. Goldstein, Y. Grad, and M. Lipstich, “Projecting the transmission dynamics of SARS-COV-2 through the post pandemic period,” *Science (80-.).*, vol. 5793, no. April, pp. 1–17, 2020.

[7] Jan Erik Solem, “Programming Computer Vision with Python,” *Program. Comput. Vis. with Python*, p. 264, 2012, doi: 10.1017/CBO9781107415324.004.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 580–587, 2014, doi: 10.1109/CVPR.2014.81.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *Proc. IEEE Comput. Soc. Conf. Comput3.. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.

[10] R. Gandhi, “R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms,” *Medium*, 2018. https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e.

[11] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” 2018, [Online]. Available: http://arxiv.org/abs/1804.02767.

[12] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” 2020, [Online]. Available: http://arxiv.org/abs/2004.10934.

[13] G. R. Valiati and D. Menotti, “Detecting Pedestrians with YOLOv3 and Semantic Segmentation Infusion,” *Int. Conf. Syst. Signals, Image Process.*, vol. 2019-June, pp. 95–100, 2019, doi: 10.1109/IWSSIP.2019.8787210.

[14] N. S. Punn, S. K. Sonbhadra, and S. Agarwal, “Monitoring COVID-19 social distancing with person detection and tracking via fine-tuned YOLO v3 and Deepsort techniques,” pp. 1–10, 2020, [Online]. Available: http://arxiv.org/abs/2005.01385.

[15] Q. C. Mao, H. M. Sun, Y. B. Liu, and R. S. Jia, “Mini-YOLOv3: Real-Time Object Detector for Embedded Applications,” *IEEE Access*, vol. 7, pp. 133529–133538, 2019, doi: 10.1109/ACCESS.2019.2941547.

[16] A. Wong, M. Famuori, M. J. Shafiee, F. Li, B. Chwyl, and J. Chung, “YOLO Nano: a Highly Compact You Only Look Once Convolutional Neural Network for Object Detection,” pp. 1–5, 2019, [Online]. Available: http://arxiv.org/abs/1910.01271.

[17] S. Cass, “Hands on,” *Spectr. IEEE Organ.*, no. c, pp. 14–16, 2020.

[18] V. Mazzia, A. Khaliq, F. Salvetti, and M. Chiaberge, “Real-time apple detection system using embedded systems with hardware accelerators: An edge AI application,” *IEEE Access*, vol. 8, pp. 9102–9114, 2020, doi: 10.1109/ACCESS.2020.2964608.

[19] World Health Organization, “Origin of SARS-CoV-2,” no. March, pp. 1–2, 2020, [Online]. Available: https://www.who.int/publications-detail/origin-of-sars-cov-2.

[20] L. Su *et al.*, “The different clinical characteristics of corona virus disease cases between children and their families in China–the character of children with COVID-19,” *Emerg. Microbes Infect.*, vol. 9, no. 1, pp. 707–713, 2020, doi: 10.1080/22221751.2020.1744483.

[21] G. David, R. S. Rye, and M. P. Agbulos, “COVID-19 FORECASTS IN THE PHILIPPINES: Insights for Policy making,” *Univ. Philipp.*, vol. 53, no. 9, pp. 1689–1699, 2020, doi: 10.1017/CBO9781107415324.004.

APPENDIX

Project Source code:

```
import numpy as np
import cv2
import imutils
import os
import time
def Check(a,  b):
    dist = ((a[0] - b[0]) ** 2 + 550 / ((a[1] + b[1]) / 2) * (a[1] - b[1]) ** 2) ** 0.5
    calibration = (a[1] + b[1]) / 2
    if 0 < dist < 0.25 * calibration:
        return True
    else:
        return False
def Setup(yolo):
    global net, ln, LABELS
    weights = os.path.sep.join([yolo, "yolov3.weights"])
    config = os.path.sep.join([yolo, "yolov3.cfg"])
    labelsPath = os.path.sep.join([yolo, "coco.names"])
    LABELS = open(labelsPath).read().strip().split("\n")
    net = cv2.dnn.readNetFromDarknet(config, weights)
    ln = net.getLayerNames()
    ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]
def ImageProcess(image):
    global processedImg
    (H, W) = (None, None)
    frame = image.copy()
    if W is None or H is None:
        (H, W) = frame.shape[:2]
    blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416), swapRB=True, 
crop=False)
    net.setInput(blob)
    starttime = time.time()
    layerOutputs = net.forward(ln)
    stoptime = time.time()
    print("Video is Getting Processed at {:.4f} seconds per frame".format((stoptime-
starttime)))
    confidences = []
    outline = []
    for output in layerOutputs:
        for detection in output:
            scores = detection[5:]
            maxi_class = np.argmax(scores)
            confidence = scores[maxi_class]
            if LABELS[maxi_class] == "person":
```

```
                if confidence > 0.5:
                    box = detection[0:4] * np.array([W, H, W, H])
                    (centerX, centerY, width, height) = box.astype("int")
                    x = int(centerX - (width / 2))
                    y = int(centerY - (height / 2))
                    outline.append([x, y, int(width), int(height)])
                    confidences.append(float(confidence))
    box_line = cv2.dnn.NMSBoxes(outline, confidences, 0.5, 0.3)
    if len(box_line) > 0:
        flat_box = box_line.flatten()
        pairs = []
        center = []
        status = []
        for i in flat_box:
            (x, y) = (outline[i][0], outline[i][1])
            (w, h) = (outline[i][2], outline[i][3])
            center.append([int(x + w / 2), int(y + h / 2)])
            status.append(False)
        for i in range(len(center)):
            for j in range(len(center)):
                close = Check(center[i], center[j])
                if close:
                    pairs.append([center[i], center[j]])
                    status[i] = True
                    status[j] = True
        index = 0
        for i in flat_box:
            (x, y) = (outline[i][0], outline[i][1])
            (w, h) = (outline[i][2], outline[i][3])
            if status[index] == True:
                cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 150), 2)
            elif status[index] == False:
                cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
            index += 1
        for h in pairs:
            cv2.line(frame, tuple(h[0]), tuple(h[1]), (0, 0, 255), 2)
    processedImg = frame.copy()
create = None
frameno = 0
filename = "newVideo.mp4"
yolo = "yolov3/"
opname = "output2.avi"
cap = cv2.VideoCapture(filename)
time1 = time.time()
while(True):
    ret, frame = cap.read()
    if not ret:
        break
    current_img = frame.copy()
    current_img = imutils.resize(current_img, width=480)
```

```
    video = current_img.shape
    frameno += 1
    if(frameno%2 == 0 or frameno == 1):
        Setup(yolo)
        ImageProcess(current_img)
        Frame = processedImg
        cv2.imshow("Image", Frame)
        if create is None:
            fourcc = cv2.VideoWriter_fourcc(*'XVID')
            create = cv2.VideoWriter(opname, fourcc, 30, (Frame.shape[1], 
Frame.shape[0]), True)
    create.write(Frame)
    if cv2.waitKey(1) & 0xFF == ord('s'):
        break
time2 = time.time()
print("Completed. Total Time Taken: {} minutes".format((time2-time1)/60))
cap.release()
cv2.destroyAllWindows()
```