**Report of Industrial Training**

at

**INTEL UNNATI PROGRAM**

**(ONLINE)**

***In partial fulfilment of the requirements for the award of the degree of***

**Bachelor of Technology**

**in**

**Electronics and Communication Engineering**

*Submitted by*

# Student Name: ANKUSH VERMA

Register Number: 200907058

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

## MANIPAL INSTITUTE OF TECHNOLOGY

(A Constituent institute of MAHE, Manipal)

MANIPAL – 576104, KARNATAKA, INDIA

**JULY 2023**

**PROJECT: SOCIAL DISTANCING USING COMPUTER VISION AND DEEP LEARNING**

**I. INTRODUCTION:**

In this project, I aim to address the crucial issue of social distancing in public spaces. By harnessing the power of computer vision and deep learning techniques, my system detects instances where the distance between individuals falls below a specified threshold.

By analysing video frames, my system identifies and quantifies violations of social distancing guidelines. Through a user-friendly interface, you can observe real-time feedback on the number of violations within each frame.

To optimize my solution, I conducted a comparative analysis between two versions: an optimized implementation leveraging the Intel Open VINO toolkit and an unoptimized version without Open VINO.

Using Open VINO, I achieved significant enhancements in performance. With the optimized version, you can conveniently monitor the frames per second (FPS) in the top left corner, ensuring efficient processing. Furthermore, the bottom section of the display provides an accurate count of social distancing violations.

My system employs the person-detection-0202 model, which utilizes the MobileNetV2 backbone, and two SSD heads based on 1/16 and 1/8 scale feature maps. This model, pretrained using the Intel Open VINO toolkit, is specifically tailored for person detection at a resolution of 512x512.

To visually represent social distancing, my system draws a red line between individuals who fail to maintain the specified threshold distance. It is important to note that in cases where multiple individuals are detected within the threshold distance, each pair is considered a separate violation. This approach may lead to a slight increase in the cumulative violation count.

**II. PROPOSED SYSTEM OVERVIEW:**

The proposed system, a Social Distance detector and management system, is designed using various components including Computer Vision techniques (CV), a Deep learning structure, and the Python.

1. Subject Detection: To detect humans in video frames or images, we utilize the YOLOv3 object detection algorithm. The video stream is processed frame by frame.

2. Human Classification: By applying YOLO object detection, we specifically identify the "Humans" class using a convolutional neural network that has been trained for object detection.

3. Bounding Boxes and Pairwise Distances: We deploy bounding boxes around humans and calculate pairwise distances based on the centroids of these rectangular boxes.

4. Distance Threshold Check: Our system verifies whether the pixel distance between adjacent individuals or boxes falls below a predefined threshold value.

5. Violation Count and Database Storage: We maintain a count of the detected violations and store this information in a local database. This data serves as the foundation for generating the dashboard through the Flask framework.

A diagram of a flowchart

Description automatically generated

WORKFLOW OF PROPOSED SYSTEM

**IV. METHODOLOGY SYNOPSIS:**

This section outlines the detailed design methodology of the project, focusing on the essential components and working principles.

1. INPUT FRAMES:

The project begins by feeding the model with image or video data obtained from a closed-circuit television (CCTV) system. The data is segmented into individual frames for processing. The frames are transformed into a 2D bird's eye view with dimensions of 448 pixels on all sides using OpenCV. This transformation enables improved depth perception and facilitates distance calculations between individuals.

2. OBJECT DETECTION AND TRACKING:

Convolutional Neural Networks (CNN) are employed to efficiently recognize objects. The model focuses on identifying areas belonging to the "Person" class, utilizing region proposals and the Intersection Over Union (IOU) score to extract relevant regions. The You Only Look Once (YOLO) model, specifically YOLOv3, is used for real-time object detection. It predicts bounding box coordinates, class probabilities, and object confidence simultaneously. Non-maximum suppression eliminates overlapping bounding boxes, ensuring accurate detections.

A diagram of a flowchart

Description automatically generated

DETECTION OF PEOPLE USING YOLOV3

3. DISTANCE CALCULATION:

After person recognition and bounding box creation, centroid coordinates are determined. Euclidean distance estimation is used to calculate the distances between centroids, representing the distances between individuals. Violation pairs, where the distance is less than a specified threshold, are identified with red bounding boxes and counted as violations. Non-violation pairs are highlighted in green.

The process is repeated for each frame of the video, accumulating the total number of violations. The violation count is stored in a local database for reference in the Flask dashboard.

By following these steps, the project achieves accurate social distance violation detection and provides comprehensive output for analysis.

A diagram of a complex function

Description automatically generated

DISTANCE MEASURED BETWEEN TWO DETECTED BOXES

**IV. FINAL ACCURACY:** By incorporating the Open VINO toolkit, my system experiences a remarkable eight-fold increase in FPS compared to the non-Open VINO model, enhancing overall efficiency and responsiveness. The FPS of the output video significantly increased from 2-5 to 15-20, surpassing the performance without Open VINO.

WITH OPENVINO WITHOUT OPENVINO

A group of people walking on a street

Description automatically generated

A group of people walking on a street

Description automatically generated