SDD (Social Distancing Detection)

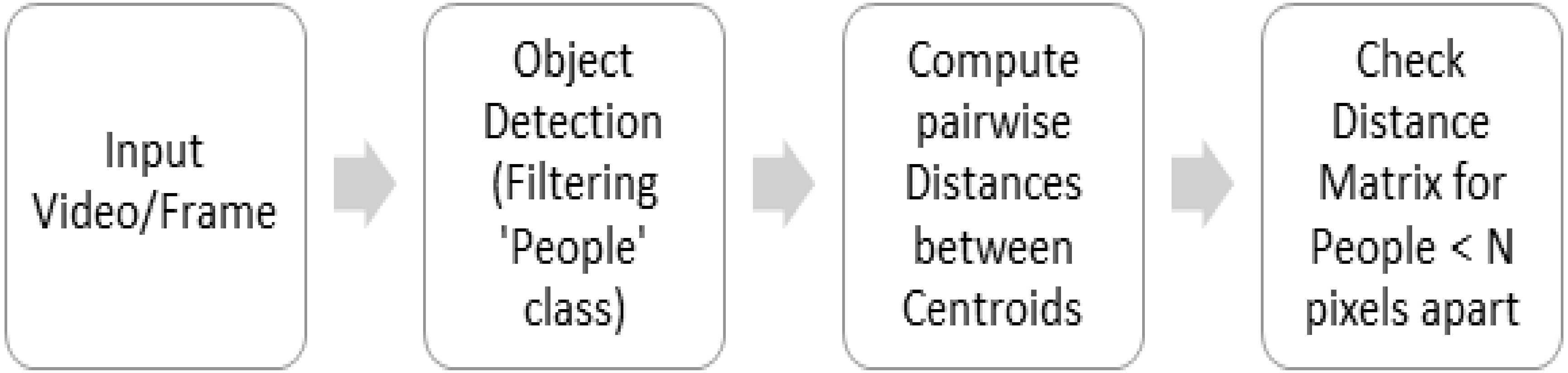
[(Computer Vision- Deep Learning-OpenCV based]

Objective:

Development of Social Distancing Detection model for monitoring healthy distance among people. And hence one can maintain good health.

Benefits:

- Object detection (person)
- Distance measurement
- Draw different colour boxes for people following safe distance or not
- Display social distancing output with count of violations
- Manual inspection if required



Input Video/Frame

Object Detection (Filtering 'People' class)

Compute pairwise Distances between Centroids

Check Distance Matrix for People < N pixels apart

The steps involved in an OpenCV-based social distancing application.

--Python Programming Language and frameworks such as NumPy, imutils and OS

--*PyCharm* is used a IDE

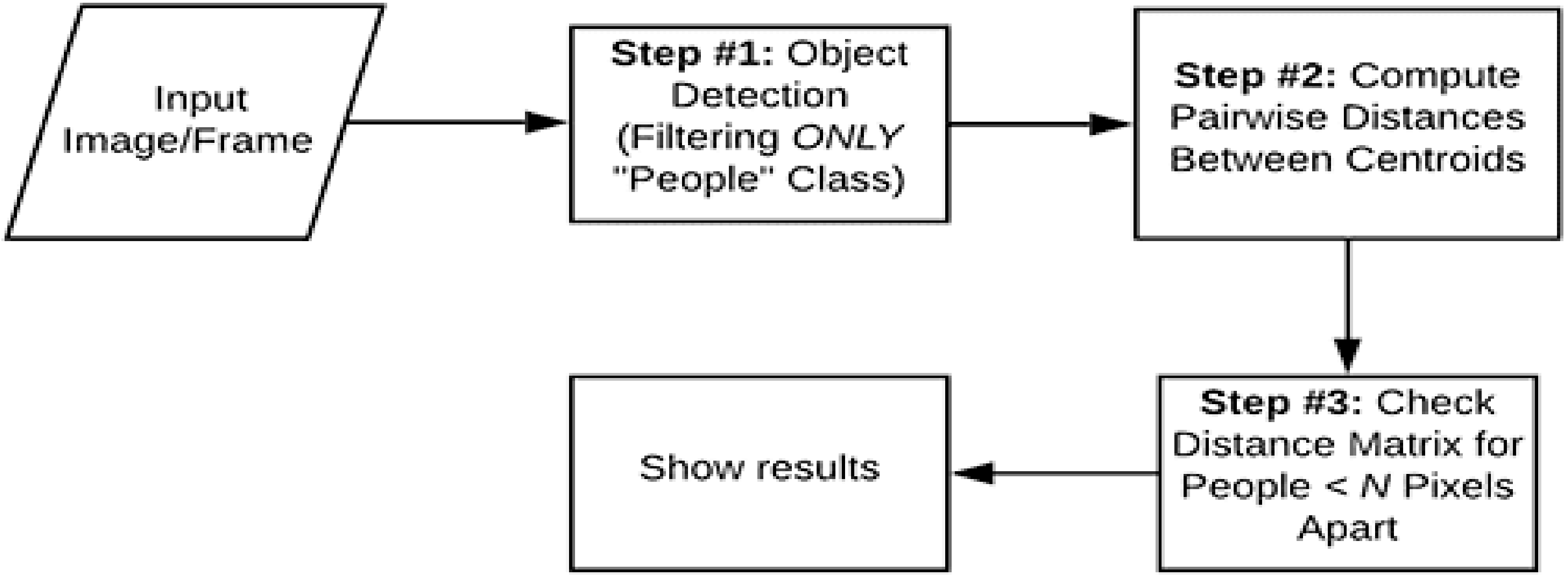
--*GitHub* is used for making all works available to the public

--*OpenCV* is used for people detection

---*Yolo* module gives necessary files to load CNN model for people detection.

---*Colab* notebook for speed up the process.

Methodology



Input Image/Frame

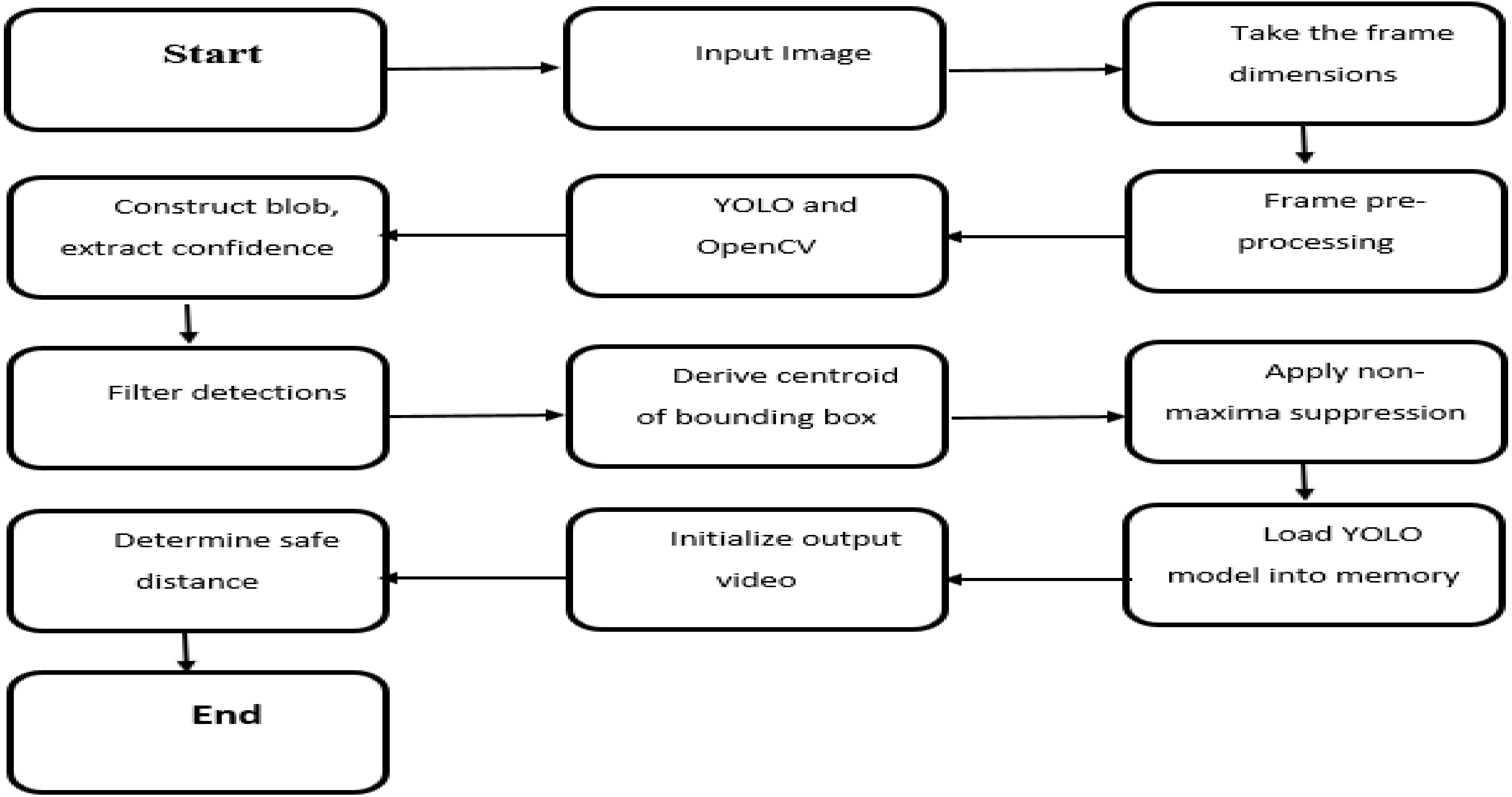
Step #1: Object Detection (Filtering *ONLY* "People" Class)

Step #2: Compute Pairwise Distances Between Centroids

Step #3: Check Distance Matrix for People < *N* Pixels Apart

Show results

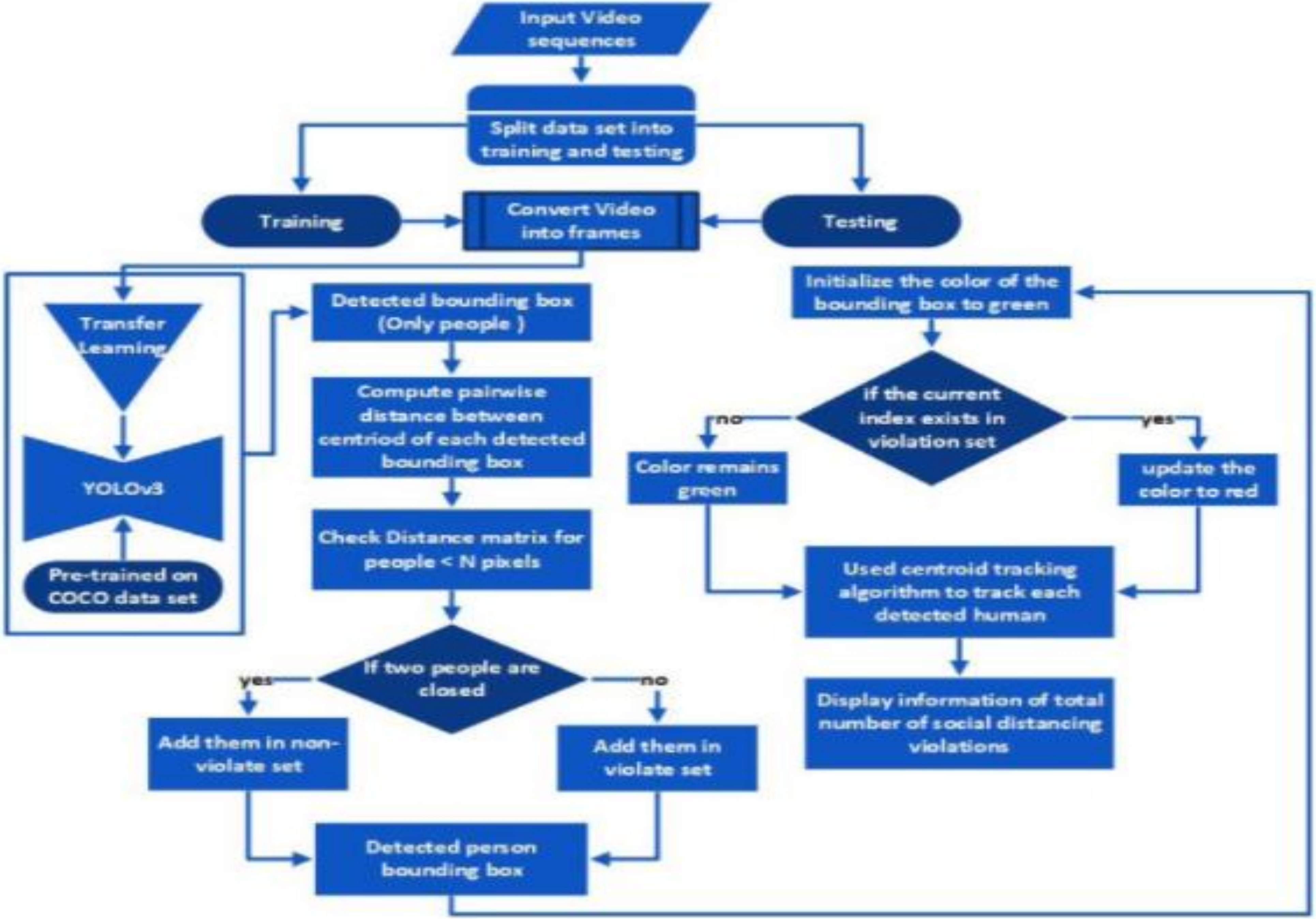
Architecture


Start
Input Image
Take the frame dimensions
Frame pre-processing
YOLO and OpenCV
Construct blob, extract confidence
Filter detections
Derive centroid of bounding box
Apply non-maxima suppression
Load YOLO model into memory
Initialize output video
Determine safe distance
End

Key performance indicators (KPI)

- Person detection
- Displaying healthy and unhealthy distances among people
- Saving output for future reference
- Successful logging
- Modular coding
- Exception’s handling

Workflow



Input Video sequences

Split data set into training and testing

Training

Convert Video into frames

Testing

Transfer Learning

YOLOv3

Pre-trained on COCO data set

Detected bounding box (Only people)

Compute pairwise distance between centriod of each detected bounding box

Check Distance matrix for people < N pixels

If two people are closed

yes

Add them in non-violate set

no

Add them in violate set

Detected person bounding box

Initialize the color of the bounding box to green

if the current index exists in violation set

no

Color remains green

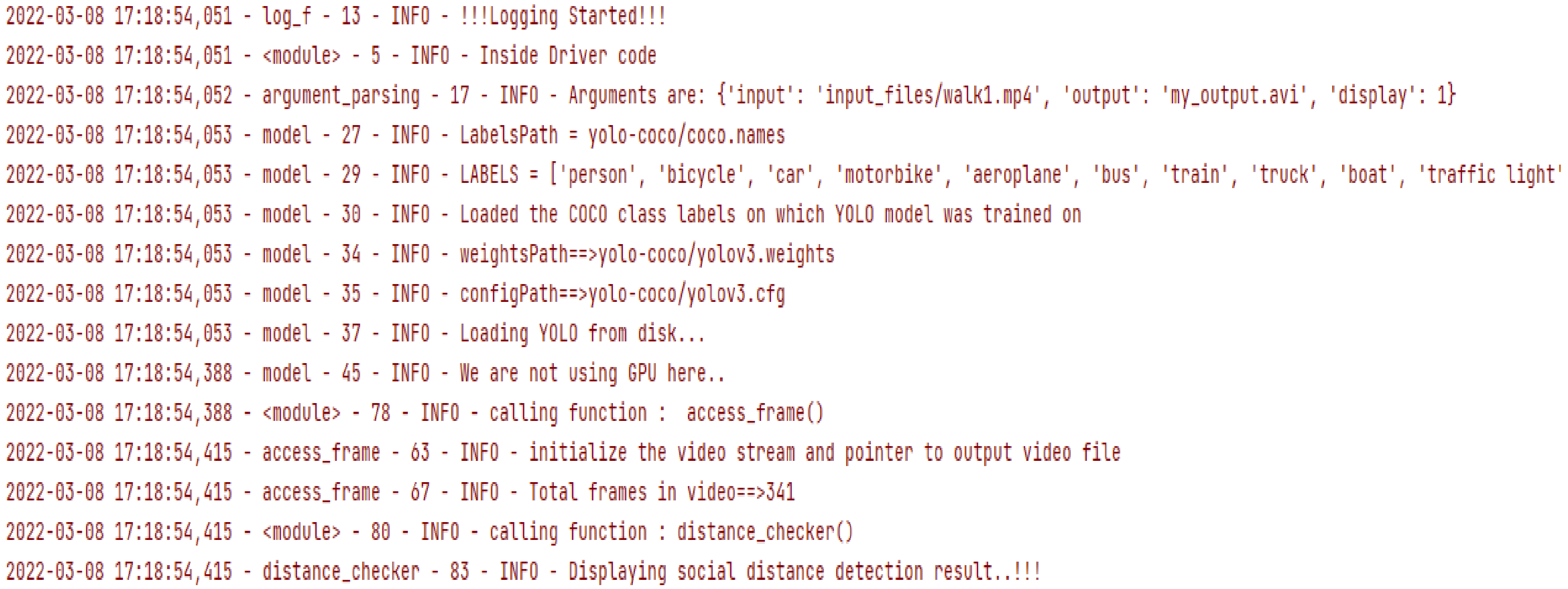
yes

update the color to red

Used centroid tracking algorithm to track each detected human

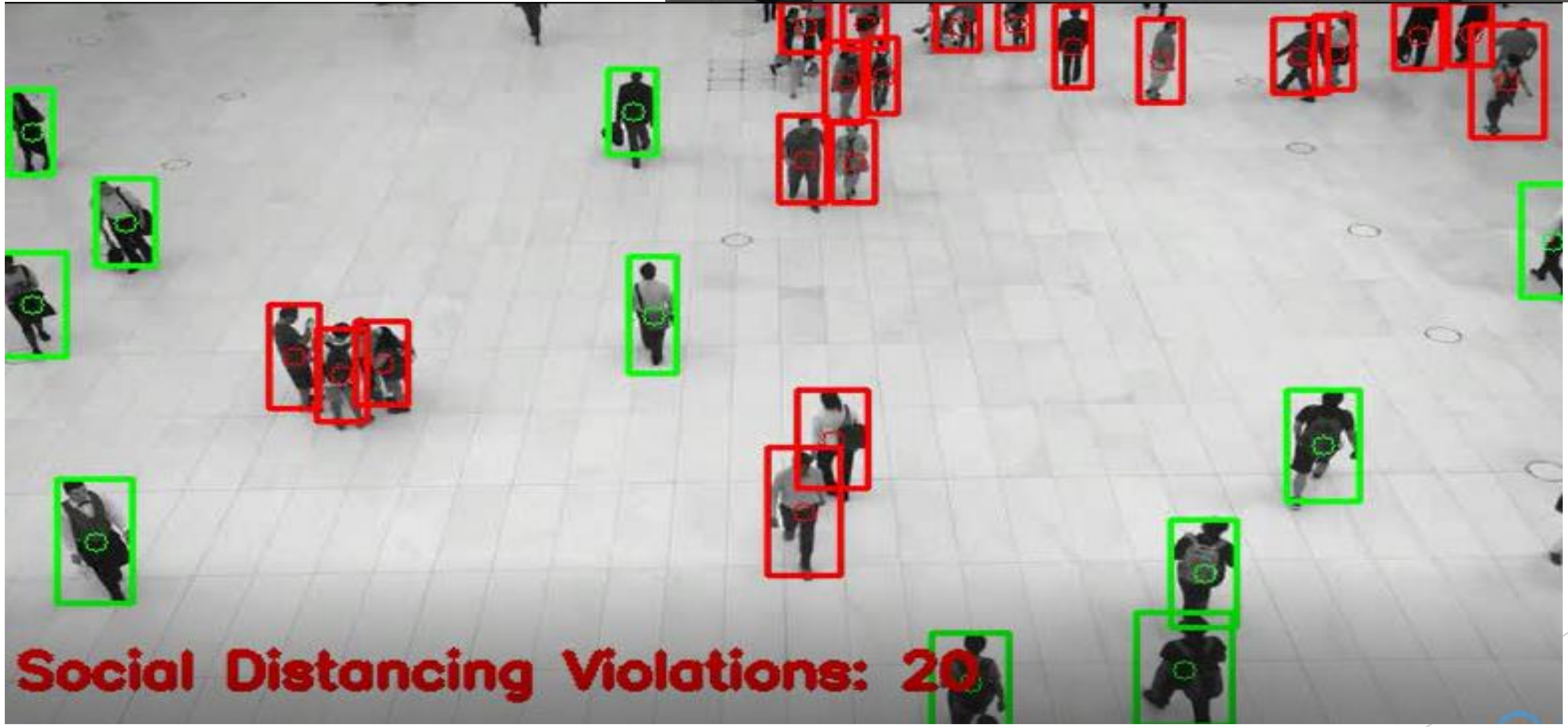
Display information of total number of social distancing violations

Log file



```
2022-03-08 17:18:54,051 - log_f - 13 - INFO - !!!Logging Started!!!
2022-03-08 17:18:54,051 - <module> - 5 - INFO - Inside Driver code
2022-03-08 17:18:54,052 - argument_parsing - 17 - INFO - Arguments are: {'input': 'input_files/walk1.mp4', 'output': 'my_output.avi', 'display': 1}
2022-03-08 17:18:54,053 - model - 27 - INFO - LabelsPath = yolo-coco/coco.names
2022-03-08 17:18:54,053 - model - 29 - INFO - LABELS = ['person', 'bicycle', 'car', 'motorbike', 'aeroplane', 'bus', 'train', 'truck', 'boat', 'traffic light'
2022-03-08 17:18:54,053 - model - 30 - INFO - Loaded the COCO class labels on which YOLO model was trained on
2022-03-08 17:18:54,053 - model - 34 - INFO - weightsPath==>yolo-coco/yolov3.weights
2022-03-08 17:18:54,053 - model - 35 - INFO - configPath==>yolo-coco/yolov3.cfg
2022-03-08 17:18:54,053 - model - 37 - INFO - Loading YOLO from disk...
2022-03-08 17:18:54,388 - model - 45 - INFO - We are not using GPU here..
2022-03-08 17:18:54,388 - <module> - 78 - INFO - calling function :  access_frame()
2022-03-08 17:18:54,415 - access_frame - 63 - INFO - initialize the video stream and pointer to output video file
2022-03-08 17:18:54,415 - access_frame - 67 - INFO - Total frames in video==>341
2022-03-08 17:18:54,415 - <module> - 80 - INFO - calling function : distance_checker()
2022-03-08 17:18:54,415 - distance_checker - 83 - INFO - Displaying social distance detection result..!!!
```

Sample input and output



Q & A

- **What are the tools & techniques used here?**

OpenCV, Deep learning, computer vision

- **Explain YOLO object detection**

[Refer LLD report] “The YOLO framework (You Only Look Once) deals with object detection in a different way, It takes the entire image in a single instance and predicts the bounding box coordinates and class probabilities for these boxes. **The biggest advantage of using YOLO is its superb speed** — it’s incredibly fast and can process 45 frames per second. YOLO also understands generalized object representation”

- **How OpenCV is applied to this project?**

- **What are KPIs?**

Refer HLD report

- **Can we use webcam input?**

Yes.

- **What are possible future enhancements?**

Finding algorithm for finding distance in meters or feet's in place of Pixel.

- **How logs are maintained?**

By console logging and file logging.

- **What is computer vision?**

“**Computer vision** is a process by which we can understand the images and videos how they are stored and how we can manipulate and retrieve data from them. Computer Vision is the base or mostly used for Artificial Intelligence. Computer-Vision is playing a major role in self-driving cars, robotics as well as in photo correction apps.”

- **Define OpenCV?**

“OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human.”

- Can you explain input and output of **detect_people(frame, net, ln, personIdx = 0)** line of code?

- **frame :** The frame from your video file or directly from the webcam
- **net :** The pre-initialized and pre-trained YOLO object detection model
- **ln :** The YOLO CNN output layer names
- **personIdx :** The YOLO model can detect many types of objects; this index is specifically for the person class, as we won’t be considering other objects

The results consist of (1) the person prediction probability, (2) bounding box coordinates for the detection, and (3) the **centroid** of the object.

- **What comes next to people_detection?**
- Compute the Euclidean distance between all pairs of centroids
- Loop over the upper triangular of distance matrix (since the matrix is symmetrical)
- Check to see if the distance violates our minimum social distance set forth by public health professionals. If two people are too close, we add them to the *violate* set.

In detail :

- Looping over the results , we proceed to:
- Extract the bounding box and centroid coordinates
- Initialize the color of the bounding box to *green*
- Check to see if the current index exists in the *violate* set, and if so, update the color to *red*
- Draw both the bounding box of the person and their object centroid. Each is color-coordinated, so we'll see which people are too close.
- Display information on the total number of social distancing violations (the length of the *violate* set) .