Real-Time People Tracking and Counting

Prepare By Rashid Ali

Relevance

Manually analyzing people in videos is impractical

Automatic process using tracking programs can be easier and faster

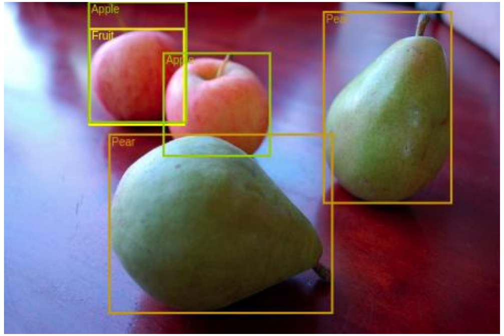
People Tracking: what is it?

Starting point: object detection

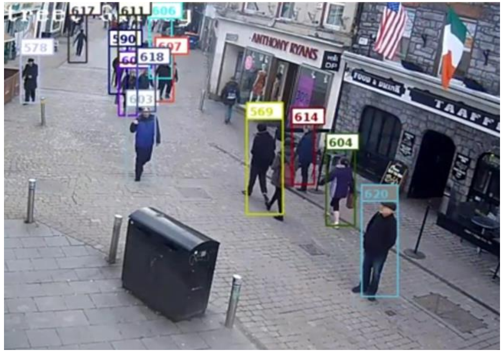
- detect & locate the object
- classify the object
- 1 frame

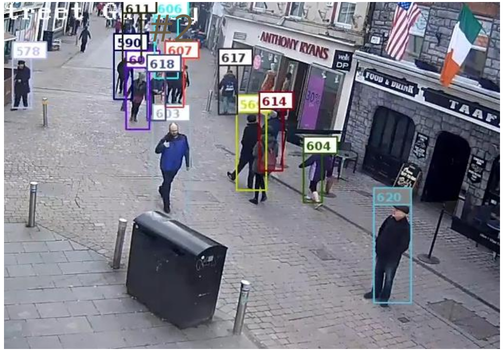
People tracking

- object is a person
- the person is tracked across multiple frames using IDs



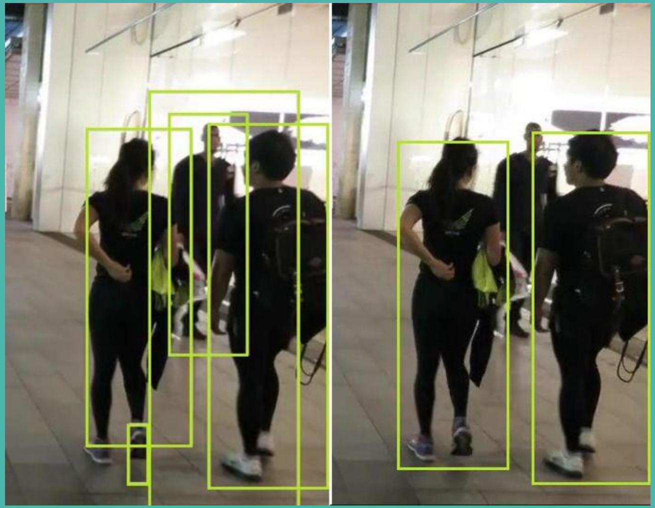
#1

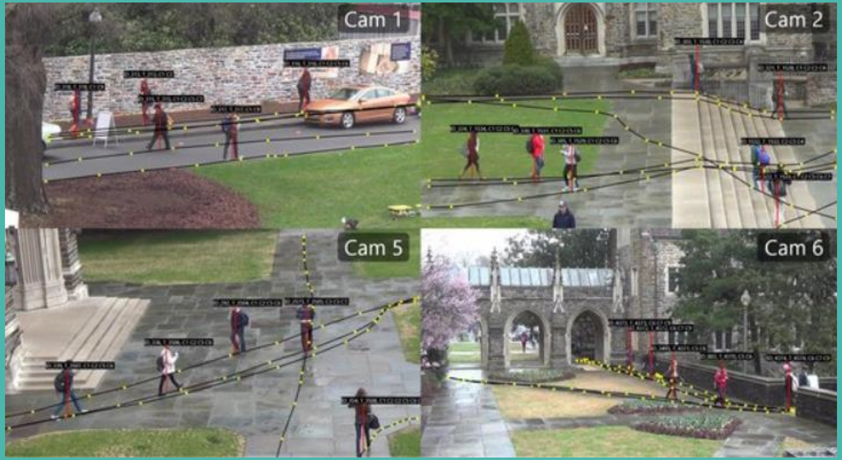




Challenges

- People that look alike
- Occlusion
- Variation in viewpoints





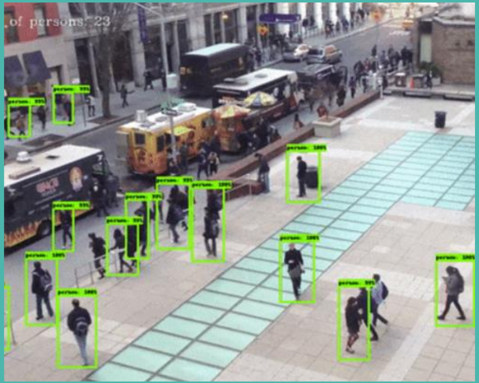
Applications

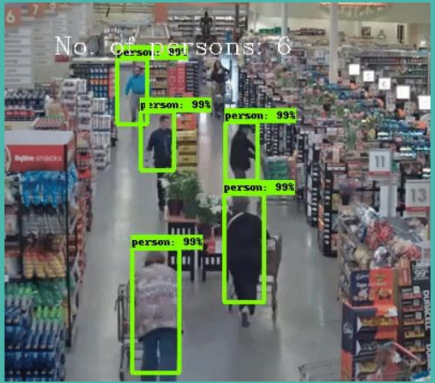
Infrastructure Planning

Retail

Security

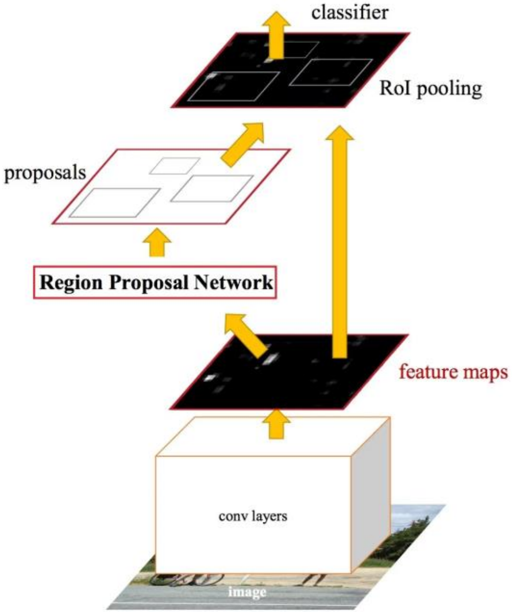
Safety



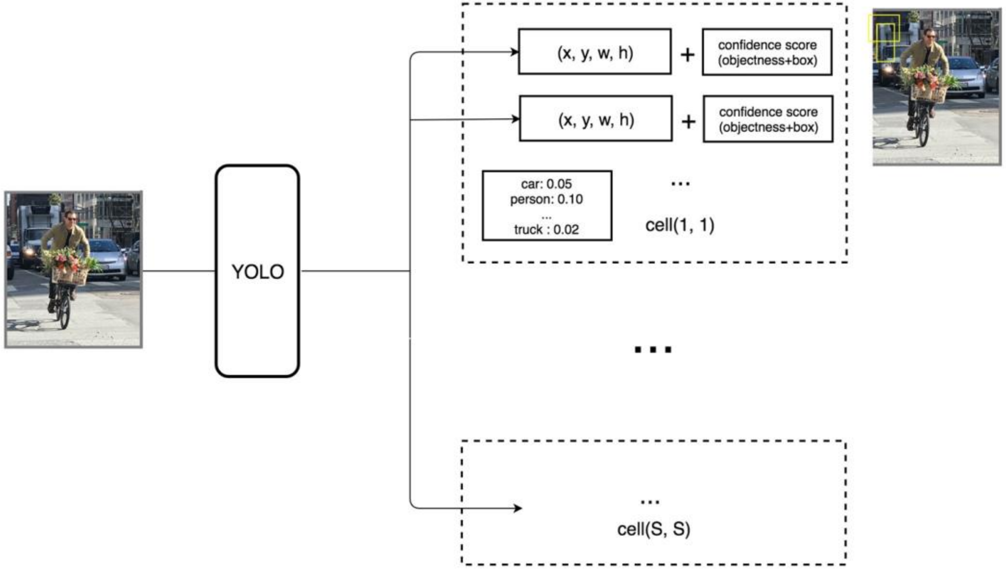


YOLO v3

YOLO : You only look once -- one-stage object detection

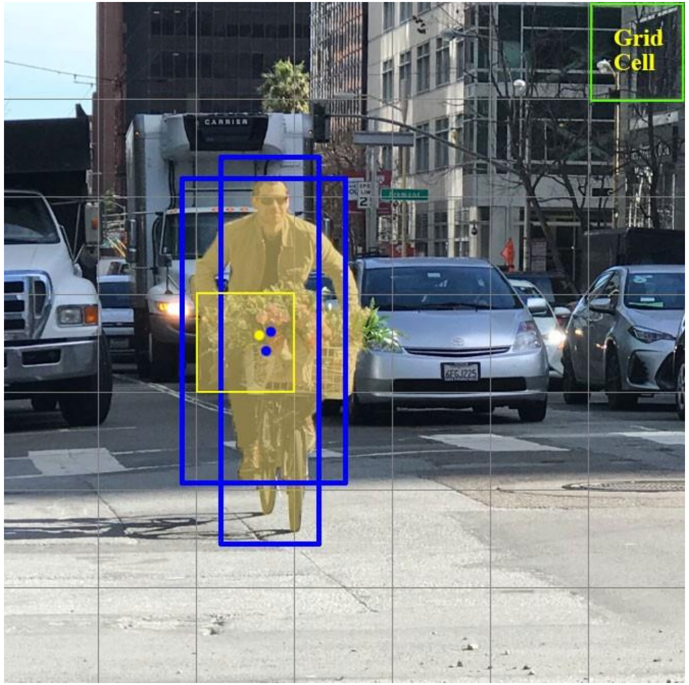


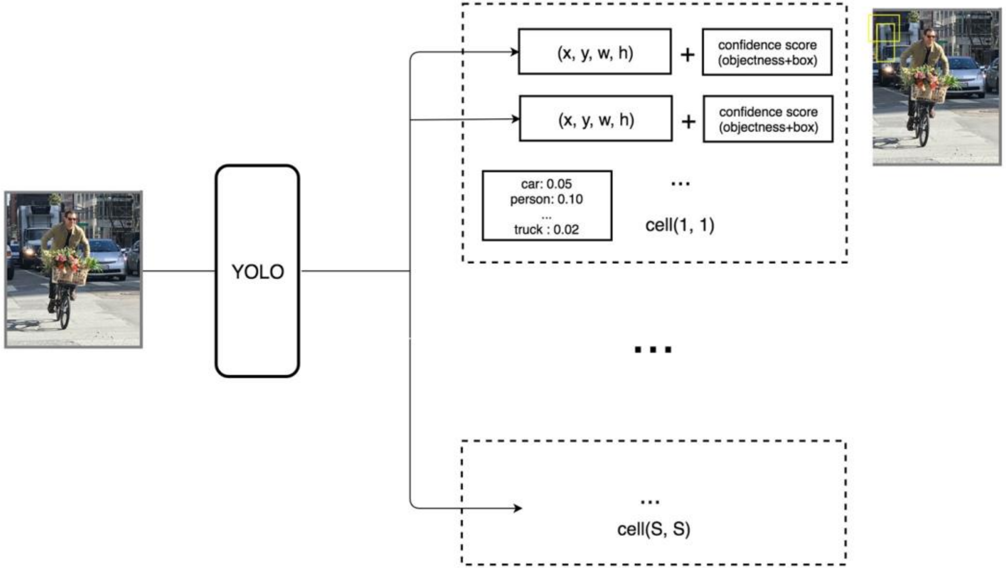
two-stage object detection -- Faster R-CNN



YOLO v3

YOLO : You only look once -- one-stage object detection





YOLO

(x, y, w, h) + confidence score (objectness+box)

(x, y, w, h) + confidence score (objectness+box)

car: 0.05
person: 0.10
...
truck : 0.02

...

cell(1, 1)

...

...

cell(S, S)

Pretrain Model

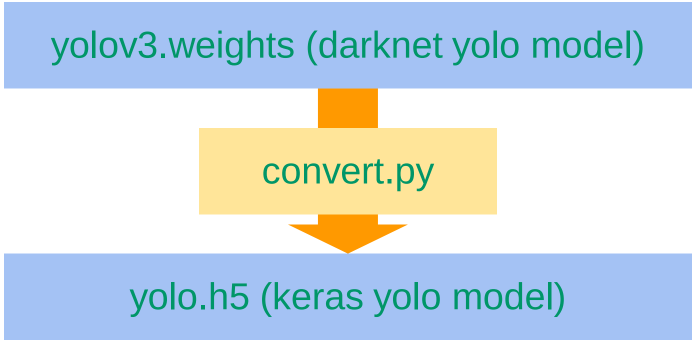
Model Pretrain by COCO dataset
(Common Objects in Context)

Source code : qqwweee/keras-yolo3 (Github)

Environment

Python 3.6.9 Tensorflow 1.14

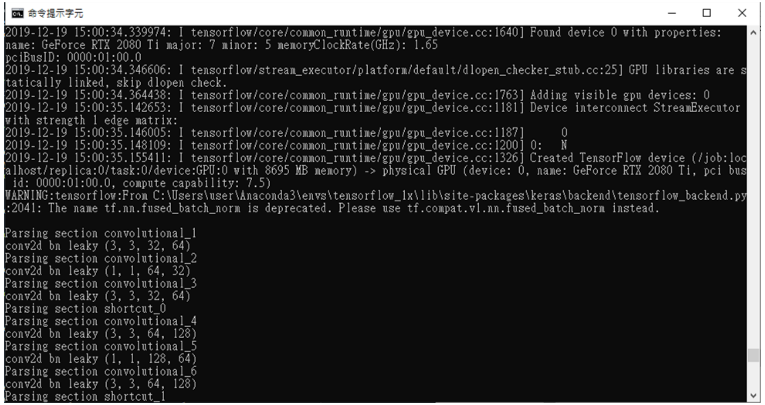
Keras 2.2.5 Open CV 3.4.2



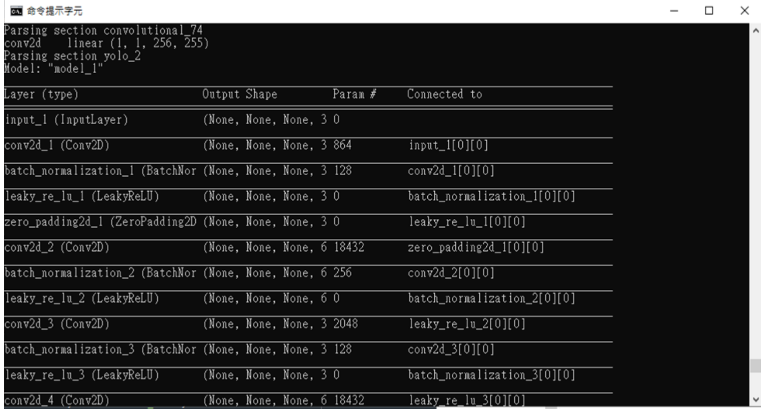
yolov3.weights (darknet yolo model)

convert.py

yolo.h5 (keras yolo model)

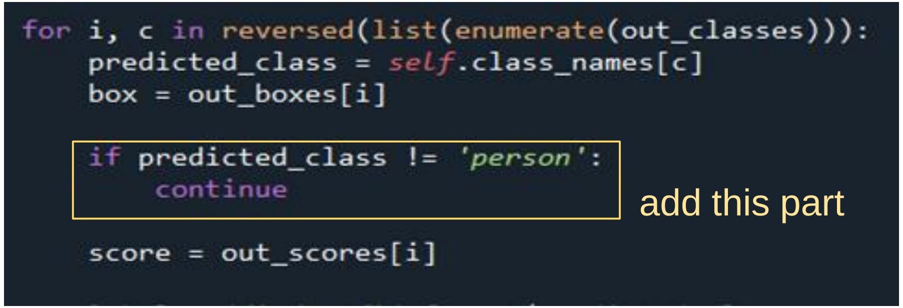


```
2019-12-19 15:00:34.339974: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1640] Found device 0 with properties:
name: GeForce RTX 2080 Ti major: 7 minor: 5 memoryClockRate(GHz): 1.65
pciBusID: 0000:01:00.0
2019-12-19 15:00:34.346606: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are s
tatically linked, skip dlopen check.
2019-12-19 15:00:34.364438: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1763] Adding visible gpu devices: 0
2019-12-19 15:00:35.142653: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1181] Device interconnect StreamExecutor
with strength 1 edge matrix:
2019-12-19 15:00:35.146005: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1187]      0
2019-12-19 15:00:35.148109: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1200] 0:   N
2019-12-19 15:00:35.155411: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1326] Created TensorFlow device (/job:loc
alhost/replica:0/task:0/device:GPU:0 with 8695 MB memory) -> physical GPU (device: 0, name: GeForce RTX 2080 Ti, pci bus
 id: 0000:01:00.0, compute capability: 7.5)
WARNING:tensorflow:From C:\Users\user\Anaconda3\envs\tensorflow_1x\lib\site-packages\keras\backend\tensorflow_backend.py
:2041: The name tf.nn.fused_batch_norm is deprecated. Please use tf.compat.v1.nn.fused_batch_norm instead.

Parsing section convolutional_1
conv2d bn leaky (3, 3, 32, 64)
Parsing section convolutional_2
conv2d bn leaky (1, 1, 64, 32)
Parsing section convolutional_3
conv2d bn leaky (3, 3, 32, 64)
Parsing section shortcut_0
Parsing section convolutional_4
conv2d bn leaky (3, 3, 64, 128)
Parsing section convolutional_5
conv2d bn leaky (1, 1, 128, 64)
Parsing section convolutional_6
conv2d bn leaky (3, 3, 64, 128)
Parsing section shortcut_1
```



```
Parsing section convolutional_74
conv2d    linear (1, 1, 256, 255)
Parsing section yolo_2
Model: "model_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, None, None, 3 0
__________________________________________________________________________________________________
conv2d_1 (Conv2D)               (None, None, None, 3 864         input_1[0][0]
__________________________________________________________________________________________________
batch_normalization_1 (BatchNor (None, None, None, 3 128         conv2d_1[0][0]
__________________________________________________________________________________________________
leaky_re_lu_1 (LeakyReLU)       (None, None, None, 3 0           batch_normalization_1[0][0]
__________________________________________________________________________________________________
zero_padding2d_1 (ZeroPadding2D (None, None, None, 3 0           leaky_re_lu_1[0][0]
__________________________________________________________________________________________________
conv2d_2 (Conv2D)               (None, None, None, 6 18432       zero_padding2d_1[0][0]
__________________________________________________________________________________________________
batch_normalization_2 (BatchNor (None, None, None, 6 256         conv2d_2[0][0]
__________________________________________________________________________________________________
leaky_re_lu_2 (LeakyReLU)       (None, None, None, 6 0           batch_normalization_2[0][0]
__________________________________________________________________________________________________
conv2d_3 (Conv2D)               (None, None, None, 3 2048        leaky_re_lu_2[0][0]
__________________________________________________________________________________________________
batch_normalization_3 (BatchNor (None, None, None, 3 128         conv2d_3[0][0]
__________________________________________________________________________________________________
leaky_re_lu_3 (LeakyReLU)       (None, None, None, 3 0           batch_normalization_3[0][0]
__________________________________________________________________________________________________
conv2d_4 (Conv2D)               (None, None, None, 6 18432       leaky_re_lu_3[0][0]
```

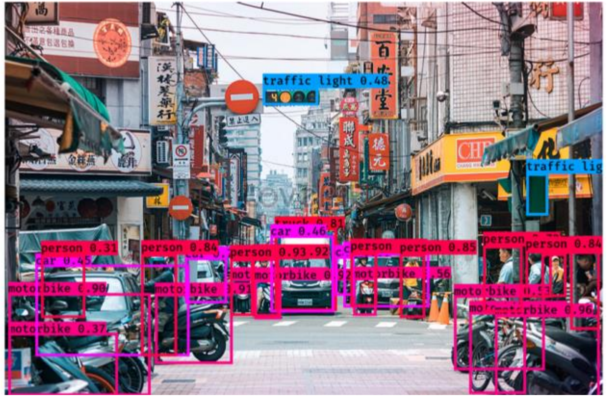
Detect person



```
for i, c in reversed(list(enumerate(out_classes))):
    predicted_class = self.class_names[c]
    box = out_boxes[i]

    if predicted_class != 'person':
        continue

    score = out_scores[i]
```

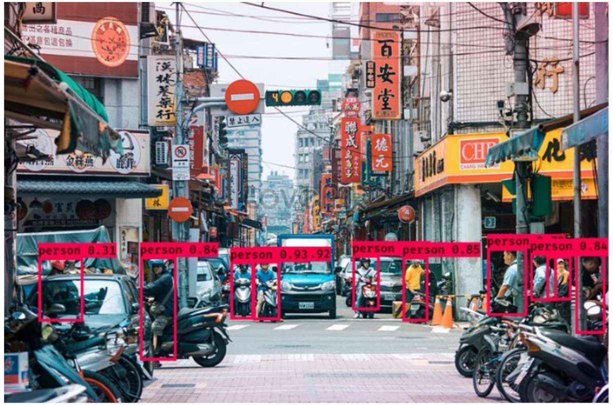
add this part

output image (original)

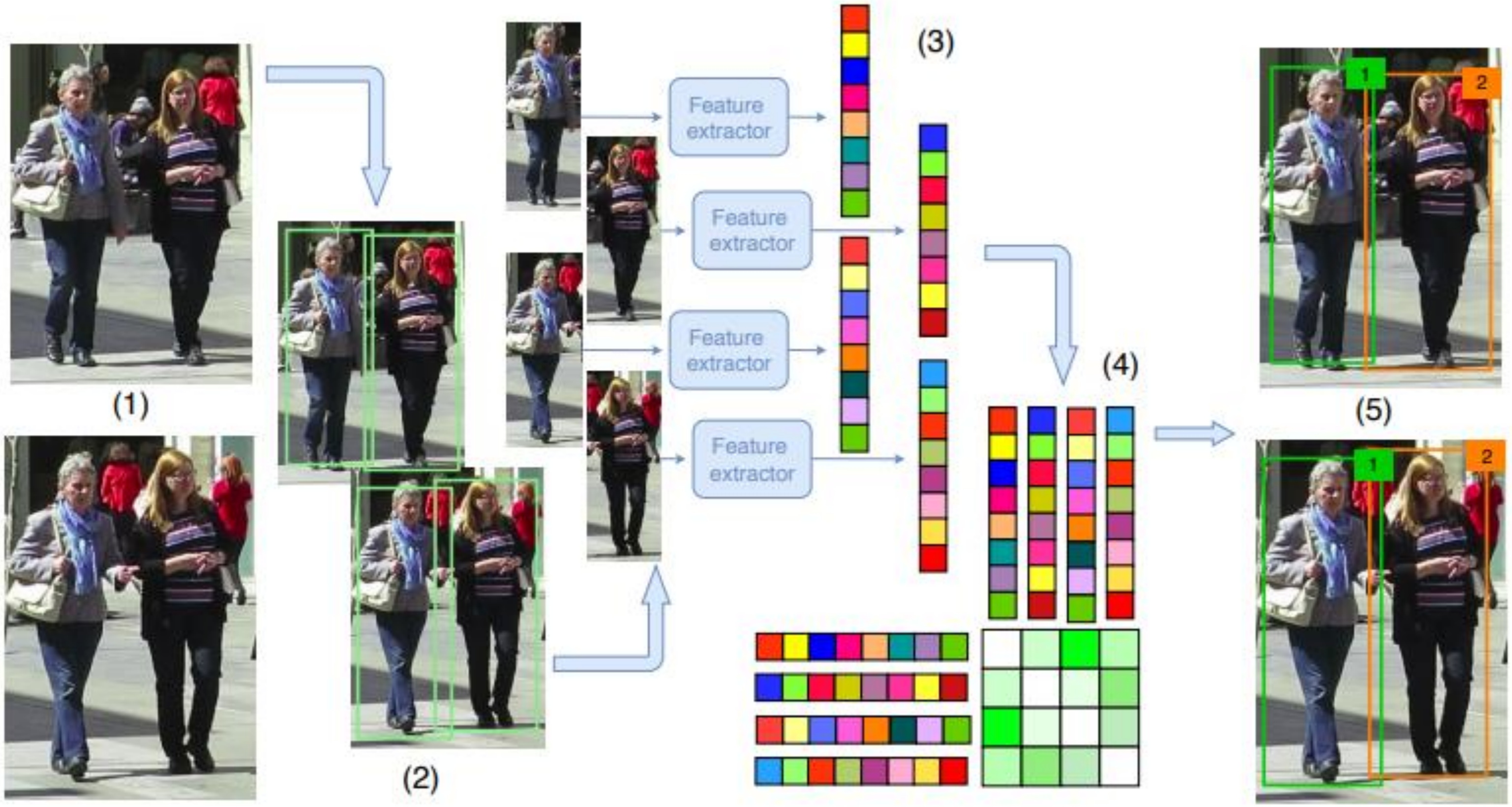


input image

output image (Just detect person)



Tracking Overview



Tracking using Deep Sort

Deep Sort define the bounding boxes as independant and track them through time. How it tracks?

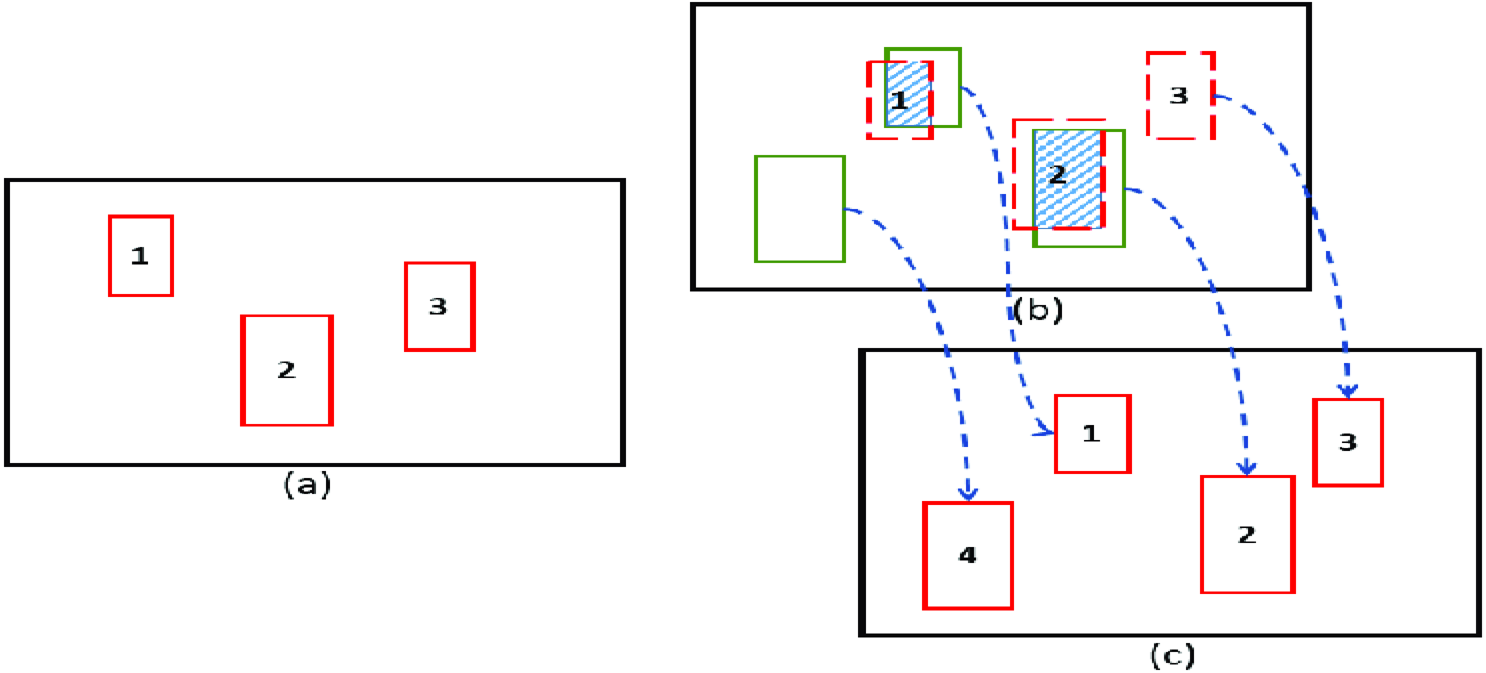
- State Estimation using Kalman filter
  - A Kalman Filter is an algorithm that can **predict future positions based on current position**.
- Assignment problem
  - A distance metric to quintify the association
  - A Hungarian algorithm can **tell if an object in current frame is the same as the one in previous frame.**
  - It will be used for association and id attribution.
- Creation and Deletion of Track Identities
  - The new tracks (Unique IDs) are created or destroyed when object enter and leave the video frame
- Deep Sort Source code: https://github.com/Qidian213/deep_sort_yolov3

State Estimation using Kalman filter

- A Kalman filter is used to estimate the state of a dynamic system (evolving with time) in the present (filtering) or the future (prediction)
- The kalman filter estimate the dynamic system state **X** at time **t** based on its state at time **t-1**
- The state **X** is vector composed of position (bounding box) and velolcity of objects
- Each esitimate is associated with a measure of uncertainty **P**
- State **X** and uncertainty **P** are represented by Gaussians.
- For each detection a track identity is created

The Assignment Problem

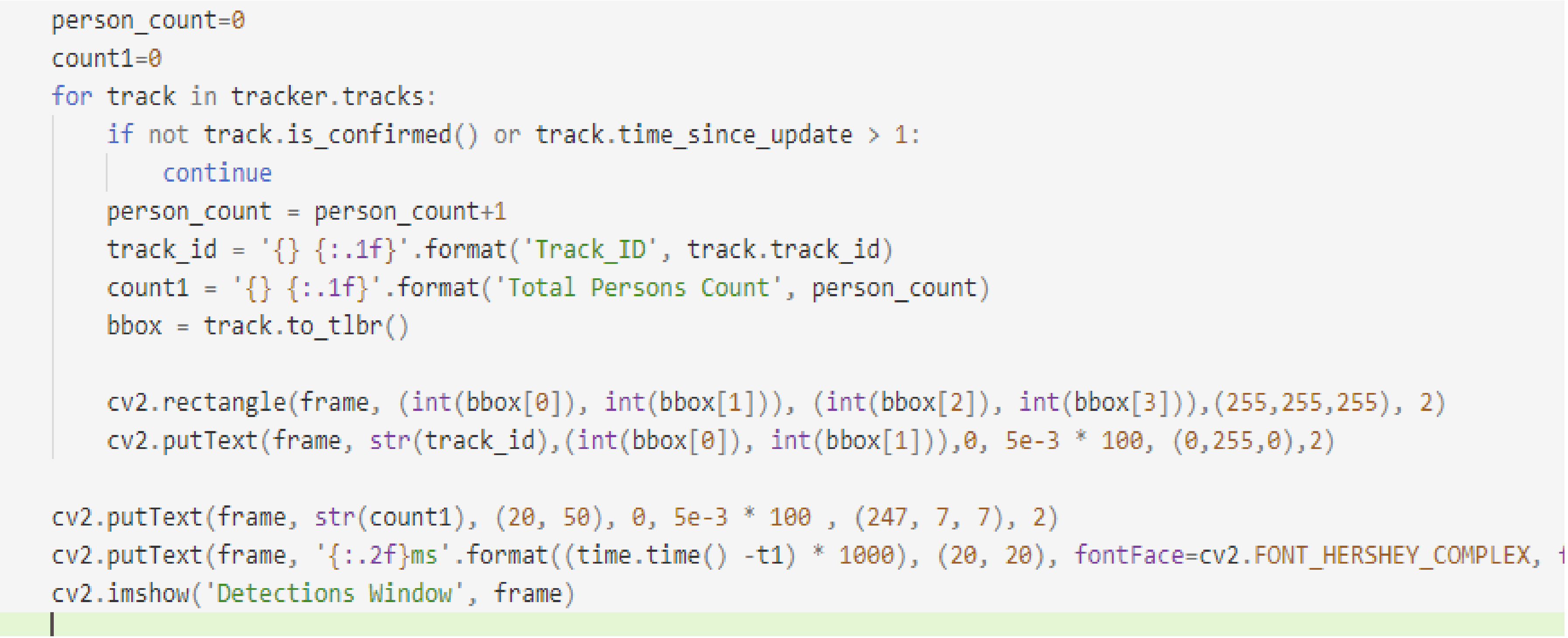
- Now we have tracked the bounding boxes in previous frame using kalman filter, how associate new detections with tracked bounding boxes?



The Assignment Problem

- The association between the new detections and tracked bounding boxes can be done based on IOU, distance metric and matching appearance features
- Assignment process works in following steps:
  - We have two lists of boxes from YOLO : a tracking list (t-1) and a detection list (t).
  - Calculate the IOU score, distance score, appearance score between tracks and new detections and store in matrix
  - Use Hungarian algorithm to calculate match matrix
  - What we get from that is matrix of which element in detection matches with which element in tracking
  - We can output matched detections, unmatched detections and unmatched trackings from this matrix
- Matched detections - Assign same ID
- Unmatched detection - Assign New ID
- Unmatched trackings - Delete ID

Counting people in each frame

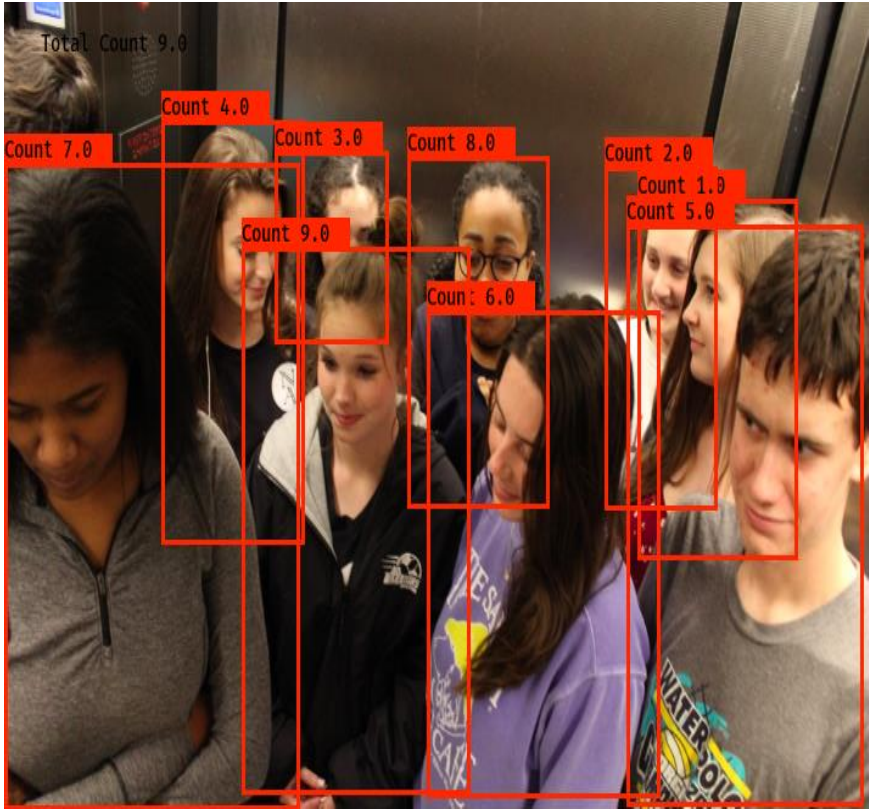


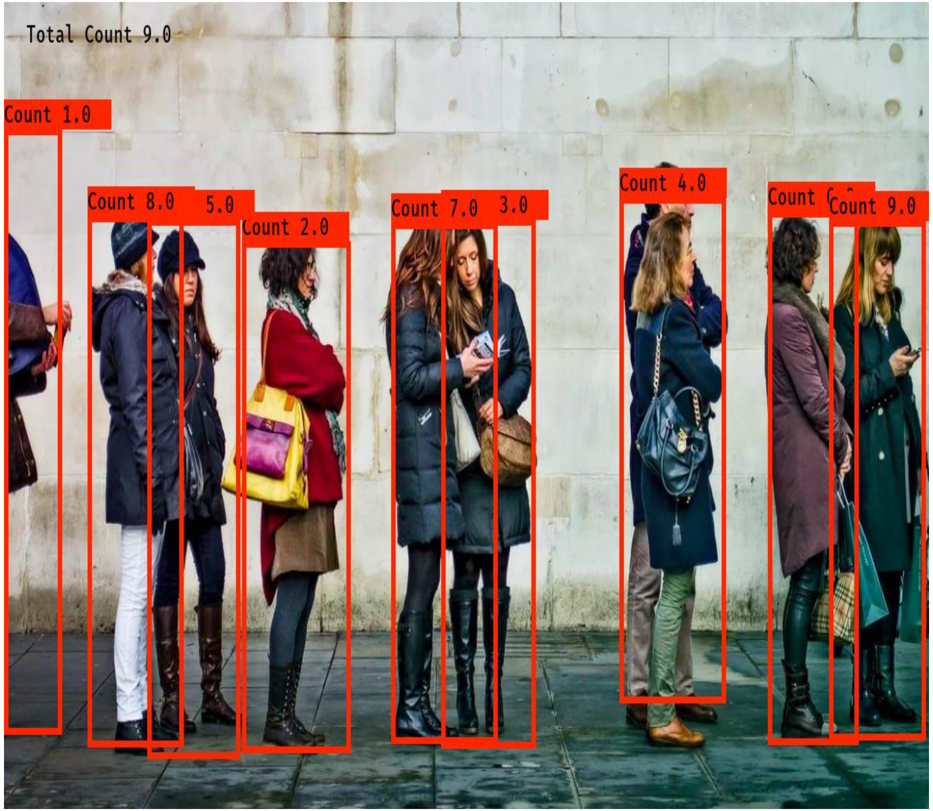
```
person_count=0
count1=0
for track in tracker.tracks:
    if not track.is_confirmed() or track.time_since_update > 1:
        continue
    person_count = person_count+1
    track_id = '{} {:.1f}'.format('Track_ID', track.track_id)
    count1 = '{} {:.1f}'.format('Total Persons Count', person_count)
    bbox = track.to_tlbr()

    cv2.rectangle(frame, (int(bbox[0]), int(bbox[1])), (int(bbox[2]), int(bbox[3])),(255,255,255), 2)
    cv2.putText(frame, str(track_id),(int(bbox[0]), int(bbox[1])),0, 5e-3 * 100, (0,255,0),2)

cv2.putText(frame, str(count1), (20, 50), 0, 5e-3 * 100 , (247, 7, 7), 2)
cv2.putText(frame, '{:.2f}ms'.format((time.time() -t1) * 1000), (20, 20), fontFace=cv2.FONT_HERSHEY_COMPLEX, f
cv2.imshow('Detections Window', frame)
```

Code Contributions

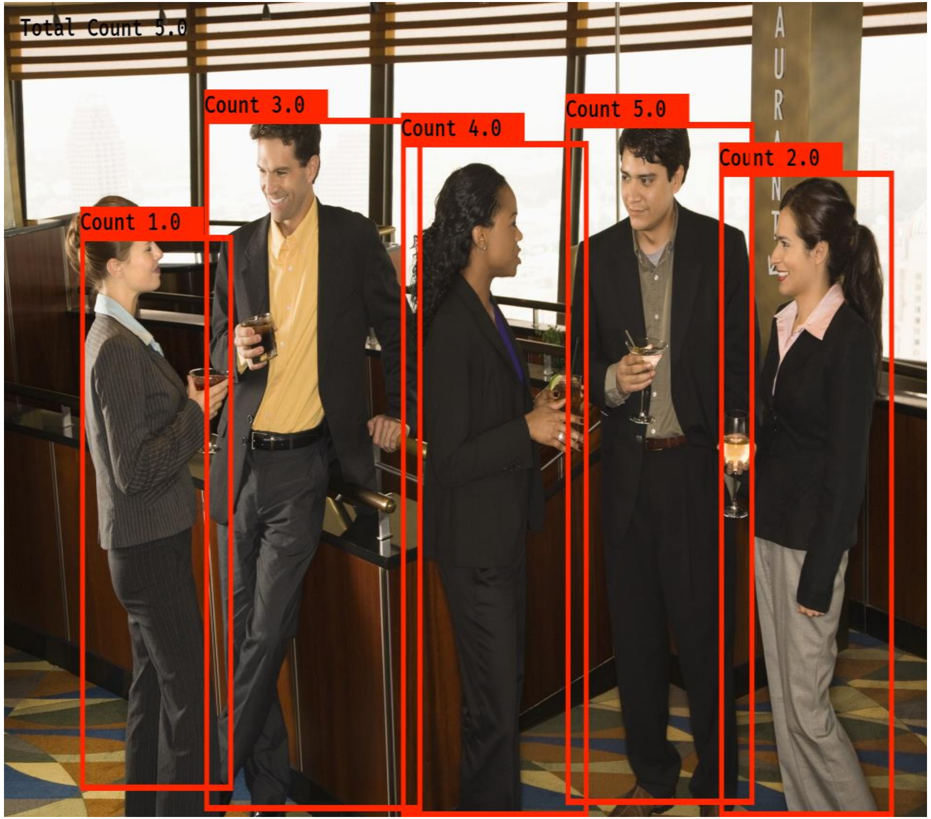
- Person class detection only from COCO dataset
- Person counting and tracking using Webcam
- Person counting from singel Image
- Person counting and tracking from video file
- Inclusion of Tiny YOLOv3 architecture in addition to YOLOv3

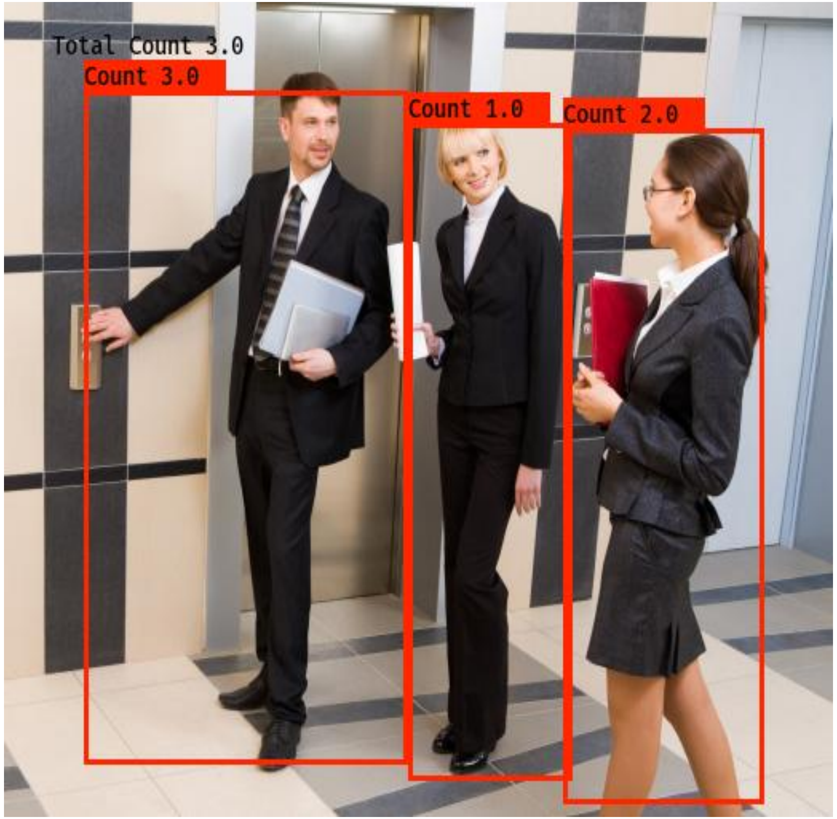
Person Counting from Single Image





Person Counting from Single Image

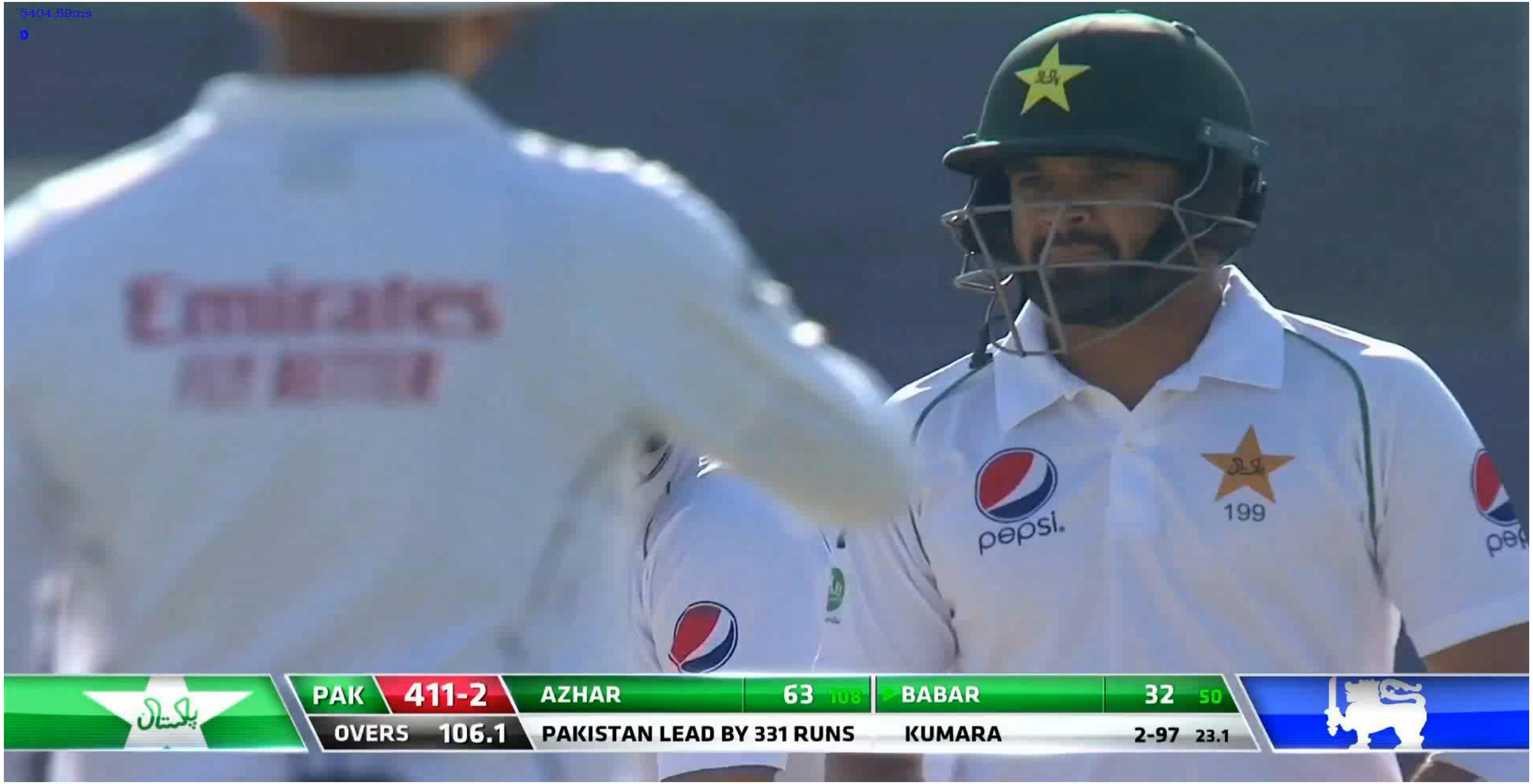




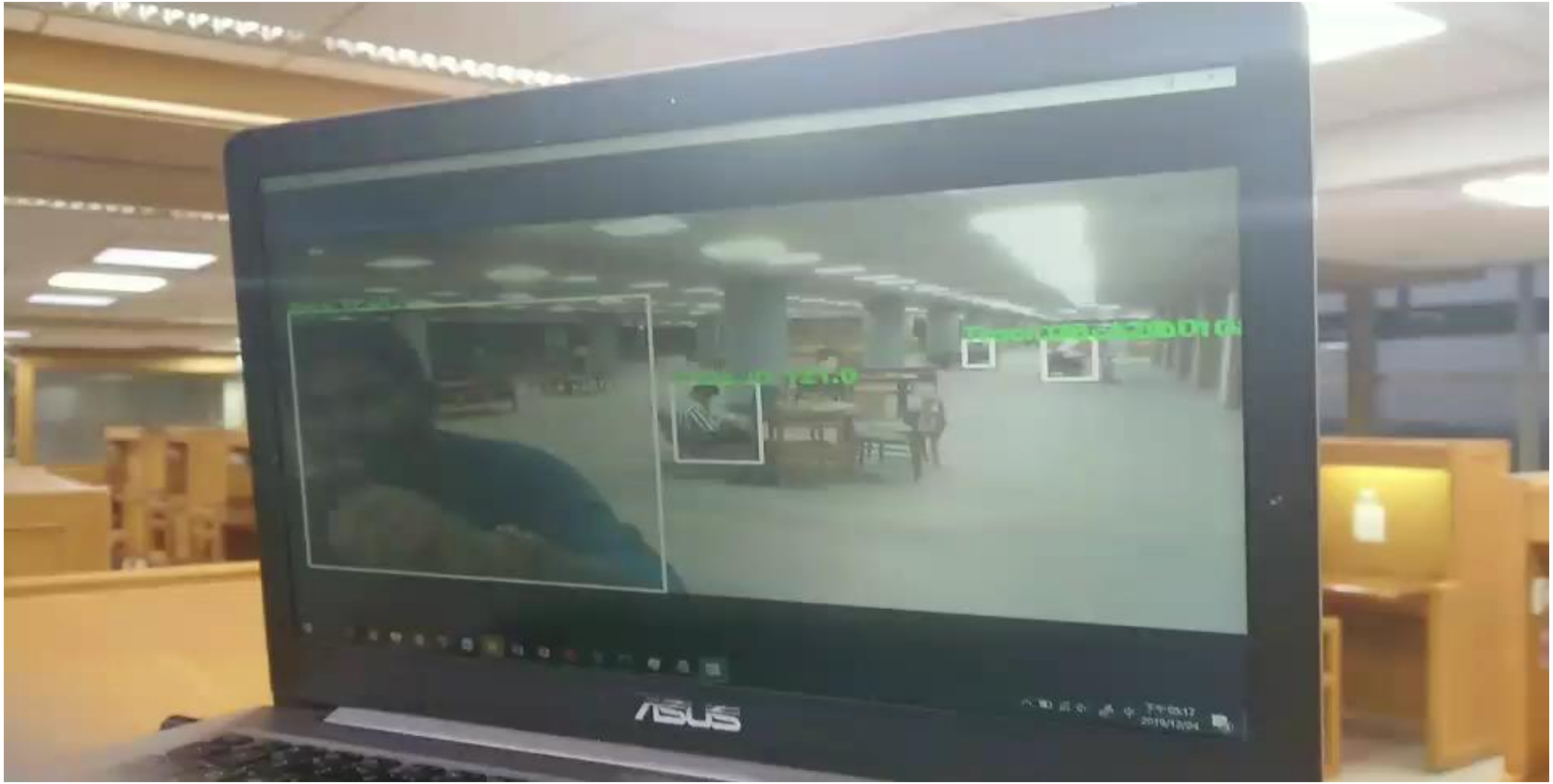
Person Tracking and Counting from Video

Person Tracking and Counting from Video

Person Tracking and Counting from Video



Person Tracking and Counting from Webcam



QUESTIONS