Object Detection and Tracking for People Counting

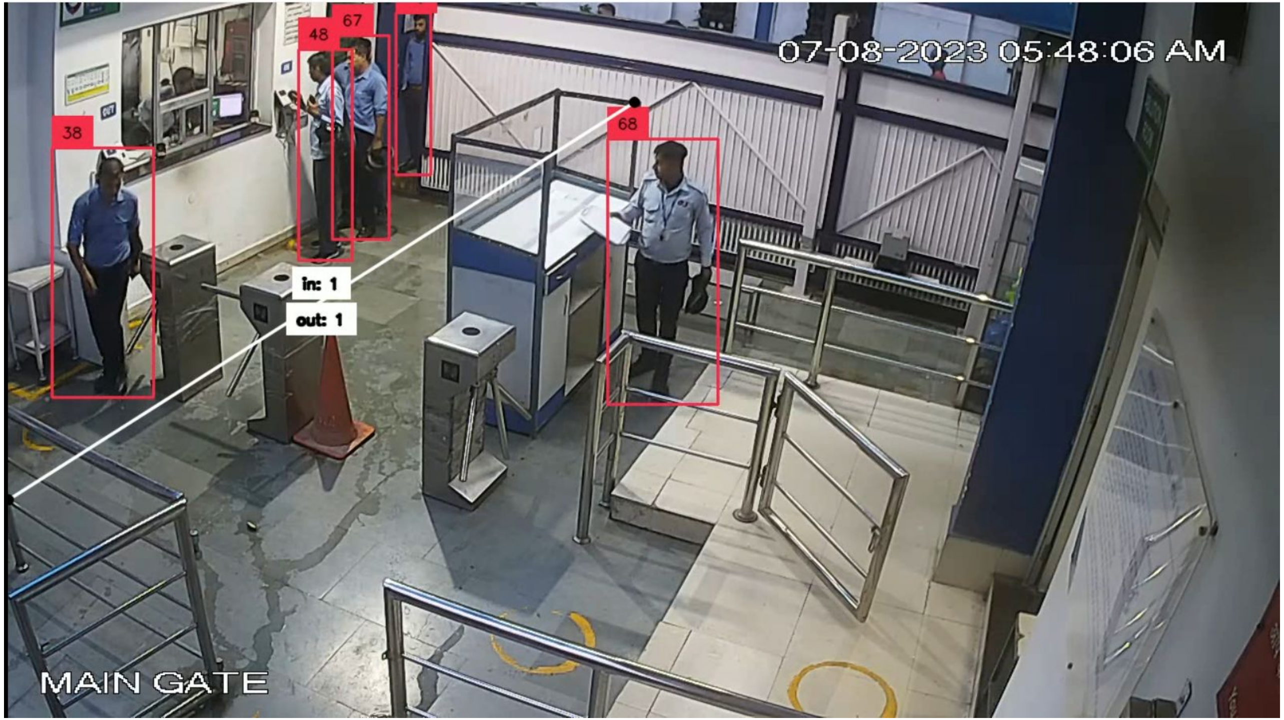
Reference line

- Focus on the id-15 before turnstile.
- In the next slide when he passes through the turnstile the id changes to 29.

Reference Line

The algorithm that was used to count the “entry and exit” is based on simple principle; that is the id of a person assigned should not change before and after the reference line.

Reference Line



This reference line has given me the better results because there is enough gap so that the id remains same before and after the line.


07-08-2023 05:48:14 AM
95
75
90
48
67
68
in: 2
out: 1
38
MAIN GATE

Object Detection Model Selection

For threshold = 0.7 and frame_buffer = 30 and GPU = RTX-3070

Model	FPS	Accuracy
YOLO-v8-n	~43	In: 67
YOLO-v8-s	~38	In: 83
YOLO-v8-m	~34	In: 108
YOLO-v8-L	~26	In: 106

Since there is not much change in the entries when using both the medium and large model, I am using medium model.

Tracker (ByteTrack)

Algorithm 1: Pseudo-code of BYTE.

Input: A video sequence `V`; object detector `Det`; detection score threshold τ
Output: Tracks $\mathcal{T}$ of the video

Initialization: $\mathcal{T} \leftarrow \emptyset$
for *frame* f_k *in* `V` **do**
    `/* Figure 2(a) */`
    `/* predict detection boxes & scores */`
    $\mathcal{D}_k \leftarrow \texttt{Det}(f_k)$
    $\mathcal{D}_{high} \leftarrow \emptyset$
    $\mathcal{D}_{low} \leftarrow \emptyset$
    for d *in* $\mathcal{D}_k$ **do**
        if $d.score > \tau$ **then**
            $\mathcal{D}_{high} \leftarrow \mathcal{D}_{high} \cup \{d\}$
        end
        else
            $\mathcal{D}_{low} \leftarrow \mathcal{D}_{low} \cup \{d\}$
        end
    end

    `/* predict new locations of tracks */`
    for t *in* $\mathcal{T}$ **do**
        $t \leftarrow \texttt{KalmanFilter}(t)$
    end

    `/* Figure 2(b) */`
    `/* first association */`
    Associate $\mathcal{T}$ and $\mathcal{D}_{high}$ using `Similarity#1`
    $\mathcal{D}_{remain} \leftarrow$ remaining object boxes from $\mathcal{D}_{high}$
    $\mathcal{T}_{remain} \leftarrow$ remaining tracks from $\mathcal{T}$

    `/* Figure 2(c) */`
    `/* second association */`
    Associate $\mathcal{T}_{remain}$ and $\mathcal{D}_{low}$ using `similarity#2`
    $\mathcal{T}_{re-remain} \leftarrow$ remaining tracks from $\mathcal{T}_{remain}$

    `/* delete unmatched tracks */`
    $\mathcal{T} \leftarrow \mathcal{T} \setminus \mathcal{T}_{re-remain}$

    `/* initialize new tracks */`
    for d *in* $\mathcal{D}_{remain}$ **do**
        $\mathcal{T} \leftarrow \mathcal{T} \cup \{d\}$
    end
end
Return: $\mathcal{T}$

Track rebirth [70, 89] is not shown in the algorithm for simplicity. In green is the key of our method.

Tracker

Table-3: Model: yolo-v8-m; track_buffer: 60; GPU: RTX-3070

Threshold	Accuracy
0.7	In: 106
0.8	In: 116
0.9	In: 116

Table-4: Model= yolo-v8-m; threshold = 0.8; GPU = RTX-3070

Frame_buffer	Accuracy
30	In: 117
60	In: 116

Disadvantages

1. The line that defined in the current scenario is cut throat; changing the surrounding slightly will cause a lot of accuracy issues. For example, the change position of security guard.
2. The script that was written is dependent a lot on the API calls rather than the core models. Hence there is not much control over the outputs that we want. For instance, the libraries installed are dependent on the hardware; I cannot control the “cpu” and “cuda” with in the same machine. Detection threshold cannot be controlled in this code.

Solution

1. The total number of entries are 116 based on the reference line that I’ve chosen.
2. The total number of exits are 3 based on the reference line that I’ve chosen.
3. You can find the solution video here.