



# Tailgating Detection

*Classy-fiers – IIT Guwahati*





# Problem Overview

Given CCTV camera footage of a secured door entry, detect anyone tailgating (entry without authorization).

# Evaluation Criteria

- I. Detection &  
Segmentation  
of humans in  
video.
- II. Tailgating  
Detection



# Objectives

## Segmentation

Identifying and separating humans from the background.



## Tailgating Detection

Detecting a possible tailgater using A.I & Deep learning.



## Alerting Security

Alerting relevant personnel through mail and/or a physical alarm.



## Identifying Intruder

Providing information about the alleged intruder







**Solutions**



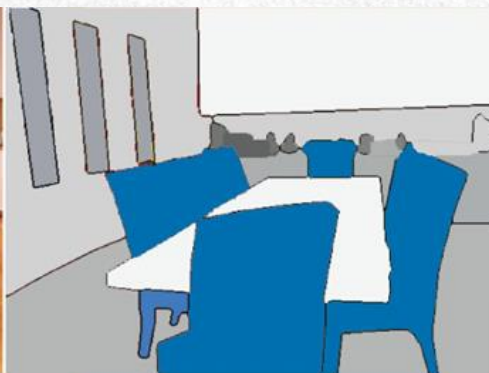
# 1. Segmentation

Two main types:

- Semantic
- Instance



Input Image

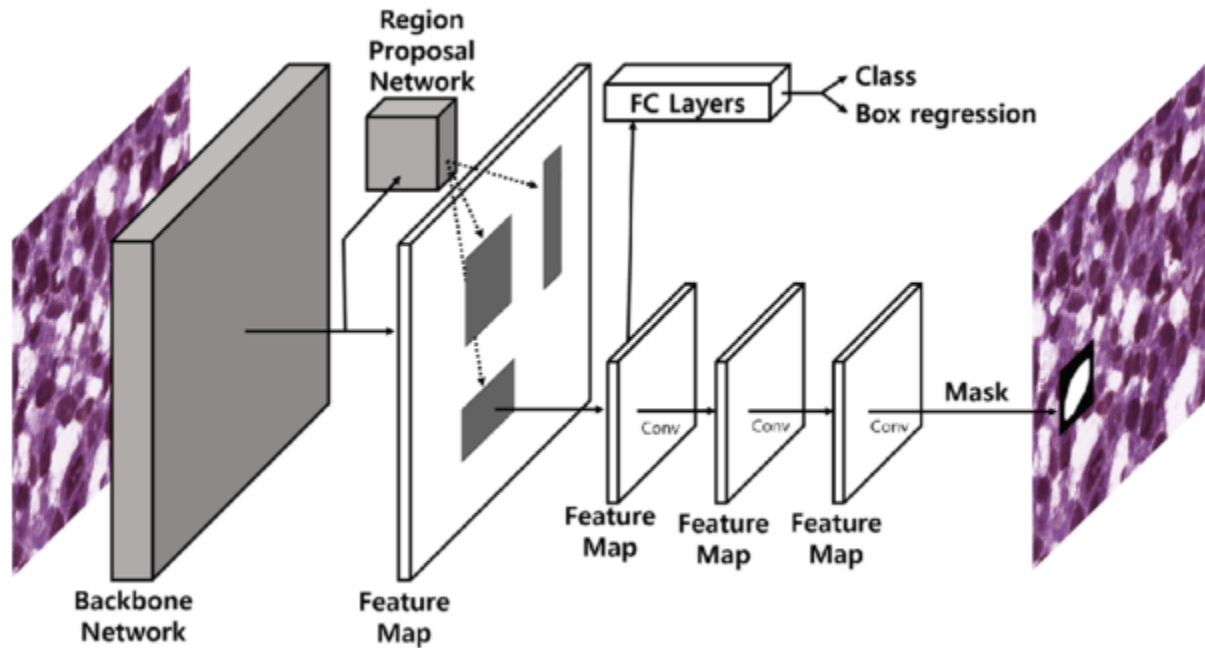


Semantic Segmentation



Instance Segmentation

- Instance segmentation will be our proposed approach as we only need to identify humans in our CCTV Video Clips.
- We generate our masks using Mask RCNN. Mask RCNN is a deep neural network aimed to solve instance segmentation problem in machine learning or computer vision. In other words, it can separate different objects in a image or a video. You give it a image, it gives you the object bounding boxes, classes and masks.
- There are two stages of Mask RCNN. First, it generates proposals about the regions where there might be an object based on the input image. Second, it predicts the class of the object, refines the bounding box and generates a mask in pixel level of the object based on the first stage proposal.



Mask R-CNN is an extension of Faster R-CNN.



- A pre-trained Mask RCNN model is used in our project, as it is very tedious to train this architecture by ourselves based on our current computational capabilities.
- The pre-trained weights are trained on the COCO Dataset, which consists of around 91 classes. But for our specific problem, we only need Humans as the class/mask output. Hence *using a pre-trained Mask RCNN would prove to be computationally inefficient.*
- To tackle this problem, we employed Transfer Learning, and finetuned the pre-existing architecture to predict only humans. Transfer Learning was achieved using the Penn Fudan Database, which is a pedestrian dataset. Hence our model now predicts and outputs only humans, thus reducing computation costs significantly.







## 2. Tailgating Detection

To detect tailgating, we need to identify people who enter behind legitimate cardholders (i.e. who swipe their cards at the entry). Assuming a signal input to our model (either from a CSV or real-time) indicating a swiped card, we have to count the number of people entering between two successive swipes.

To tackle this, we divide our approach into two stages:

- Centroid Tracking
- Surveillance Area

Let's elaborate more on these:

# STAGE 1 - Centroid Tracking:

To track the movement of humans in a frame, we aim to follow their centroid. This can be performed in two ways:

- Mask R-CNN
- KCF / MOSSE/ CSRT

## **Mask R-CNN:**

We can exploit the fact that Mask R-CNN outputs bounding boxes along with masks. Using the box coordinates, we can find out the centroid of that human and track its movement. The only drawback of this method is it's computationally expensive, as running the Mask R-CNN model on every frame (or every few frames) proves to be a task.



## KCF / MOSSE / CSRT:

To overcome the frame rate drop, we can utilise inbuilt OpenCV trackers.

Our centroid tracker worked well, but it required us to run an actual object detector on each input video frame. Ideally, we would like to apply object detection only once and then have the object tracker handle every subsequent frame, leading to a faster, more efficient object tracking pipeline.

A gist of the trackers we suggest to use:

- I. **KCF Tracker:** A novel tracking framework that utilises properties of circulant matrix to enhance the processing speed. This tracking method is an implementation of which is extended to KCF with colour-names features. It does not handle total occlusion well.
- II. **MOSSE Tracker:** The implementation is based on Visual Object Tracking using Adaptive Correlation Filters. Very, very fast. Not as accurate as CSRT or KCF but a good choice if you need pure speed.
- III. **CSRT Tracker:** The implementation is based on Discriminative Correlation Filter with Channel and Spatial Reliability. It tends to be more accurate than KCF but slightly slower.

By implementing any one of these ideas mentioned before, we can track the path of every human in a video frame. Let's move on to stage two.

## **STAGE 2 - Surveillance Area:**

Our primary goal in these two stages is to detect a tailgating incidence. To accomplish this, we propose to divide our frame into a surveillance area. We would keep track of the number of humans in this area using their centroids.

We can establish this in 2 ways:

- Utilising a line of separation:
- Utilising a closed rectangle:



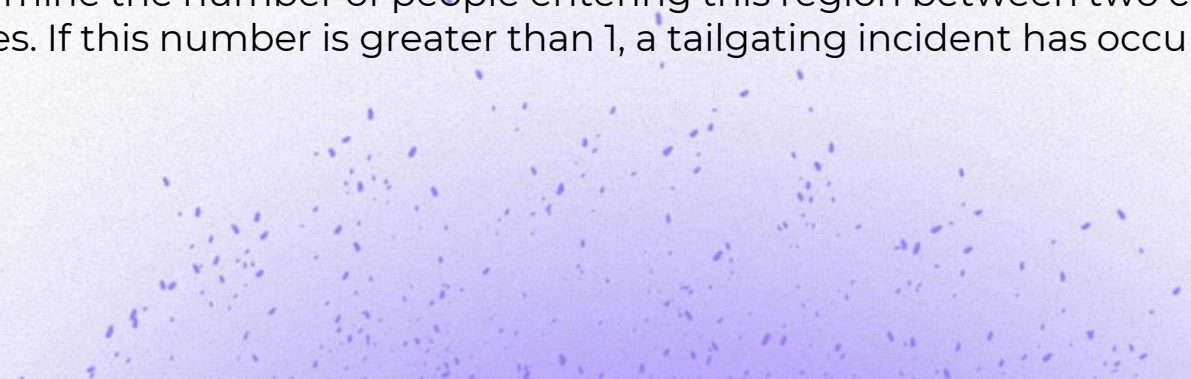


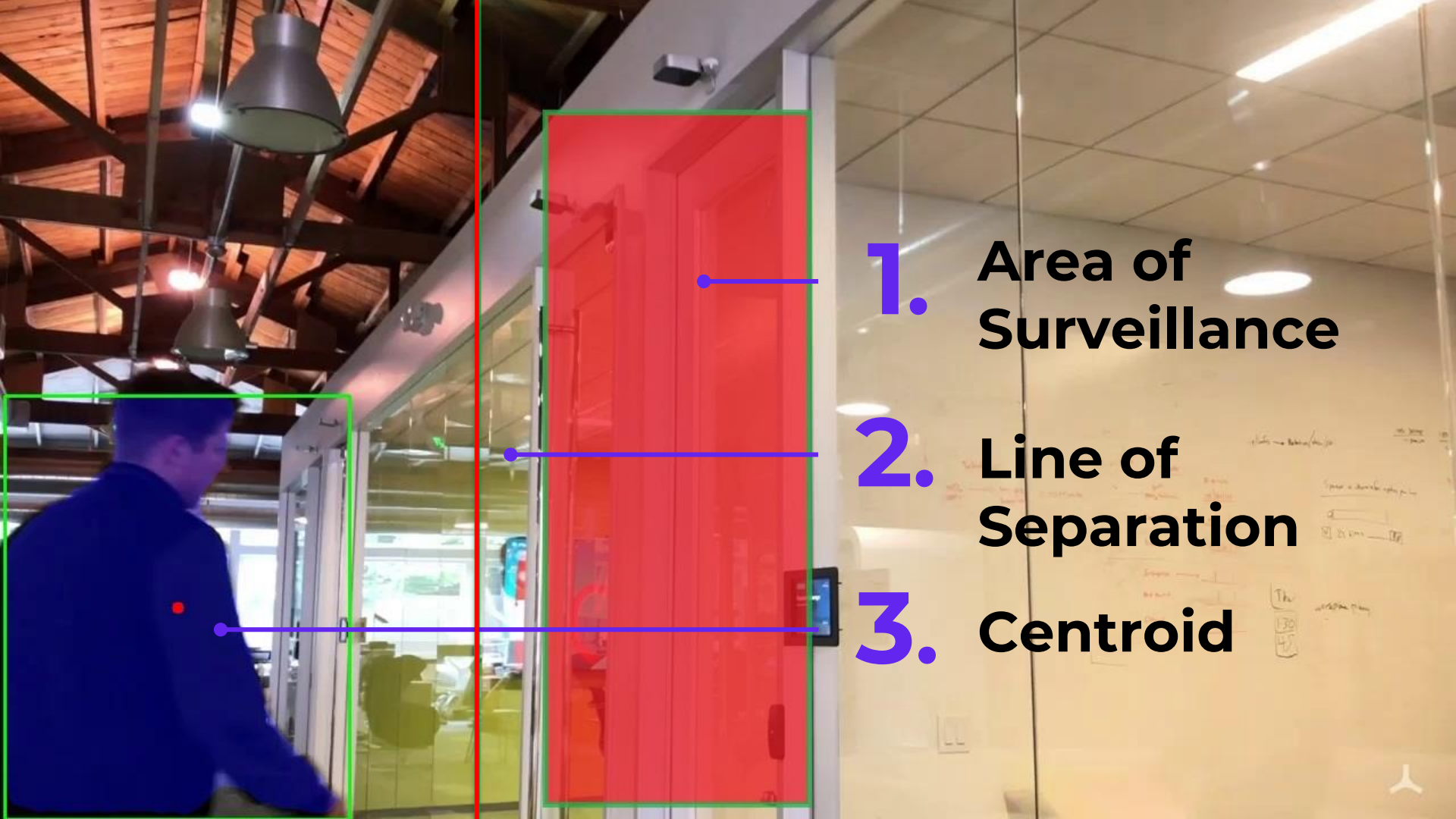
### **Utilising a line of separation:**

In this method, we track the number of humans crossing the line of separation. By drawing this line near the entry point, we can accurately determine the number of people entering between two consecutive swipes. If this number is greater than 1, a tailgating incident has occurred.

### **Utilising a closed rectangle:**

We follow the same principle and draw a rectangle on or near the entry point to determine the number of people entering this region between two consecutive swipes. If this number is greater than 1, a tailgating incident has occurred.



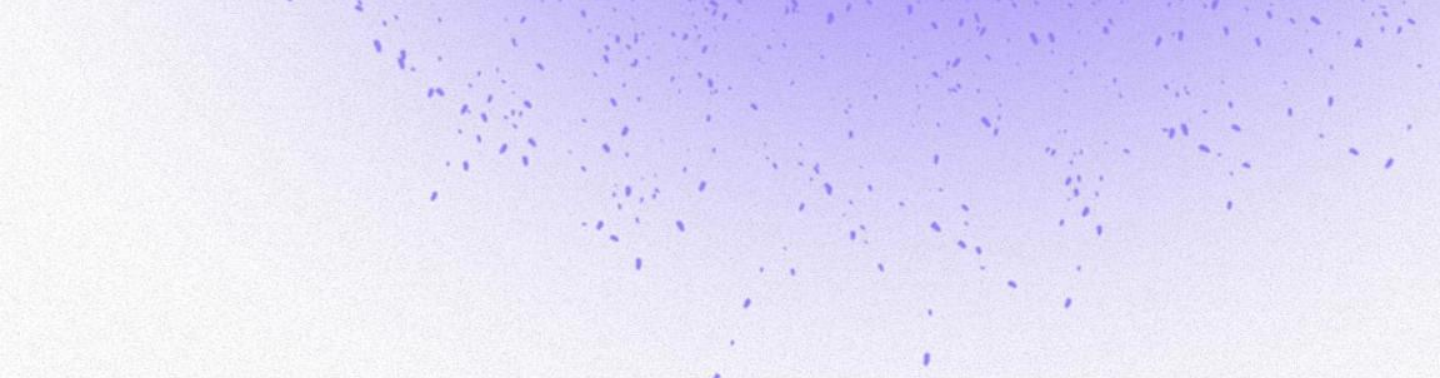


**1. Area of Surveillance**

**2. Line of Separation**

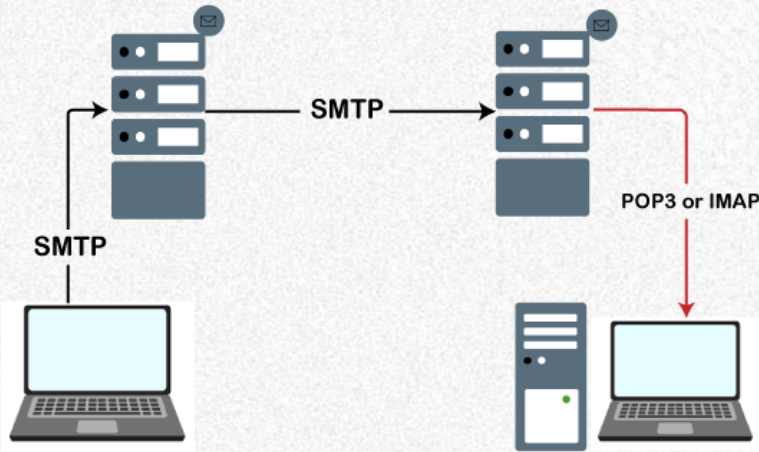
**3. Centroid**





The reason for listing multiple approaches is to make our product robust, as all of this can depend on the camera placement and the density of the crowd that enters. By using an optimal combination of the techniques mentioned earlier, tailgating can be successfully detected.

### 3. Alerting Security



- After detecting an intruder, our main objective is to raise an alarm and alert the security personnel regarding the potential security breach.
- This can be achieved using SMTP in python. We can send a mail to the security department and the person who swiped the card in the first place.
- We also propose to trigger a physical alarm, based on the employers need.





## 4. Identifying Intruder

We also propose to maintain a logger file, which would consist of time of occurrence of tailgating incidence and would also attach a cropped image of the alleged intruder, along with a short video-clip of the incident, as easily-accessible evidence.





**Thank You**



# The Team



**Rishon  
Dsouza**



**Amey  
Rambatla**



**Aishik  
Rakshit**