# Counting Crowds in Real Time using OpenCV

Sai Sandeep Manne
University of Maryland Baltimore County
CMSC 491/691 Final Report
Spring 2021, Group 29

***Abstract-***
***The primary purpose of this project is to contribute to the assessment of the feasibility of developing a technical solution that allows for the use of security cameras to count individuals in public buildings. My contribution is to analyze current real-time computer vision approaches in order to develop a system that can count the number of people in difficult surroundings reliably. I will contribute to the creation of a real-time people counter in complicated areas employing OpenCV and Deep Learning libraries in particular.***

## 1. INTRODUCTION

People's behavior must be observed, tracked, especially in public areas such as malls and stadiums, where the safety of the public and security are of paramount importance. With vast crowds, it is hard for a human monitoring operator to accurately observe the crowds' behavior. As a result, crowd management systems that can count people are being developed with the goal of assisting these human surveillance operators. Counting people on the street or at entrances to venues is useful for security, tracking, and marketing. Over the last few years, the use of cameras and digitizers for various reasons has increased dramatically. In the past, CCTV surveillance systems required a human operator to monitor the camera's input. The development of an automatic person's counter to aid in the action has become a need.

Real-time crowd counts will assist outlets in developing future plans and use this data to properly estimate and forecast crowd counts. In the event of an overcrowding situation, outlets will be able to accommodate the crowd. The dynamism of people's mobility cannot be accounted for in any of the previous methods of crowd counting. This necessitates a forward-thinking approach to the issue. Emergency circumstances such as fires, earthquakes, and other disasters can benefit from an accurate crowd counting system. In these circumstances, a crowd estimate would enable the responsible authorities to make the best decisions possible on resource supplies. Techniques like crowd counting employing detection, density, and regression have recently been proposed as answers to the challenge of an accurate estimate of crowd count. We'll use both OpenCV and Deep Learning libraries to make our people counter. For standard computer vision/image processing functions, we'll utilize OpenCV, and for people counting, we'll use the deep learning object detector. After that, we'll utilize Deep Learning libraries to construct correlation filters.

## 2. RELATED WORK

[1] The paper discusses about the awareness of the crowd counting and also about the pattern recognition. The paper does explain about the different ways of the pattern recognition available and also the different approaches of the object detections.

[2] The paper discusses about Hierarchial model approach based on counting of the people. This paper says about the use cases of the hierarchial approach and the evaluation of this approach among the other methods.

[3] The paper points the steps which are required in the detection of the objects from a picture. Like there are different steps involved in the object detection and also the object tracking. The paper explains these steps with the use-cases in each step.

[4] The paper helps to find out the total number of people in a crowded place based on the head shoulder segementation and forehead segmentation, which will be useful for the image segmentation.

[6] The paper describes the regions for the detection of the objects and also the creation of the object ID's to the detected objects in the particular frame of the input video.

[5][7] These are the papers which are helpful in finding the difference between the Object Detection and the Object tracking. And also, these papers explain about the different steps involved in the detection and tracking of the objects using various algorithms which leads to the successful implementation of the counting of the people in the real time scenario cases.

## 3. METHODOLOGY

The first step of the project will be **Video Pre- Processing** which is resizing and converting to rgb are used to pre-process frames. OpenCV is a library that may be used to do standard image processing and computer vision tasks. OpenCV will be used to do deep neural network inference, video file reading and writing, and output frame display on our screen. And the next step would be **Object Detection**, which run our object tracker during the detection phase to detect if new objects have entered our view and see if we can find objects that were "lost" during the tracking phase. We construct or update an object tracker with the new bounding box coordinates for each identified object. We only run this phase once every N frames because our object detector is more computationally intensive. There are three main object detection methods based on deep learning. 1. Faster R-CNNs 2. You Only Look Once (YOLO) 3. Single Shot Detectors (SSDs).

We achieve a quick, efficient deep learning-based object identification approach by combining the Mobile-Net architecture and the Single Shot Detector (SSD) framework. In this project, we'll be using a Caffe version of Mobile-Net SSD as our model. The Mobile-Net SSD was trained on the COCO dataset (Common Objects in Context). Mobile-net SSD contains all the pretrained deep learning model files which is used to detect the objects using a convolutional filter. In the **Object Tracking,** we construct an object tracker for each of our detected items to follow it around the frame. Compared to the object detector, our object tracker should be quicker and more efficient. We'll keep tracking until we reach frame N, at which point we'll re-run our object detector. The cycle then repeats itself. The algorithm we have used is the centroid tracking algorithm. It is based on the Euclidean distance between existing object centroids and new object centroids between video frames. The centroid tracking algorithm consists of several steps. Firstly, we calculate centroids using bounding box coordinates. Then we calculate the Euclidean distance between existing items and new bounding boxes. Then we update (x, y)-coordinates of existing objects. And then we register new objects. Lastly, we deregister old objects.

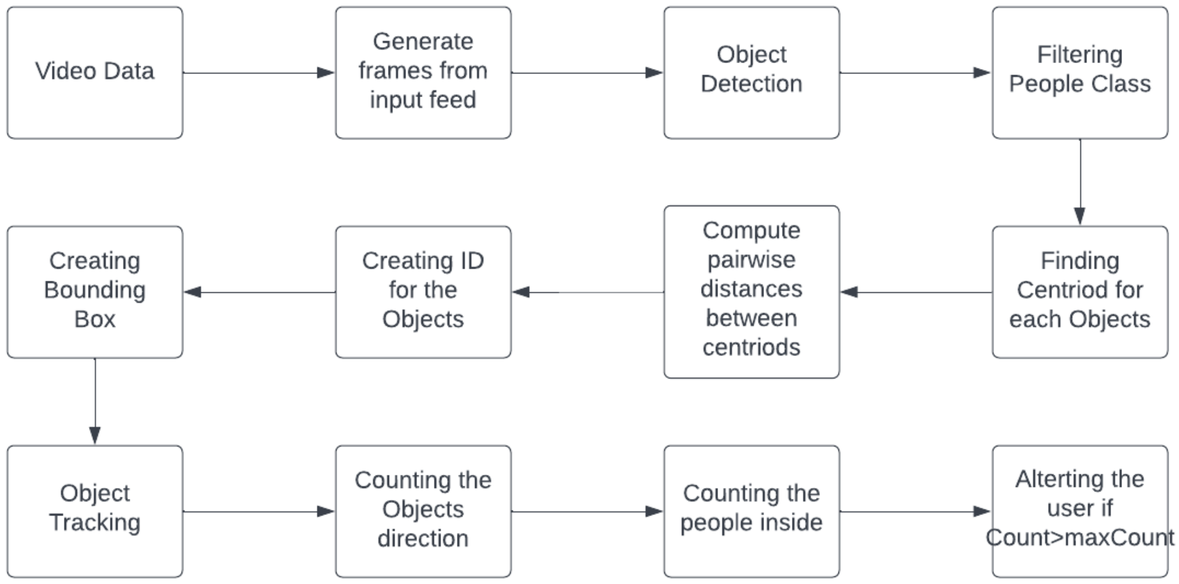

Video Data
Generate frames from input feed
Object Detection
Filtering People Class
Finding Centriod for each Objects
Compute pairwise distances between centriods
Creating ID for the Objects
Creating Bounding Box
Object Tracking
Counting the Objects direction
Counting the people inside
Alterting the user if Count>maxCount

*Figure 1. Flow diagram showing all the steps in the order required for Object Detection and Tracking of the people in the input video*

## 4. RESULTS

The quantitative parameters of this project will be in terms of FPS, Elapsed Time and the count of the people. The FPS and the Elapsed time will be found in the command prompt window. And the Count of the people will be found in the Analyzing window. And the qualitative analysis is all about the People Recognition and the people tracking, which is achieved in the below figures.

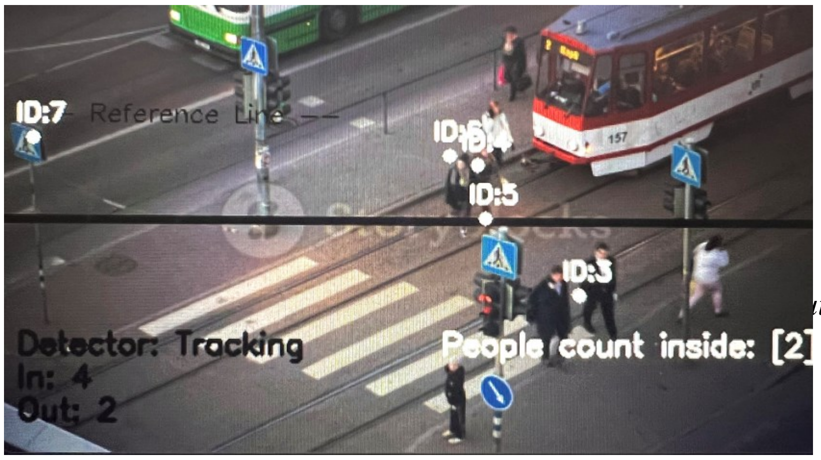


*Figure 2.* *t video 1*

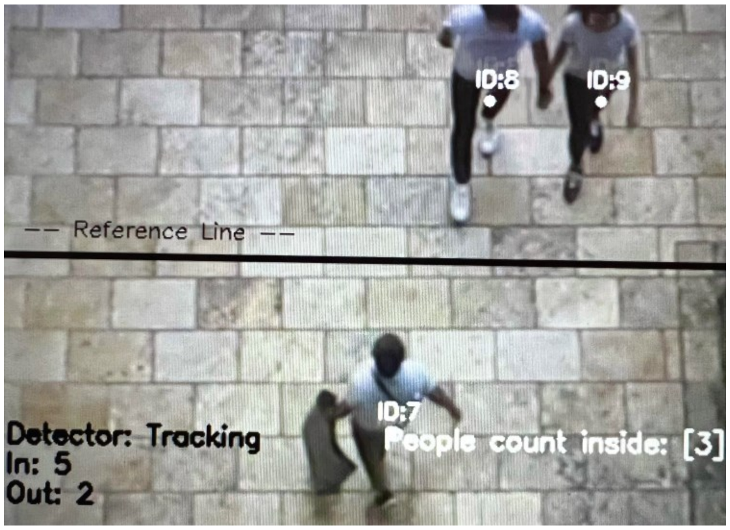
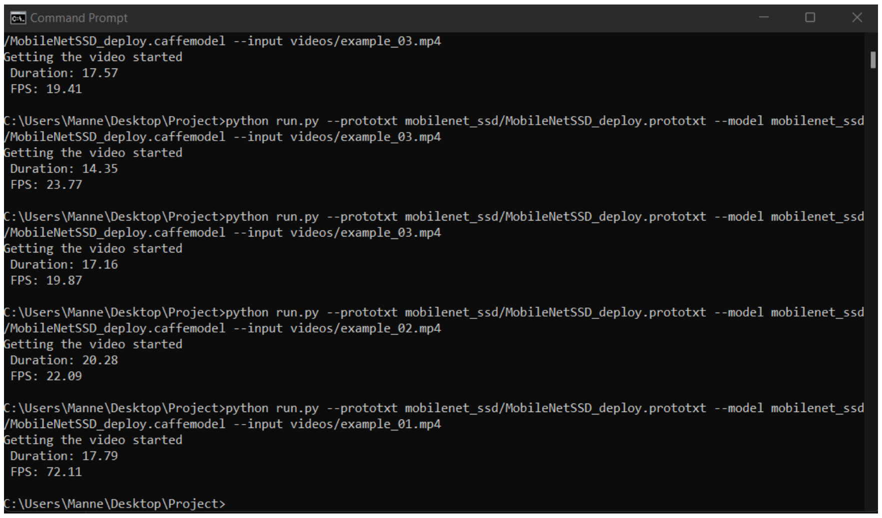


*Figure 3. Shows the detector status and the In and Out count of the input video 2 with respect to the reference line*

Figure 2 and Figure 3 represents our output snapshots of the analyzing window. It describes the status of our detector whether it is tracking or detecting. And also from figure 2, the people count inside is 2. And from the figure 3, the people count is 3 inside and the status of the detector is in the tracking mode.



```
/MobileNetSSD_deploy.caffemodel --input videos/example_03.mp4
Getting the video started
 Duration: 17.57
 FPS: 19.41

C:\Users\Manne\Desktop\Project>python run.py --prototxt mobilenet_ssd/MobileNetSSD_deploy.prototxt --model mobilenet_ssd
/MobileNetSSD_deploy.caffemodel --input videos/example_03.mp4
Getting the video started
 Duration: 14.35
 FPS: 23.77

C:\Users\Manne\Desktop\Project>python run.py --prototxt mobilenet_ssd/MobileNetSSD_deploy.prototxt --model mobilenet_ssd
/MobileNetSSD_deploy.caffemodel --input videos/example_03.mp4
Getting the video started
 Duration: 17.16
 FPS: 19.87

C:\Users\Manne\Desktop\Project>python run.py --prototxt mobilenet_ssd/MobileNetSSD_deploy.prototxt --model mobilenet_ssd
/MobileNetSSD_deploy.caffemodel --input videos/example_02.mp4
Getting the video started
 Duration: 20.28
 FPS: 22.09

C:\Users\Manne\Desktop\Project>python run.py --prototxt mobilenet_ssd/MobileNetSSD_deploy.prototxt --model mobilenet_ssd
/MobileNetSSD_deploy.caffemodel --input videos/example_01.mp4
Getting the video started
 Duration: 17.79
 FPS: 72.11

C:\Users\Manne\Desktop\Project>
```

*Figure 4. Shows the FPS and the Duration for the processing of the video*

The Figure 4 tells us about the FPS Rate and the Duration for the processing of the video. The above example has an FPS of 72.11 and the elapsed time is 17.79 seconds. So, when we check the above snapshot, we can clearly see the increase of the FPS from the different videos analyzed. This is due to the change of the different parameters in the model. So, the FPS had changed 23.7

to 20 which is a good start. And then when we tested the model with another video then we got the FPS rate to 22 and 77 and with the lesser lapsed time, which is the best FPS rates which can be achieved with our Mobile-Net model.

## 5. CONCLUSION

We created a people counter using OpenCV and Python. This system is capable of running in real time on a regular CPU. Deep learning object detectors are used to improve the accuracy of human detection. It also employs Centroid tracking and correlation filters, which are two independent object tracking techniques that allow it to discover new persons as well as recover people who may have become stuck during the tracking process, for enhanced tracking accuracy. It is possible to include a model that determines the distance between the bounding boxes, which increases the accuracy of the violation. Item identification performance in image processing is becoming increasingly important for a rising variety of real-time applications, and we can detect any sort of object with this application. At the future, we will utilize a variety of extraction process approaches for diverse objectives, and this technique can be used in airports, shopping malls, enterprises, parks, and other locations.

## REFERENCES

1. W. Liu, M. Salzmann, and P. Fua, Context-aware crowd counting, in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 50995108.

2. V. A. Sindagi and V. M. Patel, HA-CCN: Hierarchical attention based crowd counting network, IEEE Trans. Image Process., vol. 29, pp. 323335, 2020.

3. W. Ge and R. T. Collins, Marked point processes for crowd counting, in Proc. CVPR, Jun. 2009, pp. 29132920.

4. M. Li, Z. Zhang, K. Huang, and T. Tan, Estimating the number of people in crowded scenes by MID based foreground segmentation and head shoulder detection, in Proc. 19th Int.

5. Conf. Pattern Recognit., Dec. 2008

6. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, in arXiv:1506.01497v3 [cs.CV] 6 Jan 2016

7. Desen Zhou and Qian He, Cascaded Multi-Task Learning of Head Segmentation and Density Regression for RGBD Crowd Counting, Digital Object Identifier 10.1109/ACCESS.2020.2998678, June 10, 2020

8. https://arxiv.org/abs/1512.02325

9. https://betterprogramming.pub/using-python-pandas-with-excel-d5082102ca27

10. https://www.ijert.org/real-time-crowd-counting-using-opencv

11. https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11

12. https://arxiv.org/abs/1704.04861

13. https://core.ac.uk/download/pdf/47139441.pdf