LAB 3

Θεωρητικό μέρος

1. Ποια είναι η διαφορά του bounding box και του anchor box στο YOLO? (Εξηγήστε συνοπτικά)

Με τον όρο bounding box εννοούμε το παραλληλόγραμο στο οποίο περιέχεται το αντικείμενο που εντοπίζουμε.

Τα anchor boxes είναι προκαθορισμένα κουτιά διαφορετικών διαστάσεων. Ο αριθμός τους εξαρτάται απο τον αριθμό τον κλάσεων που θέλουμε να εντοπίσουμε και οι διαστάσεις τους απο το σχήμα αυτών των κλάσεων. Σκόπός των anchor boxes είναι να δημιουργήσουν πολλές πιθανές προβλέψεις για πιθανά bounding box για αντικείμενα διαφορετικών μεγεθών και σχημάτων.

2. Ποιες θα είναι οι διαστάσεις του πίνακα εξόδου (πρόβλεψης) y_hat του αλγορίθμου YOLO θεωρώντας ότι έχουμε τρία anchor boxes και 4 κλάσεις? Αναφέρετε επίσης το ρόλο για κάθε στοιχείο του πίνακα αυτού.

Κάθε anchor box έχει 5 + C στοιχεία. Αυτά είναι οι συντεταγμένες του x και y, οι διαστάσεις h(ύψος) και w(πλάτος),το pc(confidence score) και C που είναι ο αριθμός των κλάσεων. Στην περίπτωση αυτή όπου έχουμε 3 anchor boxes με 4 κλάσεις οι διαστάσεις του πίνακα εξόδου είναι (5+4)*3 = 27

3. Πώς επηρεάζουν οι διαφορετικές ρυθμίσεις για το threshold και την παράμετρο επικάλυψης την απόδοση της μεθόδου Non-max suppression?

Καθορίζοντας το threshold μόνο τα anchor boxes με confidence score πάνω απο αυτό το κατώτατο κατόφλι θα ληφθουν υπόψη. Αυξάνοντας το threshold θα απορρίπτονται περισσότερα anchor boxes και θα επιλέγονται τα κουτιά που είναι πιο βέβαιο να περιέχουν κάποιο αντικείμενο. Αντίθετα αν το μειώσουμε θα έχουμε περισσότερα anchor boxes και ίσως κάποια απο αυτά να μην περιέχουν κάποιο αντικείμενο.

Η παράμετρο επικάλυψης καθορίζει το ποσοστό επικάλυψης που θα θεωρηθεί αποδεκτό για 2 anchor boxes ώστε να θεωρηθούν ίδια αντικείμενα. Αν η επικάλυψη μεταξύ 2 anchor boxes είναι μεγαλύτερη από την παράμετρο επικάλυψης, τότε το NMS θα απορρίψει το box με το χαμηλότερο confidence score. Αν αυξησουμε την παράμετρο επικάλυψης, περισσότερα anchor boxes θα γίνουν αποδεκτά με κίνδυνο να υπάρξουν περισσότερες προβλέψεις για το ίδιο αντικείμενο.

4. Ποια είναι μερικά από τα βασικά πλεονεκτήματα της χρήσης του αλγόριθμου SORT για την παρακολούθηση αντικειμένων (object tracking);

Τα 3 βασικά πλεονεκτήματα του αλγόριθμου SORT για object tracking είναι:

- Εύκολος στην υλοποίηση σε σχέση με άλλους αλγορίθμους και στην ενσωμάτωση του σε ένα σύστημα object tracking
- Ο αλγόριθμος λειτουργεί σε πραγματικό χρόνο, καθιστώντας τον κατάλληλο για εφαρμογές όπως object tracking σε βίντεο.
- Έχει υψηλή ακρίβεια στην παρακολούθηση αντικειμένων

5. Πώς ορίζεται η μέθοδος του tracking-by-detection και πως λειτουργεί?

Η μέθοδος tracking-by-detection είναι αλγόριθμός παρακολούθησης αντικειμένων που συνδυάζει την ανίχνευση αντικειμένων (detection) και την παρακολούθηση τους (tracking) για να παράγει συνεχή πορεία και αναγνώριση των αντικειμένων σε ένα βίντεο.

Η ανίχνευση των αντικειμένων(detection) δίνεται απο τον αλγόριθμο YOLOv3.

Όταν έχουμε ένα σύνολο απο ανιχνεύσεις(detections) για έναν αριθμό απο διαδοχικά frames ενός video τότε χρησιμοποιούνται οι πληροφορίες αυτές ώστε να κάνουμε track τα αντικείμενα και να υπολογιστεί η τροχία τους. Αυτό επιτυχάνεται συγκρίνοντας τα διαδοχικά frames και παρακολουθόντας το πόσο κουνήθηκαν τα bounding boxes.

Αλλαγή υπερπαραμέτρων του αλγορίθμου

In [ ]:
```
import sys
!{sys.executable} -m pip install torch==1.5.0 torchvision==0.6.0
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
public/simple/
ERROR: Could not find a version that satisfies the requirement torch==1.5.0 (from ve
rsions: 1.11.0, 1.12.0, 1.12.1, 1.13.0, 1.13.1, 2.0.0, 2.0.1)
ERROR: No matching distribution found for torch==1.5.0
```

In [ ]:
```
!pip install matplotlib
!pip install pillow
!pip install filterpy==1.4.5
!pip install scikit-image==0.17.2
!pip install lap==0.4.0
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
public/simple/
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages
(3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib) (1.0.7)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packag
es (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib) (4.39.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib) (1.4.4)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-package
s (from matplotlib) (1.22.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib) (23.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packa
ges (from matplotlib) (8.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dis
t-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages
(from python-dateutil>=2.7->matplotlib) (1.16.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
public/simple/
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (8.
4.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
public/simple/
Collecting filterpy==1.4.5
  Downloading filterpy-1.4.5.zip (177 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 178.0/178.0 kB 4.7 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (fro
m filterpy==1.4.5) (1.22.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (fro
m filterpy==1.4.5) (1.10.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages
(from filterpy==1.4.5) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib->filterpy==1.4.5) (1.0.7)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packag
es (from matplotlib->filterpy==1.4.5) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib->filterpy==1.4.5) (4.39.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib->filterpy==1.4.5) (1.4.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib->filterpy==1.4.5) (23.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packa
ges (from matplotlib->filterpy==1.4.5) (8.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib->filterpy==1.4.5) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dis
t-packages (from matplotlib->filterpy==1.4.5) (2.8.2)
```

```
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages
(from python-dateutil>=2.7->matplotlib->filterpy==1.4.5) (1.16.0)
Building wheels for collected packages: filterpy
  Building wheel for filterpy (setup.py) ... done
  Created wheel for filterpy: filename=filterpy-1.4.5-py3-none-any.whl size=110459 s
ha256=c537dbd48b0af3ca43963c354ba34328418fdace43f8f281bd07874034dd18ee
  Stored in directory: /root/.cache/pip/wheels/0f/0c/ea/218f266af4ad626897562199fbbc
ba521b8497303200186102
Successfully built filterpy
Installing collected packages: filterpy
Successfully installed filterpy-1.4.5
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
public/simple/
Collecting scikit-image==0.17.2
  Downloading scikit-image-0.17.2.tar.gz (29.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 29.8/29.8 MB 54.4 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: numpy>=1.15.1 in /usr/local/lib/python3.10/dist-packa
ges (from scikit-image==0.17.2) (1.22.4)
Requirement already satisfied: scipy>=1.0.1 in /usr/local/lib/python3.10/dist-packag
es (from scikit-image==0.17.2) (1.10.1)
Requirement already satisfied: matplotlib!=3.0.0,>=2.0.0 in /usr/local/lib/python3.1
0/dist-packages (from scikit-image==0.17.2) (3.7.1)
Requirement already satisfied: networkx>=2.0 in /usr/local/lib/python3.10/dist-packa
ges (from scikit-image==0.17.2) (3.1)
Requirement already satisfied: pillow!=7.1.0,!=7.1.1,>=4.3.0 in /usr/local/lib/pytho
n3.10/dist-packages (from scikit-image==0.17.2) (8.4.0)
Requirement already satisfied: imageio>=2.3.0 in /usr/local/lib/python3.10/dist-pack
ages (from scikit-image==0.17.2) (2.25.1)
Requirement already satisfied: tifffile>=2019.7.26 in /usr/local/lib/python3.10/dist
-packages (from scikit-image==0.17.2) (2023.4.12)
Requirement already satisfied: PyWavelets>=1.1.1 in /usr/local/lib/python3.10/dist-p
ackages (from scikit-image==0.17.2) (1.4.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib!=3.0.0,>=2.0.0->scikit-image==0.17.2) (1.0.7)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packag
es (from matplotlib!=3.0.0,>=2.0.0->scikit-image==0.17.2) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib!=3.0.0,>=2.0.0->scikit-image==0.17.2) (4.39.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-p
ackages (from matplotlib!=3.0.0,>=2.0.0->scikit-image==0.17.2) (1.4.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pac
kages (from matplotlib!=3.0.0,>=2.0.0->scikit-image==0.17.2) (23.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pa
ckages (from matplotlib!=3.0.0,>=2.0.0->scikit-image==0.17.2) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dis
t-packages (from matplotlib!=3.0.0,>=2.0.0->scikit-image==0.17.2) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages
(from python-dateutil>=2.7->matplotlib!=3.0.0,>=2.0.0->scikit-image==0.17.2) (1.16.
0)
Building wheels for collected packages: scikit-image
  error: subprocess-exited-with-error

  × python setup.py bdist_wheel did not run successfully.
  │ exit code: 1
  ╰─> See above for output.
```

```
  note: This error originates from a subprocess, and is likely not a problem with pi
p.
  Building wheel for scikit-image (setup.py) ... error
  ERROR: Failed building wheel for scikit-image
  Running setup.py clean for scikit-image
Failed to build scikit-image
ERROR: Could not build wheels for scikit-image, which is required to install pyproje
ct.toml-based projects
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
public/simple/
Collecting lap==0.4.0
  Downloading lap-0.4.0.tar.gz (1.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.5/1.5 MB 20.1 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: lap
  Building wheel for lap (setup.py) ... done
  Created wheel for lap: filename=lap-0.4.0-cp310-cp310-linux_x86_64.whl size=165522
3 sha256=b3a7d8f825ebd151f0987e6f43d9990ba21c9453527c5b190da19cfe158edf6f
  Stored in directory: /root/.cache/pip/wheels/00/42/2e/9dfe19270eea279d79e84767ff0d
7b8082c3bf776cad00e83d
Successfully built lap
Installing collected packages: lap
Successfully installed lap-0.4.0
```

Mount to google drive. Το κάνουμε αυτό για έχουμε πρόσβαση στα python modules που χρειαζόμαστε.

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Για να μπορέσετε να έχετε πρόσβαση σε όλα τα αρχεία του φακέλου Lab_motion_tracking_exercise θα πρέπει πριν τρέξετε το παρακάτω κελί να πατε στο drive του εργαστηρίου, να κάνετε δεξί κλικ στο Lab_motion_tracking_exercise, να πατήσετε Add a shortcut to drive και μετά να πατήσετε My Drive.

In [ ]:
```
import sys
import os
sys.path.insert(0,'/content/drive/My Drive/Lab_motion_tracking_exercise')
print(os.listdir('/content/drive/My Drive/Lab_motion_tracking_exercise')) # Έλεγχος
```

```
['config', 'images', 'utils', 'models.py', 'road_trafifc.mp4', 'sort.py', '__pycache
__', 'Object_Detection_and_Tracking.ipynb']
```

Φορτώνουμε τις απαραίτητες βιβλιοθήκες

In [ ]:
```
from models import *
from utils import *

import os, sys, time, datetime, random
import torch
from torch.utils.data import DataLoader
from torchvision import datasets, transforms
```

```
from torch.autograd import Variable

import matplotlib.pyplot as plt
import matplotlib.patches as patches
from PIL import Image
```

In [ ]:
```
import os
print(os.getcwd())
```

```
/content
```

In [ ]:
```
config_path='./drive/My Drive/Lab_motion_tracking_exercise/config/yolov3.cfg'
weights_path='./drive/My Drive/Lab_motion_tracking_exercise/config/yolov3.weights'
class_path='./drive/My Drive/Lab_motion_tracking_exercise/config/coco.names'  #  φο
#img_size=416
#conf_thres=0.8
#nms_thres=0.4

img_size=416
conf_thres=0.2
nms_thres=0.4 # NMS threshold


# Load model and weights
model = Darknet(config_path, img_size=img_size)
model.load_weights(weights_path)
model.cuda()
model.eval()
classes = utils.load_classes(class_path)
Tensor = torch.cuda.FloatTensor
```

```
/usr/local/lib/python3.10/dist-packages/torch/nn/_reduction.py:42: UserWarning: size
_average and reduce args will be deprecated, please use reduction='mean' instead.
  warnings.warn(warning.format(ret))
```

In [ ]:
```
def detect_image(img):
    # scale and pad image
    ratio = min(img_size/img.size[0], img_size/img.size[1])
    imw = round(img.size[0] * ratio)
    imh = round(img.size[1] * ratio)
    img_transforms = transforms.Compose([ transforms.Resize((imh, imw)),
         transforms.Pad((max(int((imh-imw)/2),0), max(int((imw-imh)/2),0), max(int(
                        (128,128,128)),
         transforms.ToTensor(),
         ])
    # convert image to Tensor
    image_tensor = img_transforms(img).float()
    image_tensor = image_tensor.unsqueeze_(0)
    input_img = Variable(image_tensor.type(Tensor))
    # run inference on the model and get detections
    with torch.no_grad():
        detections = model(input_img)
        detections = utils.non_max_suppression(detections, 80, conf_thres, nms_thre
    return detections[0]
```

In [ ]:
```
# load image and get detections
#print(os.getcwd())
img_path = "./drive/My Drive/Lab_motion_tracking_exercise/images/Intersection-Count
prev_time = time.time()
img = Image.open(img_path)
detections = detect_image(img)
inference_time = datetime.timedelta(seconds=time.time() - prev_time)
print ('Inference Time: %s' % (inference_time))

# Get bounding-box colors
cmap = plt.get_cmap('tab20b')
colors = [cmap(i) for i in np.linspace(0, 1, 20)]

img = np.array(img)
plt.figure()
fig, ax = plt.subplots(1, figsize=(12,9))
ax.imshow(img)

pad_x = max(img.shape[0] - img.shape[1], 0) * (img_size / max(img.shape))
pad_y = max(img.shape[1] - img.shape[0], 0) * (img_size / max(img.shape))
unpad_h = img_size - pad_y
unpad_w = img_size - pad_x

if detections is not None:
    unique_labels = detections[:, -1].cpu().unique()
    n_cls_preds = len(unique_labels)
    bbox_colors = random.sample(colors, n_cls_preds)
    # browse detections and draw bounding boxes
    for x1, y1, x2, y2, conf, cls_conf, cls_pred in detections.cpu():
        box_h = ((y2 - y1) / unpad_h) * img.shape[0]
        box_w = ((x2 - x1) / unpad_w) * img.shape[1]
        y1 = ((y1 - pad_y // 2) / unpad_h) * img.shape[0]
        x1 = ((x1 - pad_x // 2) / unpad_w) * img.shape[1]
        color = bbox_colors[int(np.where(unique_labels == int(cls_pred))[0])]
        bbox = patches.Rectangle((x1, y1), box_w, box_h, linewidth=2, edgecolor=col
        ax.add_patch(bbox)
        plt.text(x1, y1, s=classes[int(cls_pred)], color='white', verticalalignment
                bbox={'color': color, 'pad': 0})
plt.axis('off')
# save image
#plt.savefig(img_path.replace(".jpg", "-det.jpg"), bbox_inches='tight', pad_inches=
plt.show()
```

```
Inference Time: 0:00:03.493347
<Figure size 640x480 with 0 Axes>
```

Μια μικρή τιμή του conf_thres, κρατώντας την τιμή για nms_trhes=0.4

In [ ]:
```
conf_thres=0.2
nms_thres=0.4 # NMS threshold
```

In [ ]:
```
%pylab inline
import cv2
from IPython.display import clear_output

videopath = './drive/My Drive/20.mp4'  # Εδώ θα αλλάξετε το path για να πάρετε το v
                                                                    # Το video θα

cmap = plt.get_cmap('tab20b')
colors = [cmap(i)[:3] for i in np.linspace(0, 1, 20)]

# initialize Sort object and video capture
from sort import *
vid = cv2.VideoCapture(videopath)
mot_tracker = Sort()
frames = vid.get(cv2.CAP_PROP_FRAME_COUNT)
#while(True):
for ii in range(int(frames)):
    ret, frame = vid.read()
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    pilimg = Image.fromarray(frame)
    detections = detect_image(pilimg)

    img = np.array(pilimg)
    pad_x = max(img.shape[0] - img.shape[1], 0) * (img_size / max(img.shape))
    pad_y = max(img.shape[1] - img.shape[0], 0) * (img_size / max(img.shape))
    unpad_h = img_size - pad_y
    unpad_w = img_size - pad_x
    if detections is not None:
        print("ii", ii)
```

```
        print(detections.shape)
        tracked_objects = mot_tracker.update(detections.cpu())
        print(tracked_objects.shape)

        unique_labels = detections[:, -1].cpu().unique()
        n_cls_preds = len(unique_labels)
        for x1, y1, x2, y2, obj_id, cls_pred in tracked_objects:
            box_h = int(((y2 - y1) / unpad_h) * img.shape[0])
            box_w = int(((x2 - x1) / unpad_w) * img.shape[1])
            y1 = int(((y1 - pad_y // 2) / unpad_h) * img.shape[0])
            x1 = int(((x1 - pad_x // 2) / unpad_w) * img.shape[1])

            color = colors[int(obj_id) % len(colors)]
            color = [i * 255 for i in color]
            cls = classes[int(cls_pred)]
            cv2.rectangle(frame, (x1, y1), (x1+box_w, y1+box_h), color, 4)
            cv2.rectangle(frame, (x1, y1-35), (x1+len(cls)*19+60, y1), color, -1)
            cv2.putText(frame, cls + "-" + str(int(obj_id)), (x1, y1 - 10), cv2.FON

    fig=figure(figsize=(12, 8))
    title("Video Stream")
    imshow(frame)
    show()
    #clear_output(wait=True)
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
/usr/local/lib/python3.10/dist-packages/IPython/core/magics/pylab.py:159: UserWarnin
g: pylab import has clobbered these variables: ['random']
`%matplotlib` prevents importing * from pylab and numpy
  warn("pylab import has clobbered these variables: %s"  % clobbered +
```

```
ii 0
torch.Size([18, 7])
(18, 6)
```


```
ii 1
torch.Size([18, 7])
```

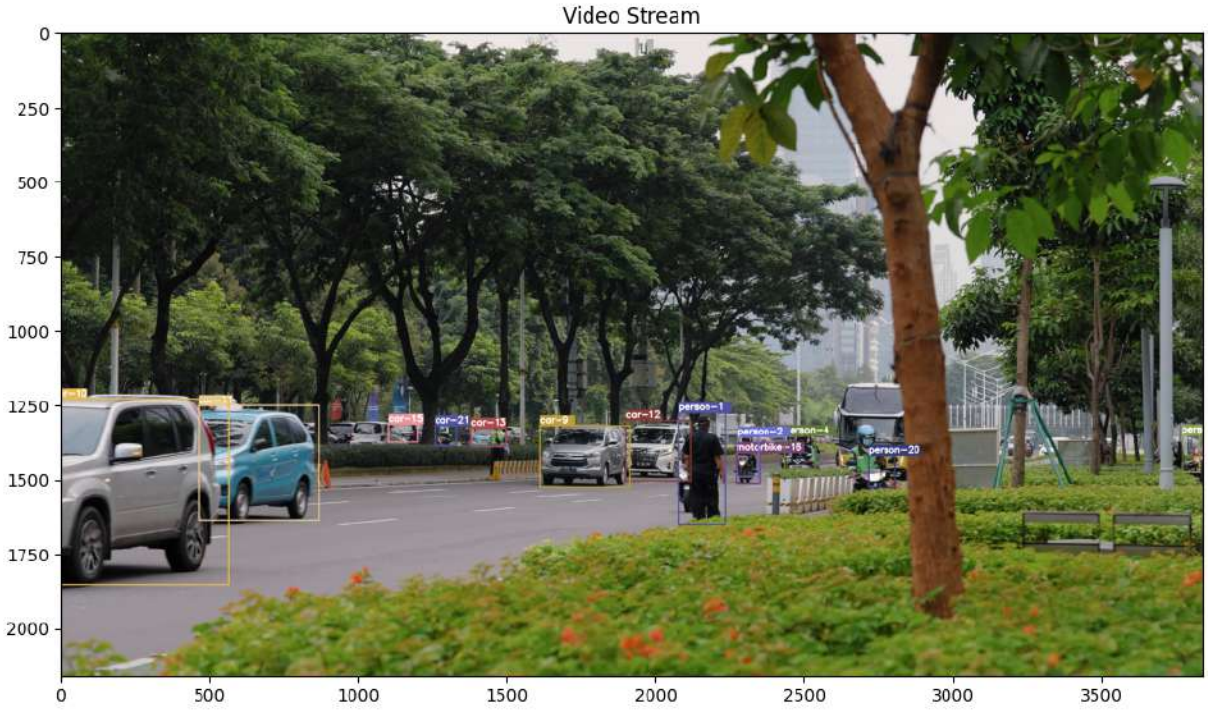
```
/content/drive/My Drive/Lab_motion_tracking_exercise/sort.py:39: NumbaWarning:
Compilation is falling back to object mode WITH looplifting enabled because Function
"iou" failed type inference due to: non-precise type pyobject
During: typing of argument at /content/drive/My Drive/Lab_motion_tracking_exercise/s
ort.py (44)

File "drive/My Drive/Lab_motion_tracking_exercise/sort.py", line 44:
def iou(bb_test,bb_gt):
    <source elided>
  """
  xx1 = np.maximum(bb_test[0], bb_gt[0])
  ^

  @jit
/usr/local/lib/python3.10/dist-packages/numba/core/object_mode_passes.py:151: NumbaW
arning: Function "iou" was compiled in object mode without forceobj=True.

File "drive/My Drive/Lab_motion_tracking_exercise/sort.py", line 40:
@jit
def iou(bb_test,bb_gt):
^

  warnings.warn(errors.NumbaWarning(warn_msg,
/usr/local/lib/python3.10/dist-packages/numba/core/object_mode_passes.py:161: NumbaD
eprecationWarning:
Fall-back from the nopython compilation path to the object mode compilation path has
been detected, this is deprecated behaviour.

For more information visit https://numba.readthedocs.io/en/stable/reference/deprecat
ion.html#deprecation-of-object-mode-fall-back-behaviour-when-using-jit

File "drive/My Drive/Lab_motion_tracking_exercise/sort.py", line 40:
@jit
def iou(bb_test,bb_gt):
^

  warnings.warn(errors.NumbaDeprecationWarning(msg,
```

```
(18, 6)
```


```
ii 2
torch.Size([21, 7])
(21, 6)
```


```
ii 3
torch.Size([23, 7])
(13, 6)
```


```
ii 4
torch.Size([18, 7])
(13, 6)
```




```
ii 5
torch.Size([24, 7])
(15, 6)
```


```
ii 6
torch.Size([20, 7])
(13, 6)
```

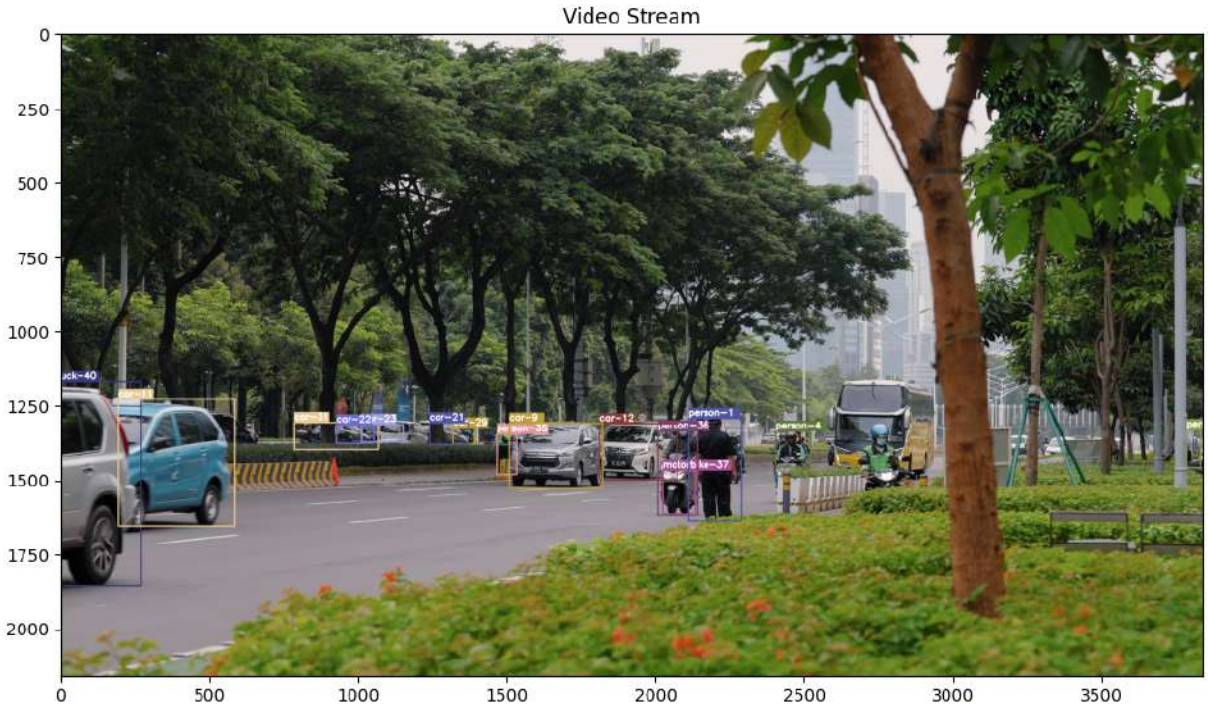

```
ii 7
torch.Size([24, 7])
(17, 6)
```


```
ii 8
torch.Size([24, 7])
(16, 6)
```


```
ii 9
torch.Size([20, 7])
(13, 6)
```


```
ii 10
torch.Size([22, 7])
(16, 6)
```


```
ii 11
torch.Size([22, 7])
(15, 6)
```




```
ii 12
torch.Size([22, 7])
(16, 6)
```


```
ii 13
torch.Size([20, 7])
(17, 6)
```


```
ii 14
torch.Size([19, 7])
(16, 6)
```


```
ii 15
torch.Size([20, 7])
(16, 6)
```

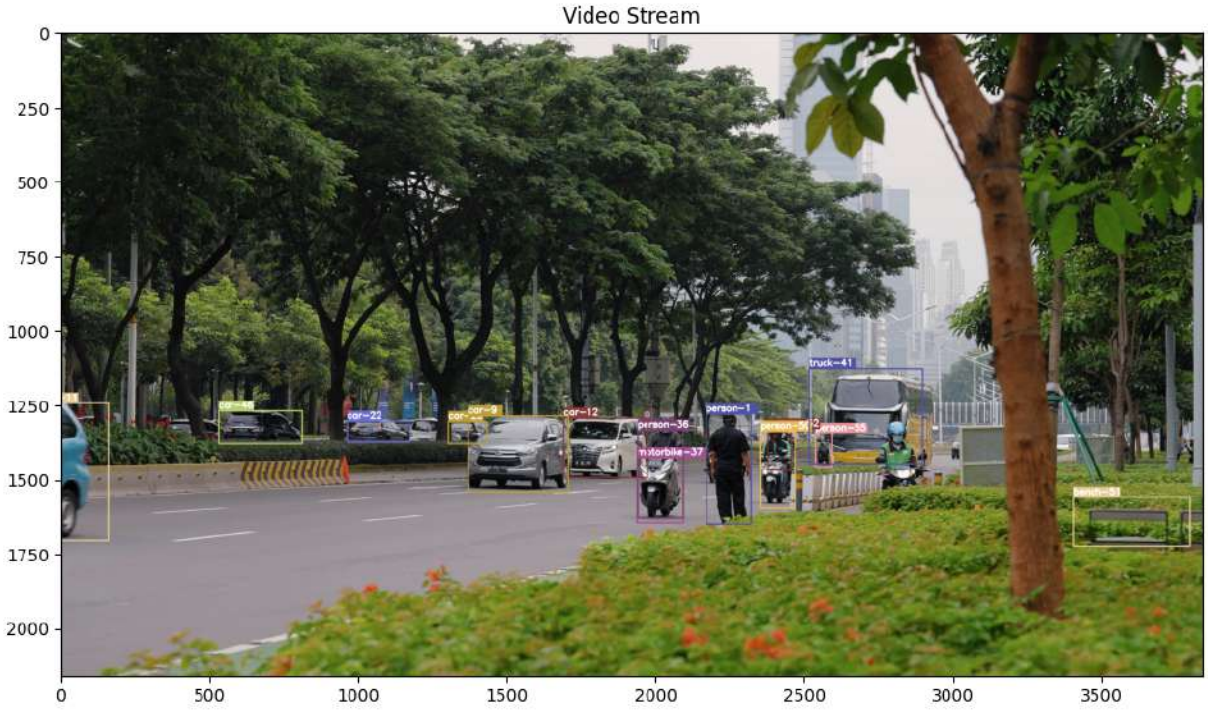

```
ii 16
torch.Size([18, 7])
(14, 6)
```


```
ii 17
torch.Size([18, 7])
(14, 6)
```


```
ii 18
torch.Size([17, 7])
(14, 6)
```


```
ii 19
torch.Size([20, 7])
(14, 6)
```


```
ii 20
torch.Size([19, 7])
(14, 6)
```




```
ii 21
torch.Size([18, 7])
(14, 6)
```


```
ii 22
torch.Size([21, 7])
(13, 6)
```


```
ii 23
torch.Size([21, 7])
(14, 6)
```


```
ii 24
torch.Size([21, 7])
(15, 6)
```


```
ii 25
torch.Size([21, 7])
(16, 6)
```


```
ii 26
torch.Size([20, 7])
(16, 6)
```


```
ii 27
torch.Size([15, 7])
(14, 6)
```


```
ii 28
torch.Size([16, 7])
(14, 6)
```


```
ii 29
torch.Size([19, 7])
(14, 6)
```


```
ii 30
torch.Size([22, 7])
(15, 6)
```


```
ii 31
torch.Size([16, 7])
(12, 6)
```


```
ii 32
torch.Size([17, 7])
(11, 6)
```


```
ii 33
torch.Size([15, 7])
(12, 6)
```

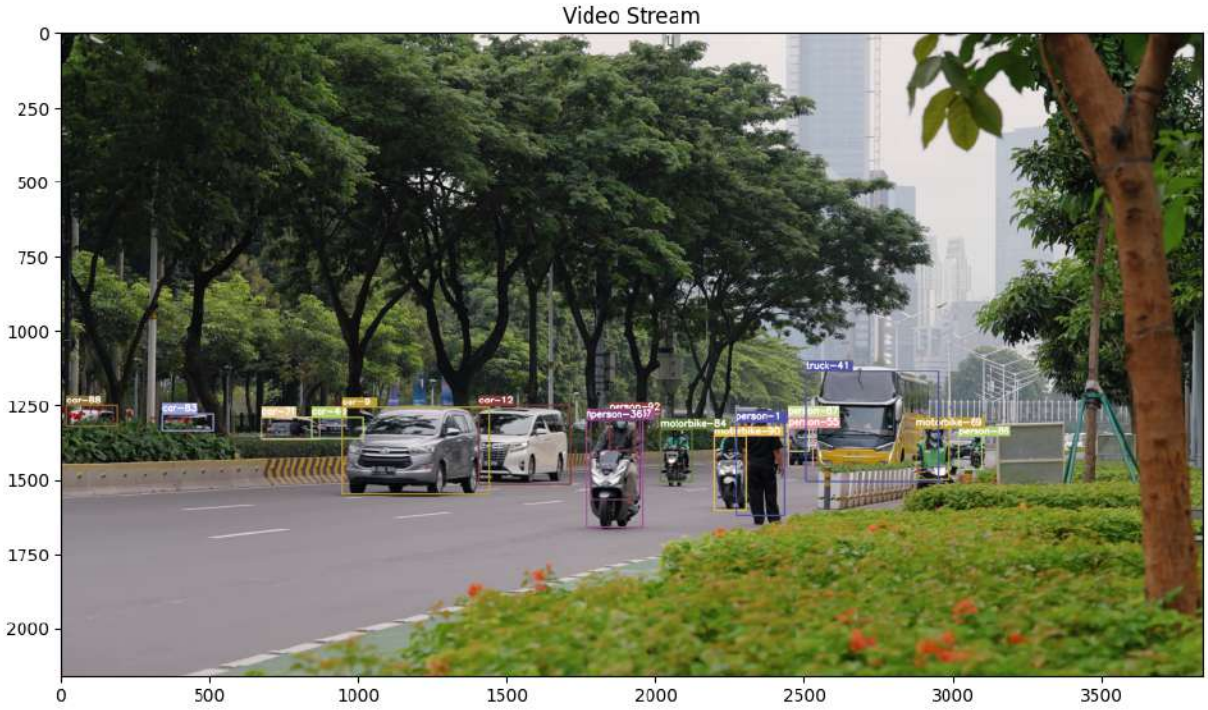

```
ii 34
torch.Size([18, 7])
(13, 6)
```


```
ii 35
torch.Size([20, 7])
(14, 6)
```


```
ii 36
torch.Size([19, 7])
(16, 6)
```


```
ii 37
torch.Size([18, 7])
(16, 6)
```


```
ii 38
torch.Size([20, 7])
(17, 6)
```

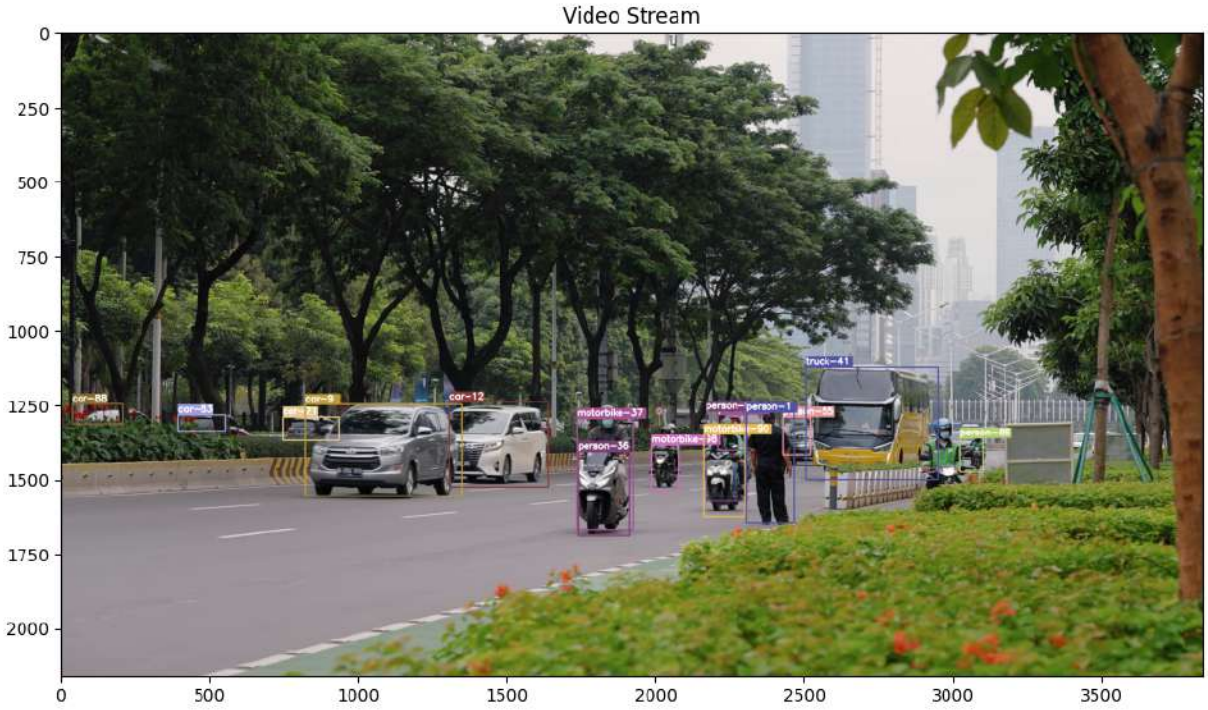



```
ii 39
torch.Size([18, 7])
(15, 6)
```


```
ii 40
torch.Size([17, 7])
(13, 6)
```


```
ii 41
torch.Size([17, 7])
(15, 6)
```


```
ii 42
torch.Size([19, 7])
(15, 6)
```




```
ii 43
torch.Size([17, 7])
(16, 6)
```

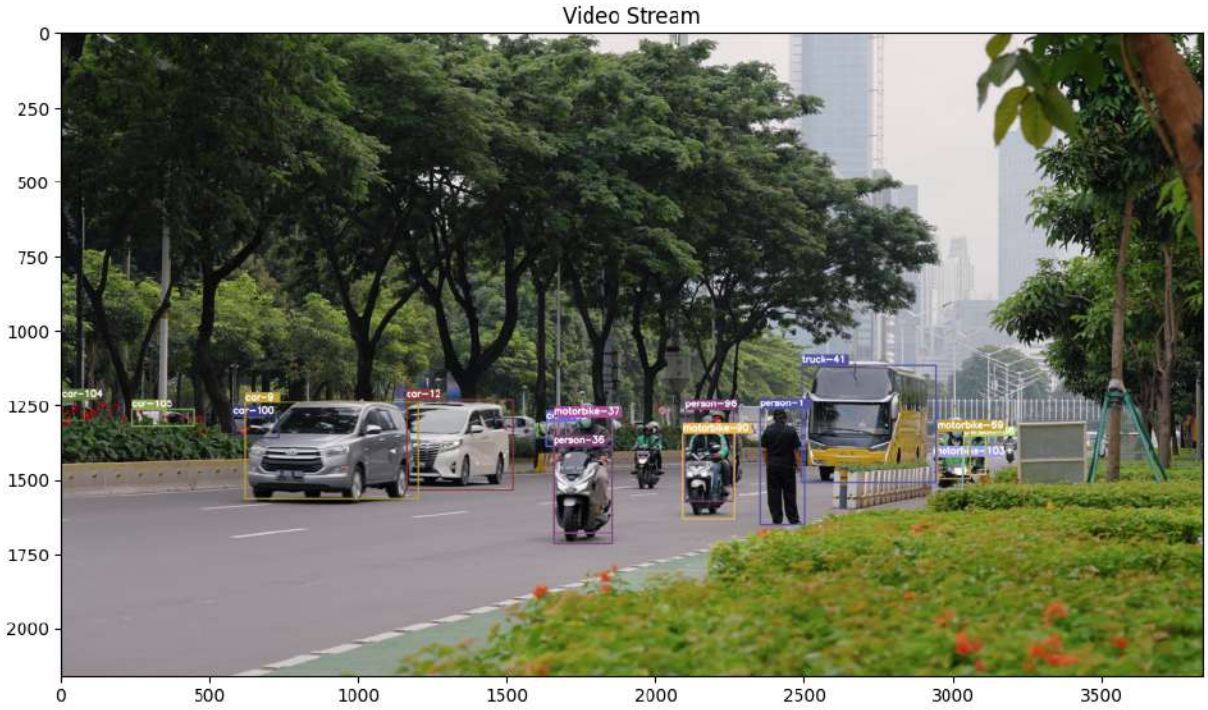

```
ii 44
torch.Size([18, 7])
(14, 6)
```


```
ii 45
torch.Size([17, 7])
(14, 6)
```


```
ii 46
torch.Size([20, 7])
(14, 6)
```


```
ii 47
torch.Size([19, 7])
(13, 6)
```


```
ii 48
torch.Size([18, 7])
(15, 6)
```




```
ii 49
torch.Size([18, 7])
(14, 6)
```


```
ii 50
torch.Size([17, 7])
(13, 6)
```


```
ii 51
torch.Size([17, 7])
(13, 6)
```


```
ii 52
torch.Size([19, 7])
(13, 6)
```


```
ii 53
torch.Size([20, 7])
(13, 6)
```


```
ii 54
torch.Size([21, 7])
(13, 6)
```


```
ii 55
torch.Size([19, 7])
(13, 6)
```


```
ii 56
torch.Size([21, 7])
(14, 6)
```


```
ii 57
torch.Size([24, 7])
(15, 6)
```


```
ii 58
torch.Size([24, 7])
(14, 6)
```


```
ii 59
torch.Size([22, 7])
(17, 6)
```


```
ii 60
torch.Size([19, 7])
(16, 6)
```


```
ii 61
torch.Size([20, 7])
(16, 6)
```


```
ii 62
torch.Size([22, 7])
(16, 6)
```


```
ii 63
torch.Size([21, 7])
(17, 6)
```


```
ii 64
torch.Size([24, 7])
(17, 6)
```


```
ii 65
torch.Size([24, 7])
(19, 6)
```


```
ii 66
torch.Size([24, 7])
(20, 6)
```


```
ii 67
torch.Size([27, 7])
(22, 6)
```


```
ii 68
torch.Size([27, 7])
(23, 6)
```


```
ii 69
torch.Size([26, 7])
(23, 6)
```


```
ii 70
torch.Size([25, 7])
(23, 6)
```


```
ii 71
torch.Size([27, 7])
(22, 6)
```


```
ii 72
torch.Size([25, 7])
(22, 6)
```


```
ii 73
torch.Size([25, 7])
(20, 6)
```


```
ii 74
torch.Size([24, 7])
(22, 6)
```


```
ii 75
torch.Size([26, 7])
(22, 6)
```


```
ii 76
torch.Size([27, 7])
(22, 6)
```


```
ii 77
torch.Size([27, 7])
(22, 6)
```


```
ii 78
torch.Size([28, 7])
(22, 6)
```


```
ii 79
torch.Size([28, 7])
(25, 6)
```


```
ii 80
torch.Size([32, 7])
(25, 6)
```


```
ii 81
torch.Size([28, 7])
(23, 6)
```

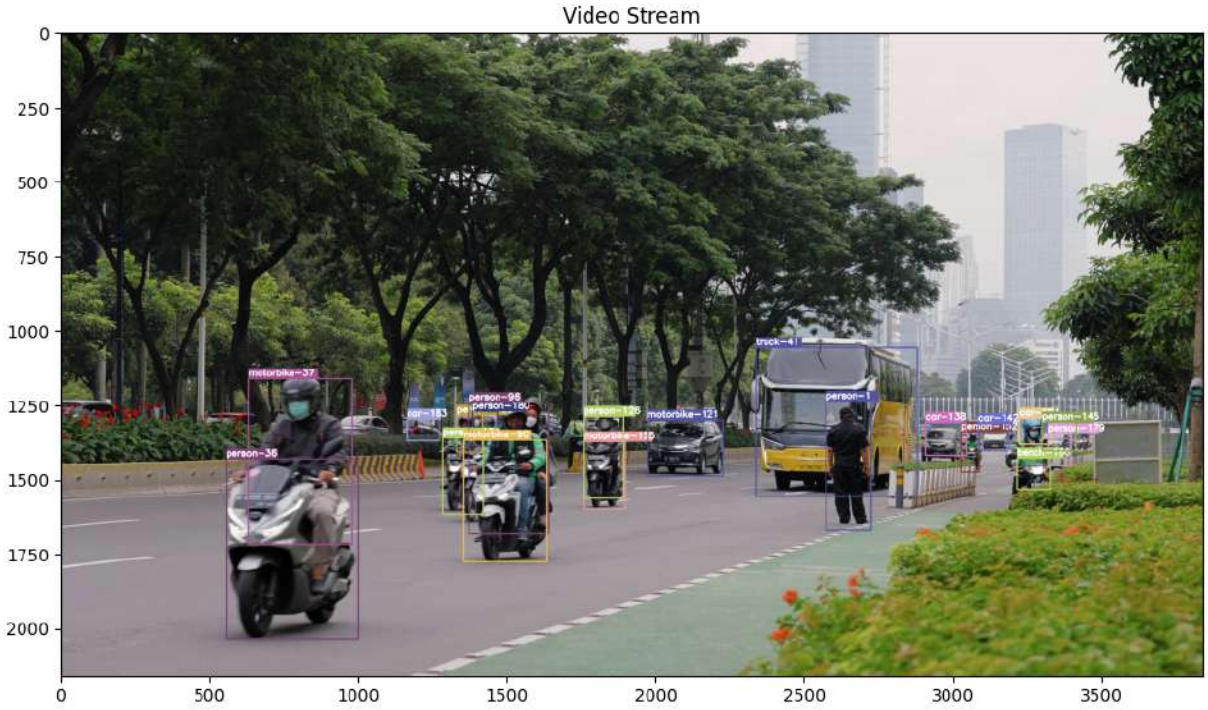

```
ii 82
torch.Size([29, 7])
(23, 6)
```


```
ii 83
torch.Size([34, 7])
(23, 6)
```


```
ii 84
torch.Size([28, 7])
(21, 6)
```


```
ii 85
torch.Size([29, 7])
(18, 6)
```


```
ii 86
torch.Size([31, 7])
(20, 6)
```



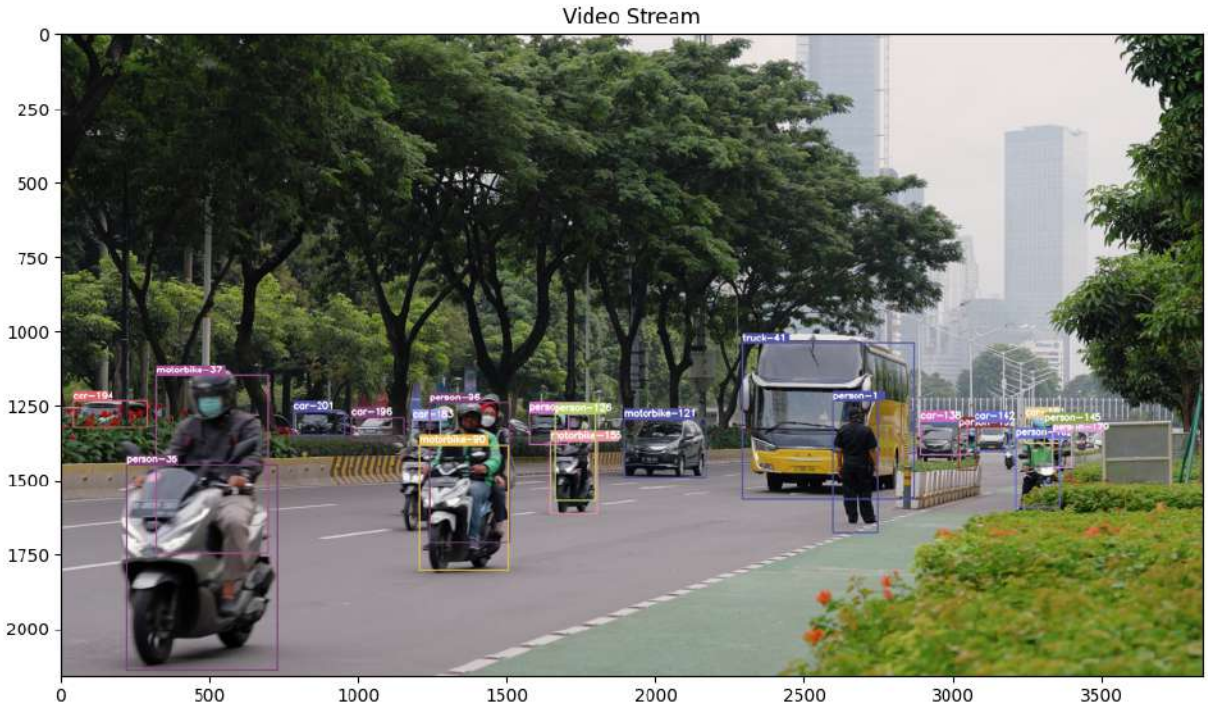

```
ii 87
torch.Size([27, 7])
(22, 6)
```


```
ii 88
torch.Size([26, 7])
(22, 6)
```

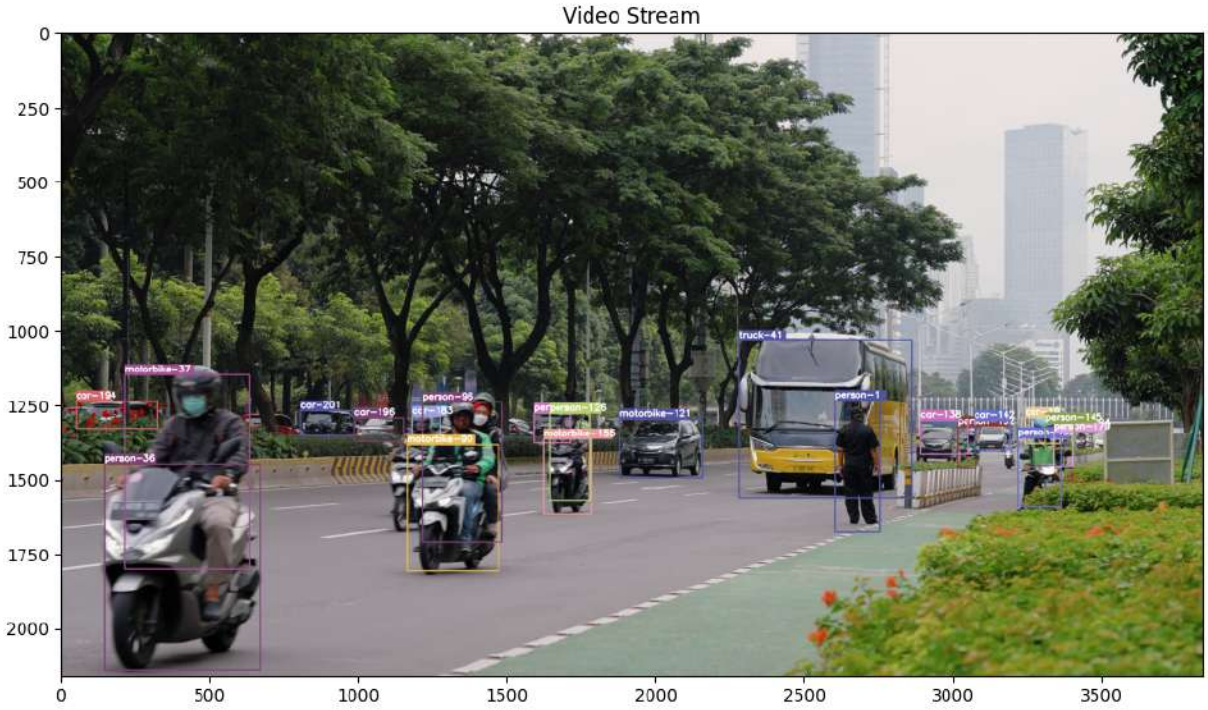

```
ii 89
torch.Size([29, 7])
(21, 6)
```


```
ii 90
torch.Size([24, 7])
(20, 6)
```

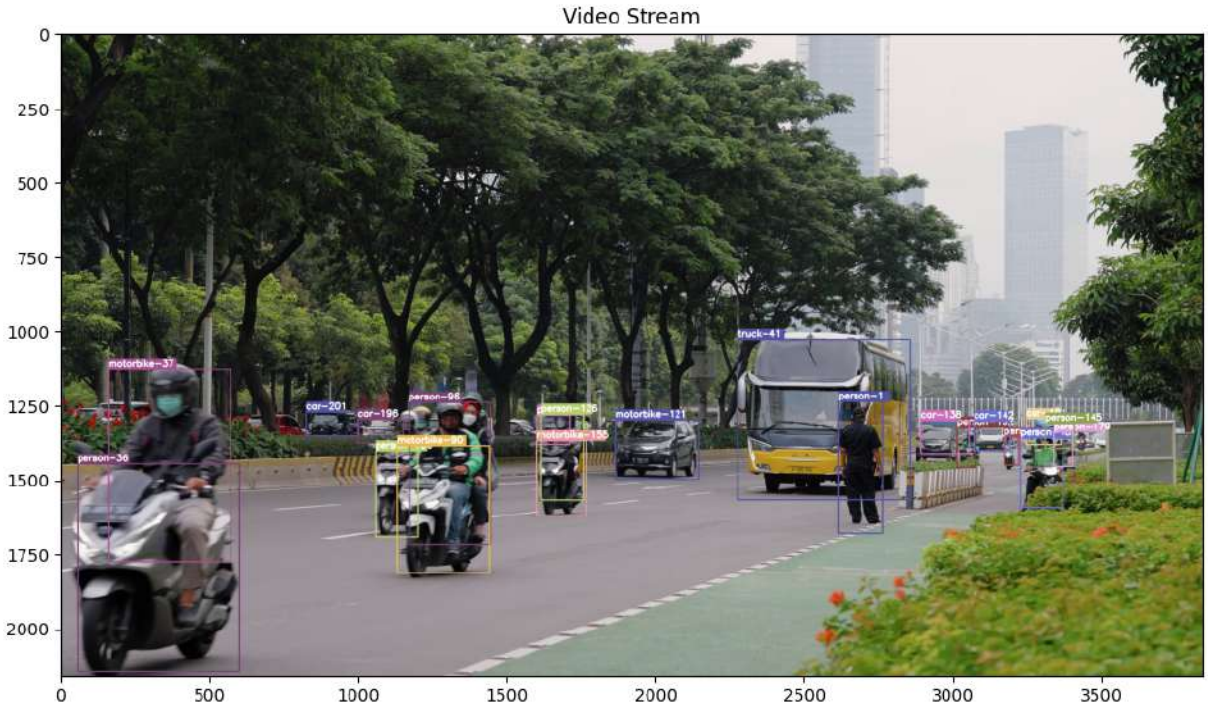

```
ii 91
torch.Size([30, 7])
(21, 6)
```



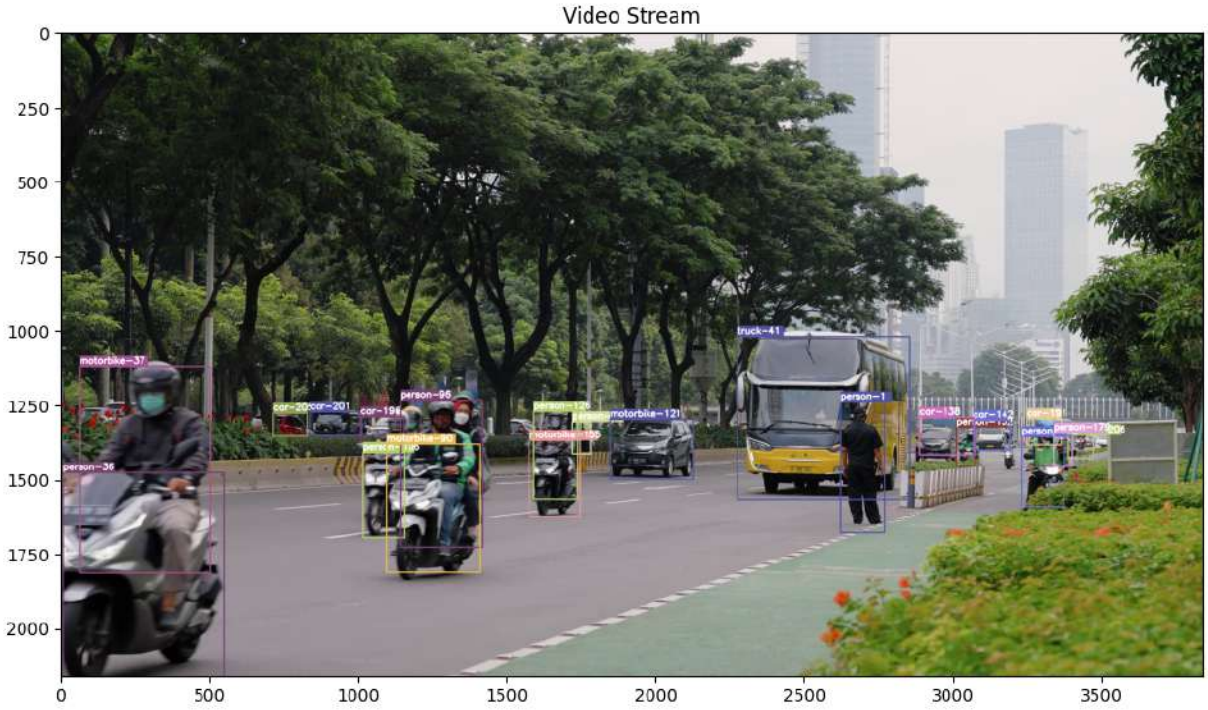

```
ii 92
torch.Size([31, 7])
(22, 6)
```



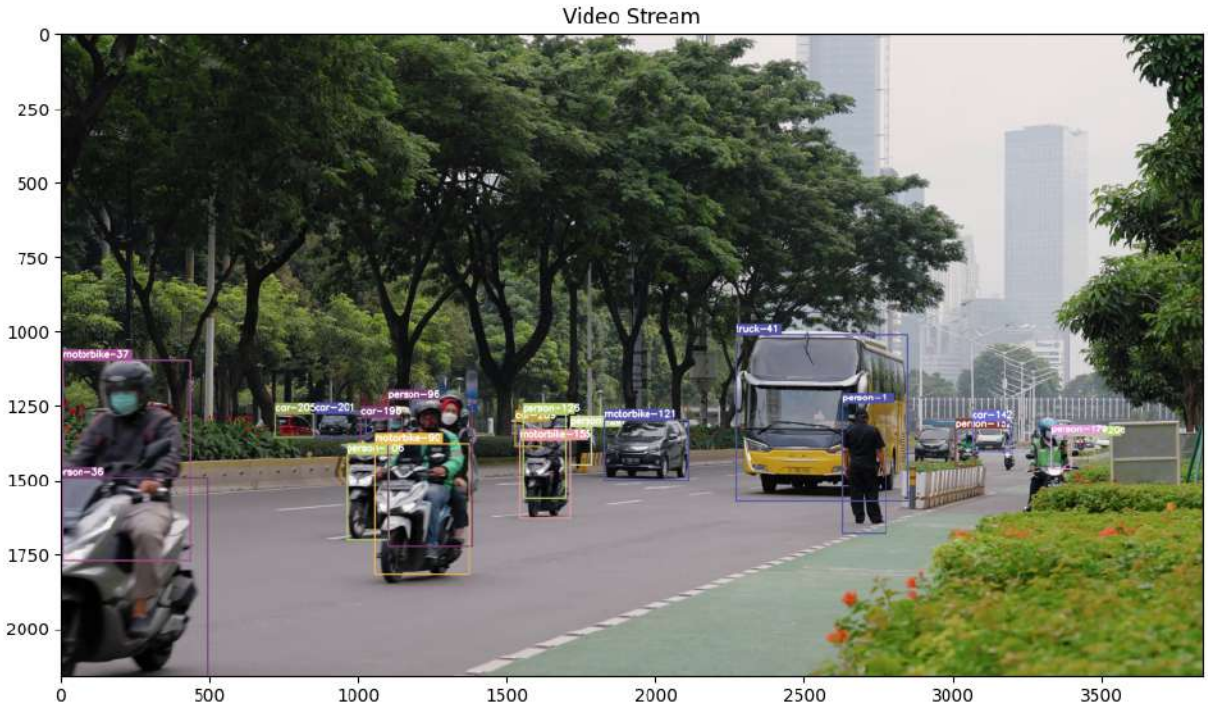

```
ii 93
torch.Size([31, 7])
(21, 6)
```



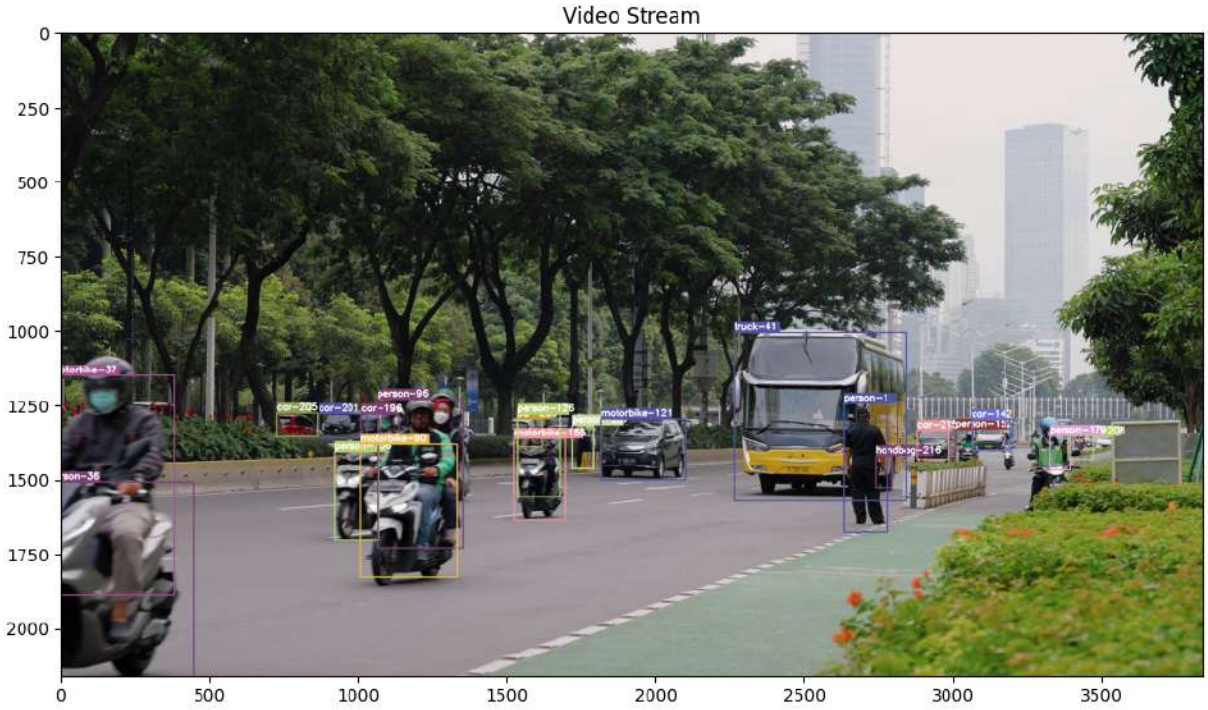

```
ii 94
torch.Size([26, 7])
(21, 6)
```




```
ii 95
torch.Size([28, 7])
(19, 6)
```



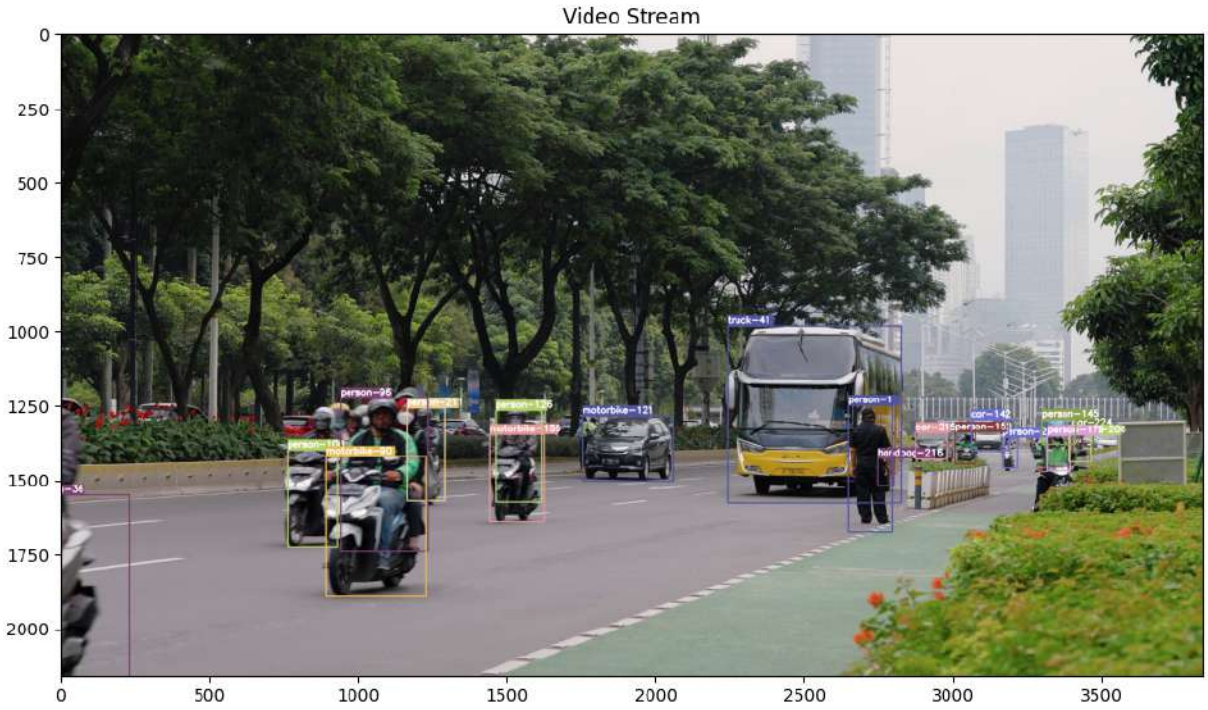

```
ii 96
torch.Size([28, 7])
(21, 6)
```



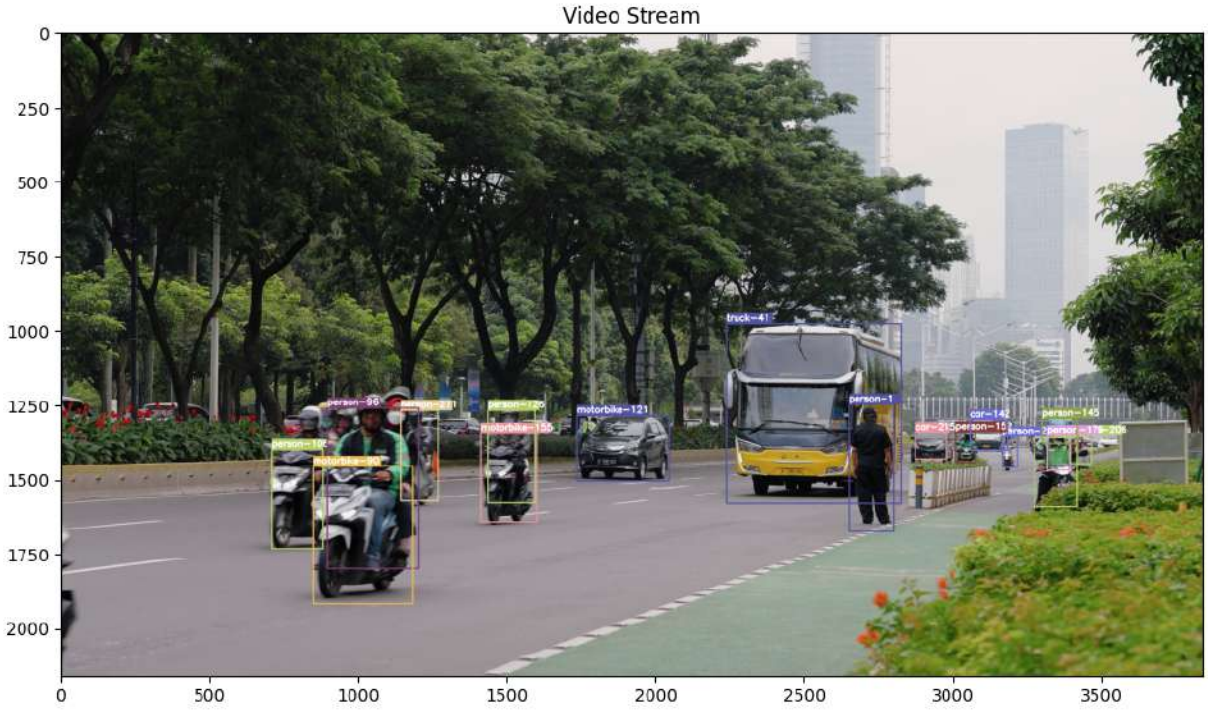

```
ii 97
torch.Size([27, 7])
(21, 6)
```


```
ii 98
torch.Size([28, 7])
(19, 6)
```

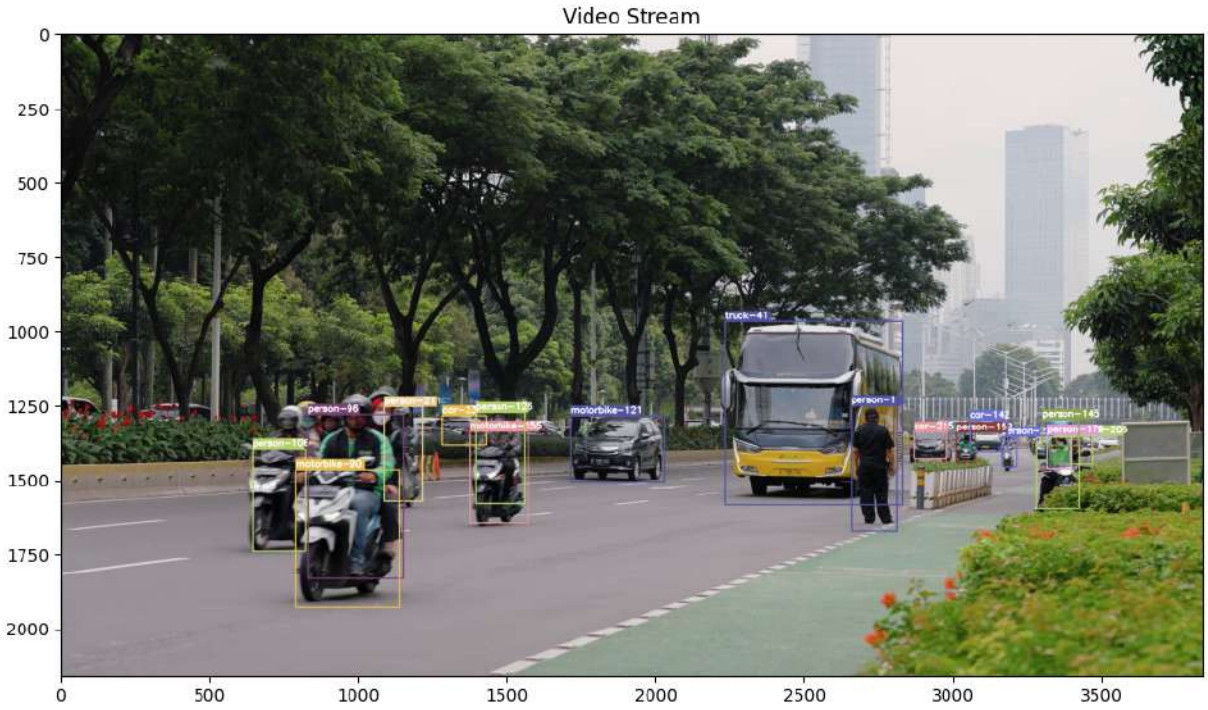

```
ii 99
torch.Size([26, 7])
(19, 6)
```




```
ii 100
torch.Size([25, 7])
(17, 6)
```

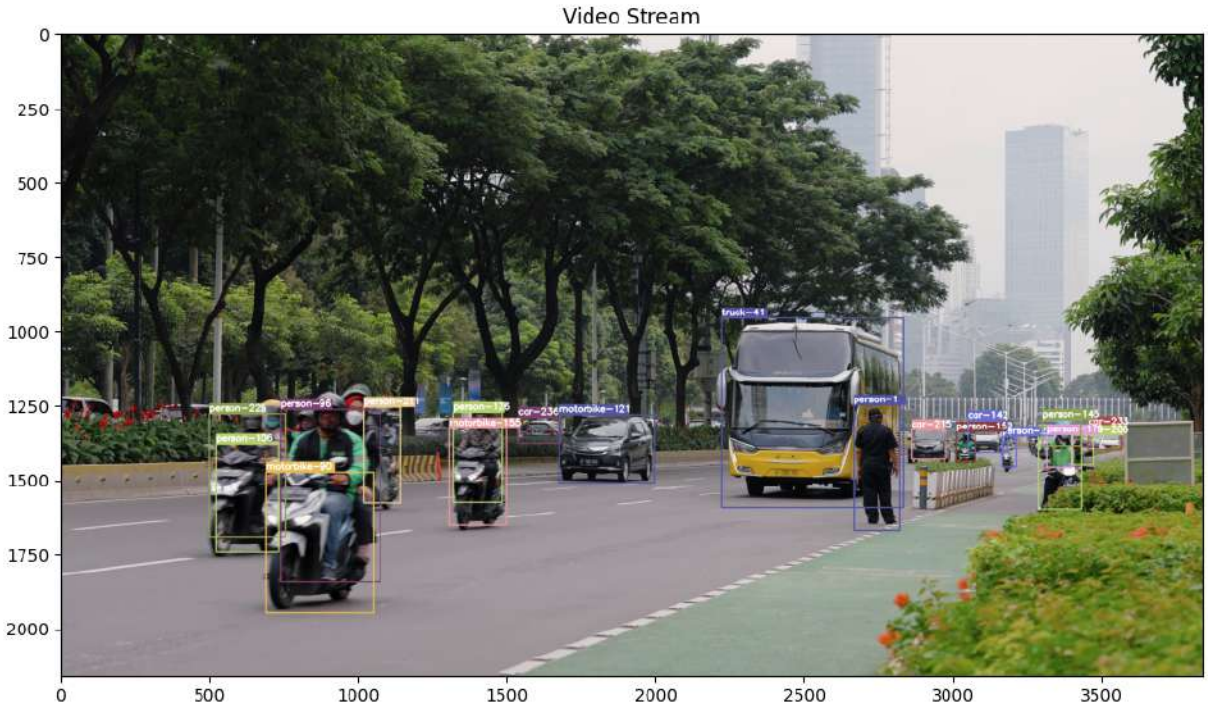



```
ii 101
torch.Size([28, 7])
(18, 6)
```

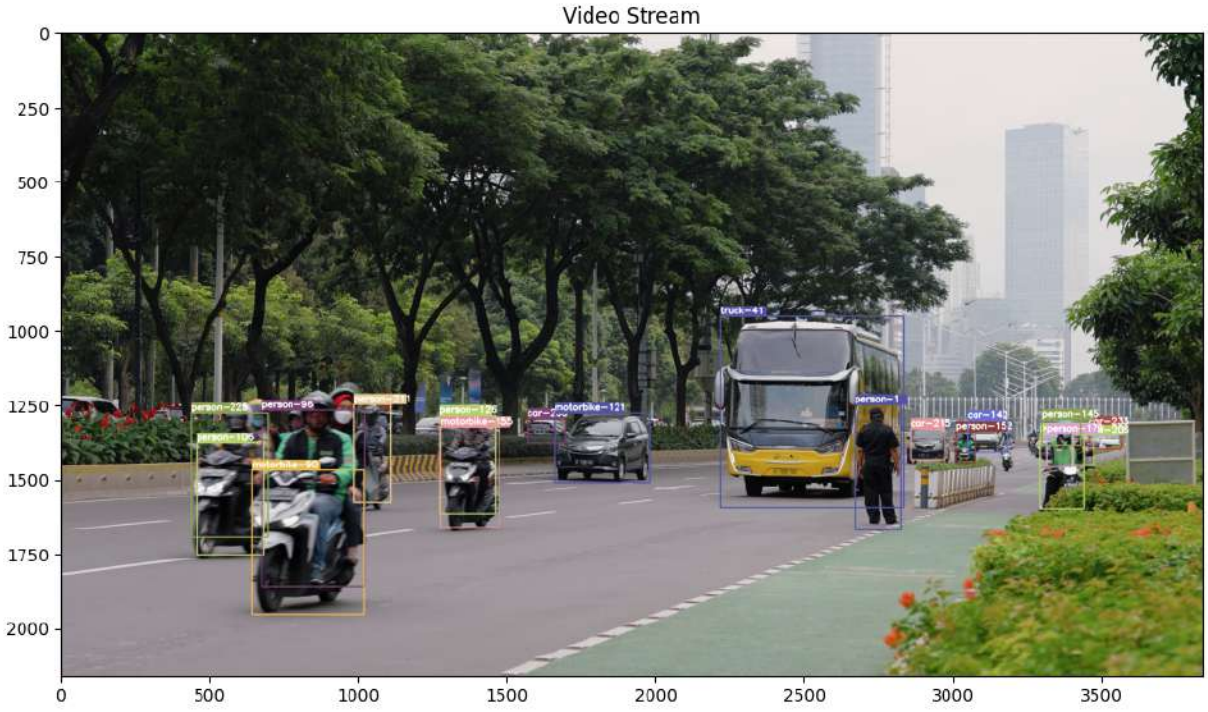



```
ii 102
torch.Size([32, 7])
(18, 6)
```

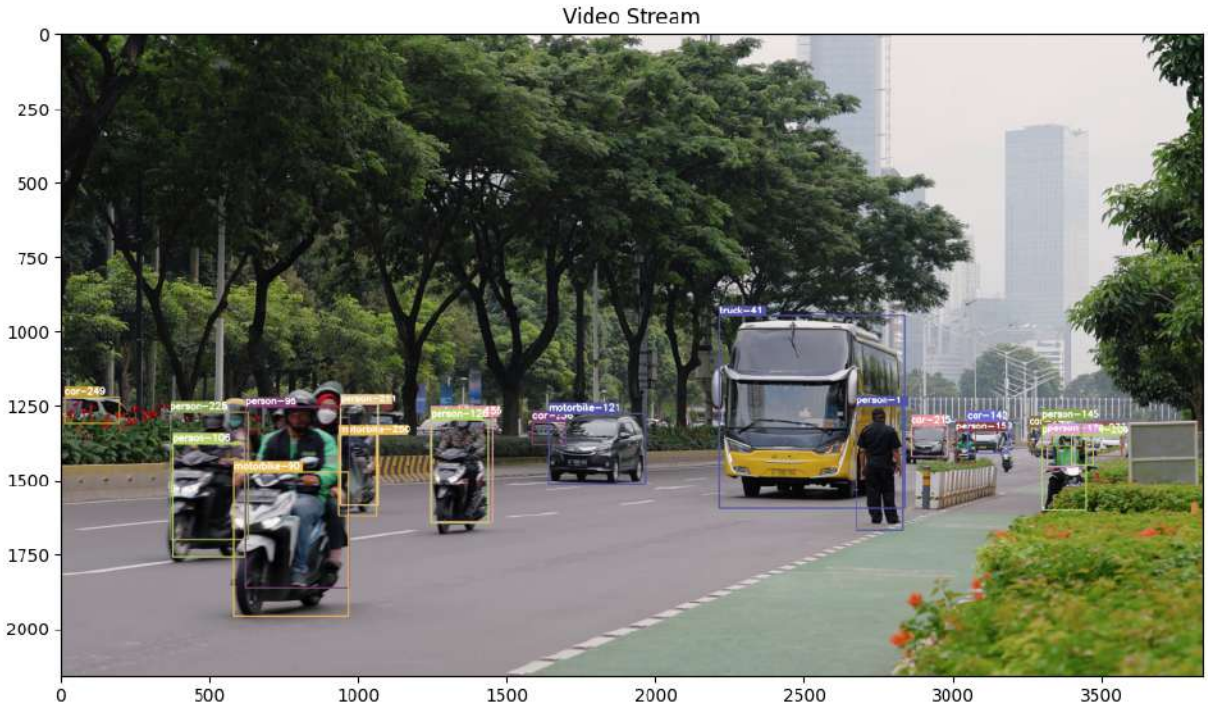

```
ii 103
torch.Size([27, 7])
(20, 6)
```



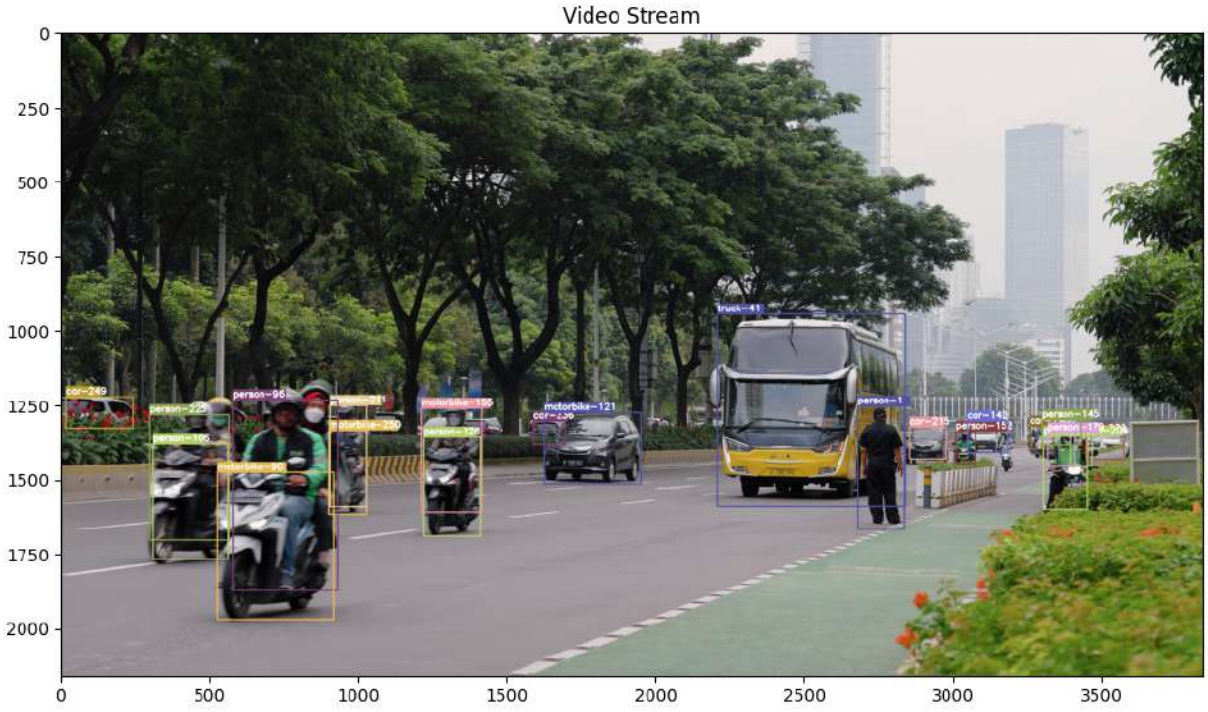

```
ii 104
torch.Size([25, 7])
(19, 6)
```



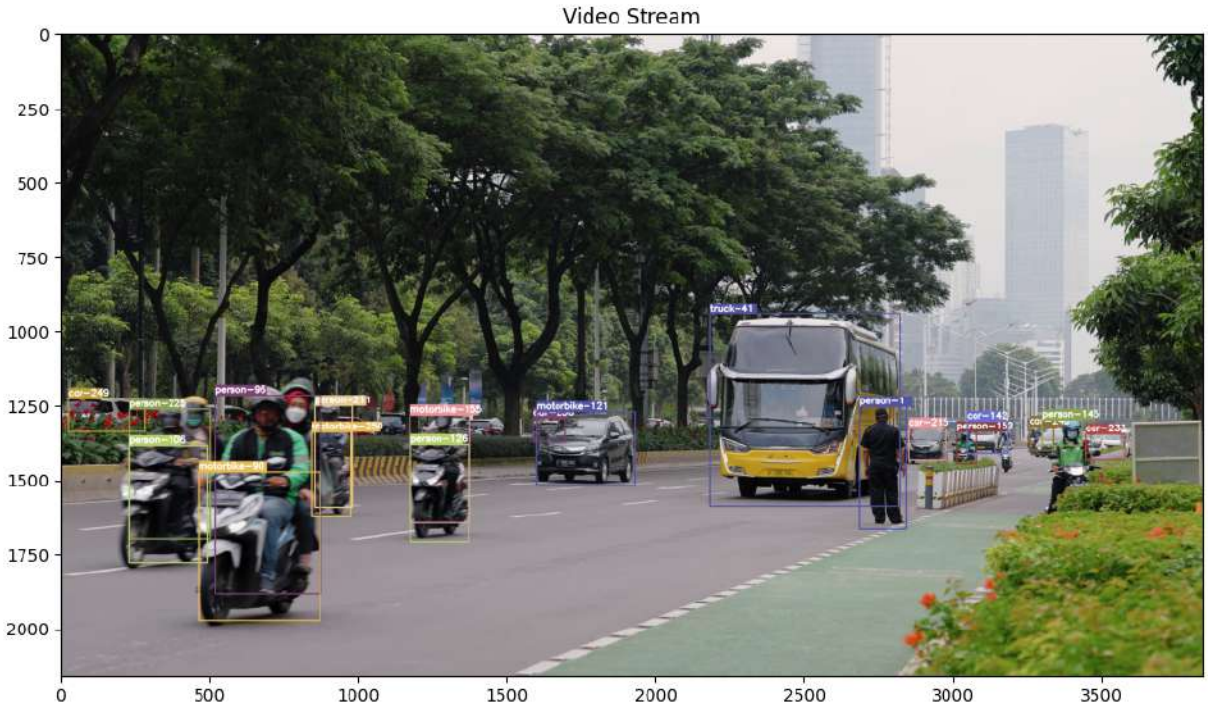

```
ii 105
torch.Size([28, 7])
(22, 6)
```



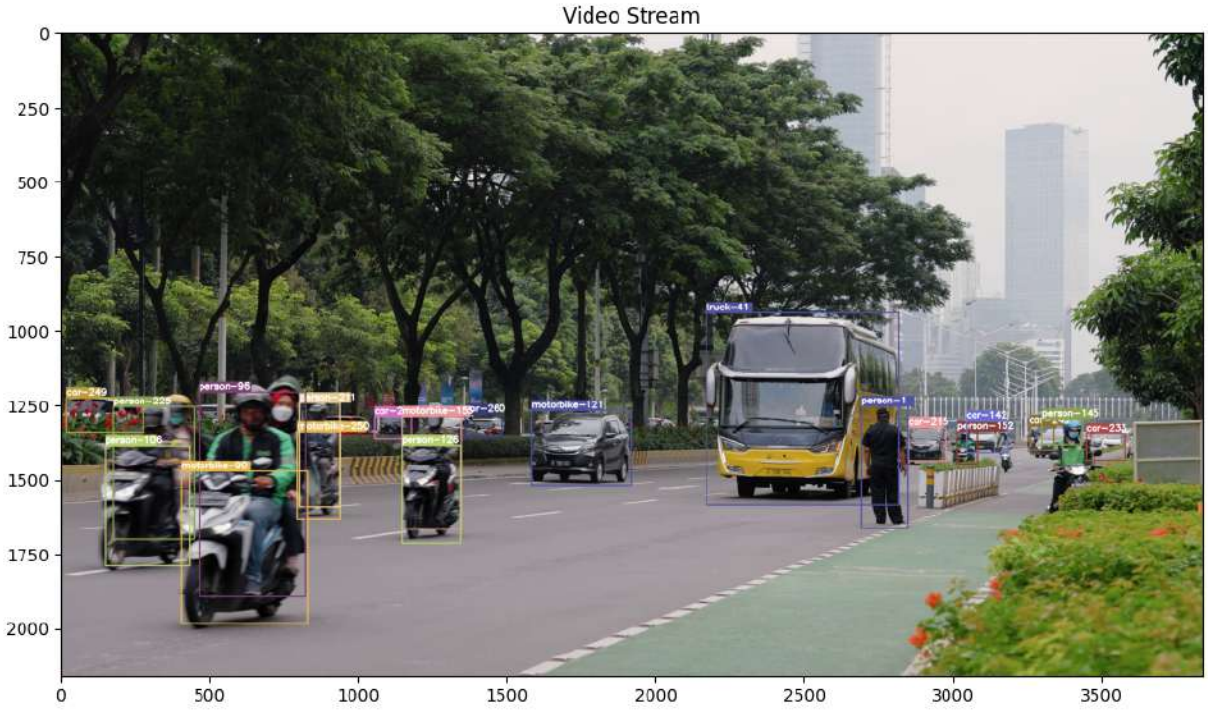

```
ii 106
torch.Size([27, 7])
(22, 6)
```



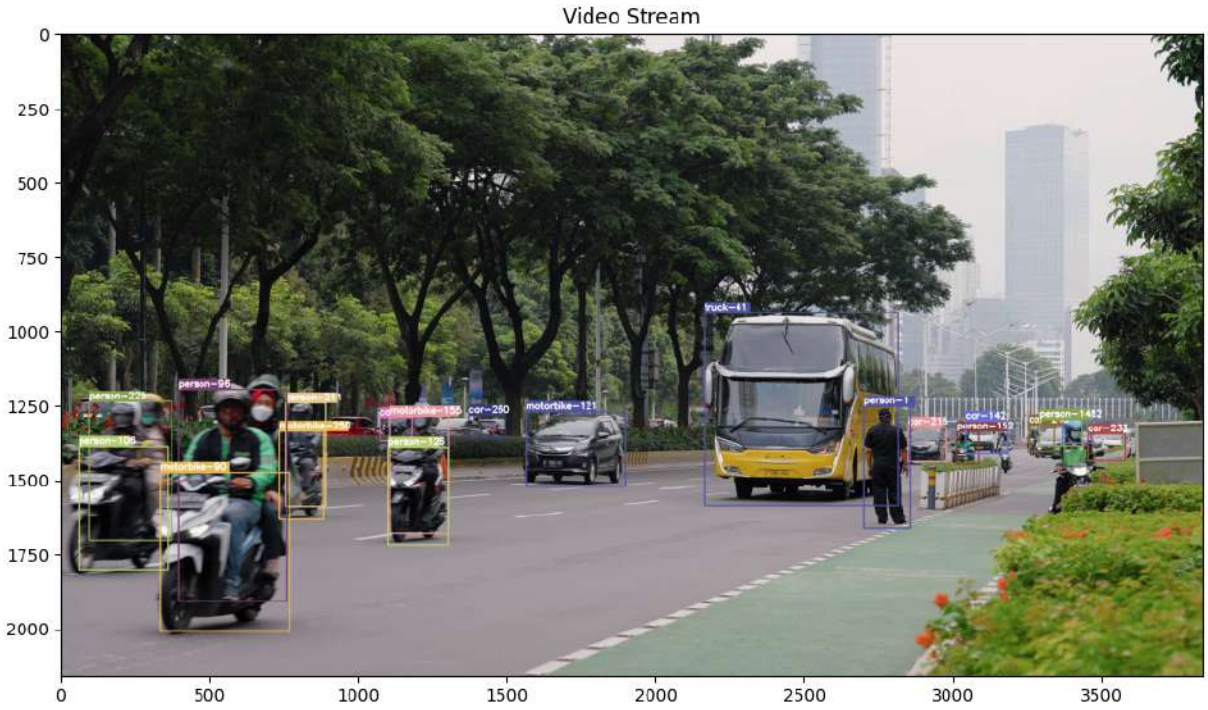

```
ii 107
torch.Size([25, 7])
(19, 6)
```



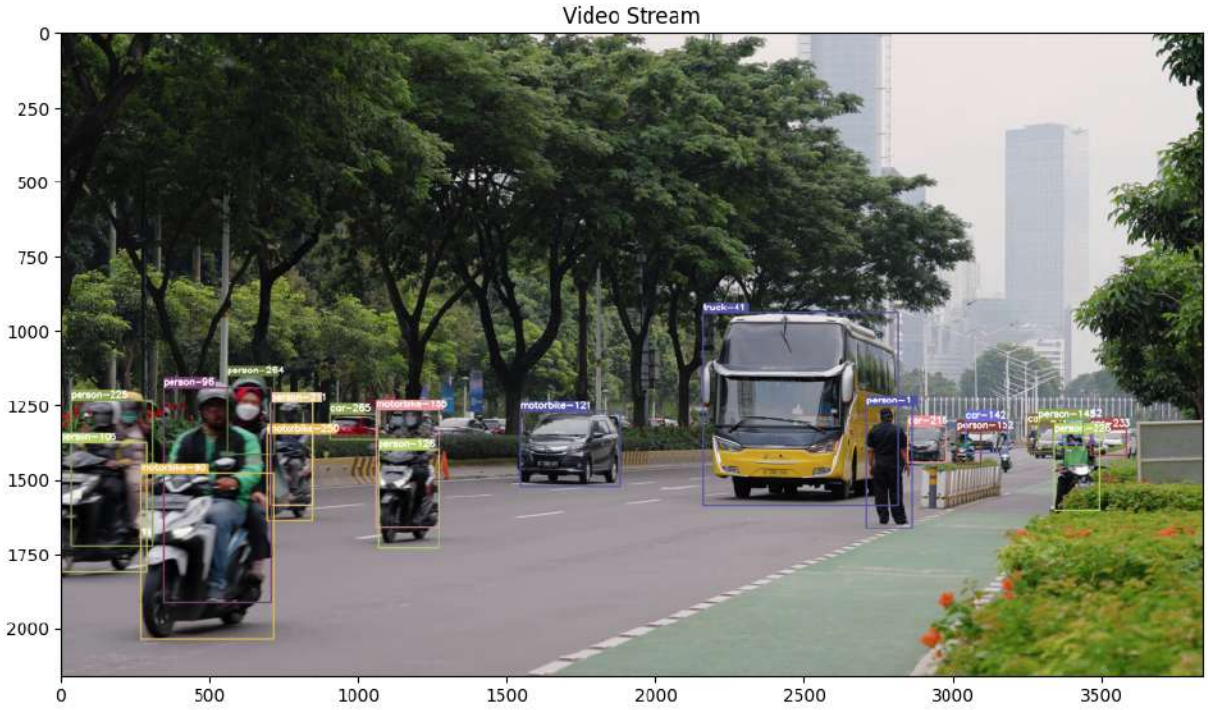

```
ii 108
torch.Size([26, 7])
(20, 6)
```



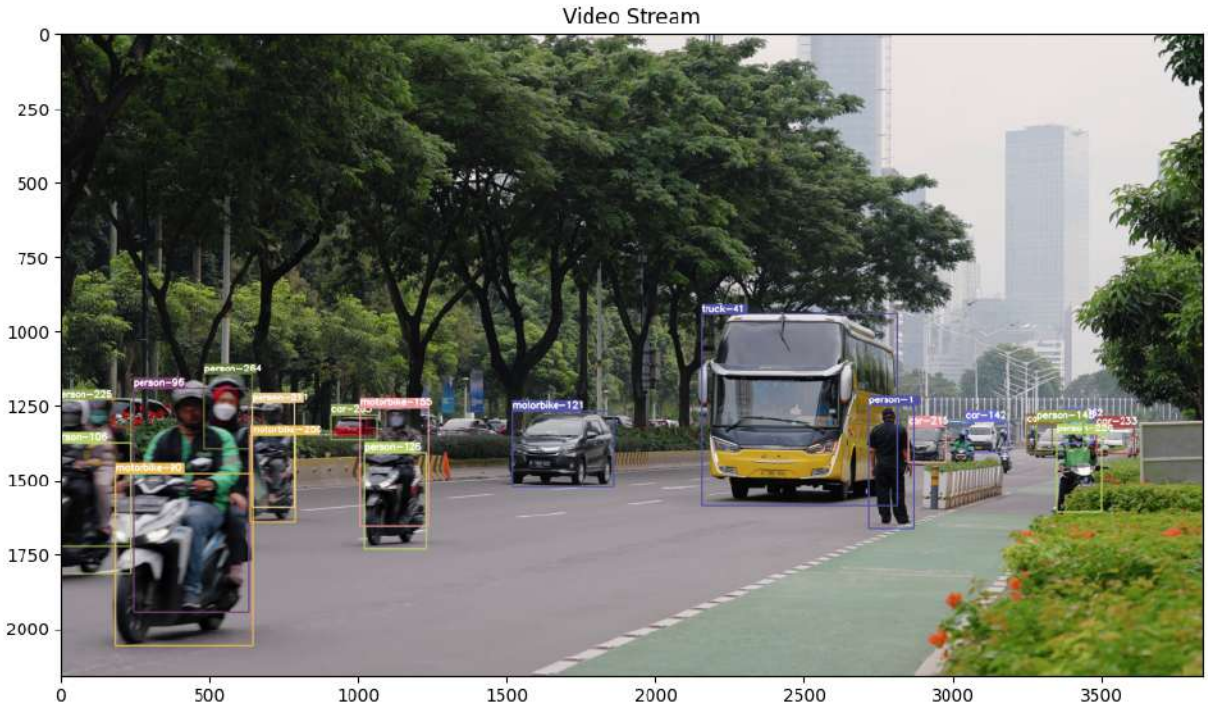

```
ii 109
torch.Size([25, 7])
(20, 6)
```



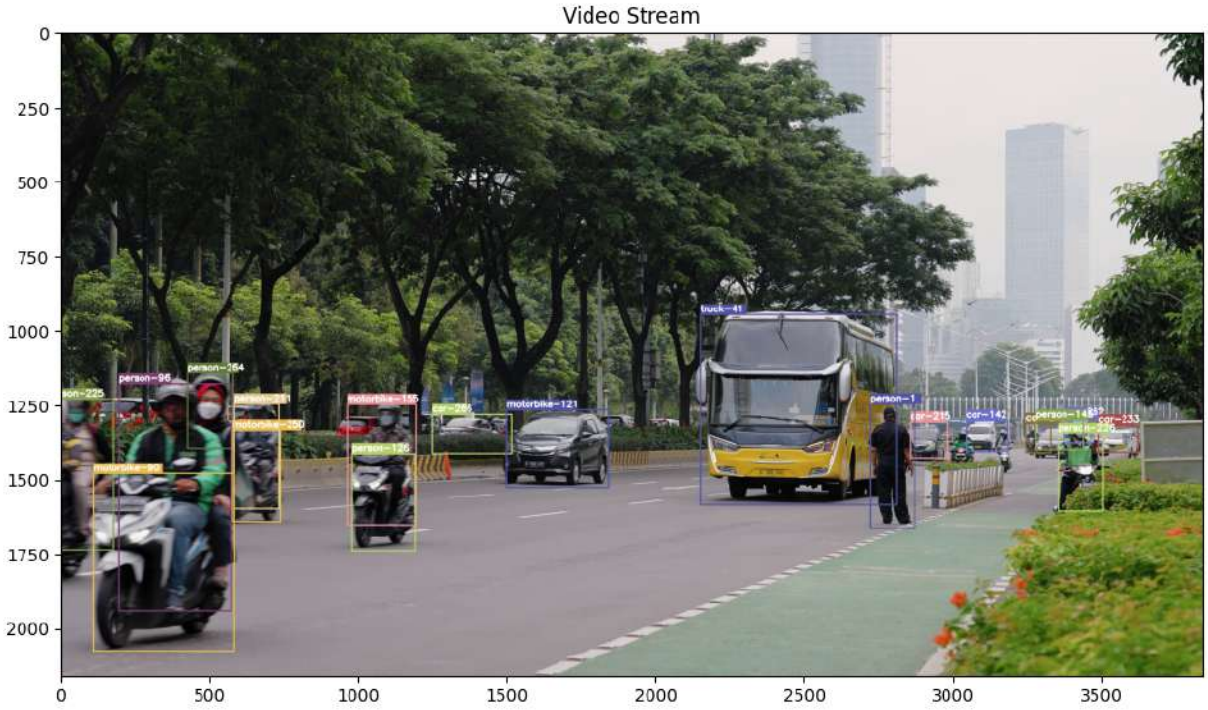

```
ii 110
torch.Size([25, 7])
(21, 6)
```



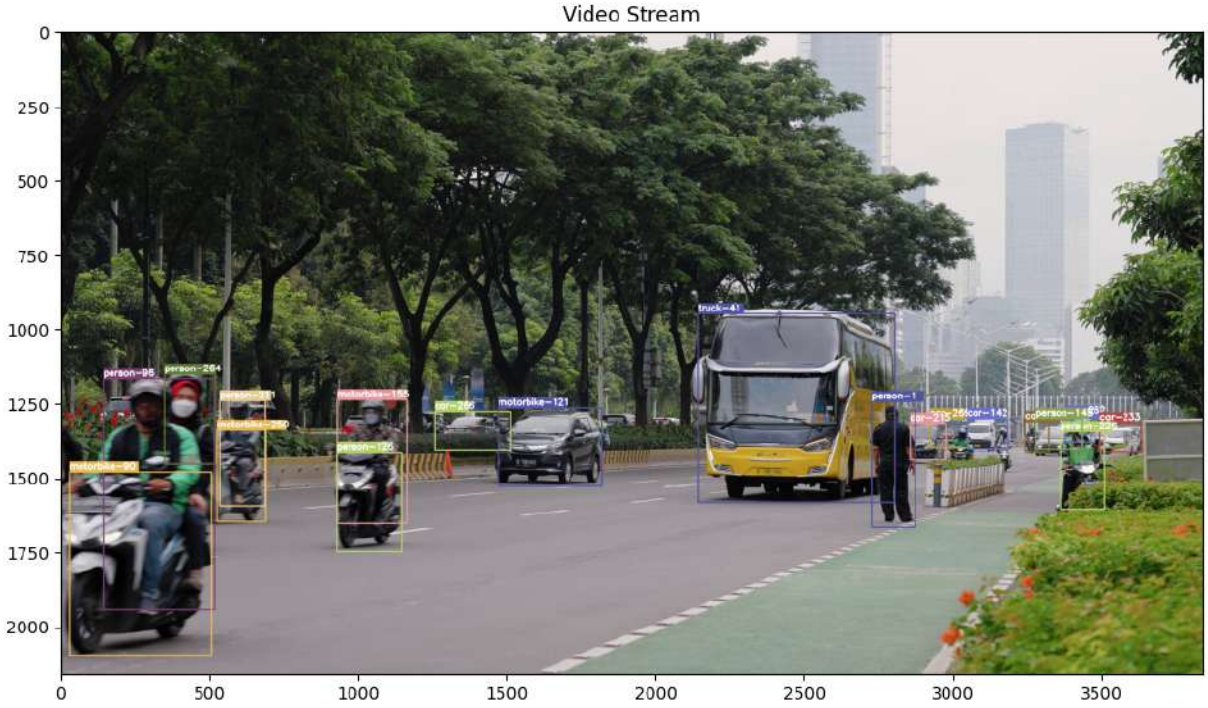

```
ii 111
torch.Size([24, 7])
(20, 6)
```



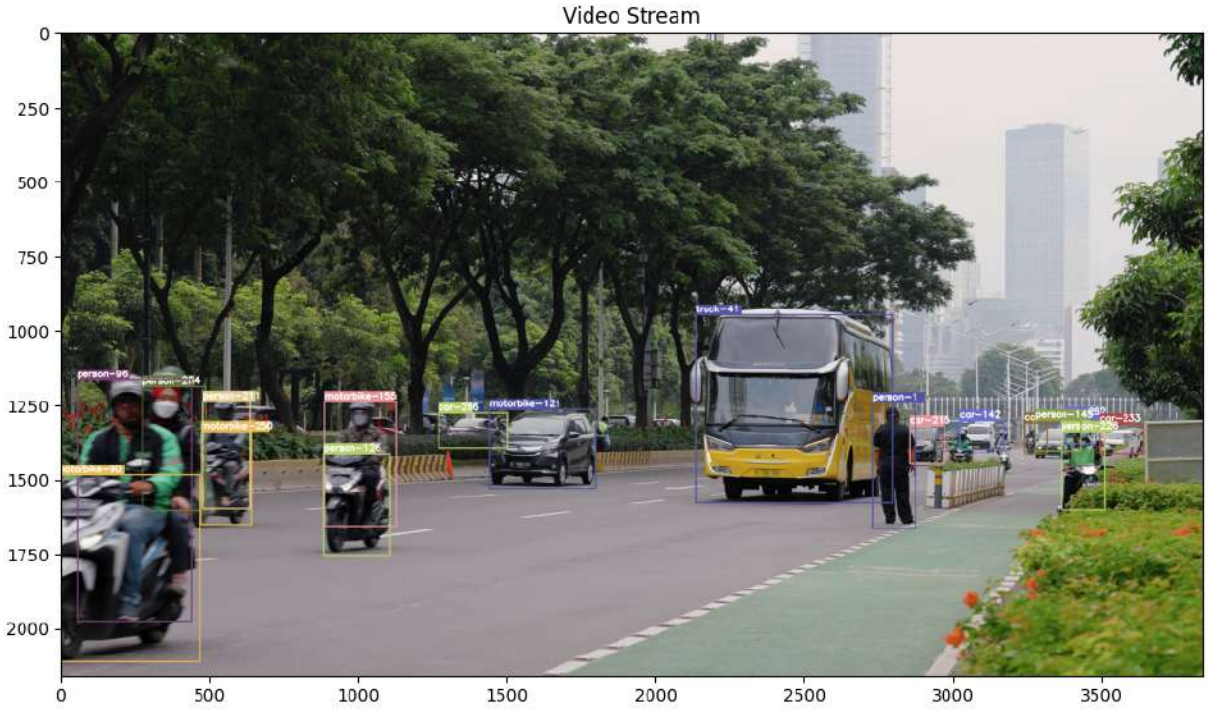

```
ii 112
torch.Size([26, 7])
(19, 6)
```



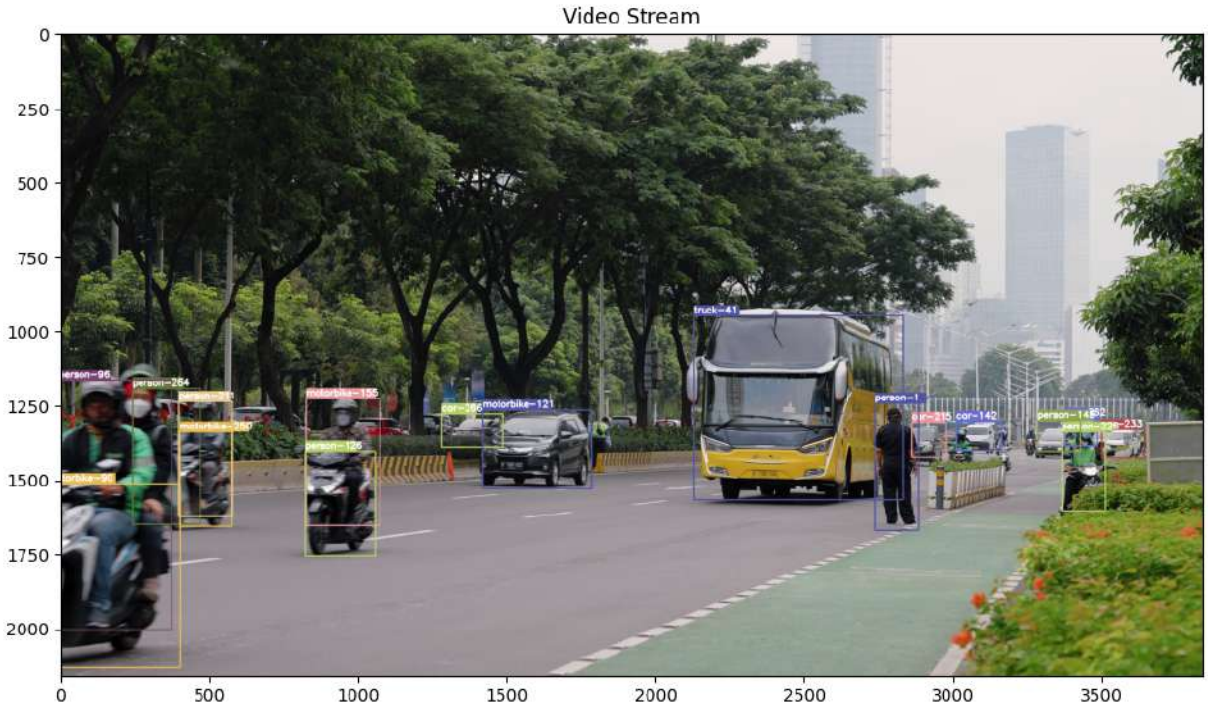

```
ii 113
torch.Size([26, 7])
(19, 6)
```



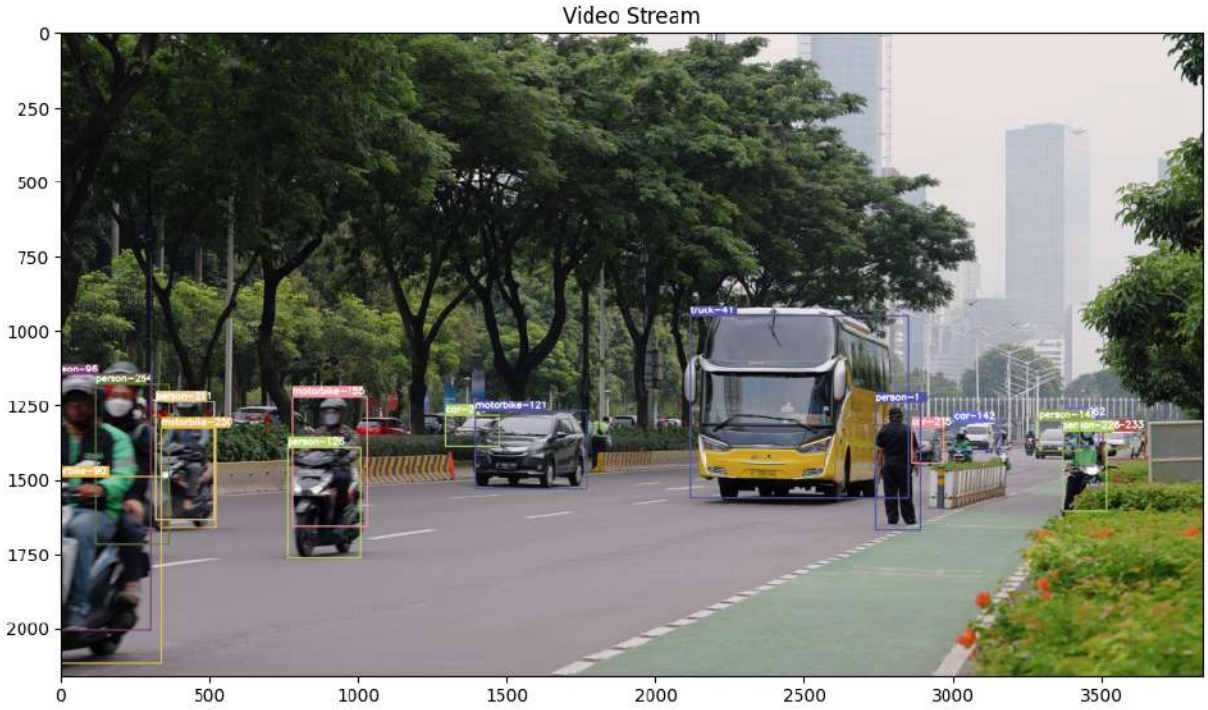

```
ii 114
torch.Size([19, 7])
(18, 6)
```



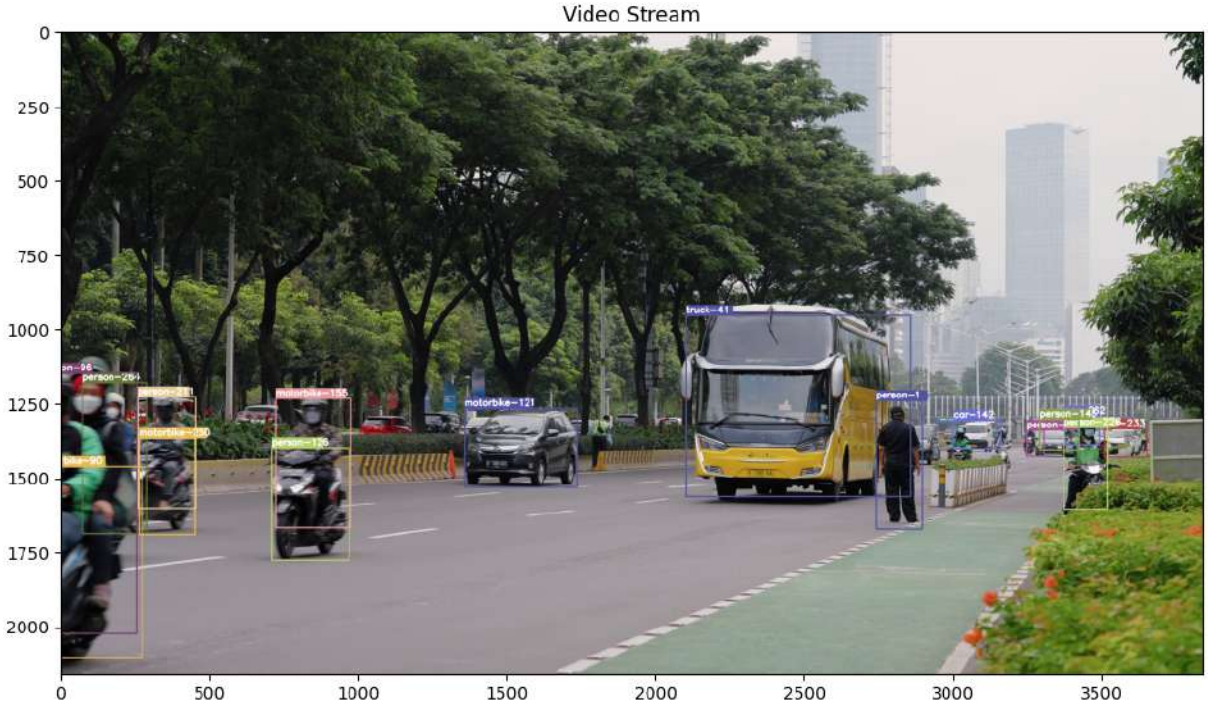

```
ii 115
torch.Size([24, 7])
(17, 6)
```



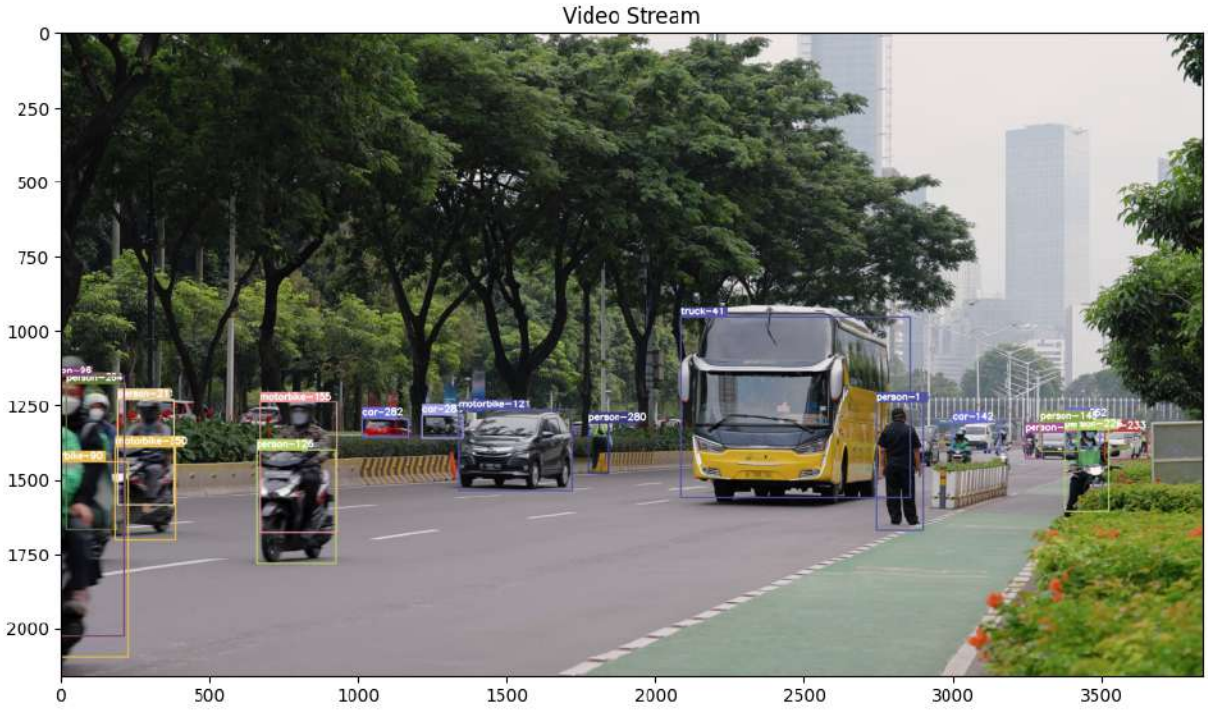

```
ii 116
torch.Size([24, 7])
(17, 6)
```



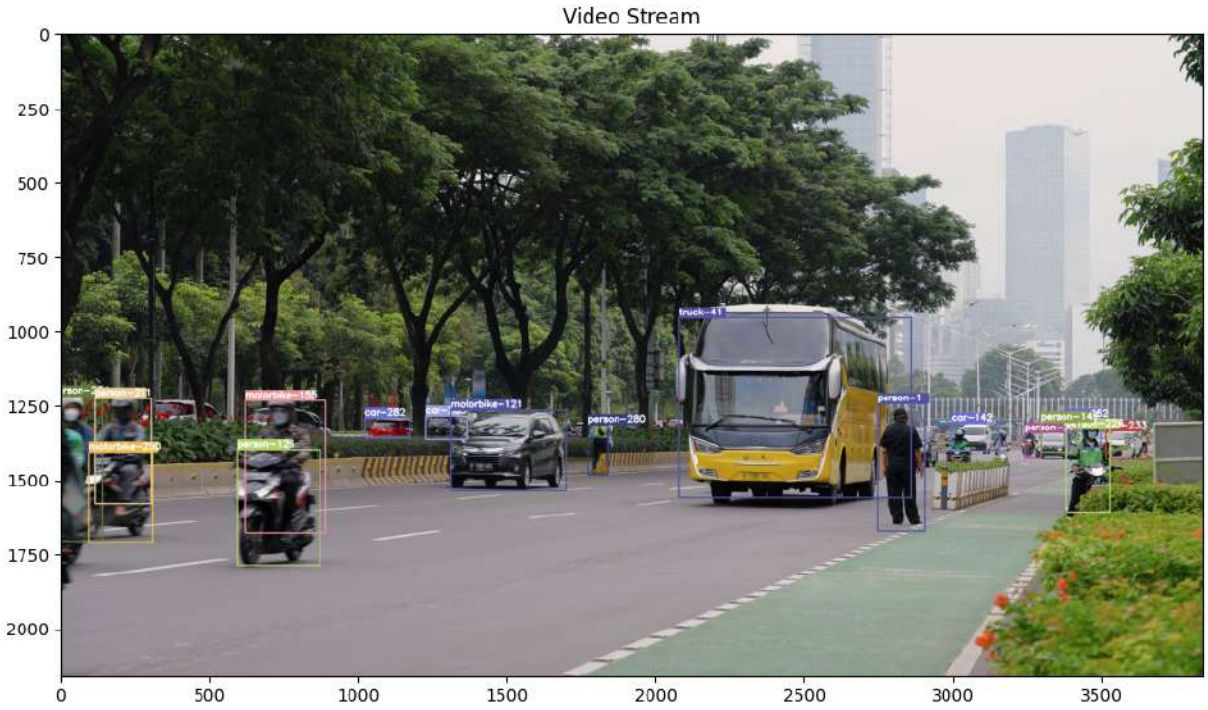

```
ii 117
torch.Size([21, 7])
(16, 6)
```



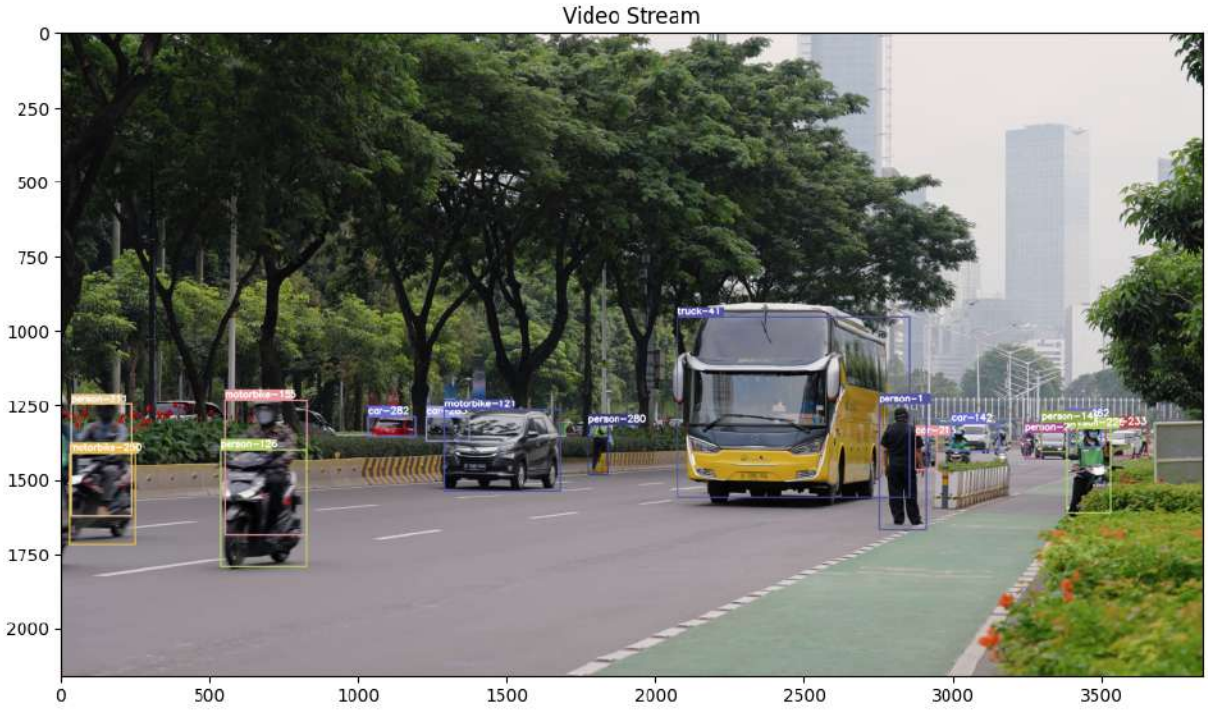

```
ii 118
torch.Size([22, 7])
(19, 6)
```



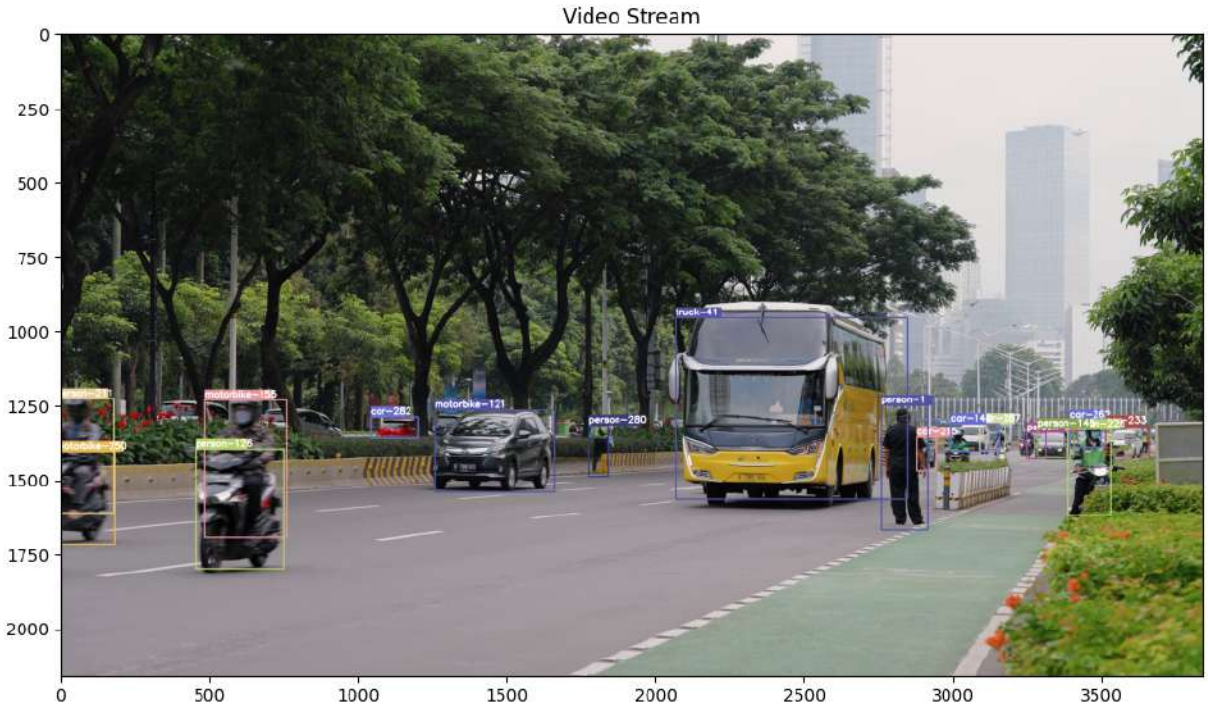

```
ii 119
torch.Size([23, 7])
(17, 6)
```



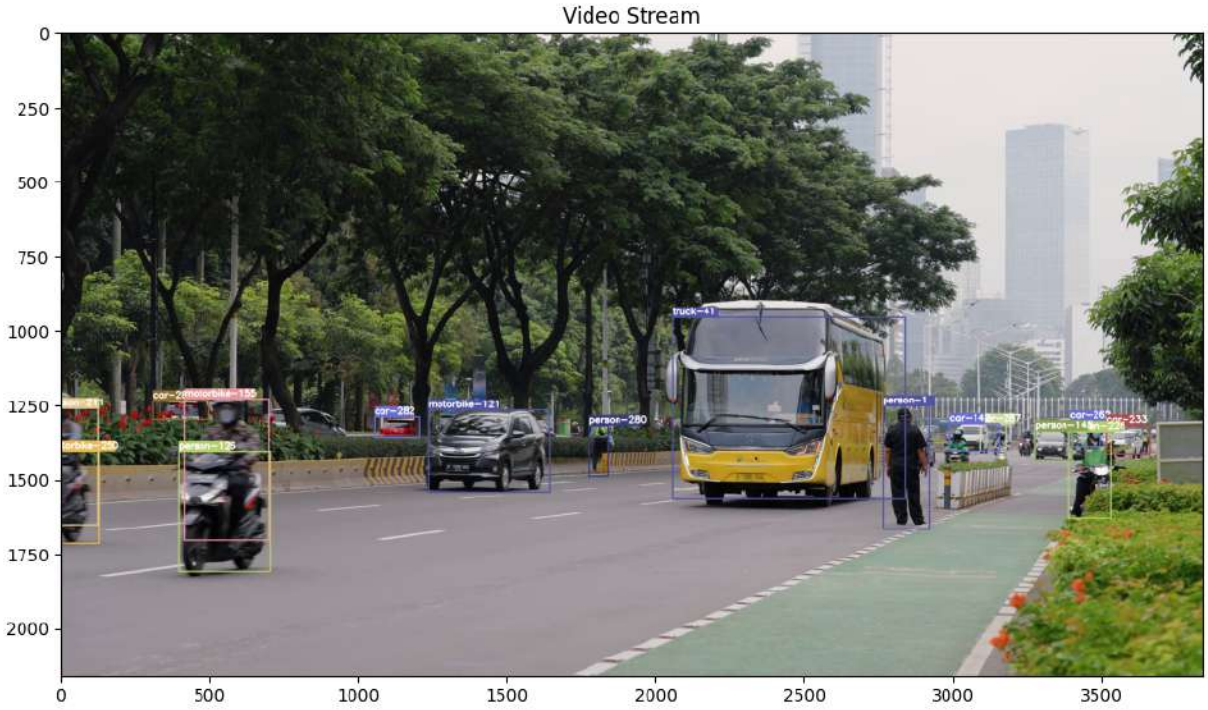

```
ii 120
torch.Size([23, 7])
(17, 6)
```



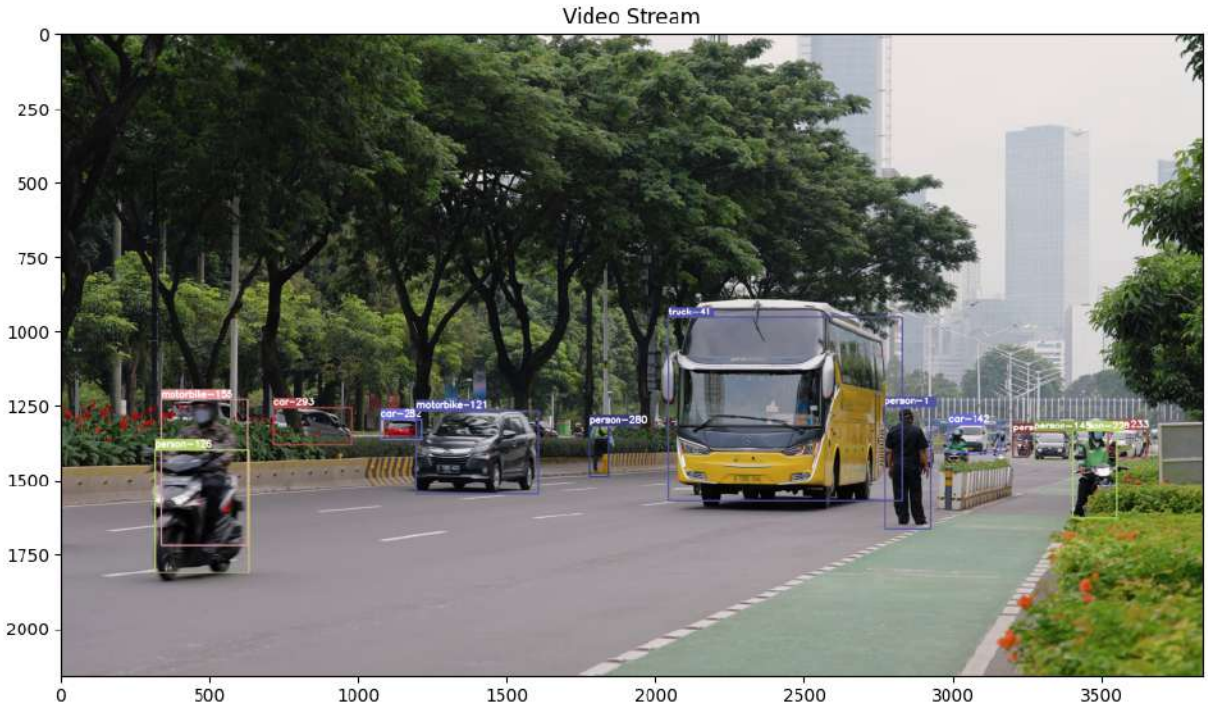

```
ii 121
torch.Size([23, 7])
(17, 6)
```



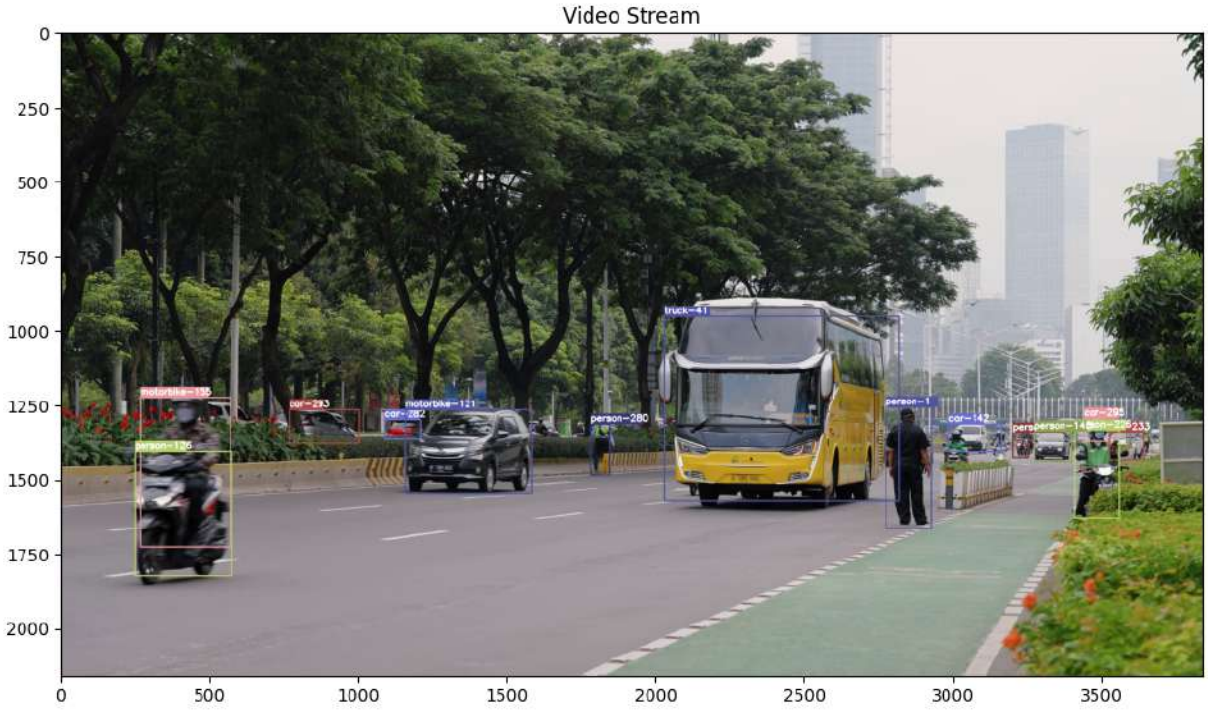

```
ii 122
torch.Size([19, 7])
(16, 6)
```



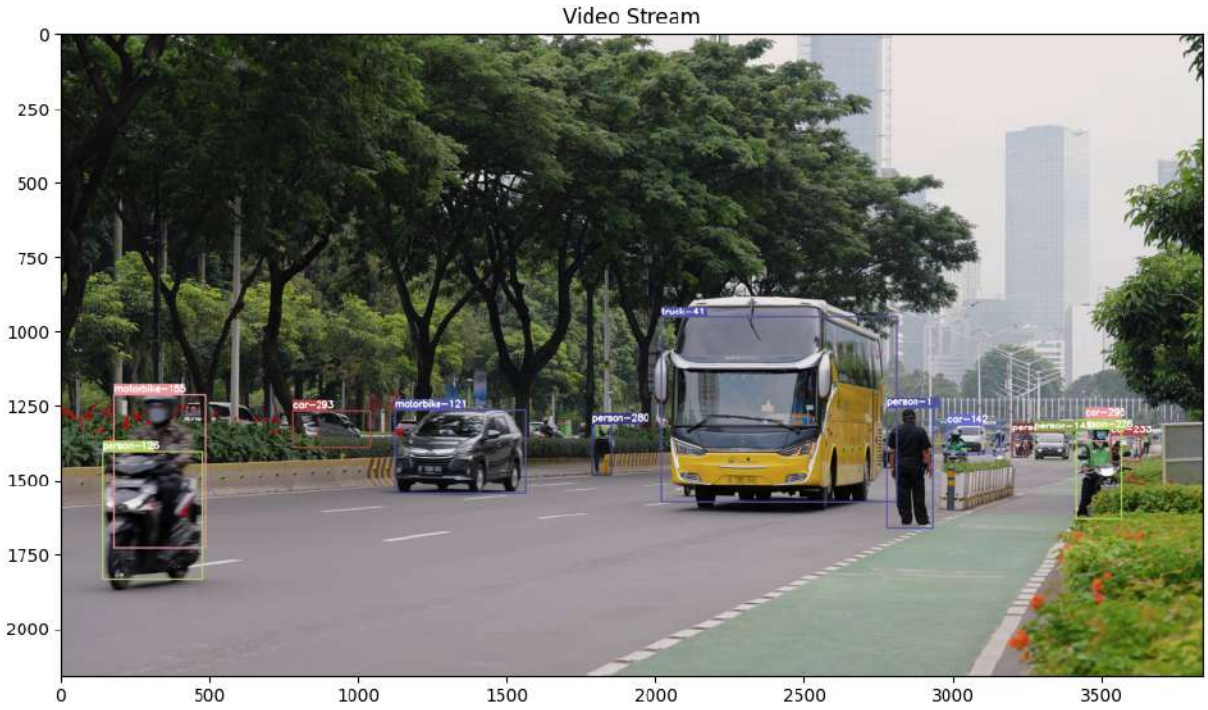

```
ii 123
torch.Size([16, 7])
(13, 6)
```



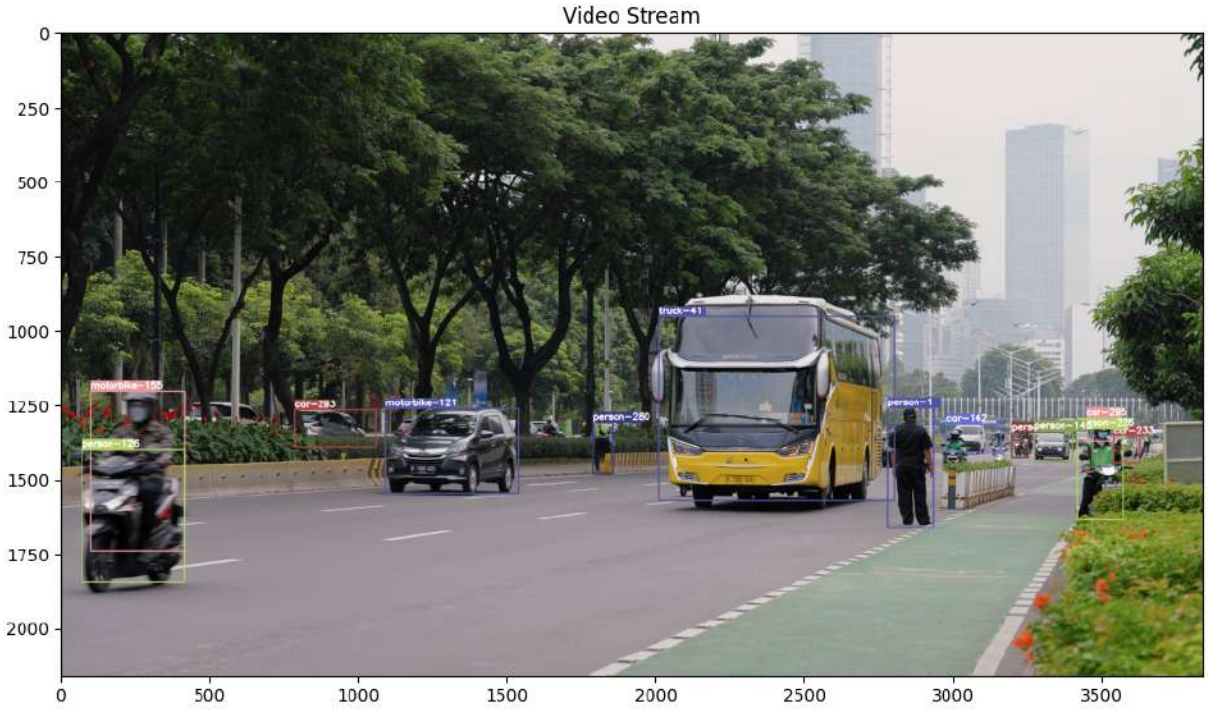

```
ii 124
torch.Size([16, 7])
(14, 6)
```



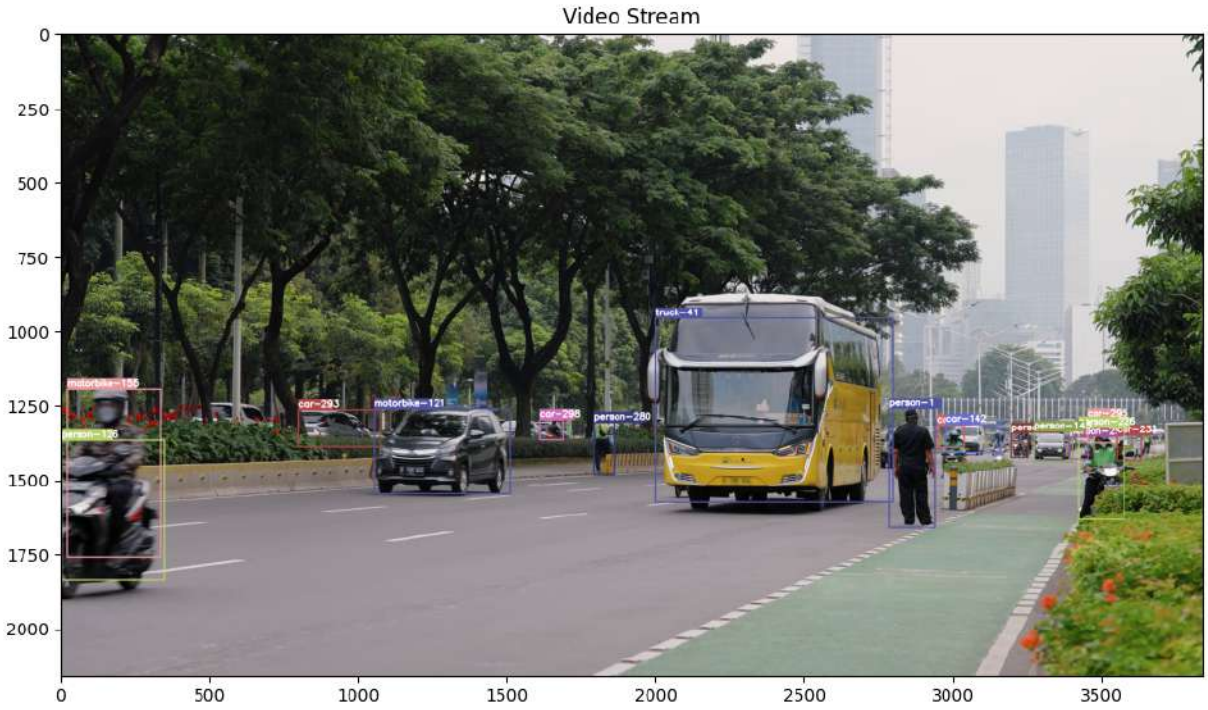

```
ii 125
torch.Size([17, 7])
(13, 6)
```



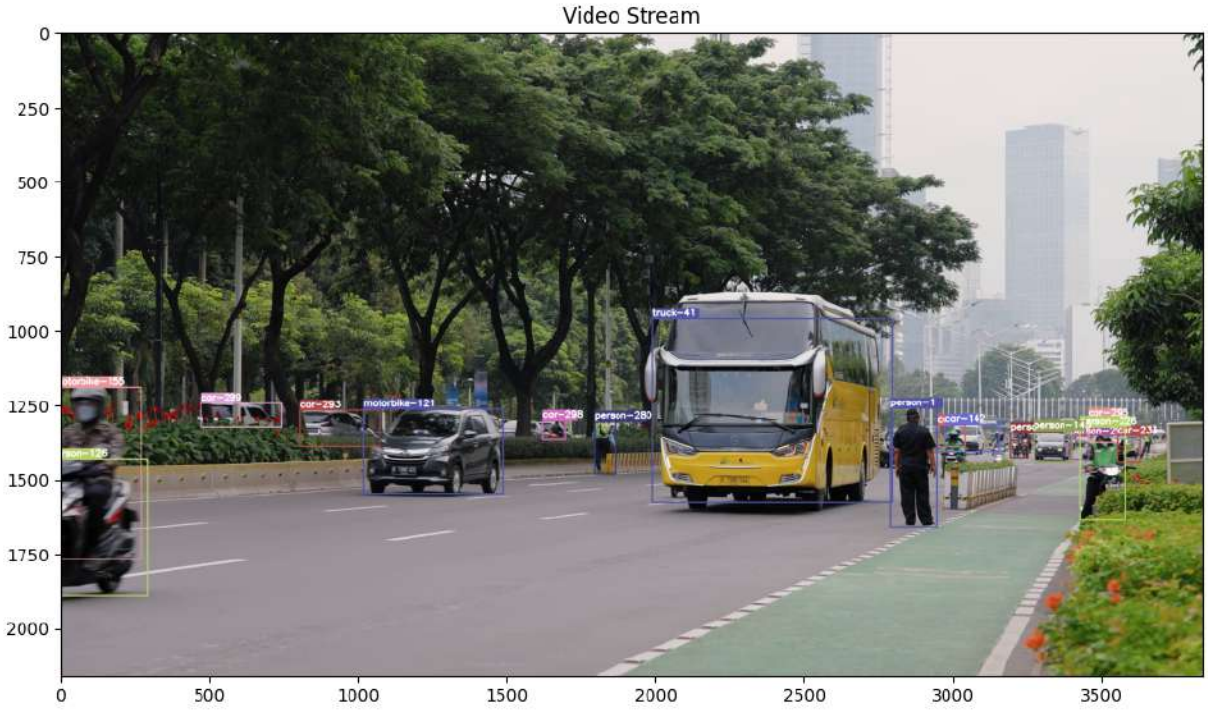

```
ii 126
torch.Size([21, 7])
(13, 6)
```



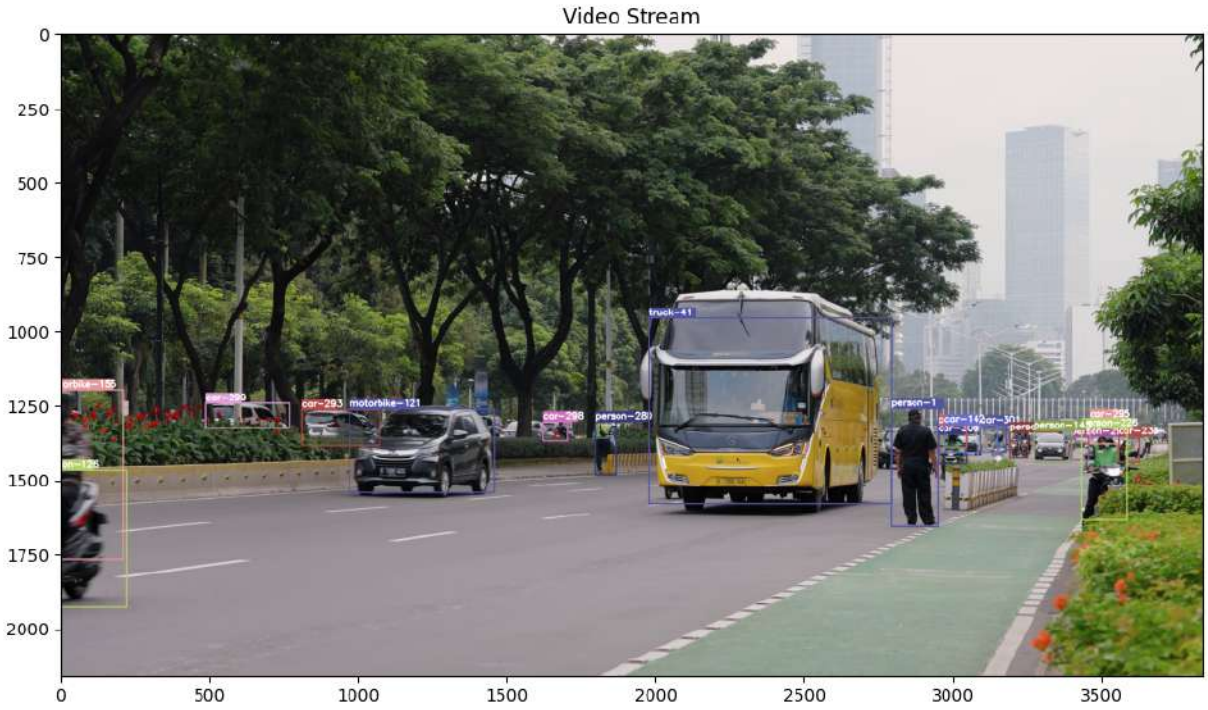

```
ii 127
torch.Size([22, 7])
(16, 6)
```



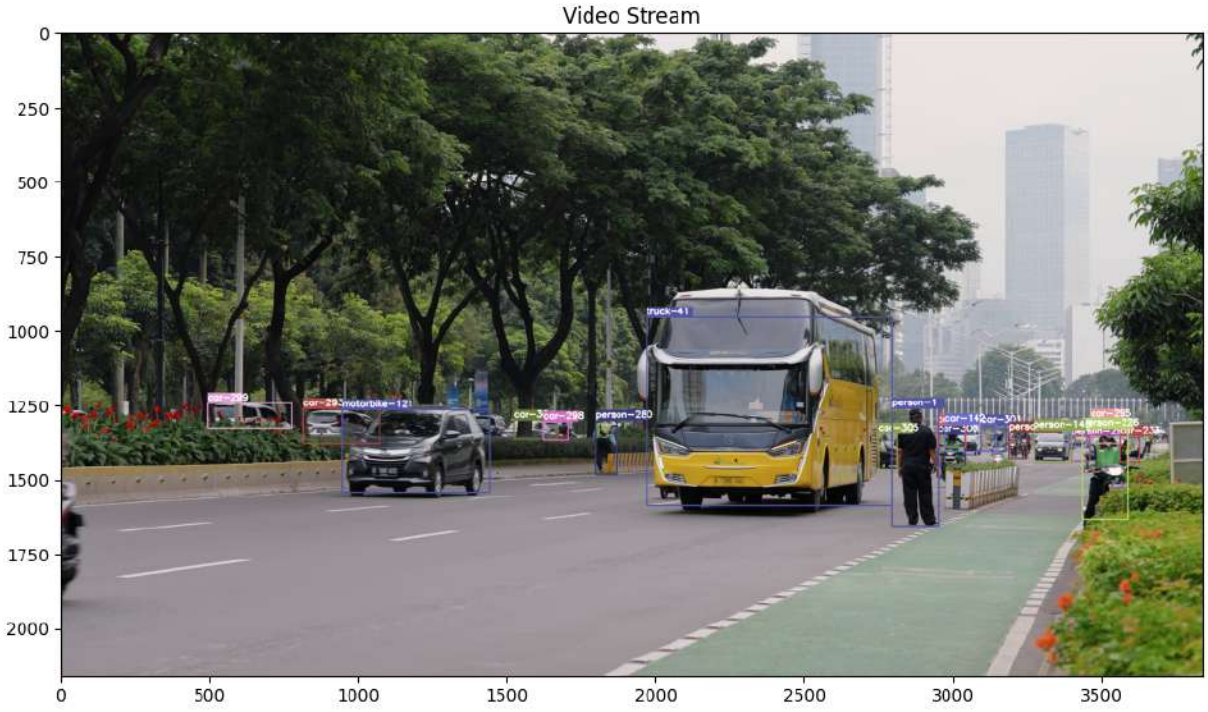

```
ii 128
torch.Size([27, 7])
(17, 6)
```



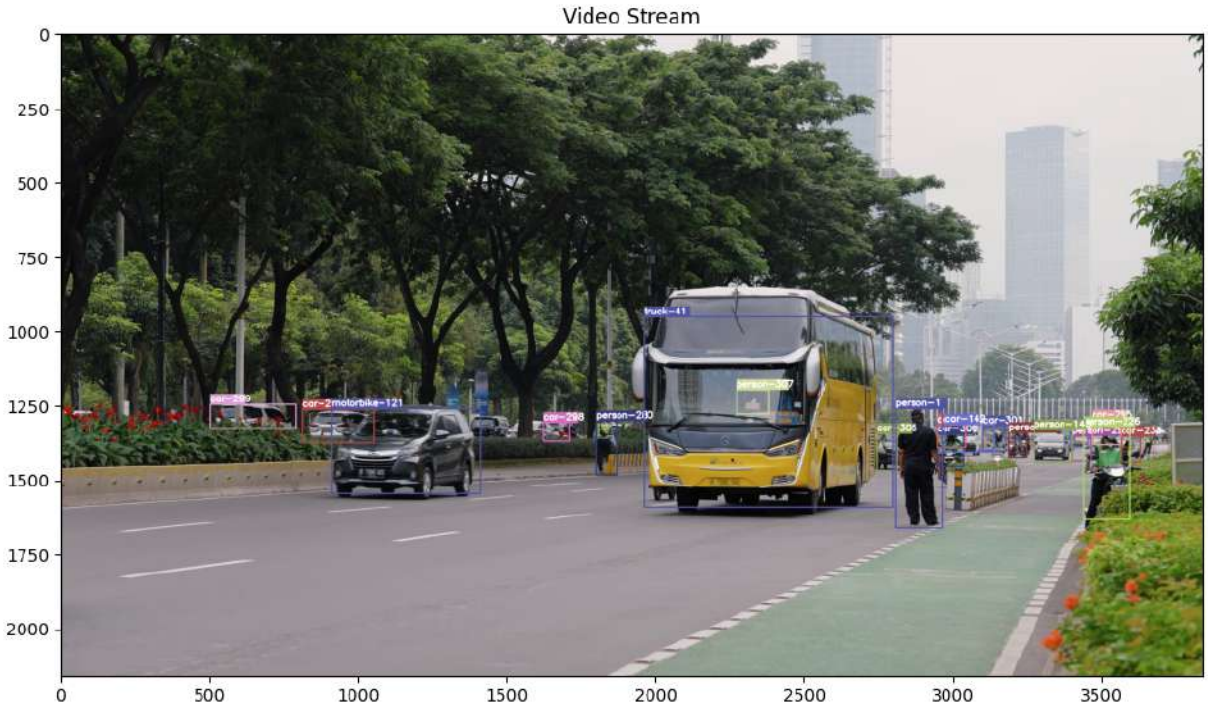

```
ii 129
torch.Size([25, 7])
(19, 6)
```



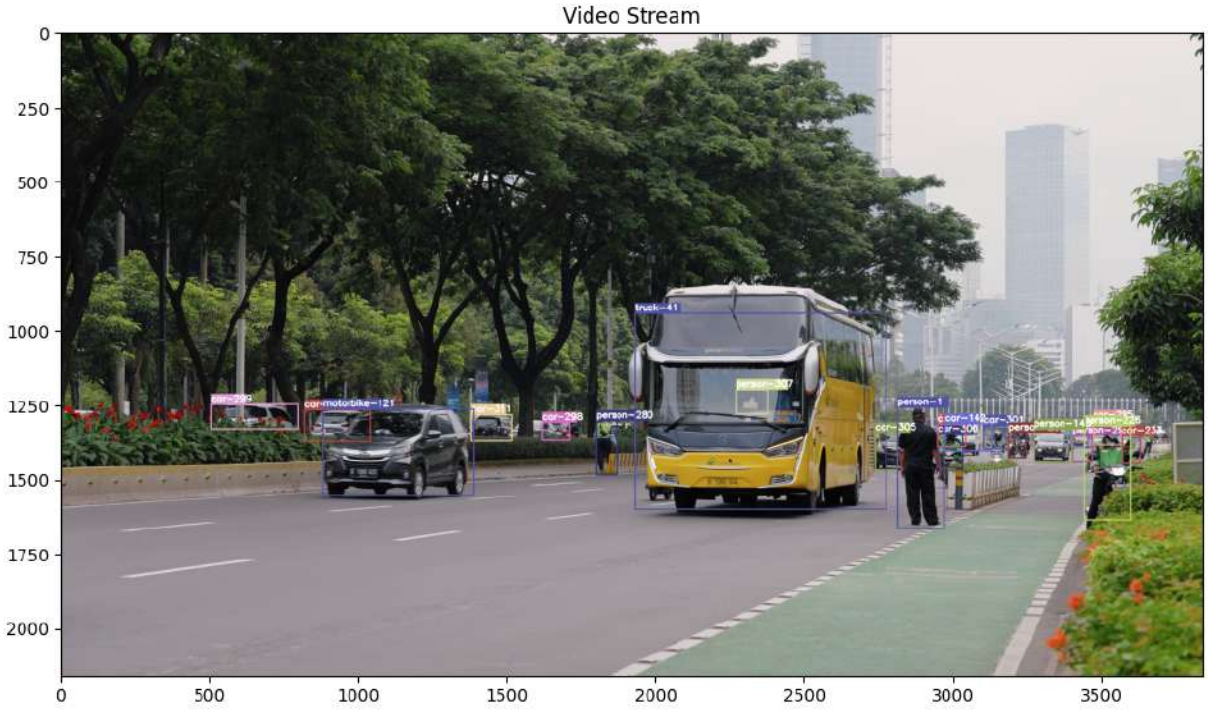

```
ii 130
torch.Size([24, 7])
(20, 6)
```



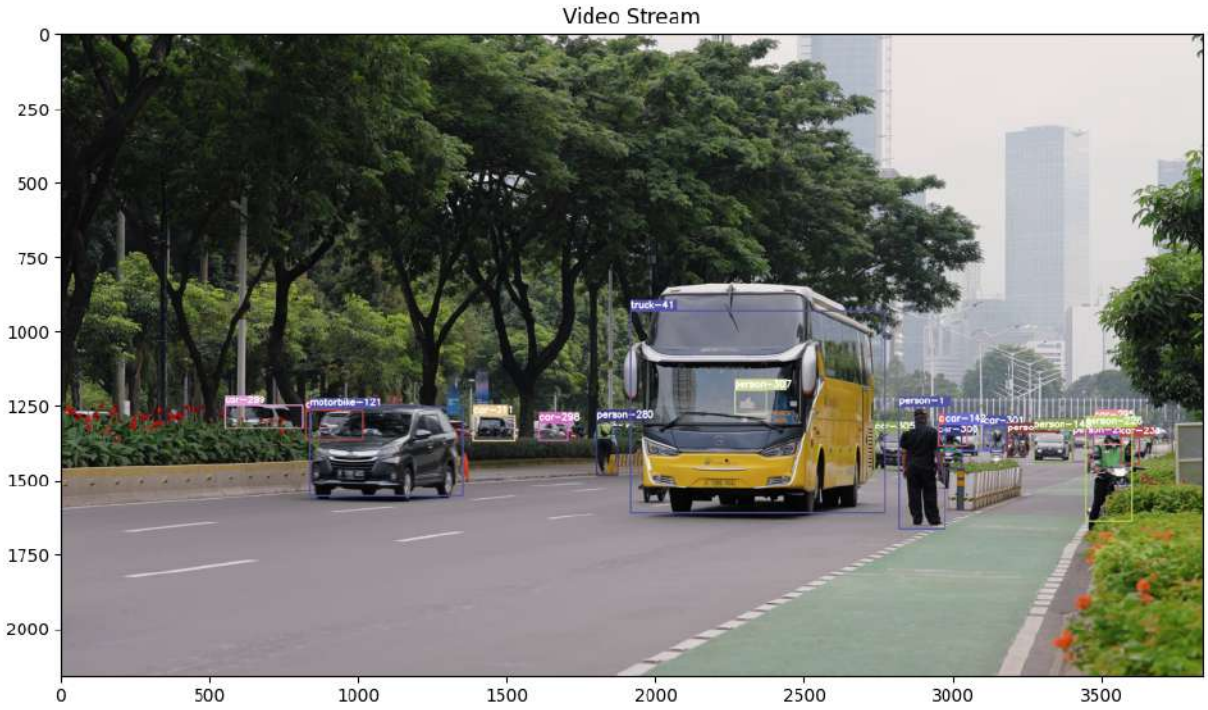

```
ii 131
torch.Size([25, 7])
(20, 6)
```




```
ii 132
torch.Size([25, 7])
(21, 6)
```

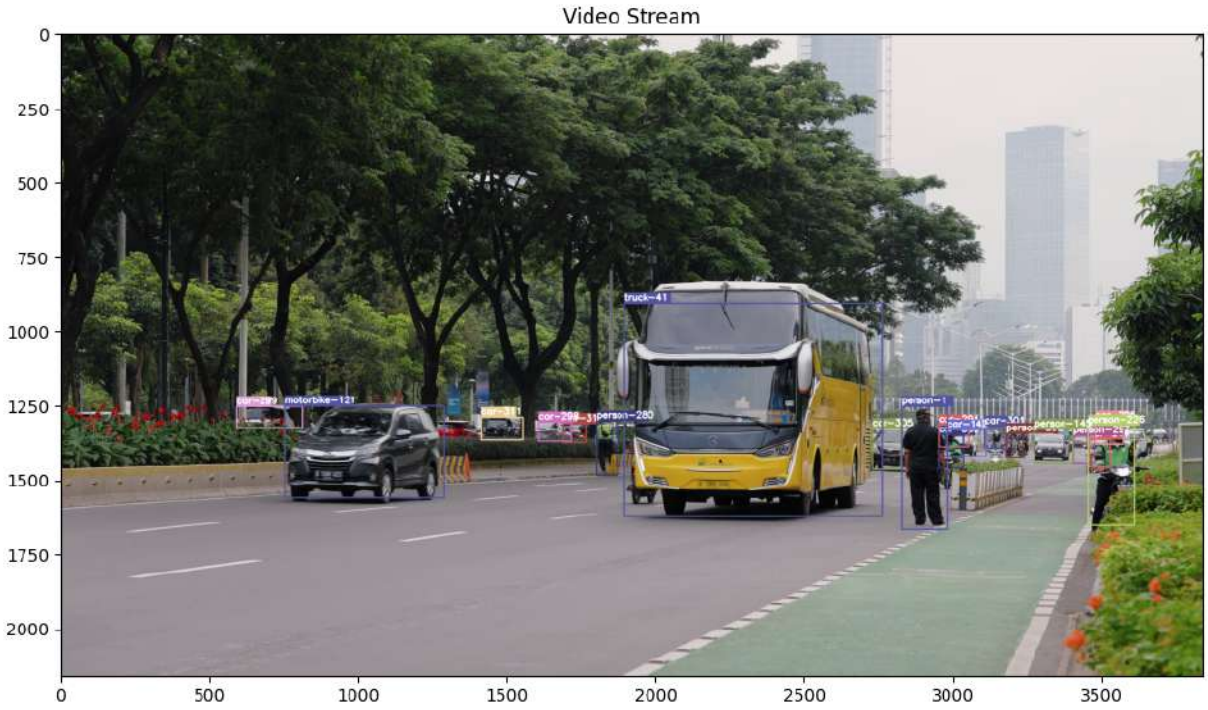



```
ii 133
torch.Size([23, 7])
(20, 6)
```

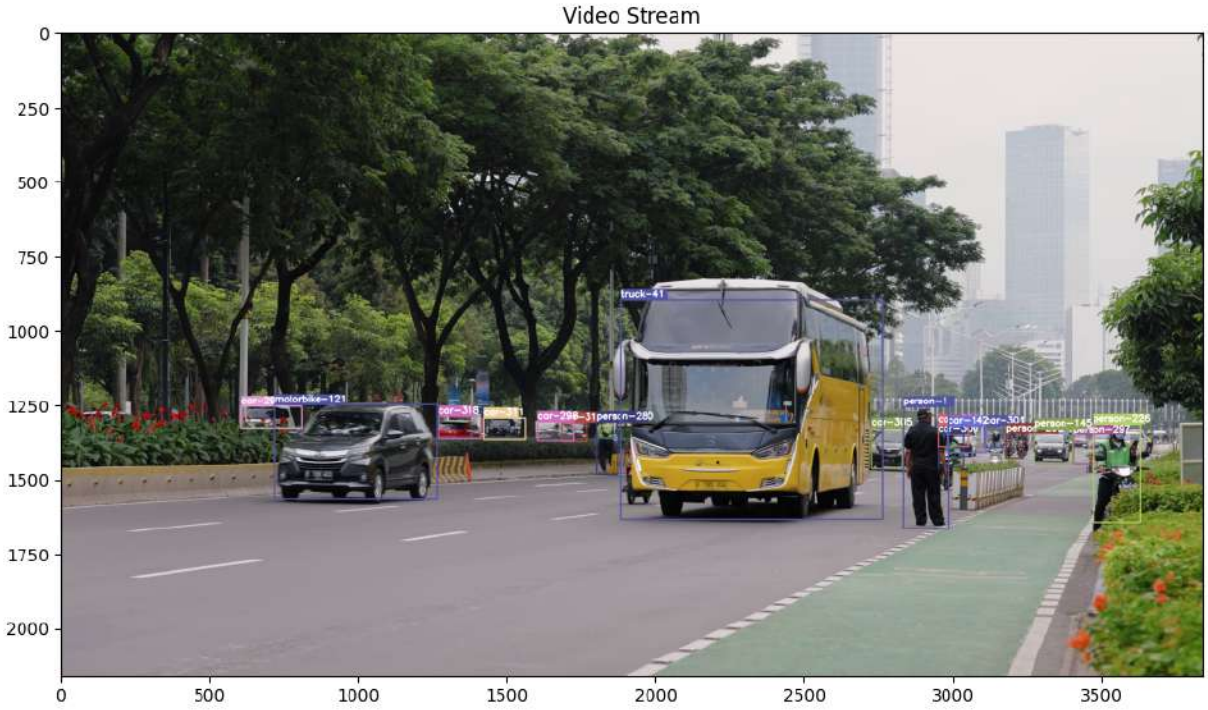



```
ii 134
torch.Size([22, 7])
(19, 6)
```

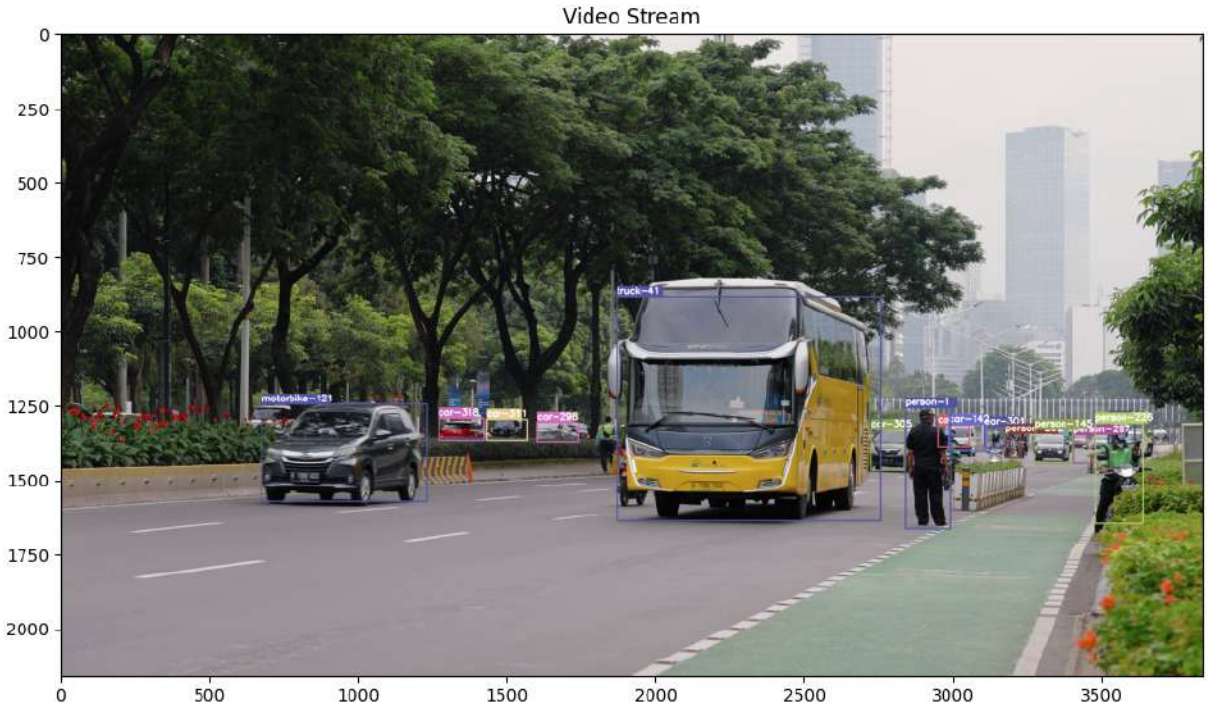

```
ii 135
torch.Size([23, 7])
(19, 6)
```



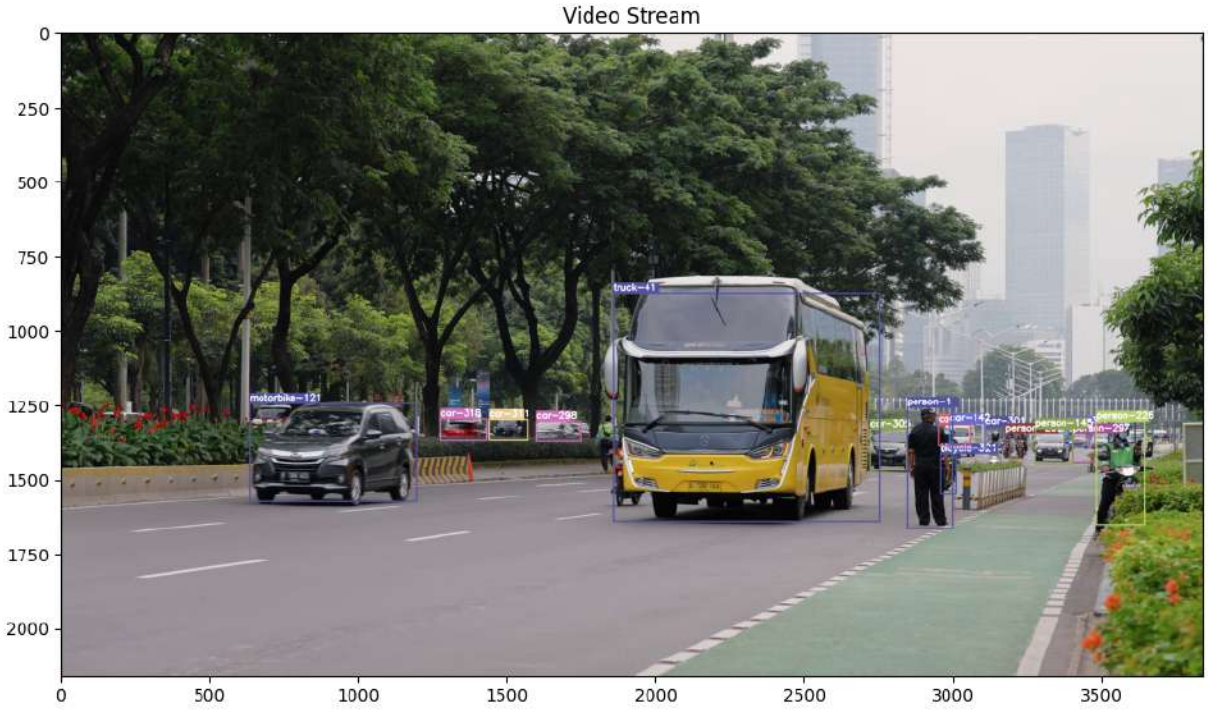

```
ii 136
torch.Size([22, 7])
(18, 6)
```



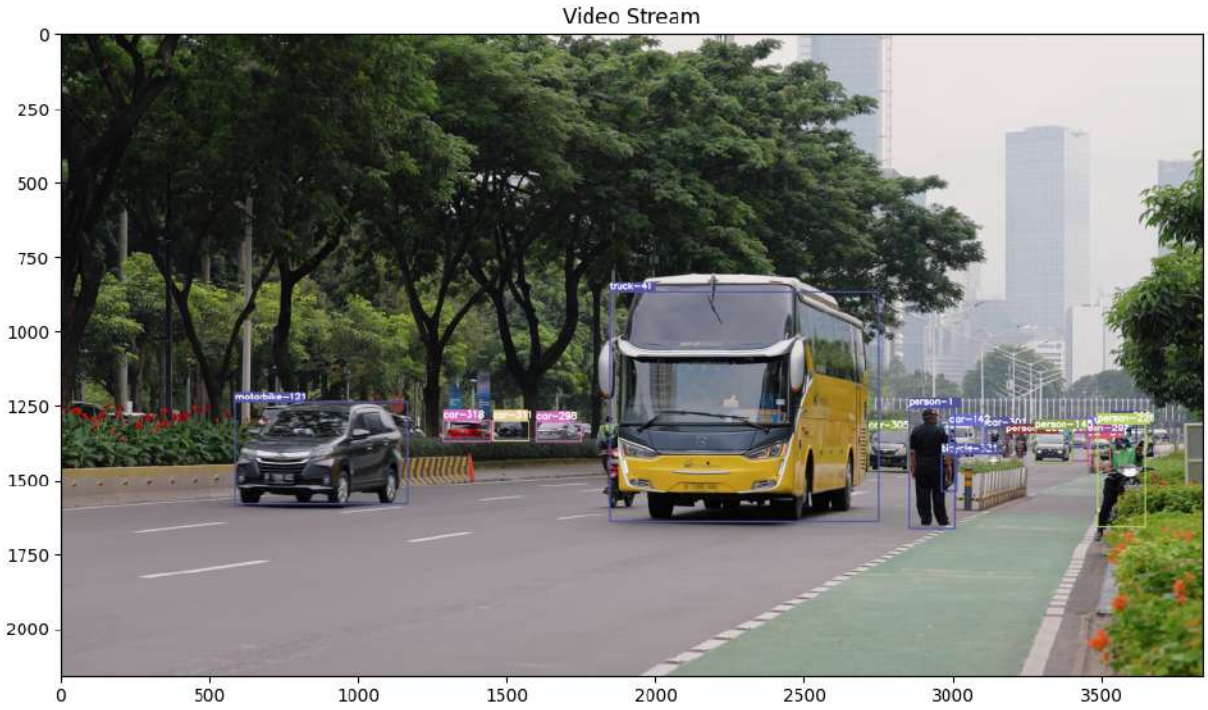

```
ii 137
torch.Size([21, 7])
(15, 6)
```



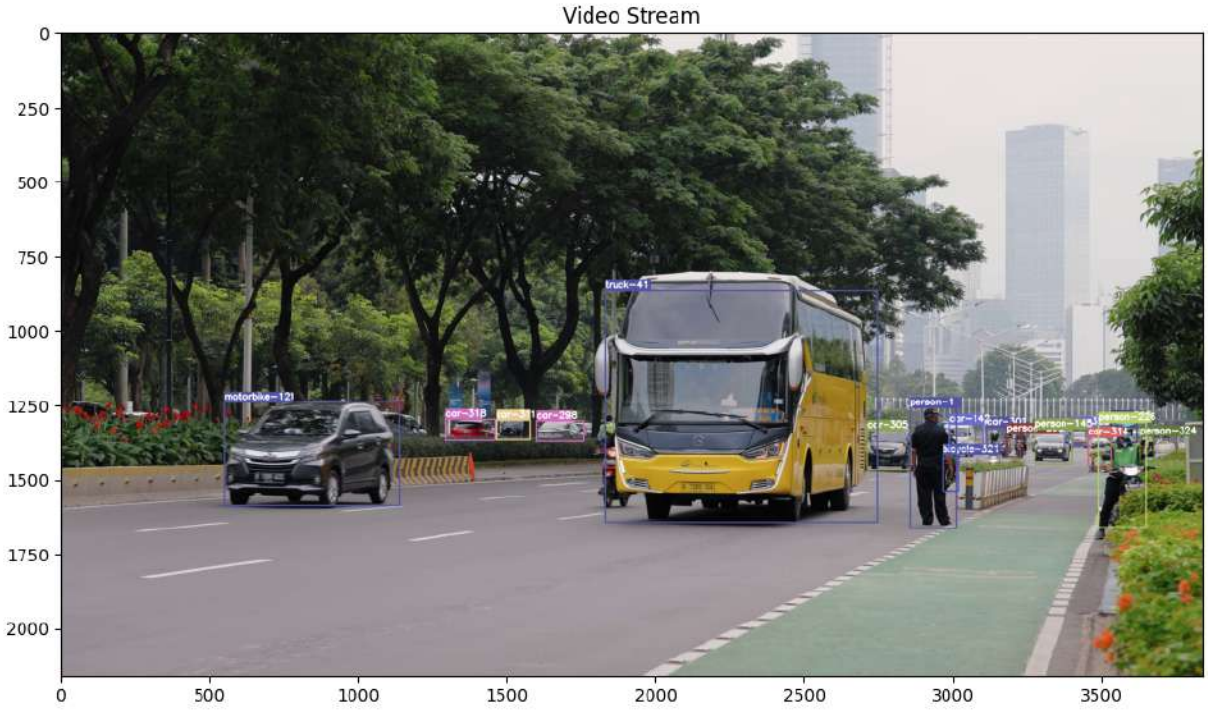

```
ii 138
torch.Size([20, 7])
(17, 6)
```



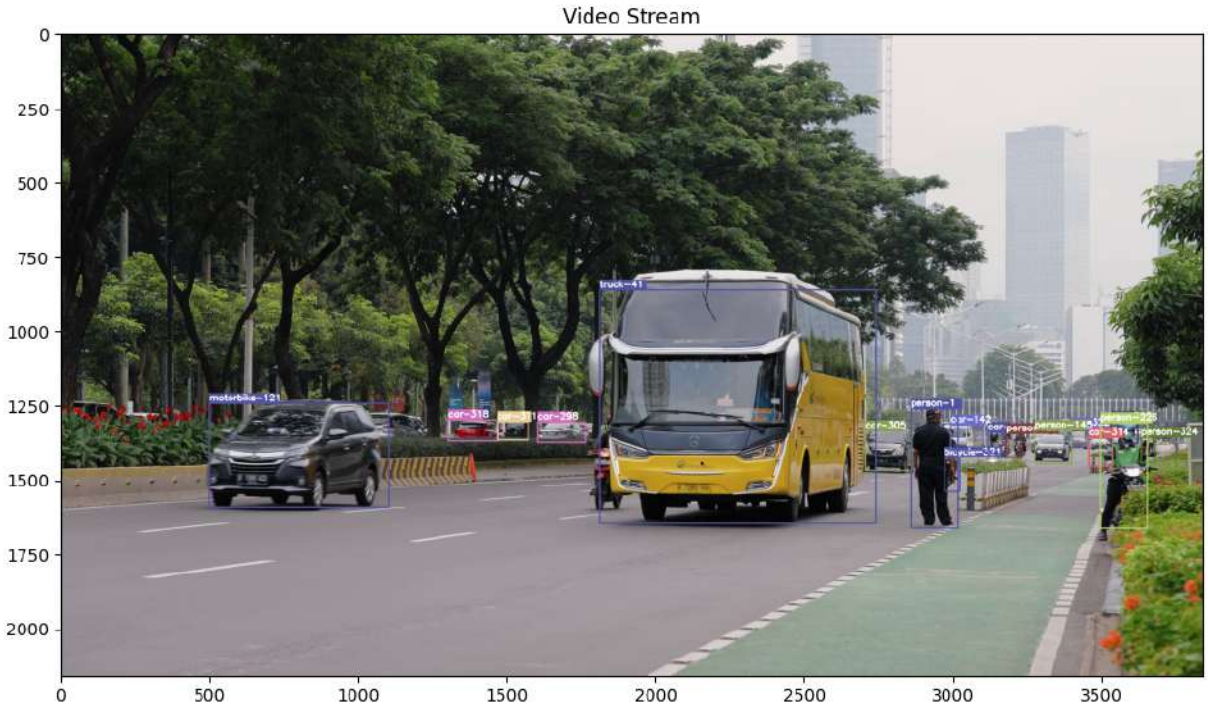

```
ii 139
torch.Size([20, 7])
(17, 6)
```



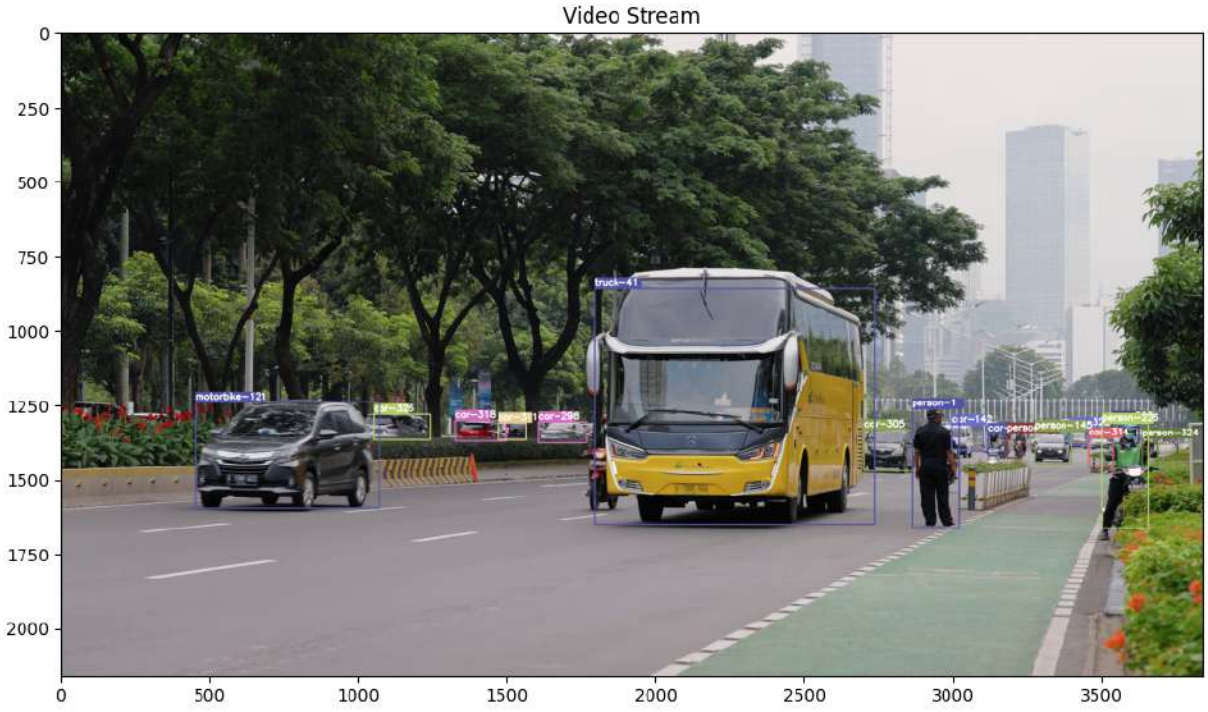

```
ii 140
torch.Size([19, 7])
(17, 6)
```




```
ii 141
torch.Size([21, 7])
(17, 6)
```

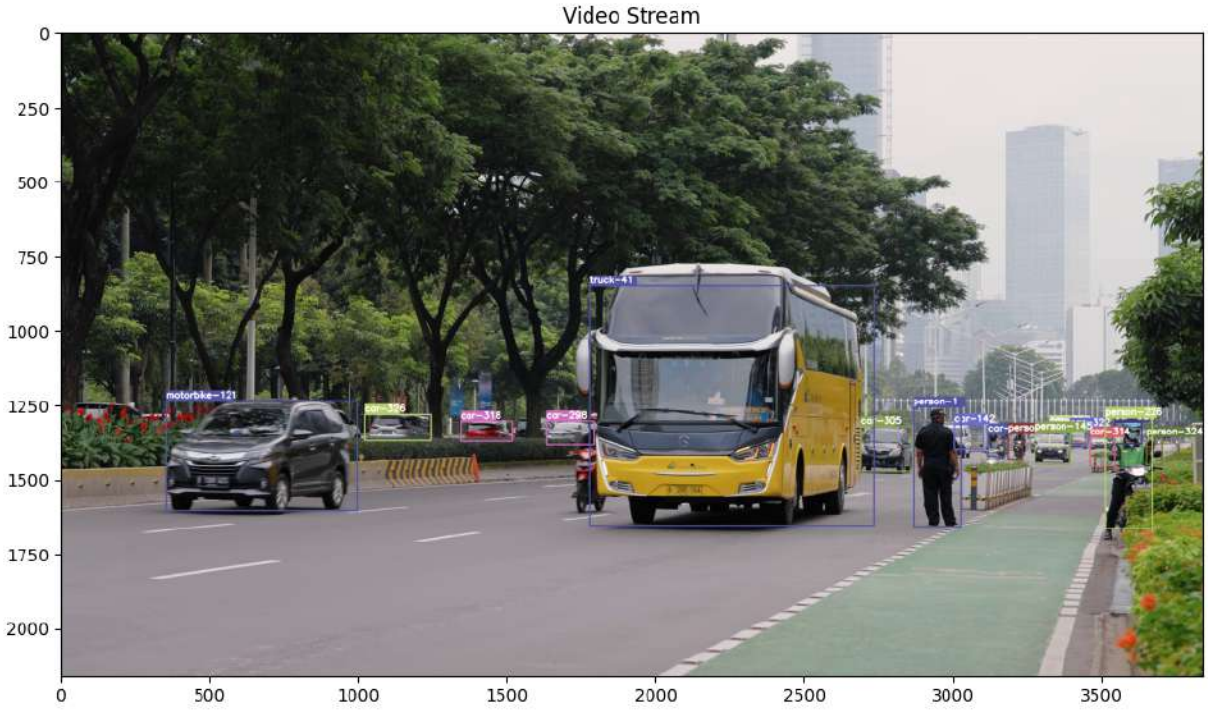



```
ii 142
torch.Size([20, 7])
(17, 6)
```



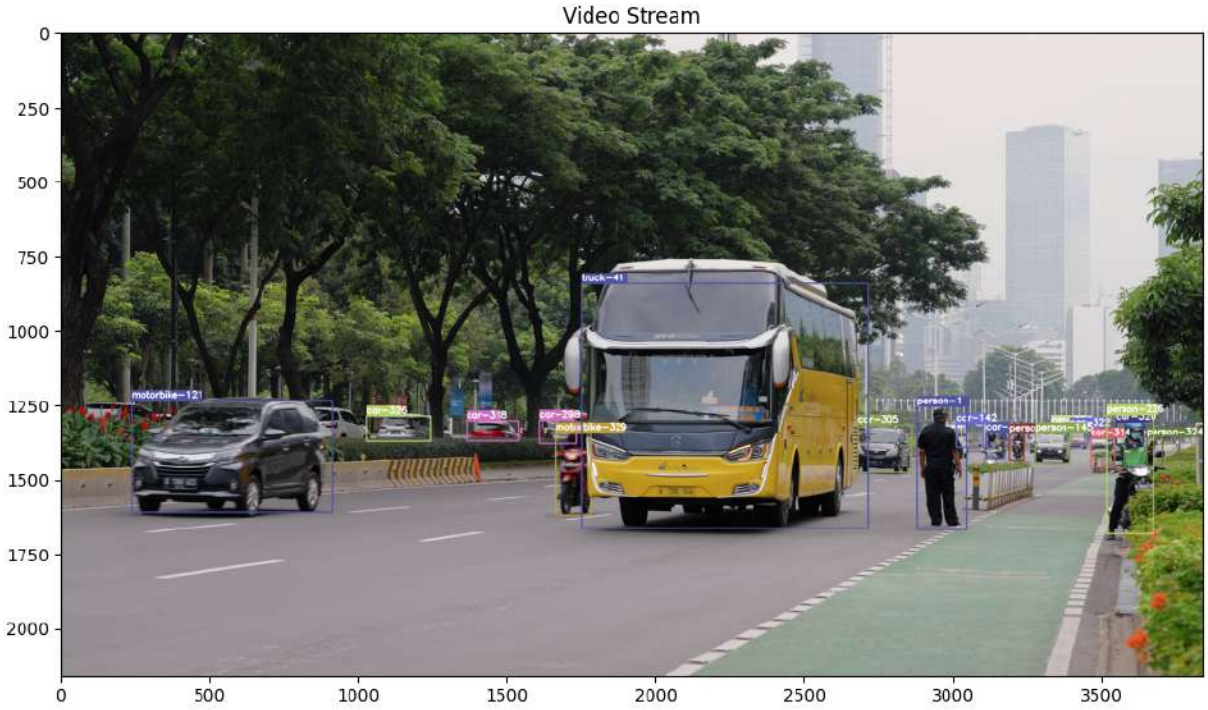

```
ii 143
torch.Size([18, 7])
(17, 6)
```


```
ii 144
torch.Size([19, 7])
(17, 6)
```

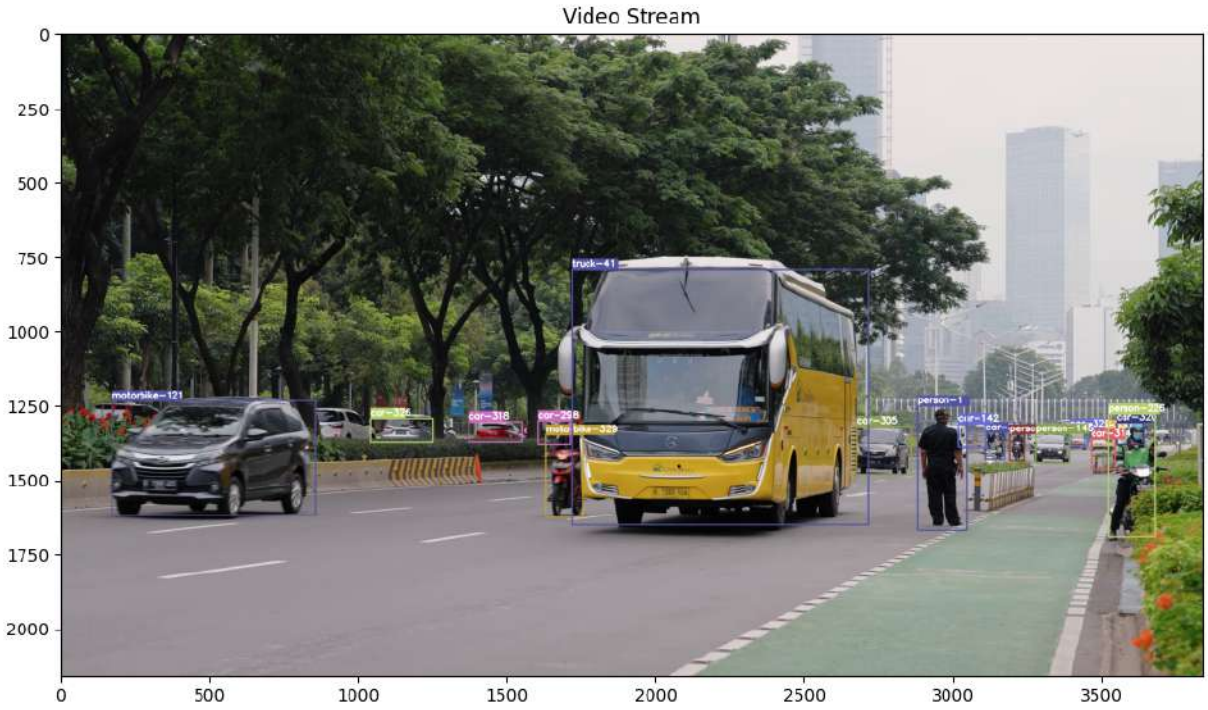



```
ii 145
torch.Size([22, 7])
(18, 6)
```

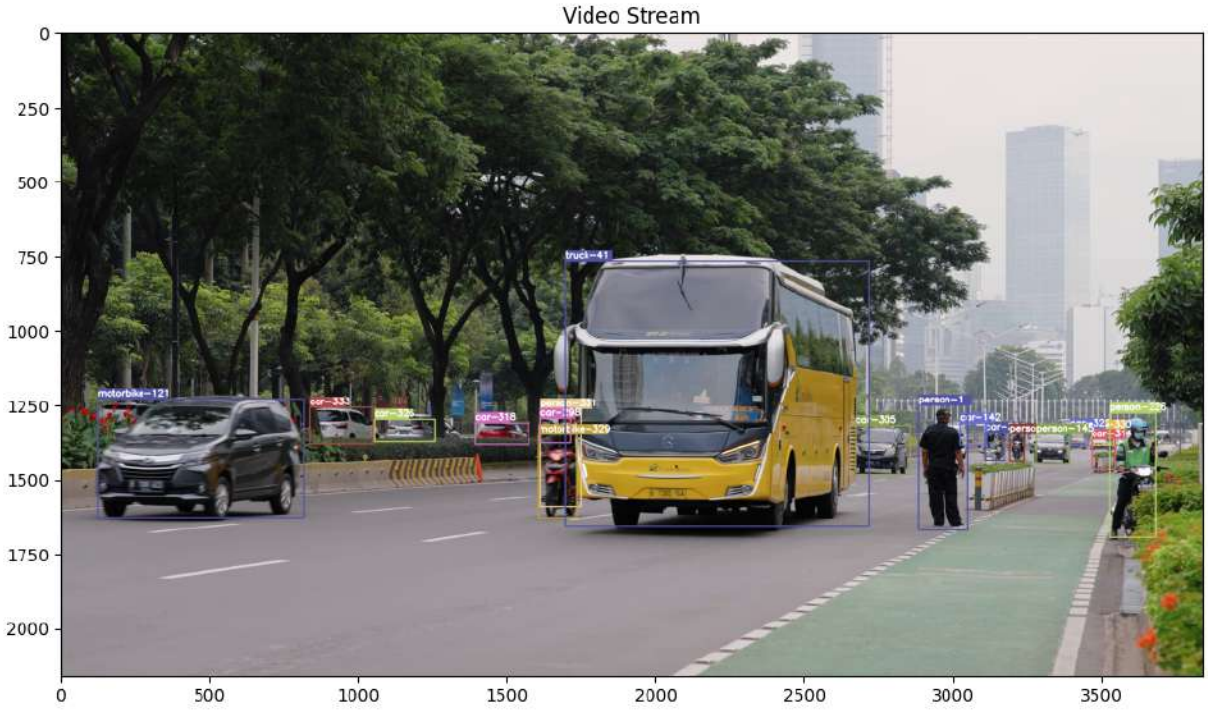

```
ii 146
torch.Size([23, 7])
(18, 6)
```




```
ii 147
torch.Size([20, 7])
(17, 6)
```




```
ii 148
torch.Size([19, 7])
(18, 6)
```

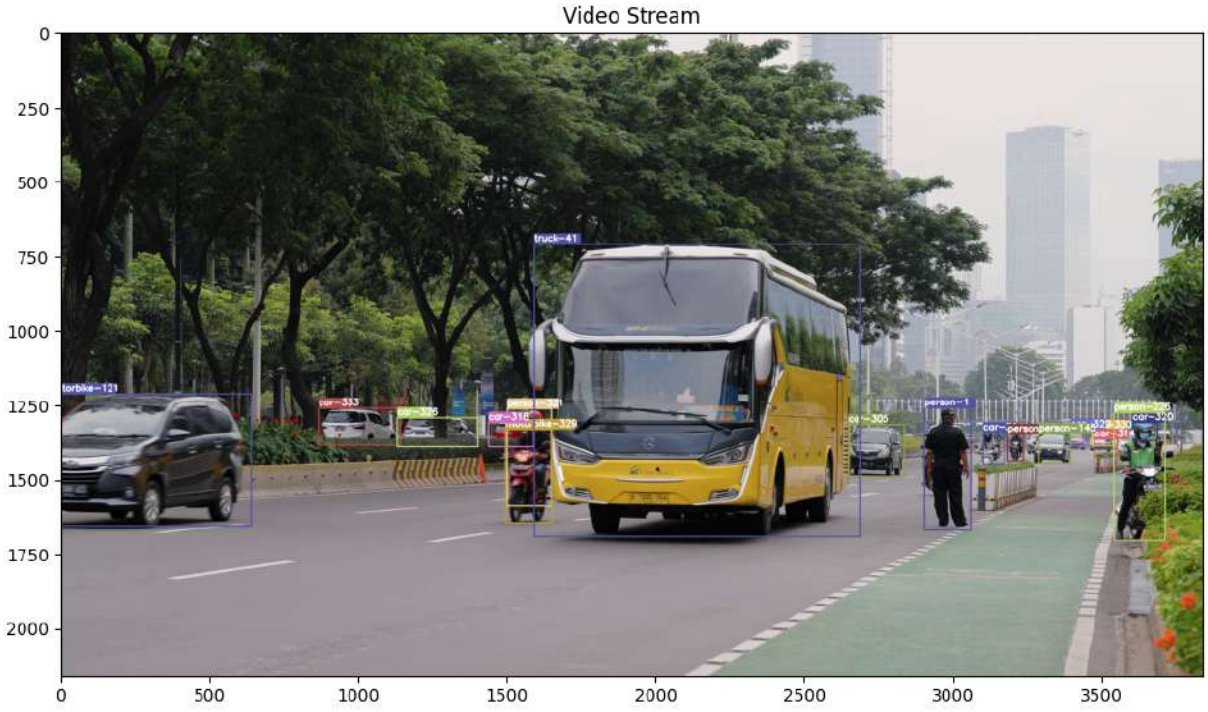



```
ii 149
torch.Size([18, 7])
(17, 6)
```

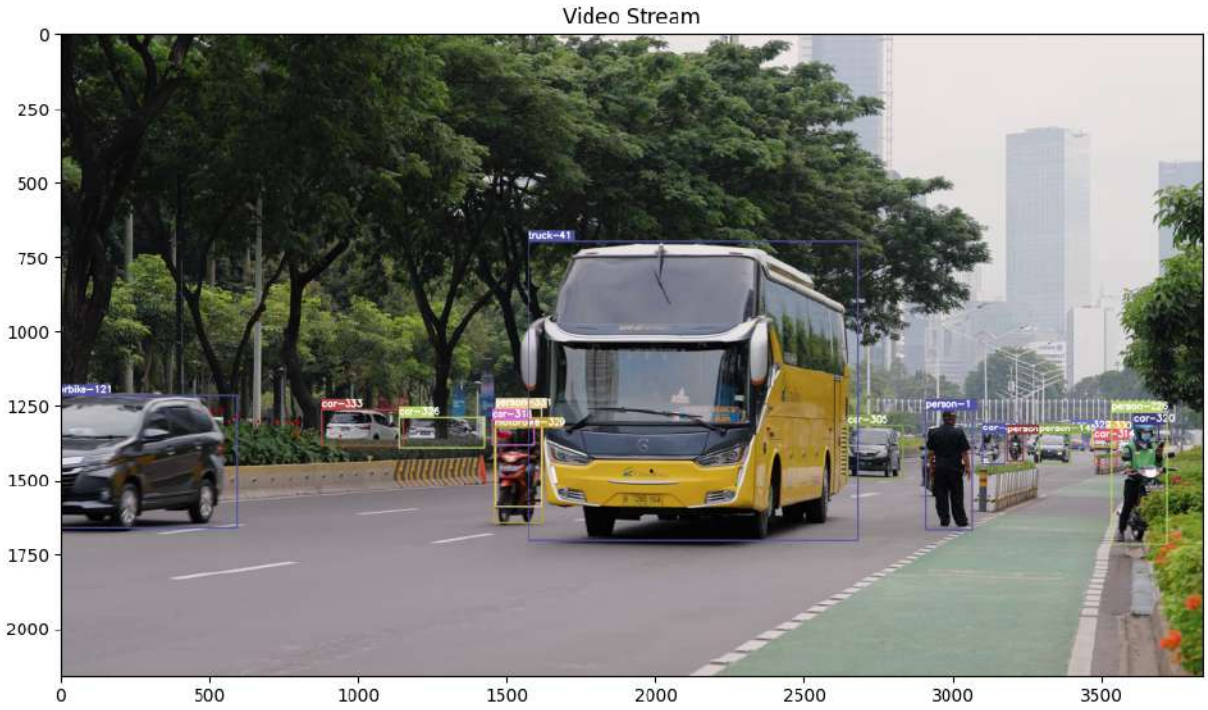

```
ii 150
torch.Size([18, 7])
(17, 6)
```


```
ii 151
torch.Size([20, 7])
(18, 6)
```

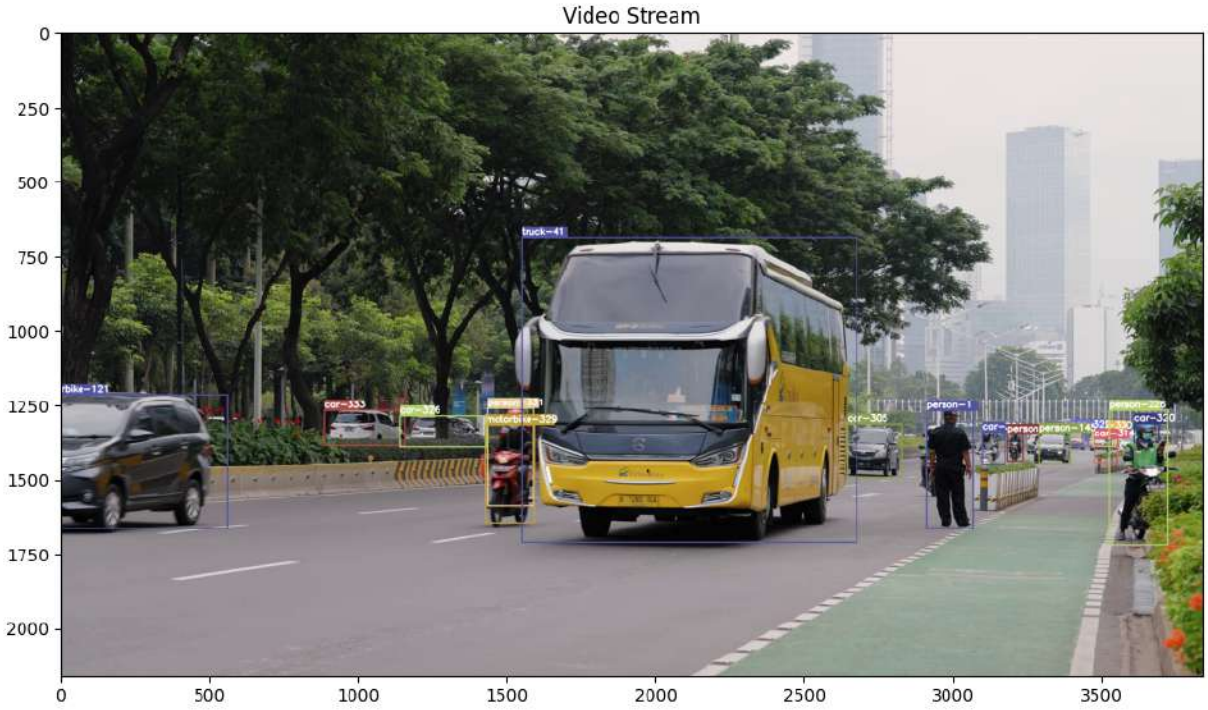

```
ii 152
torch.Size([20, 7])
(17, 6)
```



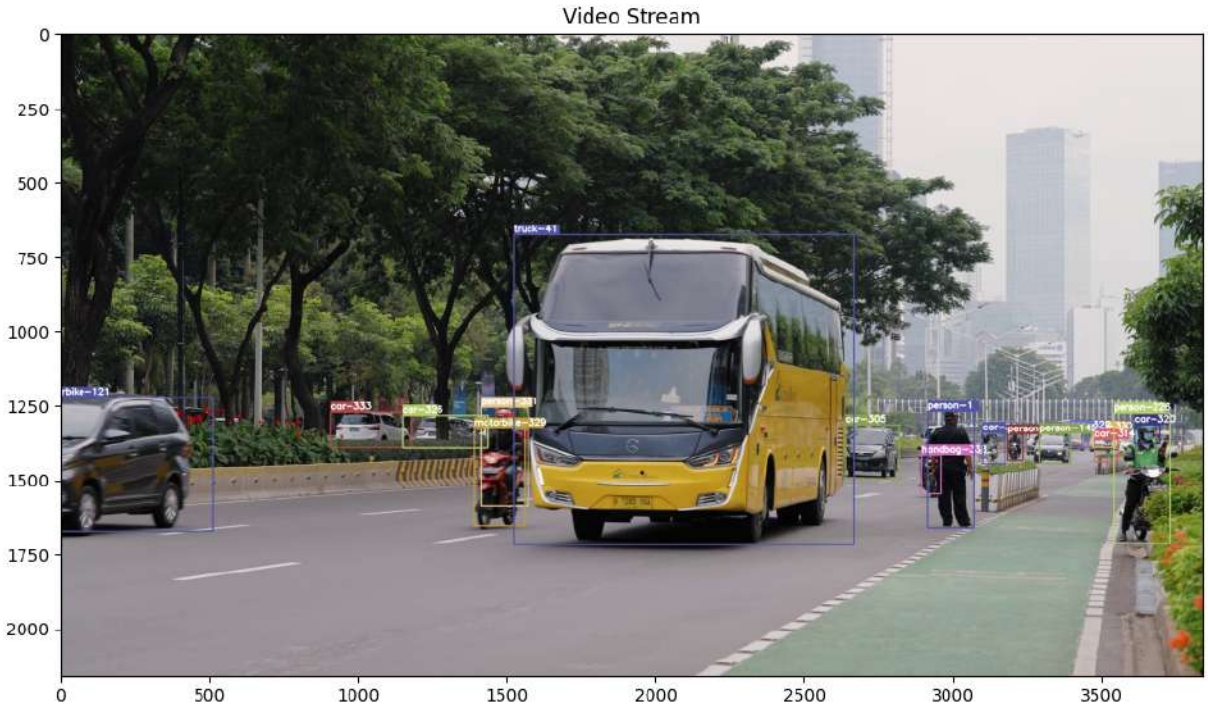

```
ii 153
torch.Size([19, 7])
(17, 6)
```



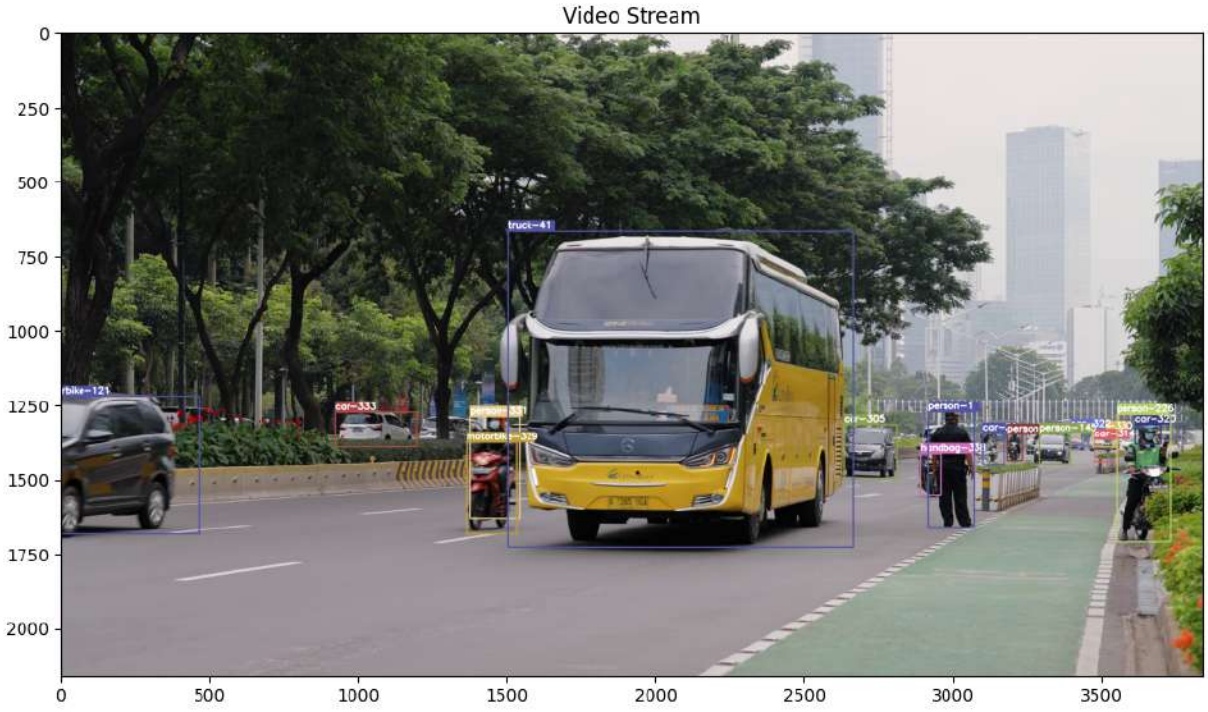

```
ii 154
torch.Size([17, 7])
(16, 6)
```




```
ii 155
torch.Size([19, 7])
(17, 6)
```



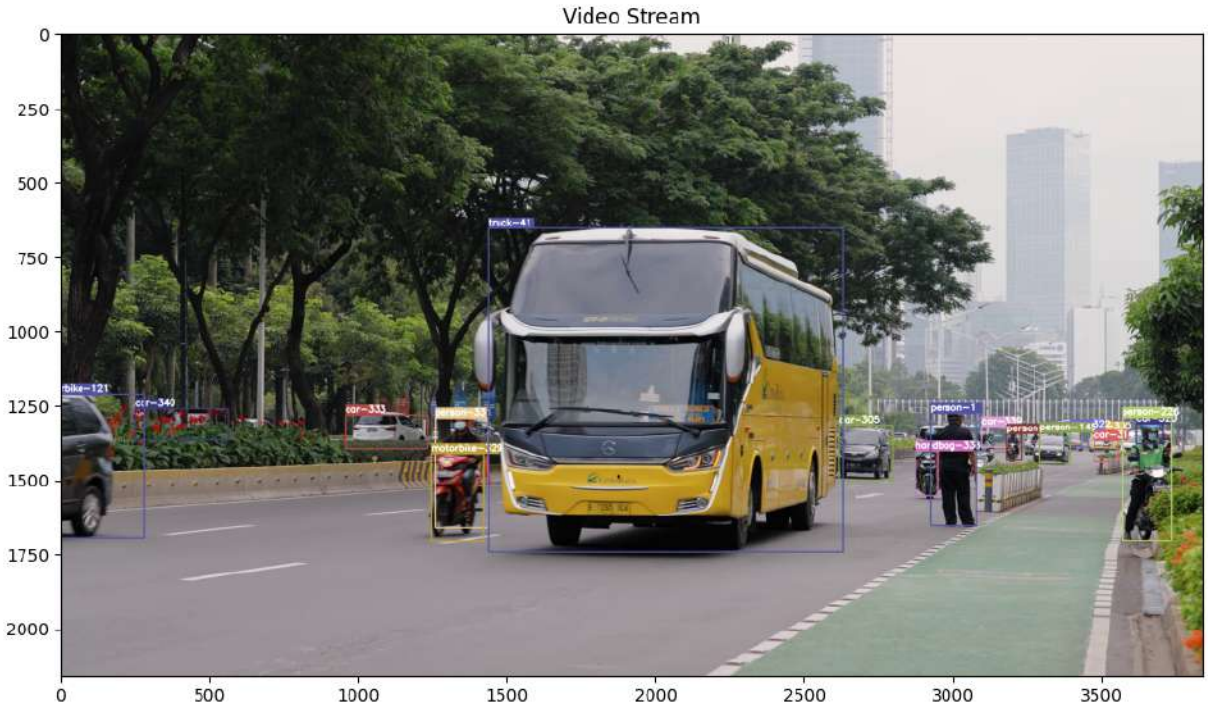

```
ii 156
torch.Size([19, 7])
(16, 6)
```



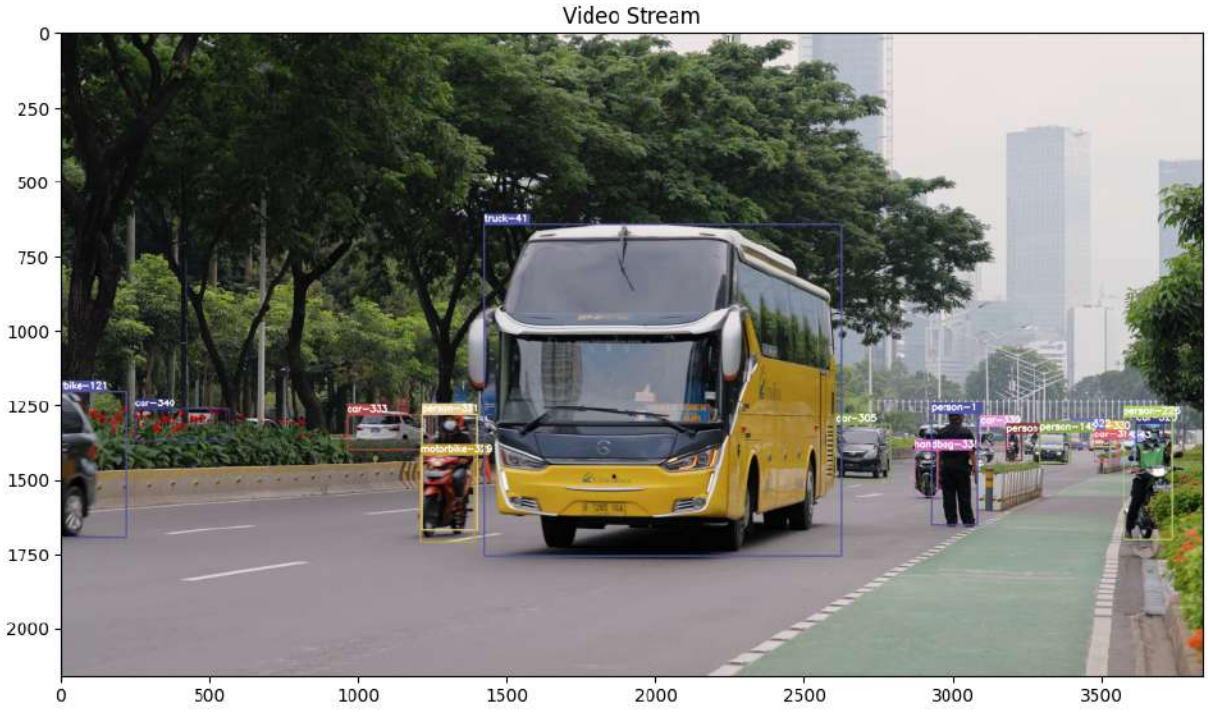

```
ii 157
torch.Size([21, 7])
(16, 6)
```


```
ii 158
torch.Size([19, 7])
(18, 6)
```

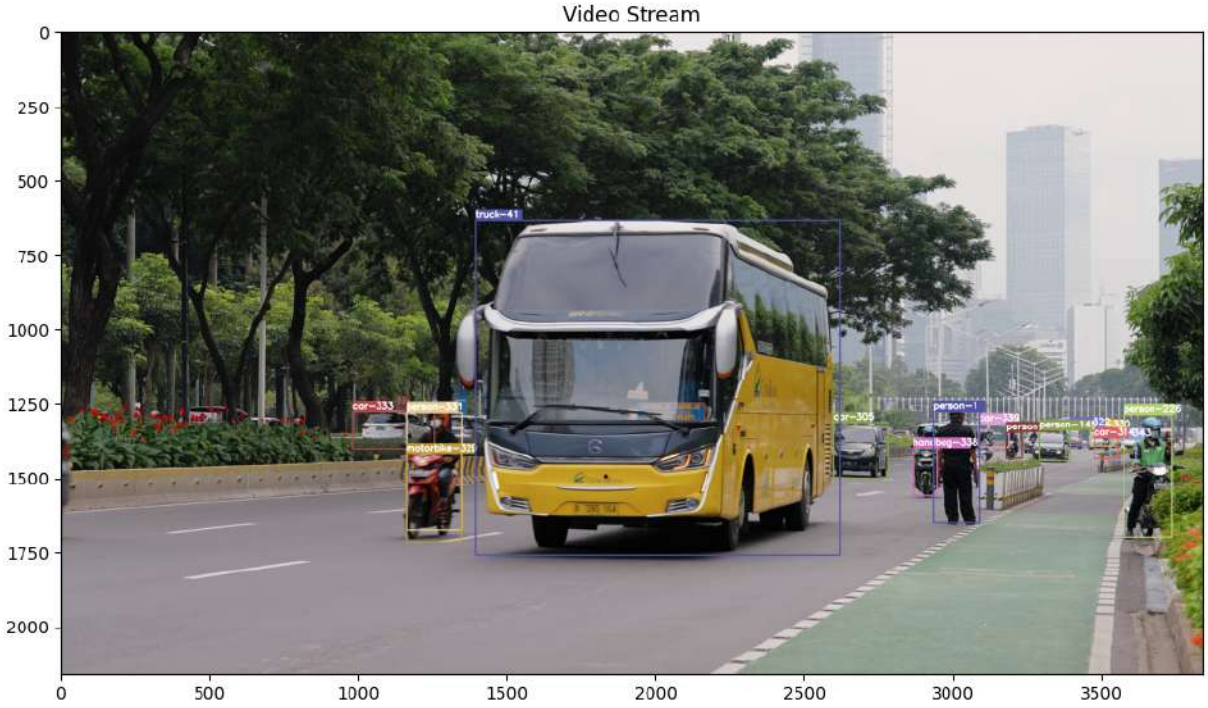

```
ii 159
torch.Size([19, 7])
(17, 6)
```



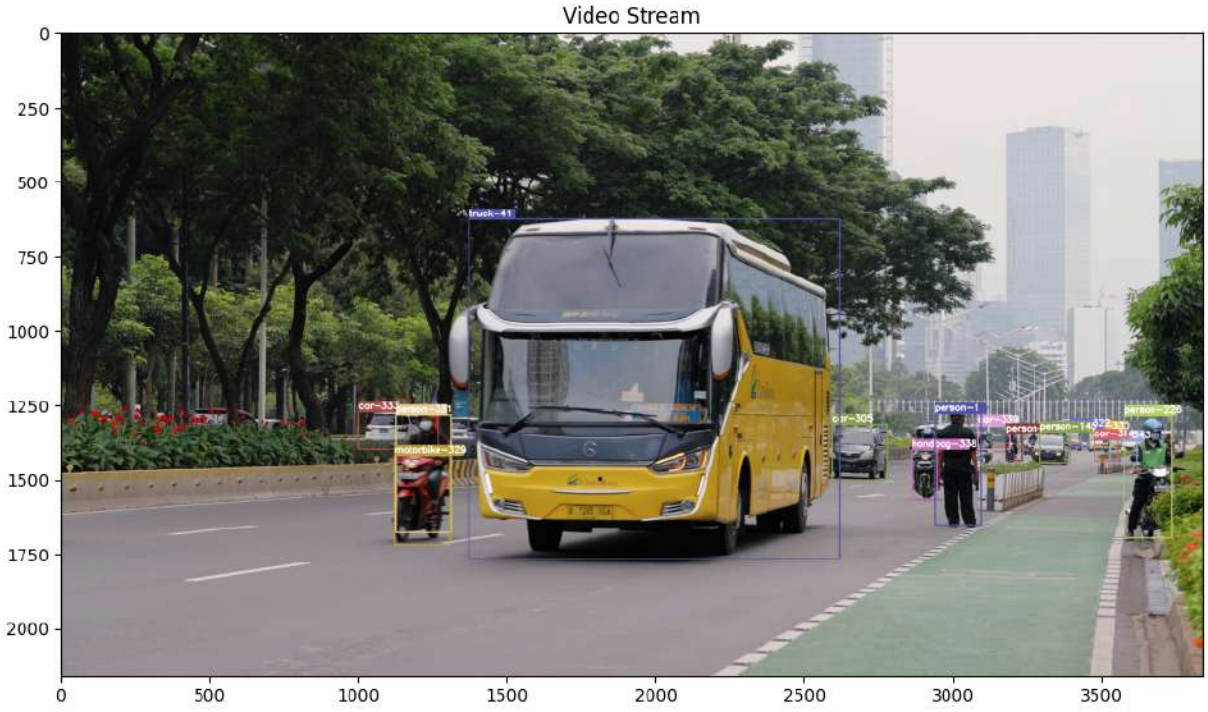

```
ii 160
torch.Size([22, 7])
(18, 6)
```



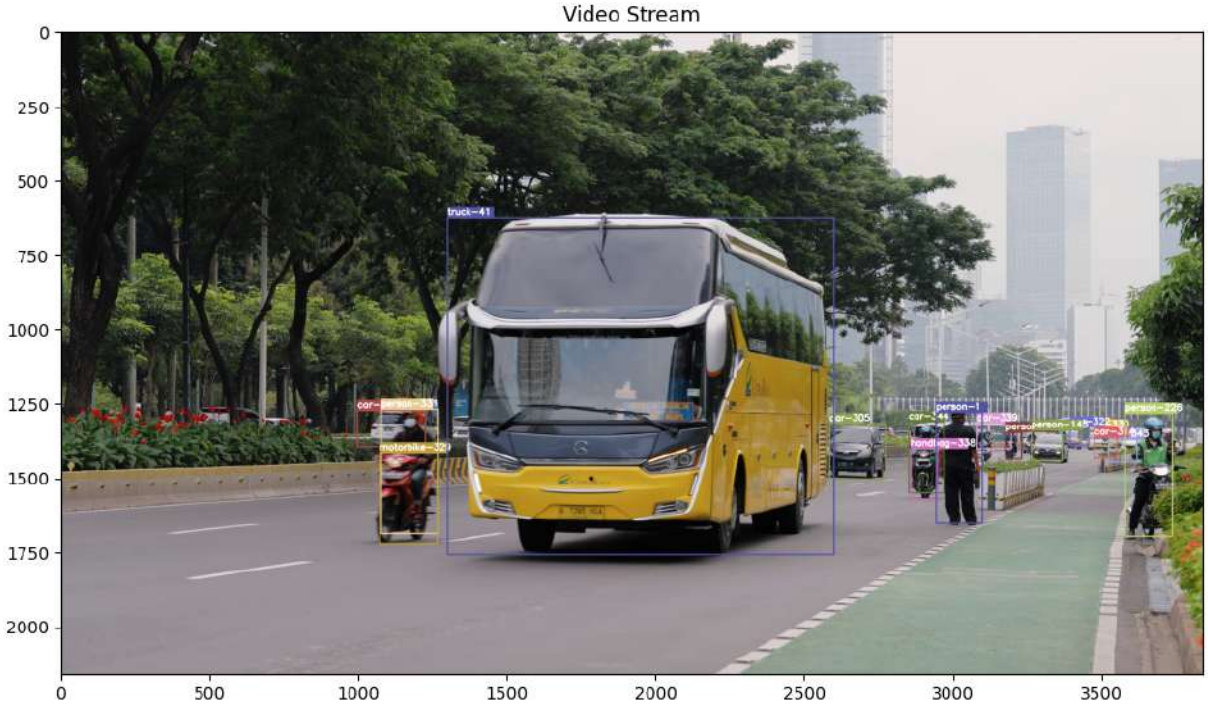

```
ii 161
torch.Size([18, 7])
(15, 6)
```



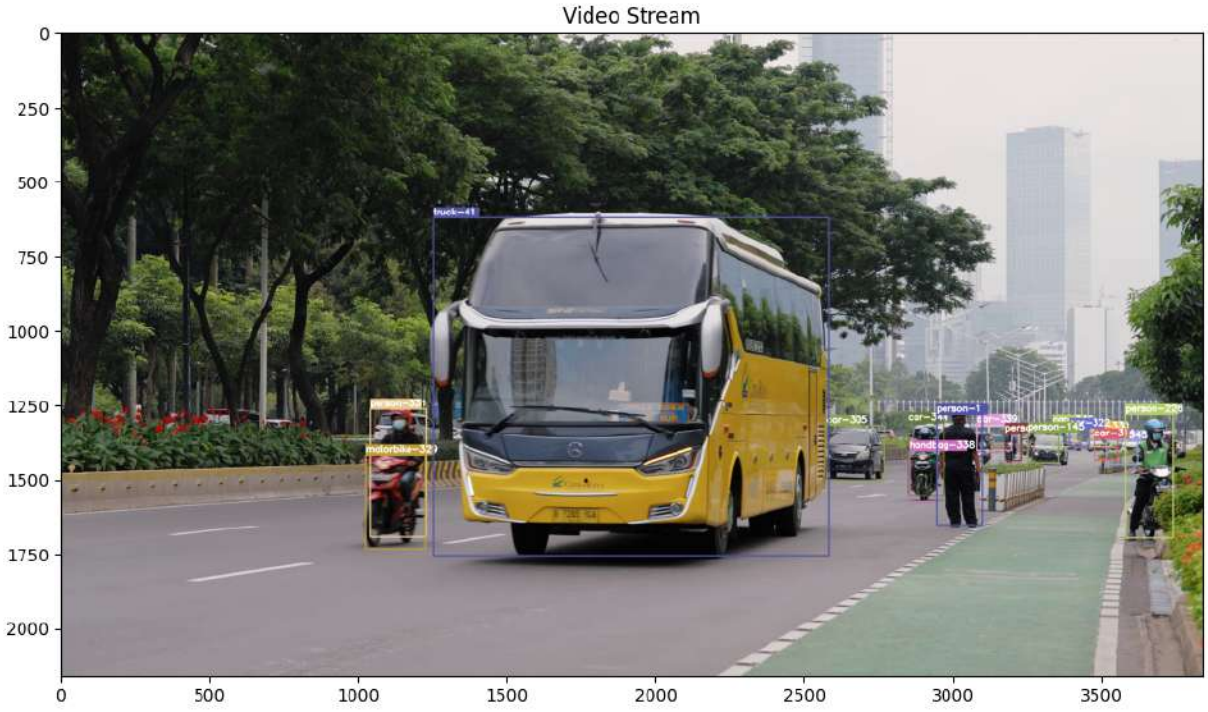

```
ii 162
torch.Size([17, 7])
(15, 6)
```



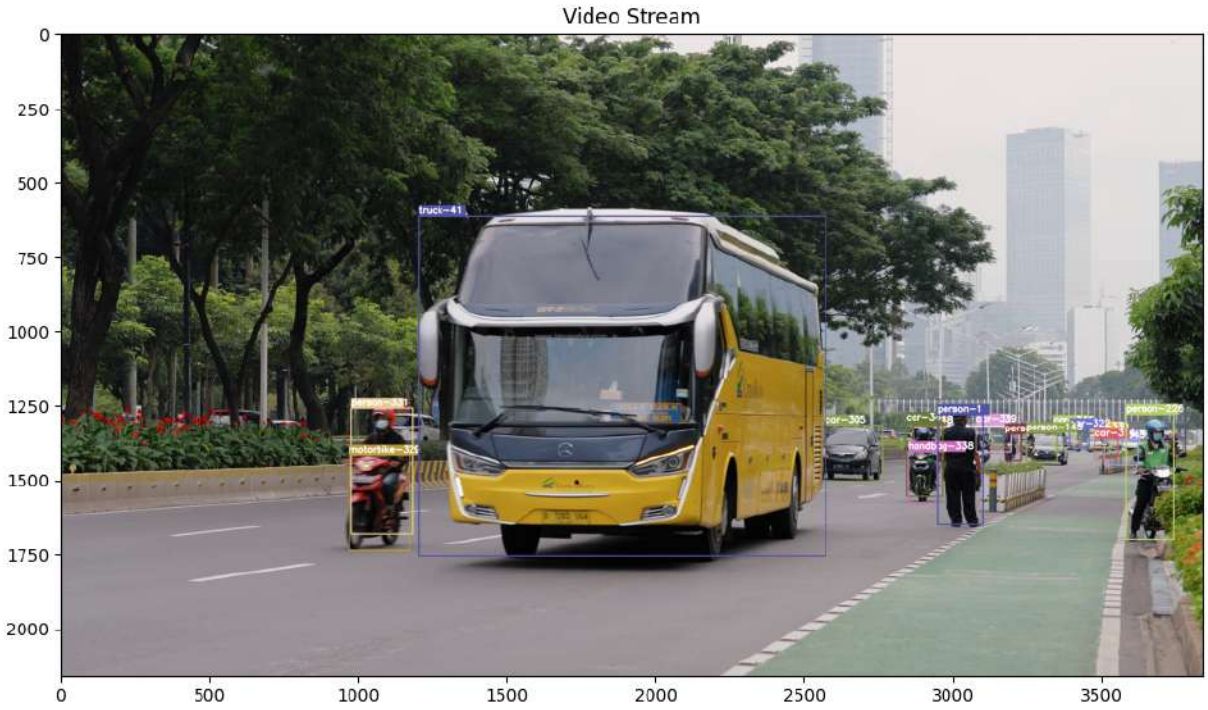

```
ii 163
torch.Size([20, 7])
(16, 6)
```



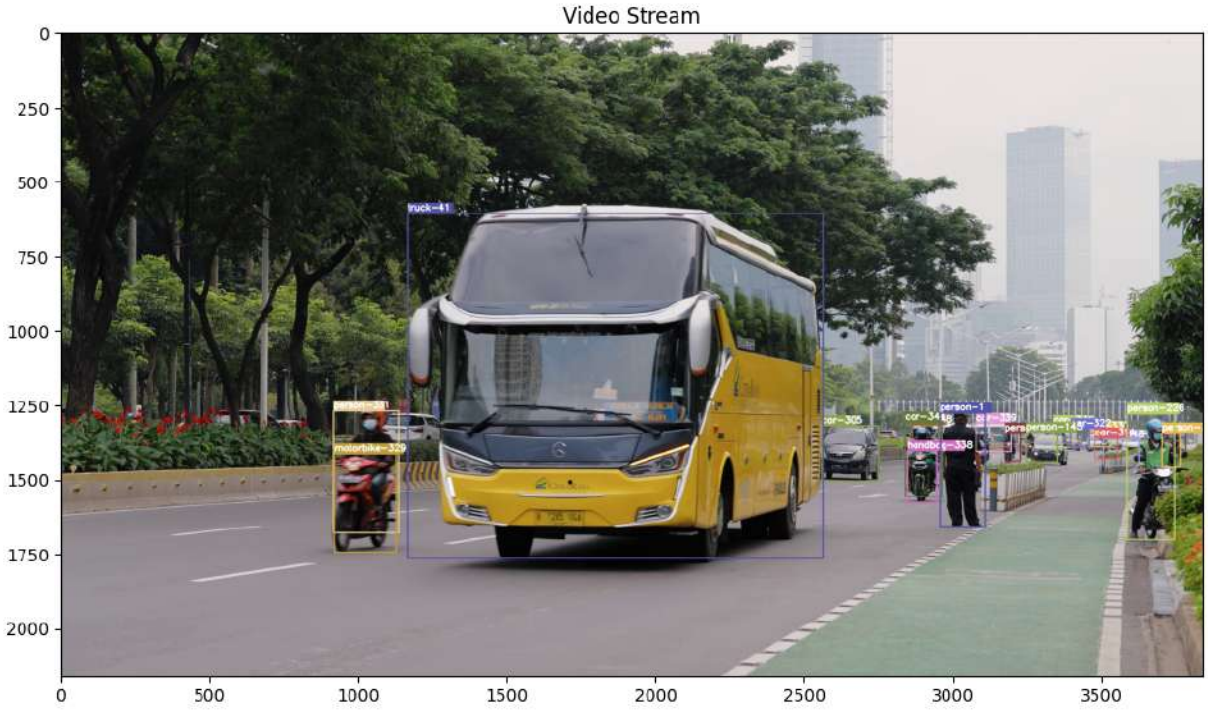

```
ii 164
torch.Size([20, 7])
(16, 6)
```




```
ii 165
torch.Size([22, 7])
(17, 6)
```




```
ii 166
torch.Size([21, 7])
(18, 6)
```




```
ii 167
torch.Size([18, 7])
(17, 6)
```

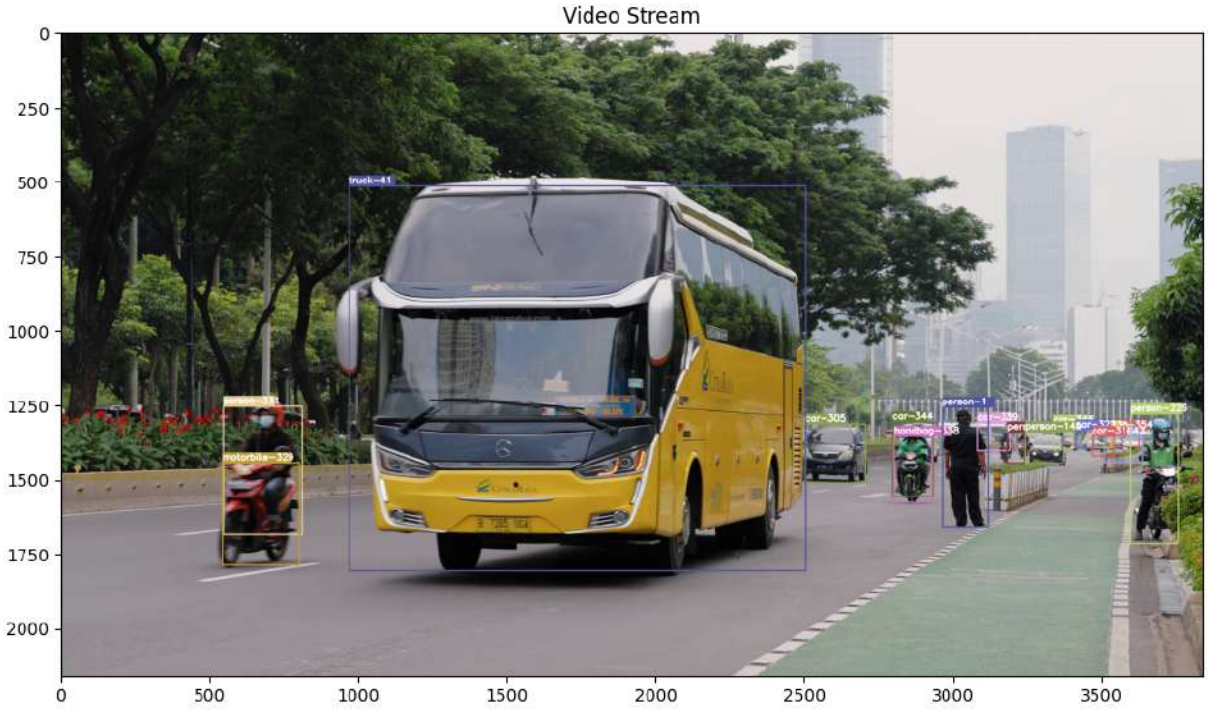

```
ii 168
torch.Size([19, 7])
(17, 6)
```


```
ii 169
torch.Size([18, 7])
(17, 6)
```


```
ii 170
torch.Size([19, 7])
(16, 6)
```

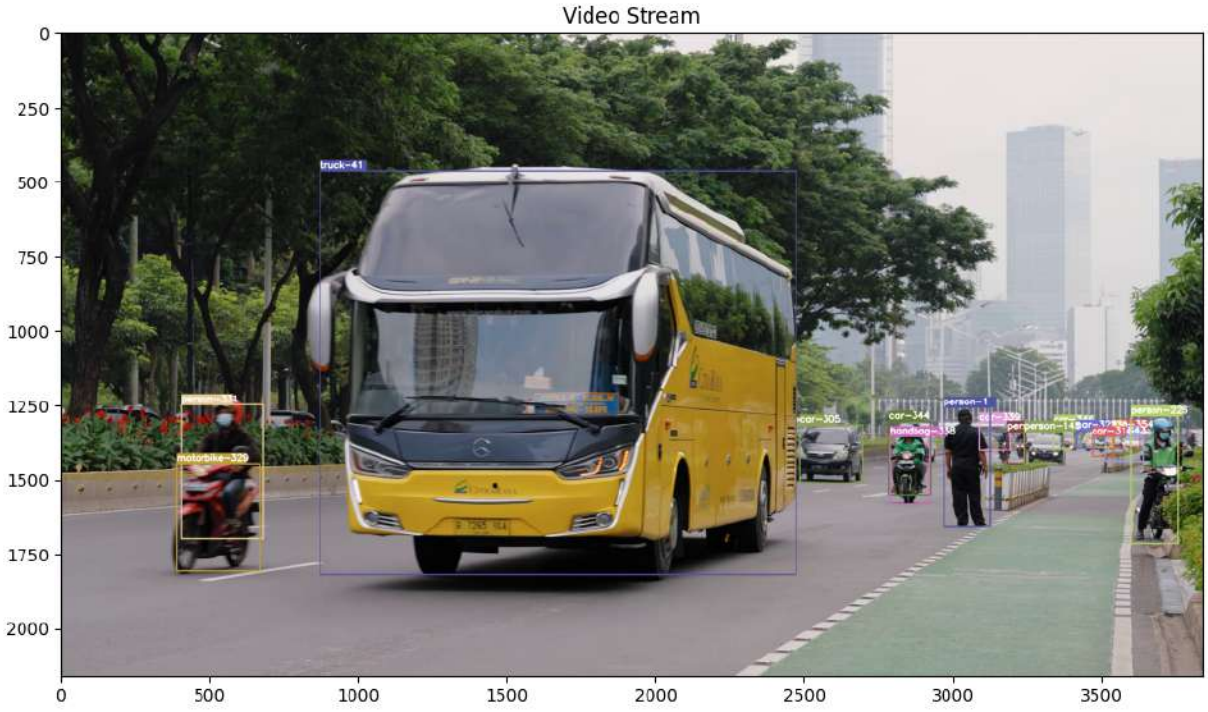

```
ii 171
torch.Size([20, 7])
(17, 6)
```


```
ii 172
torch.Size([19, 7])
(17, 6)
```

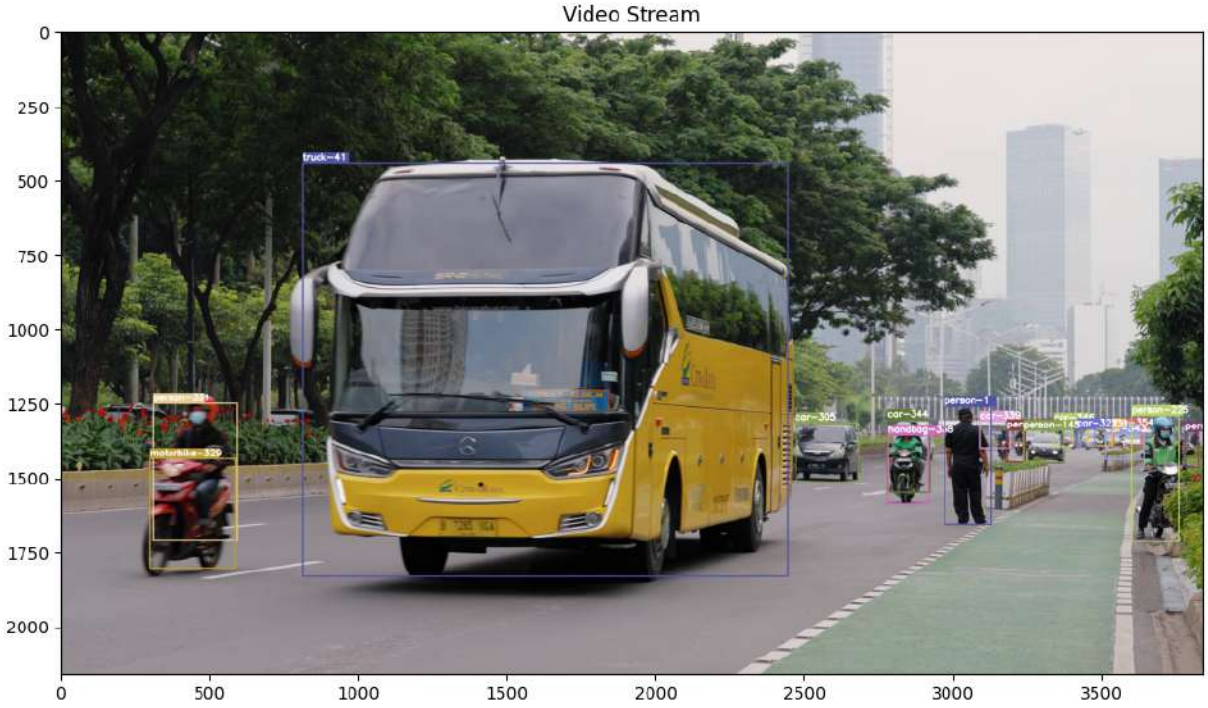



```
ii 173
torch.Size([20, 7])
(18, 6)
```

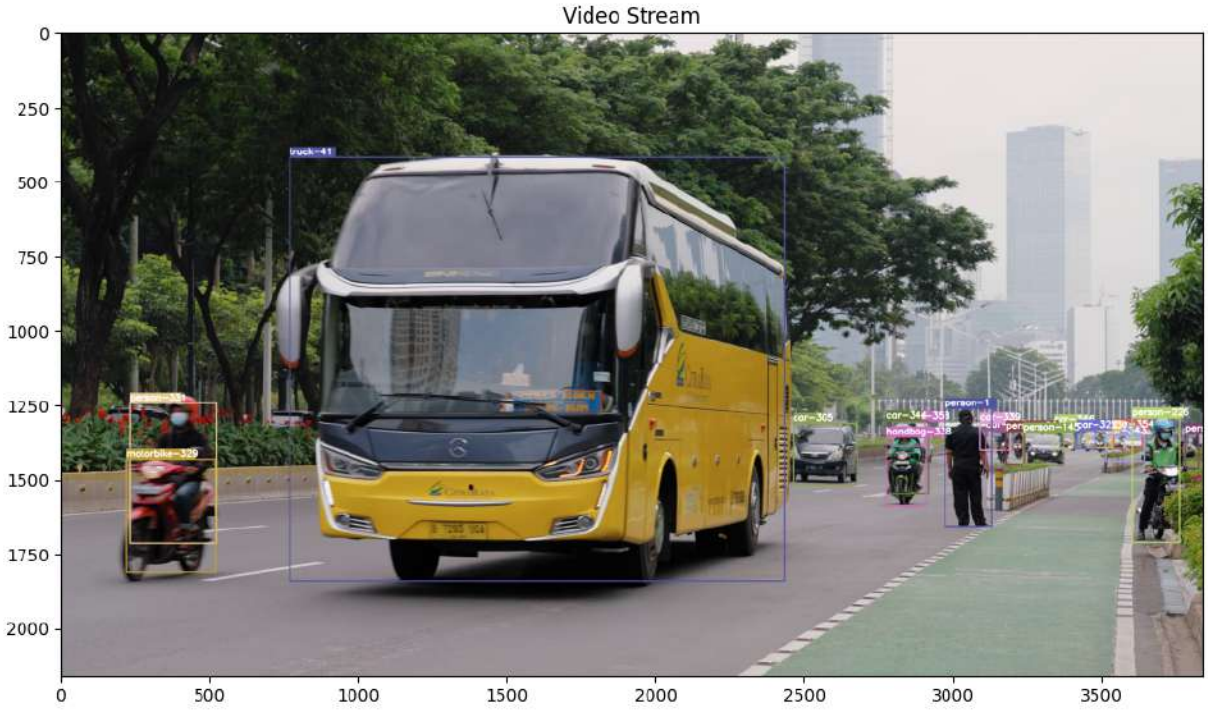

```
ii 174
torch.Size([20, 7])
(17, 6)
```



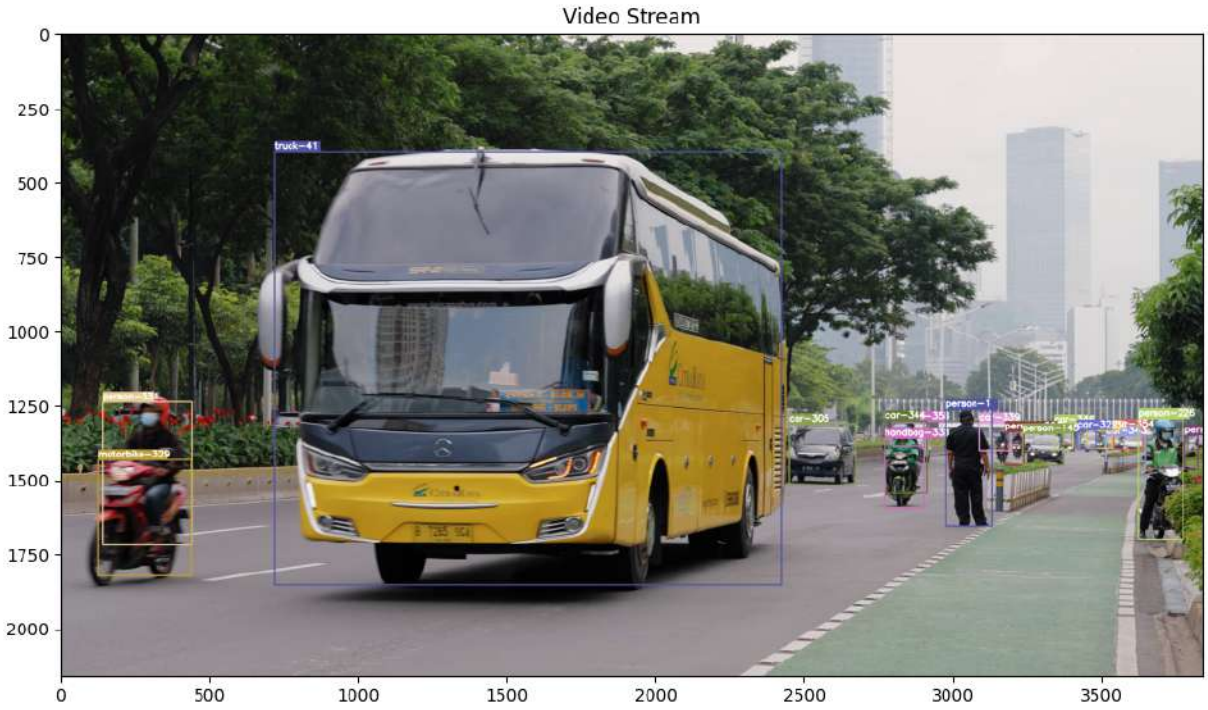

```
ii 175
torch.Size([20, 7])
(17, 6)
```



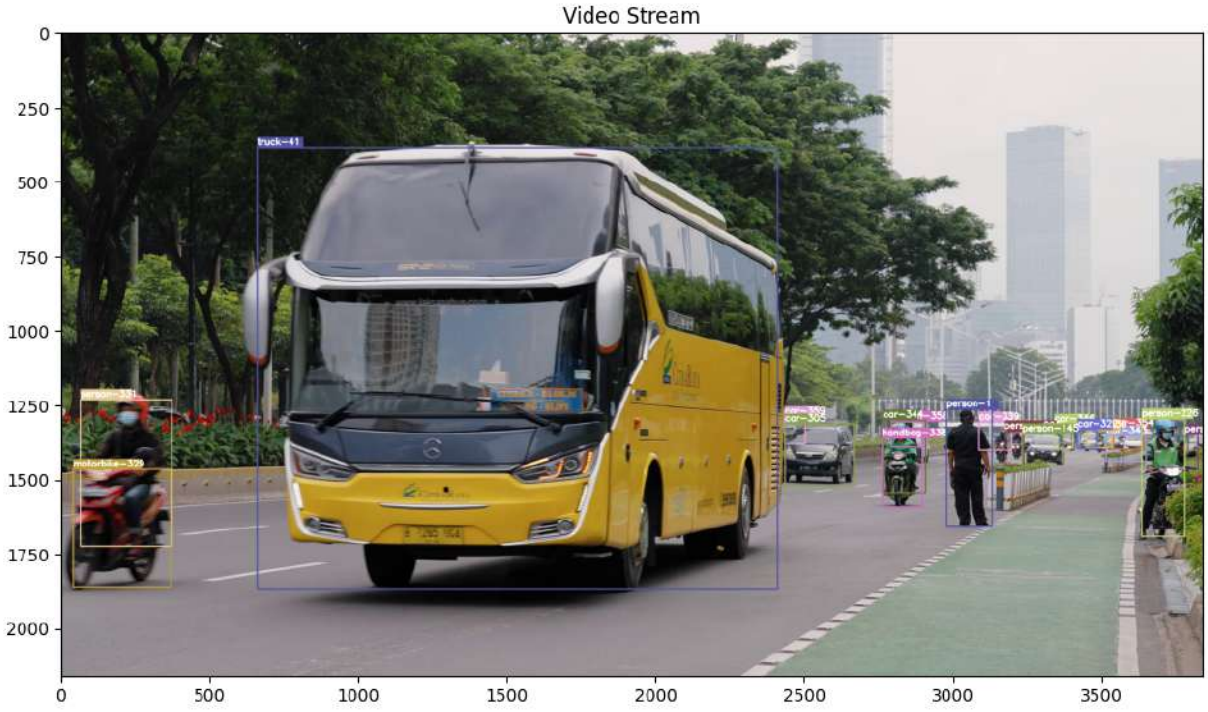

```
ii 176
torch.Size([21, 7])
(19, 6)
```



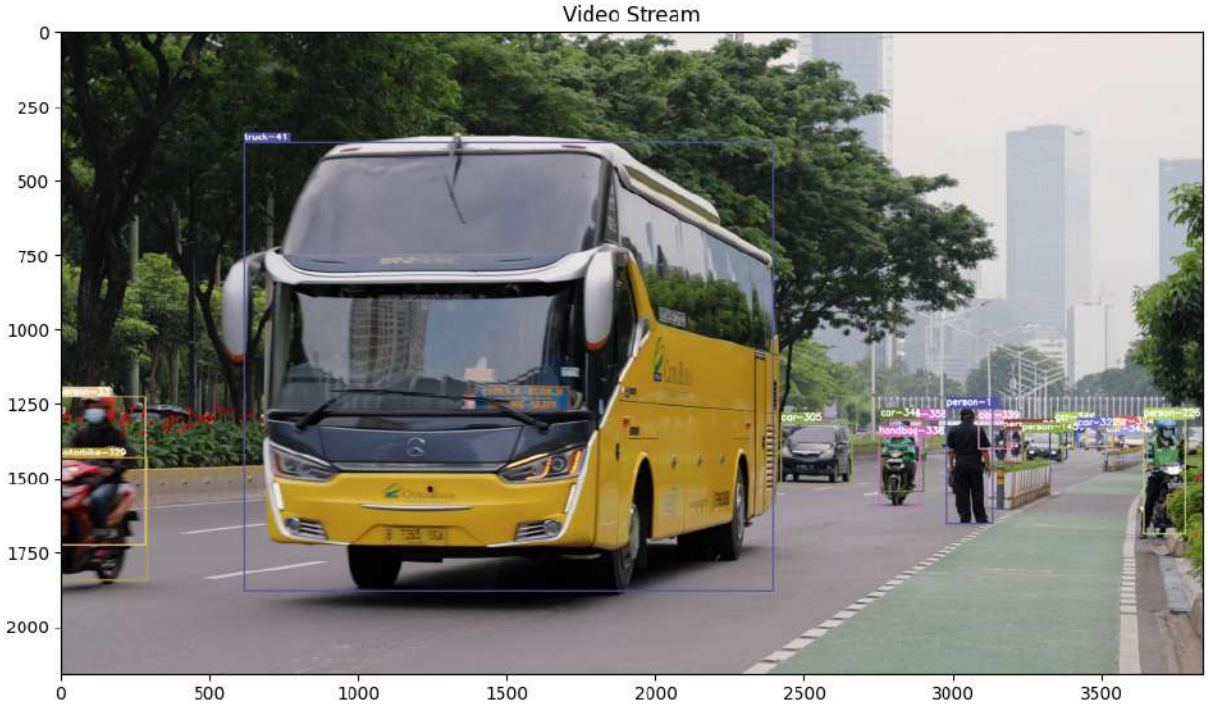

```
ii 177
torch.Size([22, 7])
(18, 6)
```



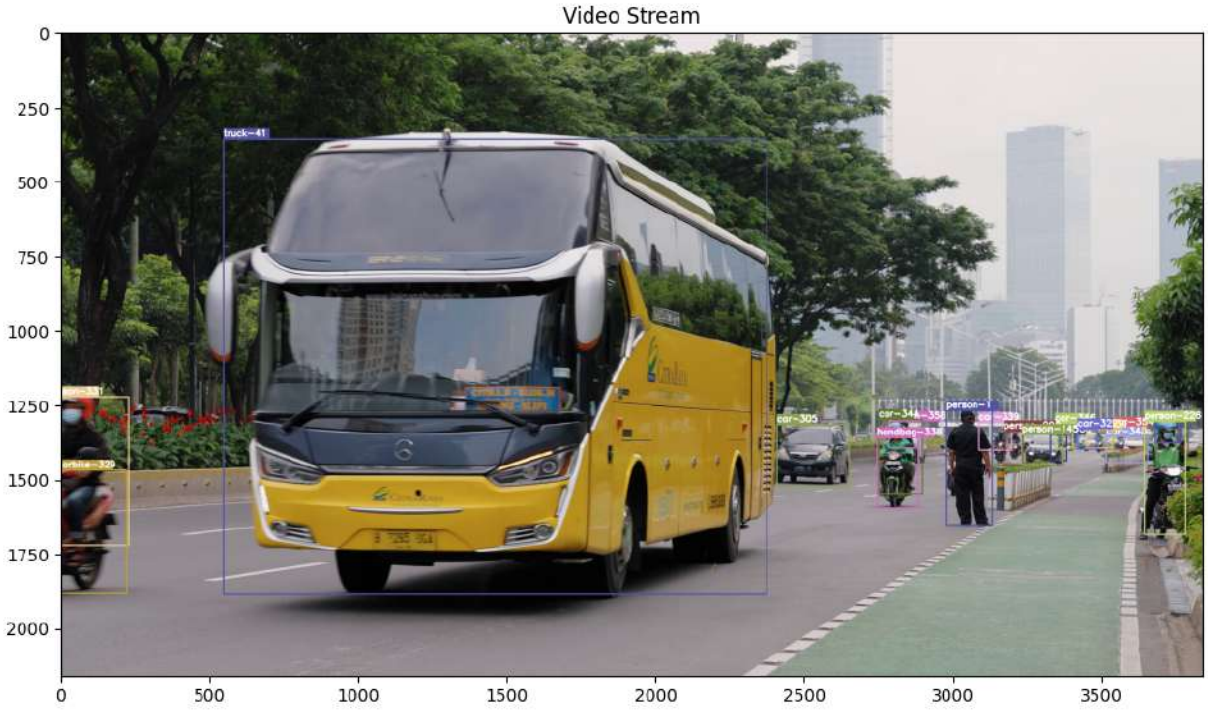

```
ii 178
torch.Size([21, 7])
(19, 6)
```



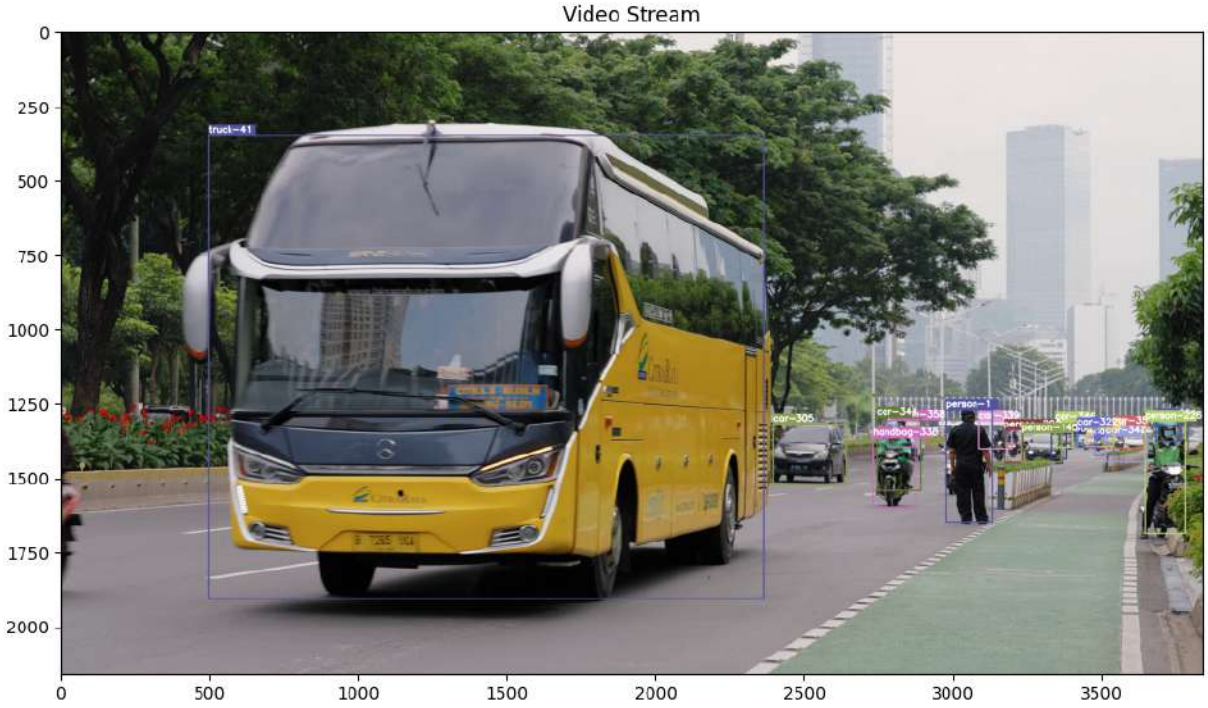

```
ii 179
torch.Size([22, 7])
(18, 6)
```



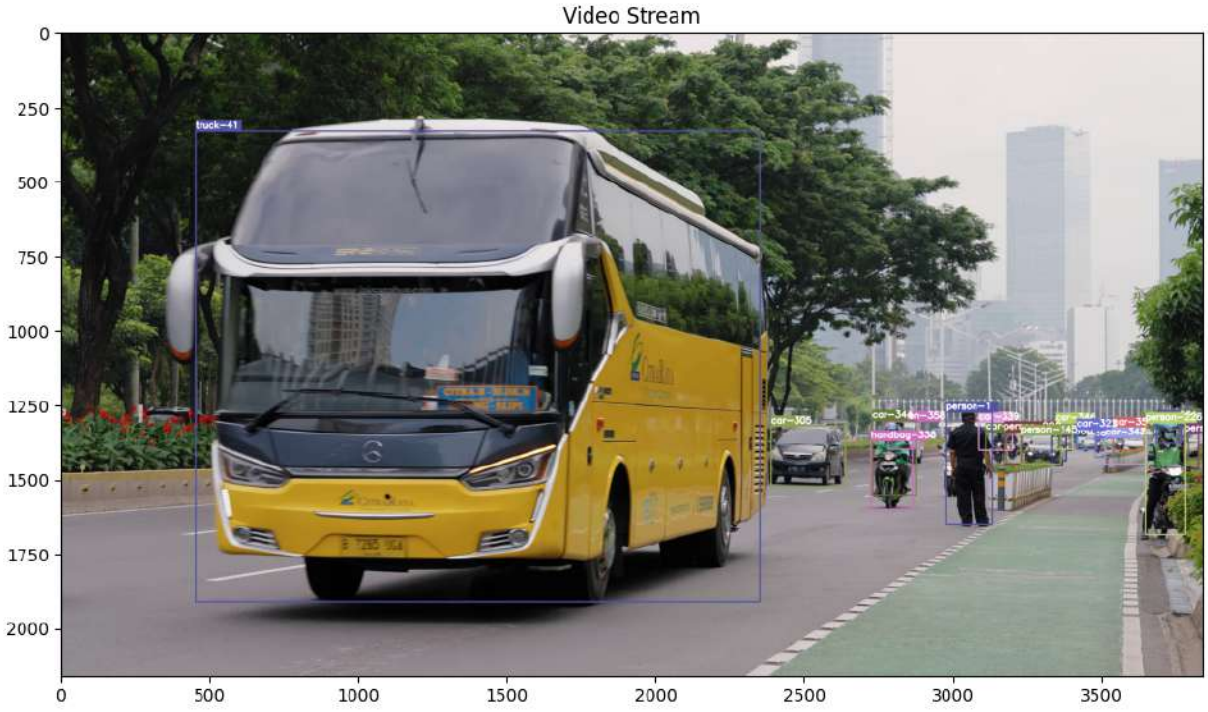

```
ii 180
torch.Size([23, 7])
(19, 6)
```



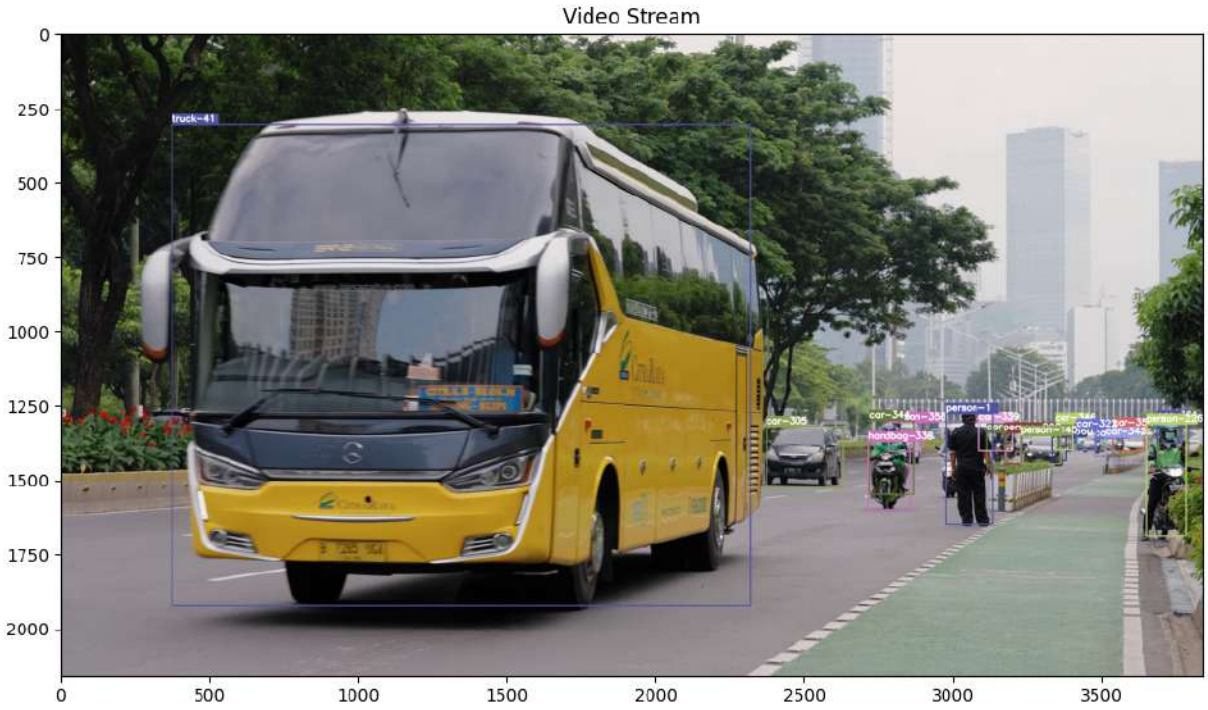

```
ii 181
torch.Size([23, 7])
(18, 6)
```




```
ii 182
torch.Size([20, 7])
(19, 6)
```




```
ii 183
torch.Size([20, 7])
(18, 6)
```




```
ii 184
torch.Size([18, 7])
(17, 6)
```

Παρατηρούμε ότι με μικρό confidence threshold, ο YOLO μας επιστρέφει πολλά bounding boxes με πολλές προβλέψεις. Από τη μία αναγνωρίζονται αυτοκίνητα στο βάθος της εικόνας, από την άλλη όμως υπάρζουν κάποιες λανθασμένες προβλέψεις (πχ στο 1ο frame, ότι υπάρχουν 2 άνθρωποι στα δεξιά της εικονας, ή το φορτηγό στη μέση, με το ροζ bounding box).

Για ίδιο nms_thres και μεγαλύτερο confidence threshold προβλέπουμε ότι οι προβλέψεις που θα επιστραφούν από τον αλγόριθμο θα είναι αρκετά λιγότερες.

Μια μεγάλη τιμή του conf_thres , κρατώντας την τιμή για nms_trhes=0.4

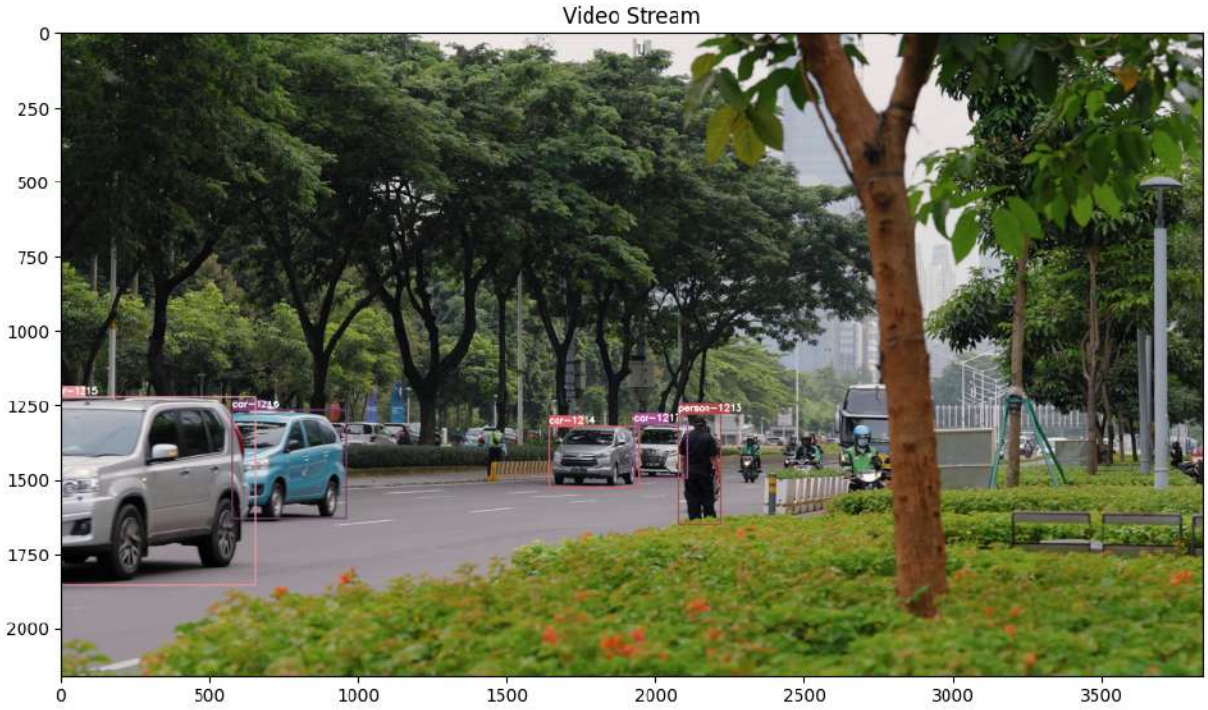
In [ ]:
```
conf_thres=0.9
nms_thres=0.4 # NMS threshold
```

In [ ]:
```
%pylab inline
import cv2
from IPython.display import clear_output

videopath = './drive/My Drive/20.mp4'  # Εδώ θα αλλάξετε το path για να πάρετε το v
                                                                    # Το video θα

cmap = plt.get_cmap('tab20b')
colors = [cmap(i)[:3] for i in np.linspace(0, 1, 20)]

# initialize Sort object and video capture
from sort import *
vid = cv2.VideoCapture(videopath)
mot_tracker = Sort()
```

```
frames = vid.get(cv2.CAP_PROP_FRAME_COUNT)
#while(True):
for ii in range(70):
    ret, frame = vid.read()
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    pilimg = Image.fromarray(frame)
    detections = detect_image(pilimg)

    img = np.array(pilimg)
    pad_x = max(img.shape[0] - img.shape[1], 0) * (img_size / max(img.shape))
    pad_y = max(img.shape[1] - img.shape[0], 0) * (img_size / max(img.shape))
    unpad_h = img_size - pad_y
    unpad_w = img_size - pad_x
    if detections is not None:
        print("ii", ii)
        print(detections.shape)
        tracked_objects = mot_tracker.update(detections.cpu())
        print(tracked_objects.shape)

        unique_labels = detections[:, -1].cpu().unique()
        n_cls_preds = len(unique_labels)
        for x1, y1, x2, y2, obj_id, cls_pred in tracked_objects:
            box_h = int(((y2 - y1) / unpad_h) * img.shape[0])
            box_w = int(((x2 - x1) / unpad_w) * img.shape[1])
            y1 = int(((y1 - pad_y // 2) / unpad_h) * img.shape[0])
            x1 = int(((x1 - pad_x // 2) / unpad_w) * img.shape[1])

            color = colors[int(obj_id) % len(colors)]
            color = [i * 255 for i in color]
            cls = classes[int(cls_pred)]
            cv2.rectangle(frame, (x1, y1), (x1+box_w, y1+box_h), color, 4)
            cv2.rectangle(frame, (x1, y1-35), (x1+len(cls)*19+60, y1), color, -1)
            cv2.putText(frame, cls + "-" + str(int(obj_id)), (x1, y1 - 10), cv2.FON

    fig=figure(figsize=(12, 8))
    title("Video Stream")
    imshow(frame)
    show()
    #clear_output(wait=True)
```

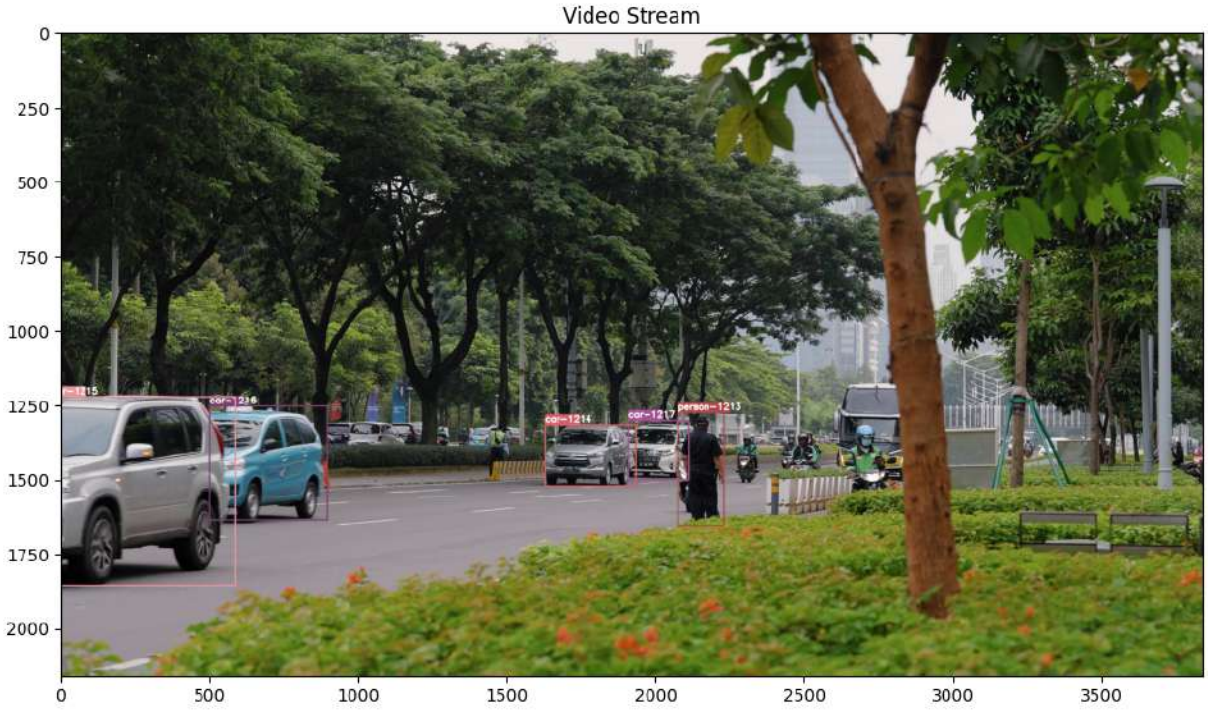
```
Populating the interactive namespace from numpy and matplotlib
ii 0
torch.Size([6, 7])
(6, 6)
```


```
ii 1
torch.Size([5, 7])
(5, 6)
```



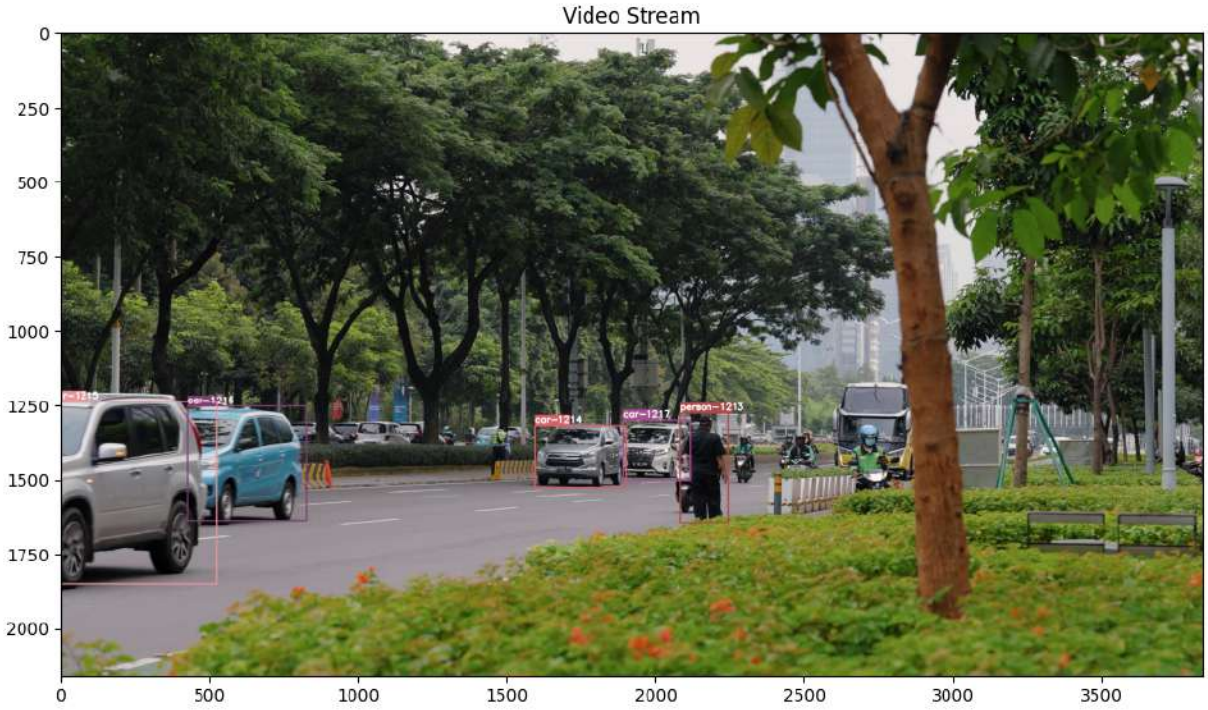

```
ii 2
torch.Size([6, 7])
(6, 6)
```


```
ii 3
torch.Size([5, 7])
(5, 6)
```

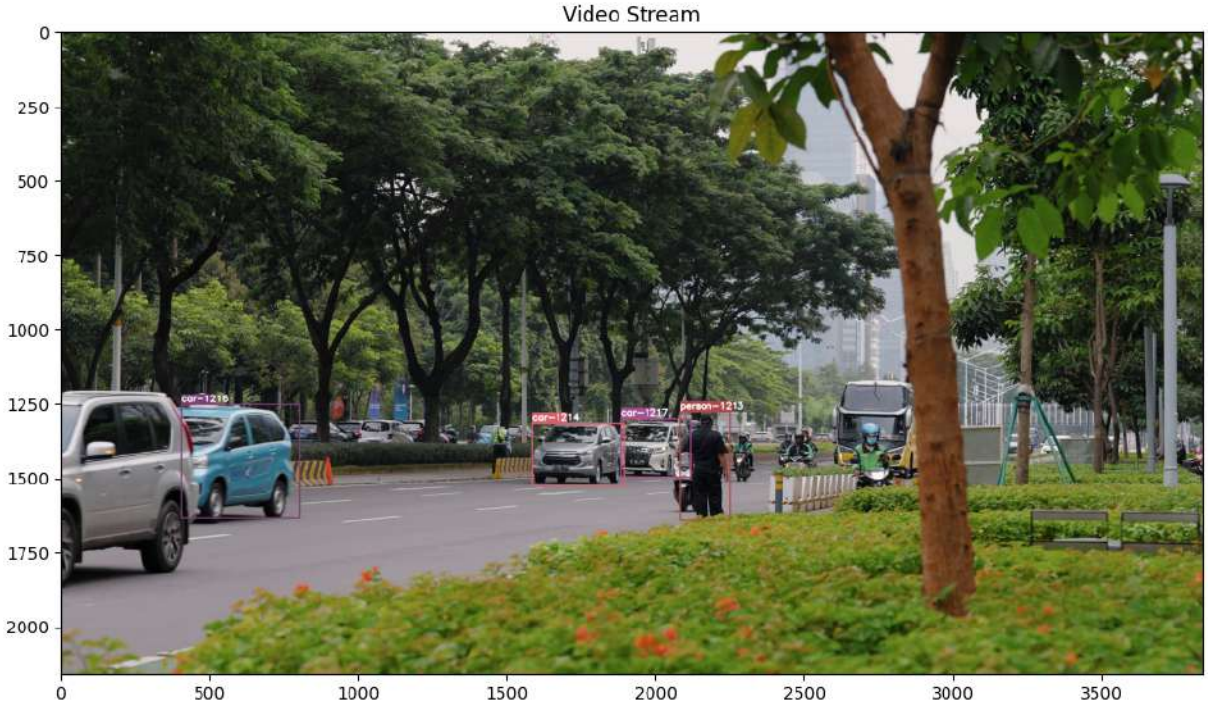



```
ii 4
torch.Size([5, 7])
(5, 6)
```

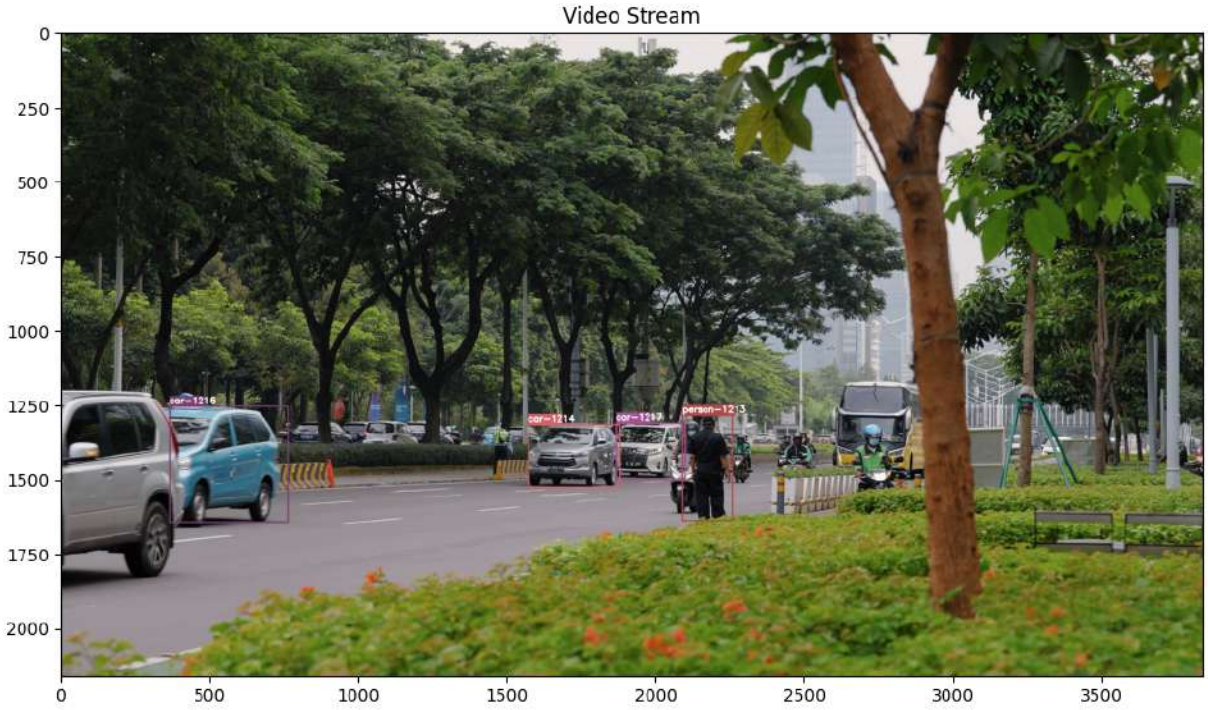

```
ii 5
torch.Size([6, 7])
(5, 6)
```




```
ii 6
torch.Size([5, 7])
(4, 6)
```




```
ii 7
torch.Size([5, 7])
(4, 6)
```

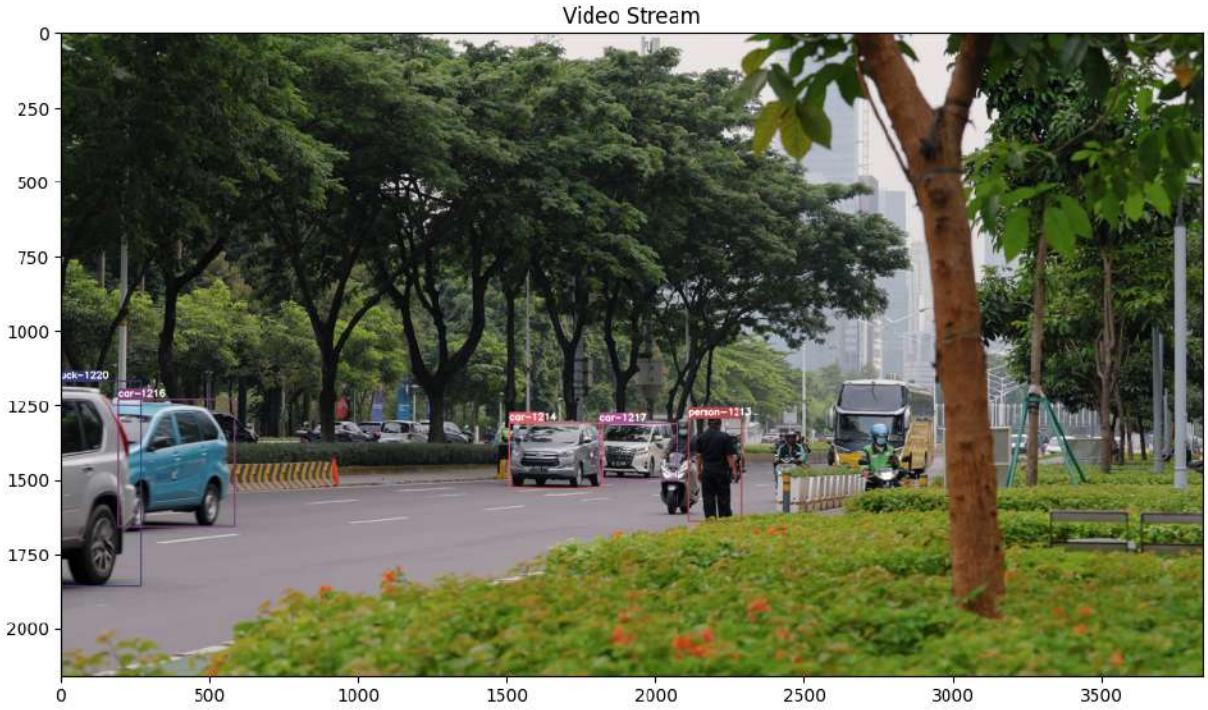



```
ii 8
torch.Size([6, 7])
(5, 6)
```


```
ii 9
torch.Size([5, 7])
(4, 6)
```


```
ii 10
torch.Size([5, 7])
(4, 6)
```


```
ii 11
torch.Size([5, 7])
(5, 6)
```




```
ii 12
torch.Size([5, 7])
(5, 6)
```


```
ii 13
torch.Size([6, 7])
(4, 6)
```


```
ii 14
torch.Size([6, 7])
(4, 6)
```


```
ii 15
torch.Size([6, 7])
(4, 6)
```

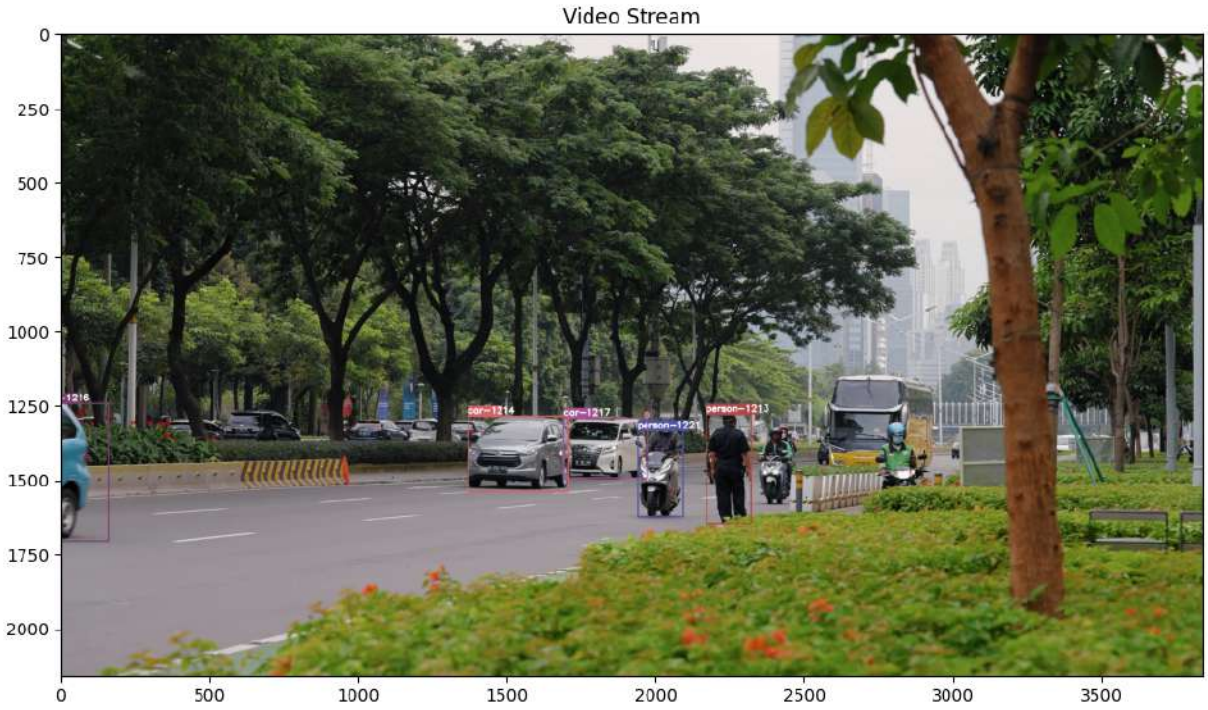

```
ii 16
torch.Size([6, 7])
(6, 6)
```


```
ii 17
torch.Size([5, 7])
(5, 6)
```


```
ii 18
torch.Size([6, 7])
(5, 6)
```

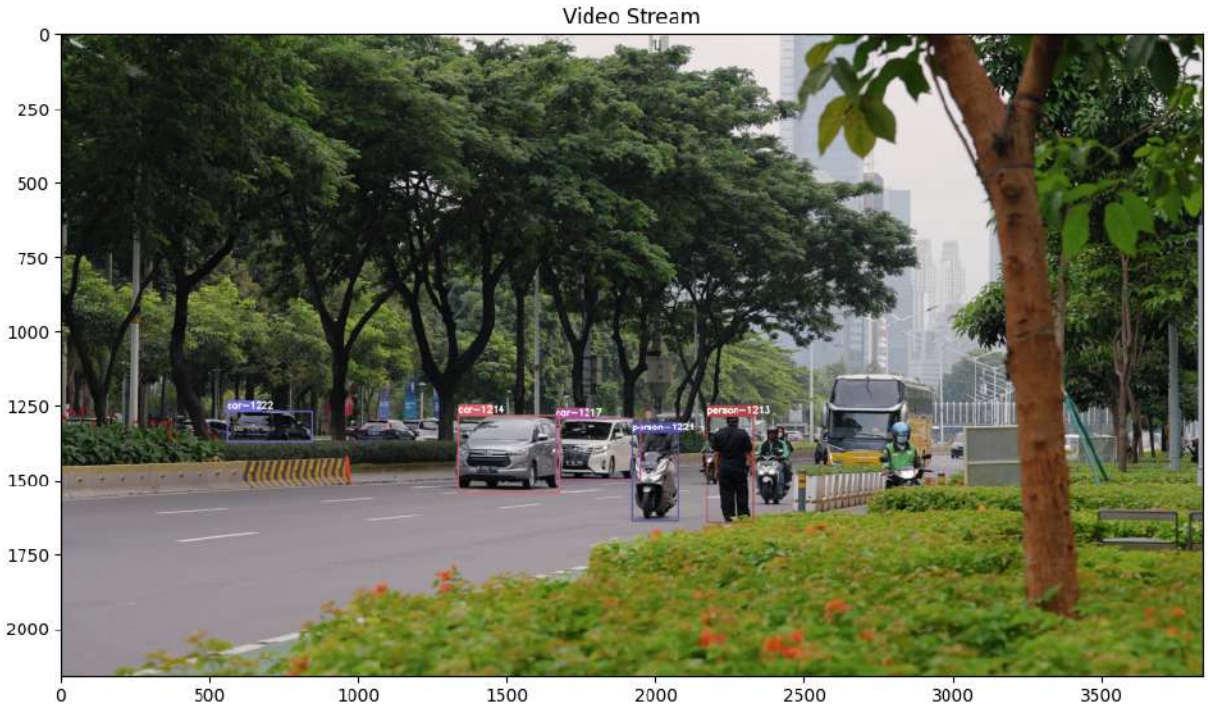

```
ii 19
torch.Size([5, 7])
(5, 6)
```


```
ii 20
torch.Size([7, 7])
(5, 6)
```

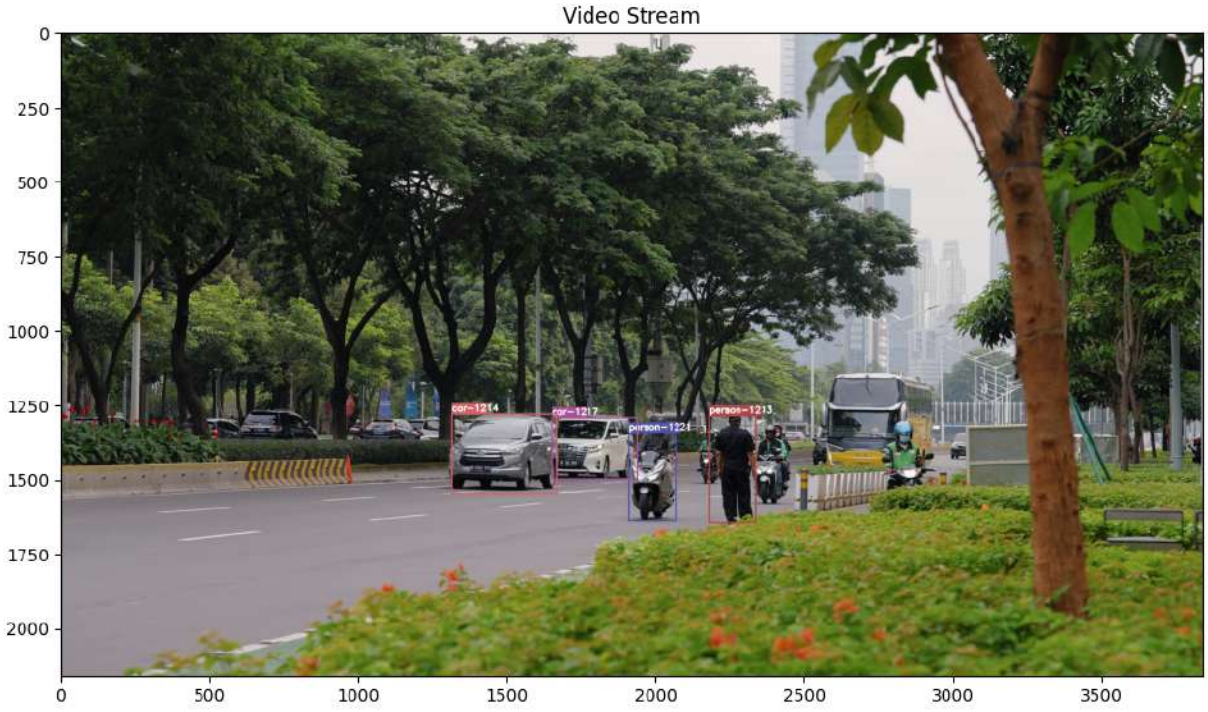



```
ii 21
torch.Size([5, 7])
(4, 6)
```

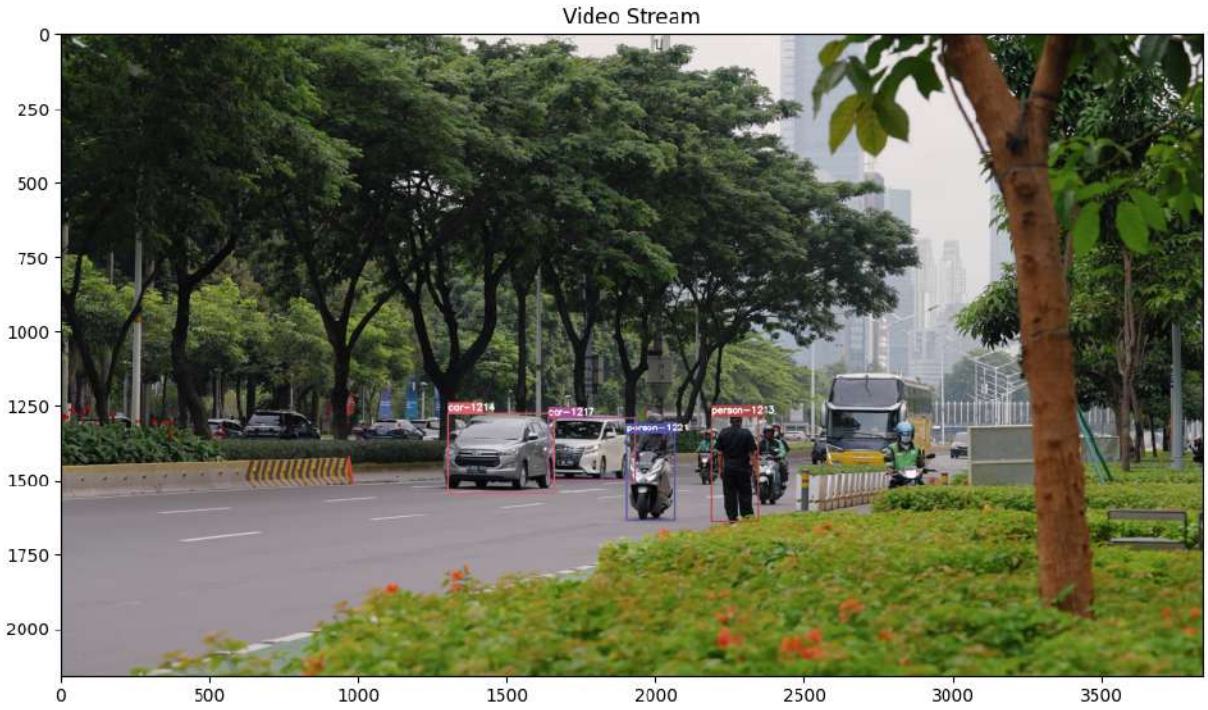

```
ii 22
torch.Size([5, 7])
(5, 6)
```




```
ii 23
torch.Size([4, 7])
(4, 6)
```




```
ii 24
torch.Size([5, 7])
(4, 6)
```




```
ii 25
torch.Size([5, 7])
(4, 6)
```


```
ii 26
torch.Size([5, 7])
(4, 6)
```

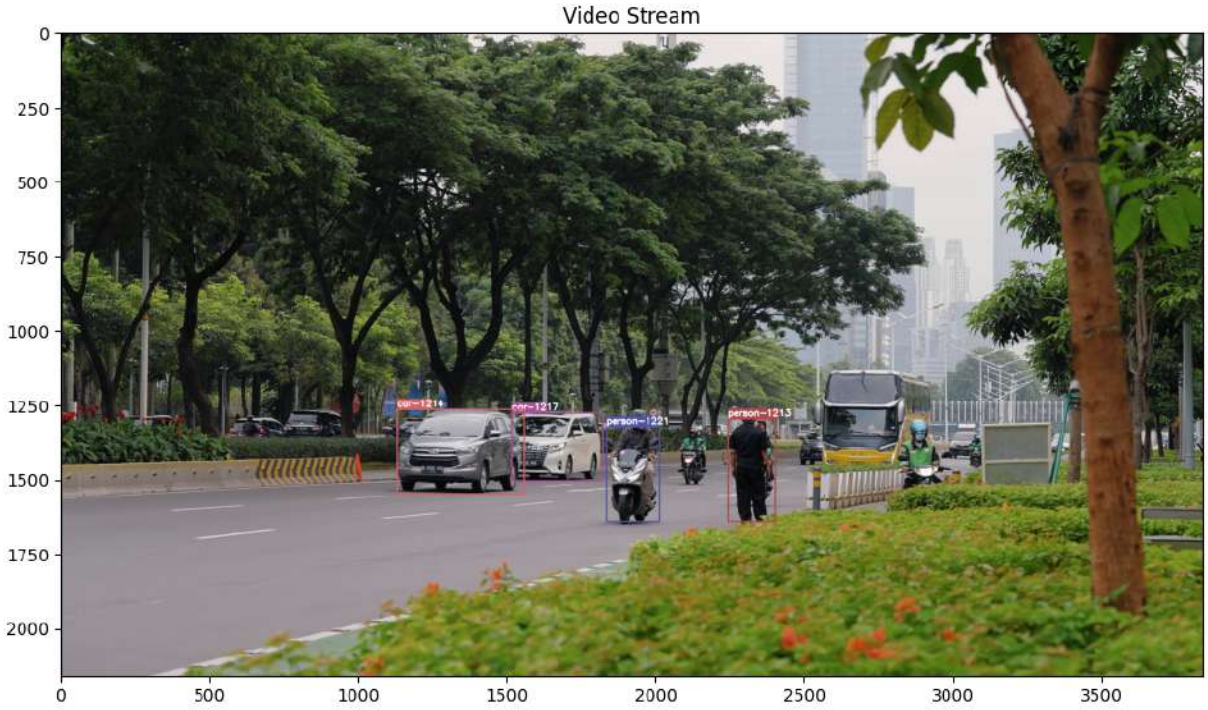

```
ii 27
torch.Size([5, 7])
(4, 6)
```


```
ii 28
torch.Size([5, 7])
(5, 6)
```


```
ii 29
torch.Size([5, 7])
(4, 6)
```


```
ii 30
torch.Size([5, 7])
(4, 6)
```


```
ii 31
torch.Size([6, 7])
(4, 6)
```




```
ii 32
torch.Size([7, 7])
(4, 6)
```


```
ii 33
torch.Size([7, 7])
(4, 6)
```


```
ii 34
torch.Size([7, 7])
(7, 6)
```


```
ii 35
torch.Size([7, 7])
(7, 6)
```


```
ii 36
torch.Size([7, 7])
(7, 6)
```


```
ii 37
torch.Size([6, 7])
(5, 6)
```


```
ii 38
torch.Size([7, 7])
(5, 6)
Buffered data was truncated after reaching the output size limit.
```

Όπως προβλέψαμε, ο αλγόριθμος μας επέστρεψε αρκετά λιγότερες προβλέψεις. Πάλι έχουμε θετικά και αρνητικά. Αρχικά ο αλγόριθμος δεν φαίνεται να έκανε κάποιο λάθος μεν, αλλά ταυτόχρονα δεν προέβλεψε αρκετά αντικείμενα, τα οποία φαίνεται να είναι σημαντικά στην εικόνα (πχ τα μηχανάκια στον δρόμο).

iii) μια μικρή τιμή του nms_thres, κρατώντας την τιμή για conf_thres=0.8

In [ ]:
```
conf_thres=0.8
nms_thres=0.2 # NMS threshold
```

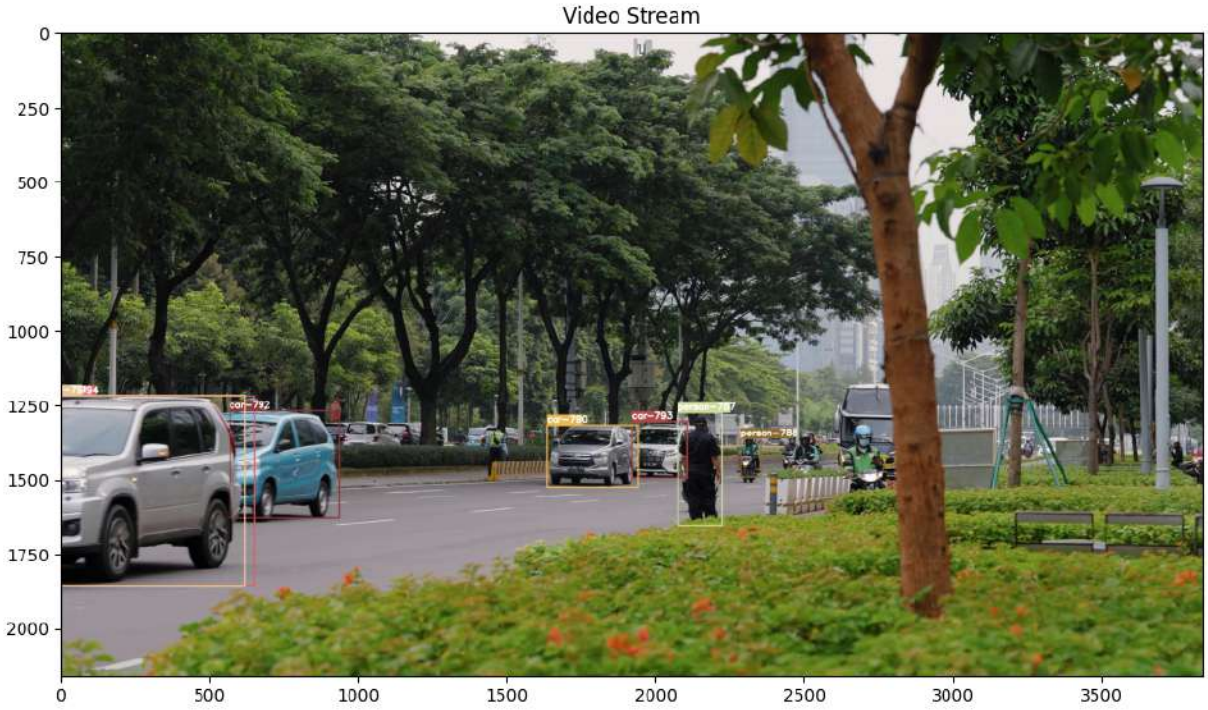
In [ ]:
```
%pylab inline
import cv2
from IPython.display import clear_output

videopath = './drive/My Drive/20.mp4'  # Εδώ θα αλλάξετε το path για να πάρετε το v
                                                                    # Το video θα

cmap = plt.get_cmap('tab20b')
colors = [cmap(i)[:3] for i in np.linspace(0, 1, 20)]

# initialize Sort object and video capture
from sort import *
vid = cv2.VideoCapture(videopath)
mot_tracker = Sort()
frames = vid.get(cv2.CAP_PROP_FRAME_COUNT)
#while(True):
for ii in range(50):
    ret, frame = vid.read()
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    pilimg = Image.fromarray(frame)
    detections = detect_image(pilimg)

    img = np.array(pilimg)
    pad_x = max(img.shape[0] - img.shape[1], 0) * (img_size / max(img.shape))
    pad_y = max(img.shape[1] - img.shape[0], 0) * (img_size / max(img.shape))
    unpad_h = img_size - pad_y
    unpad_w = img_size - pad_x
    if detections is not None:
        print("ii", ii)
        print(detections.shape)
        tracked_objects = mot_tracker.update(detections.cpu())
        print(tracked_objects.shape)

        unique_labels = detections[:, -1].cpu().unique()
        n_cls_preds = len(unique_labels)
        for x1, y1, x2, y2, obj_id, cls_pred in tracked_objects:
            box_h = int(((y2 - y1) / unpad_h) * img.shape[0])
            box_w = int(((x2 - x1) / unpad_w) * img.shape[1])
            y1 = int(((y1 - pad_y // 2) / unpad_h) * img.shape[0])
            x1 = int(((x1 - pad_x // 2) / unpad_w) * img.shape[1])

            color = colors[int(obj_id) % len(colors)]
            color = [i * 255 for i in color]
            cls = classes[int(cls_pred)]
```

```
            cv2.rectangle(frame, (x1, y1), (x1+box_w, y1+box_h), color, 4)
            cv2.rectangle(frame, (x1, y1-35), (x1+len(cls)*19+60, y1), color, -1)
            cv2.putText(frame, cls + "-" + str(int(obj_id)), (x1, y1 - 10), cv2.FON

    fig=figure(figsize=(12, 8))
    title("Video Stream")
    imshow(frame)
    show()
    #clear_output(wait=True)
```

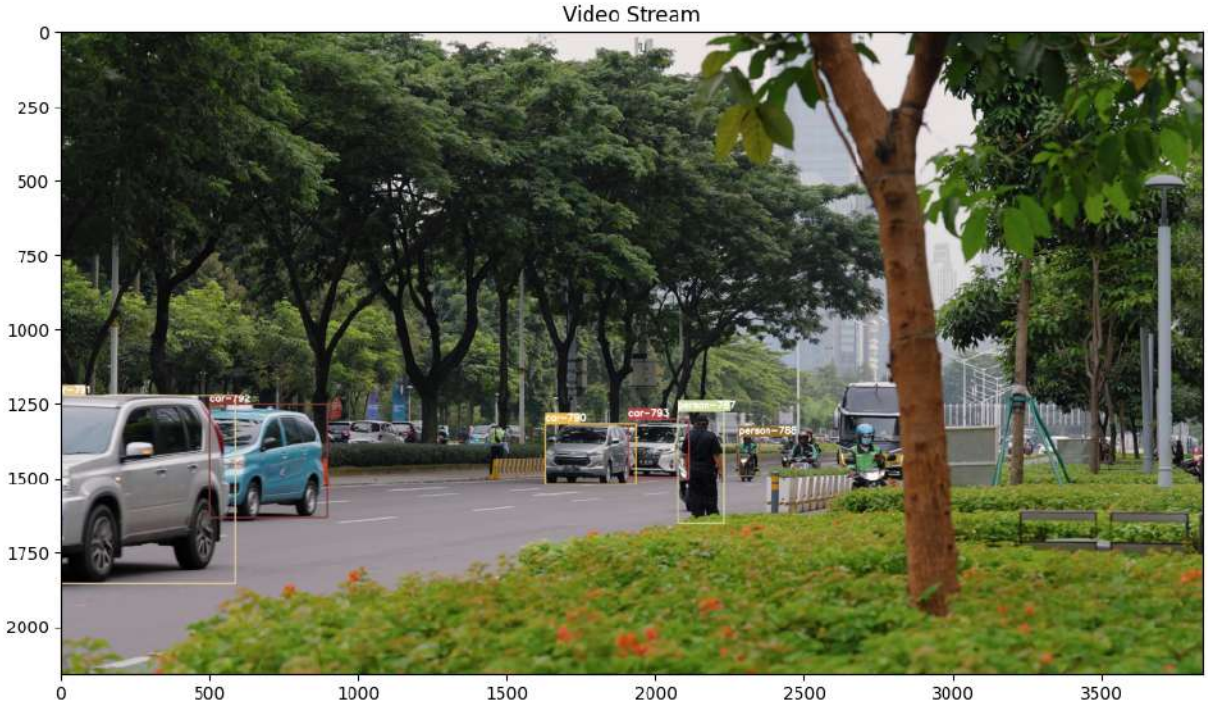
```
Populating the interactive namespace from numpy and matplotlib
ii 0
torch.Size([8, 7])
(8, 6)
```


```
ii 1
torch.Size([6, 7])
(6, 6)
```

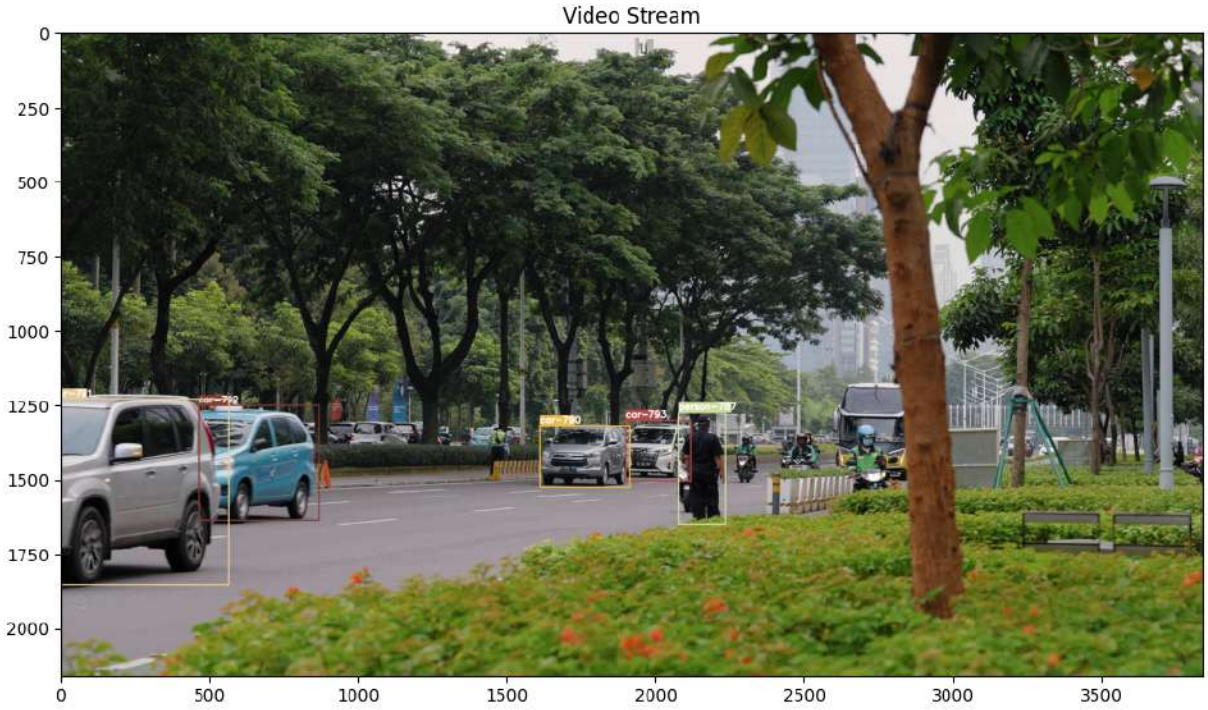

```
ii 2
torch.Size([7, 7])
(7, 6)
```



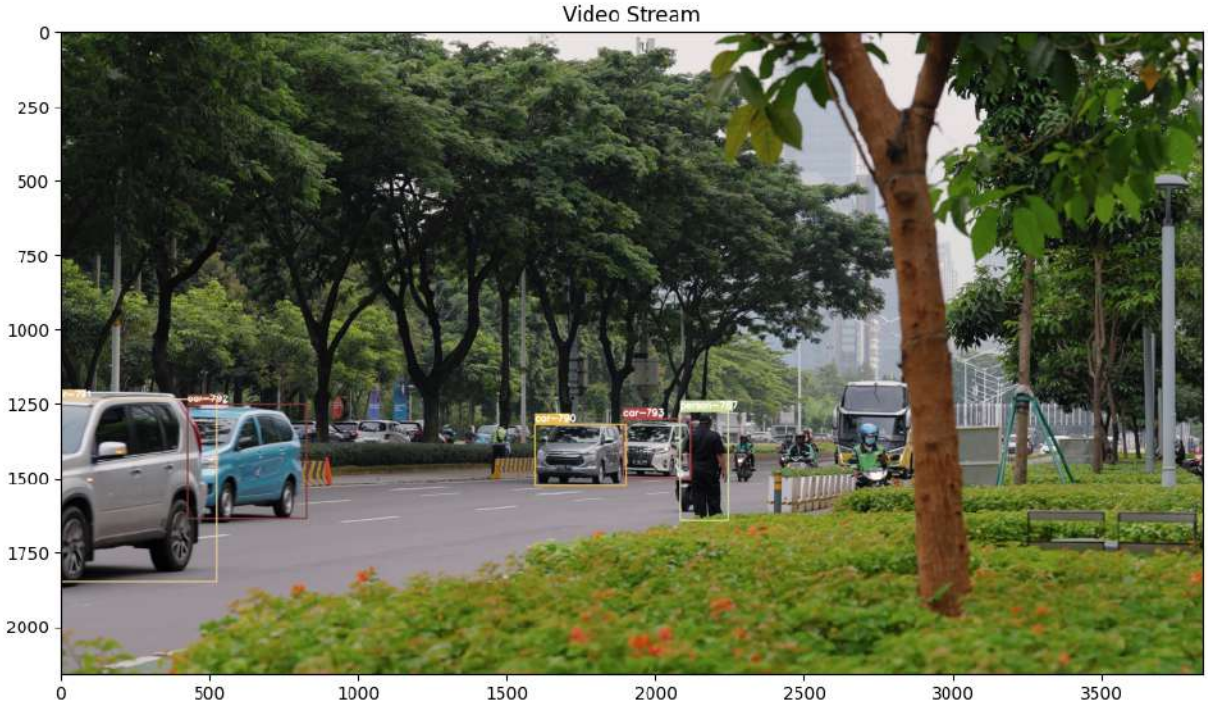

```
ii 3
torch.Size([6, 7])
(6, 6)
```



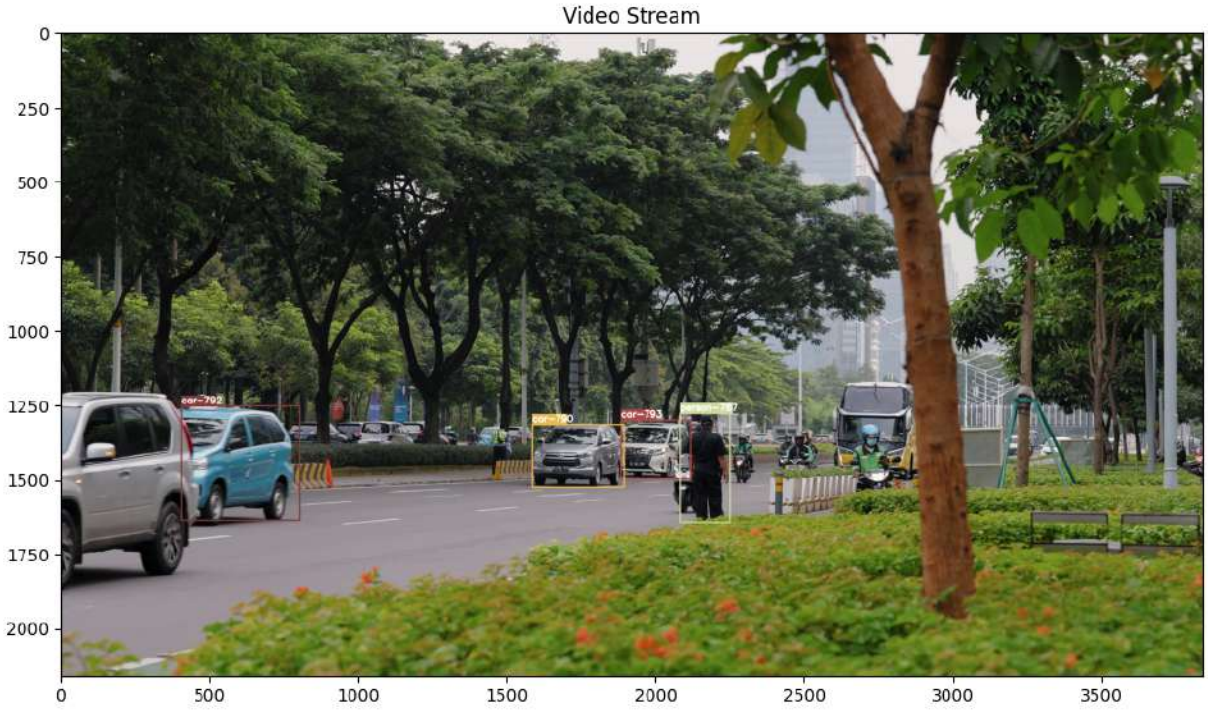

```
ii 4
torch.Size([5, 7])
(5, 6)
```



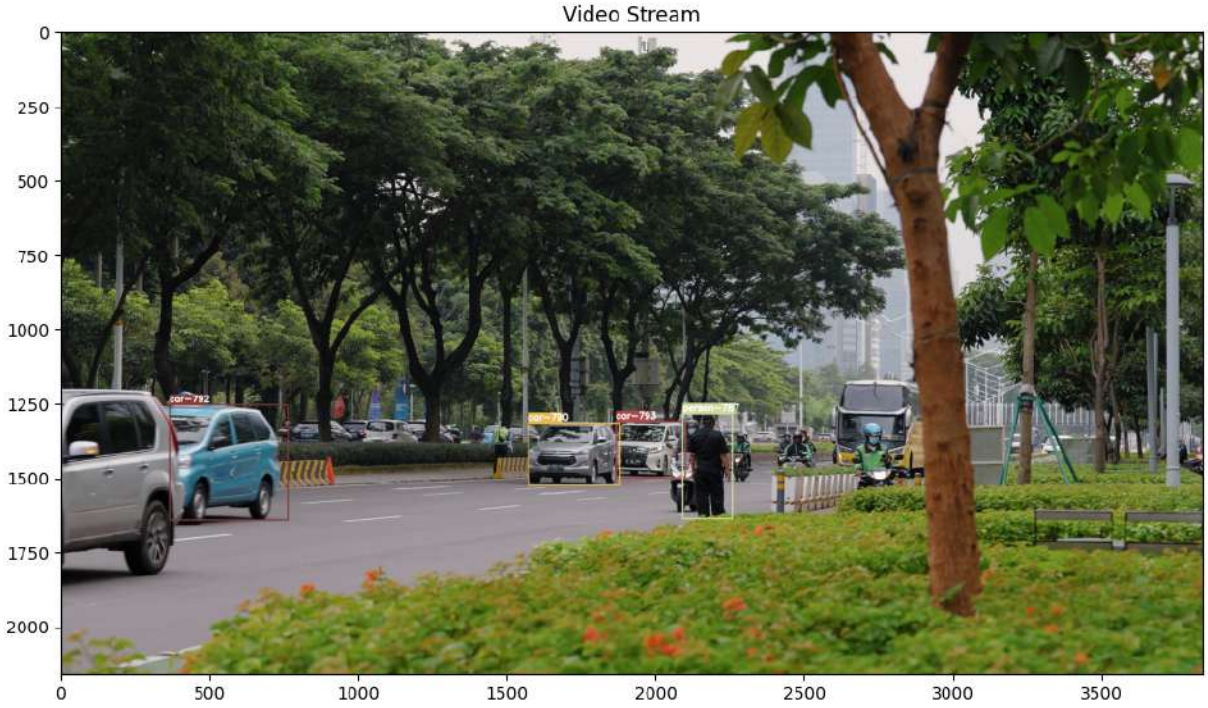

```
ii 5
torch.Size([6, 7])
(5, 6)
```



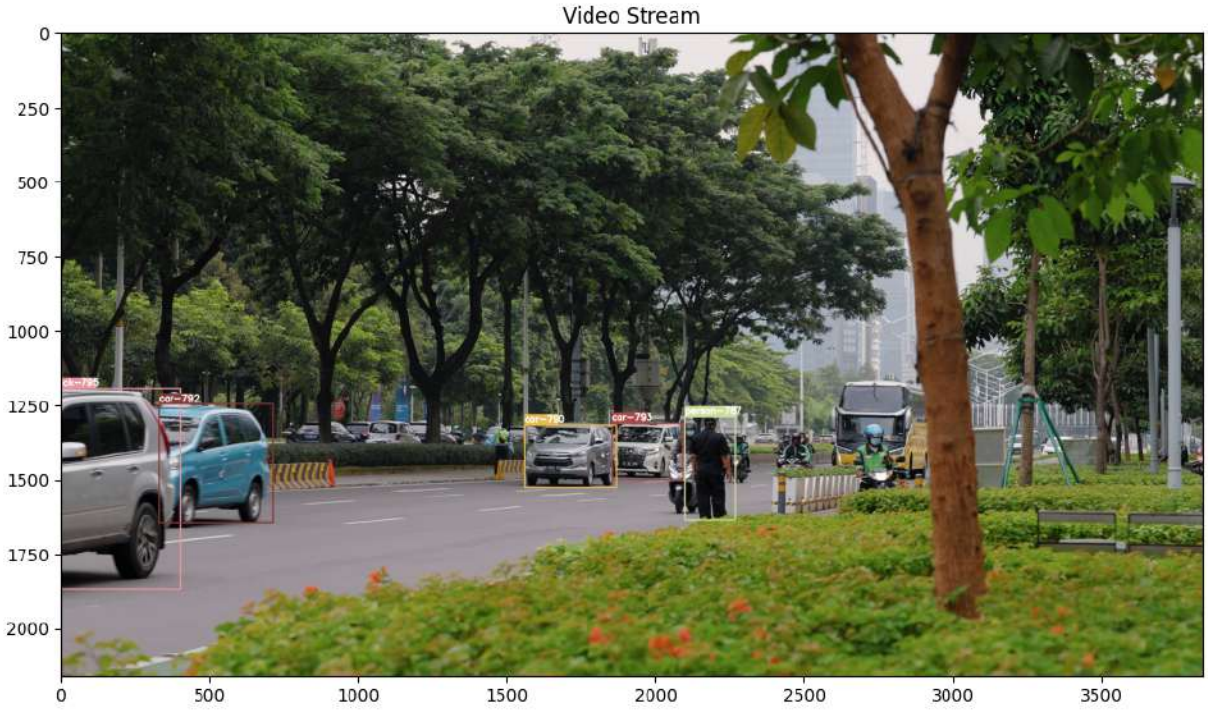

```
ii 6
torch.Size([5, 7])
(4, 6)
```




```
ii 7
torch.Size([6, 7])
(4, 6)
```

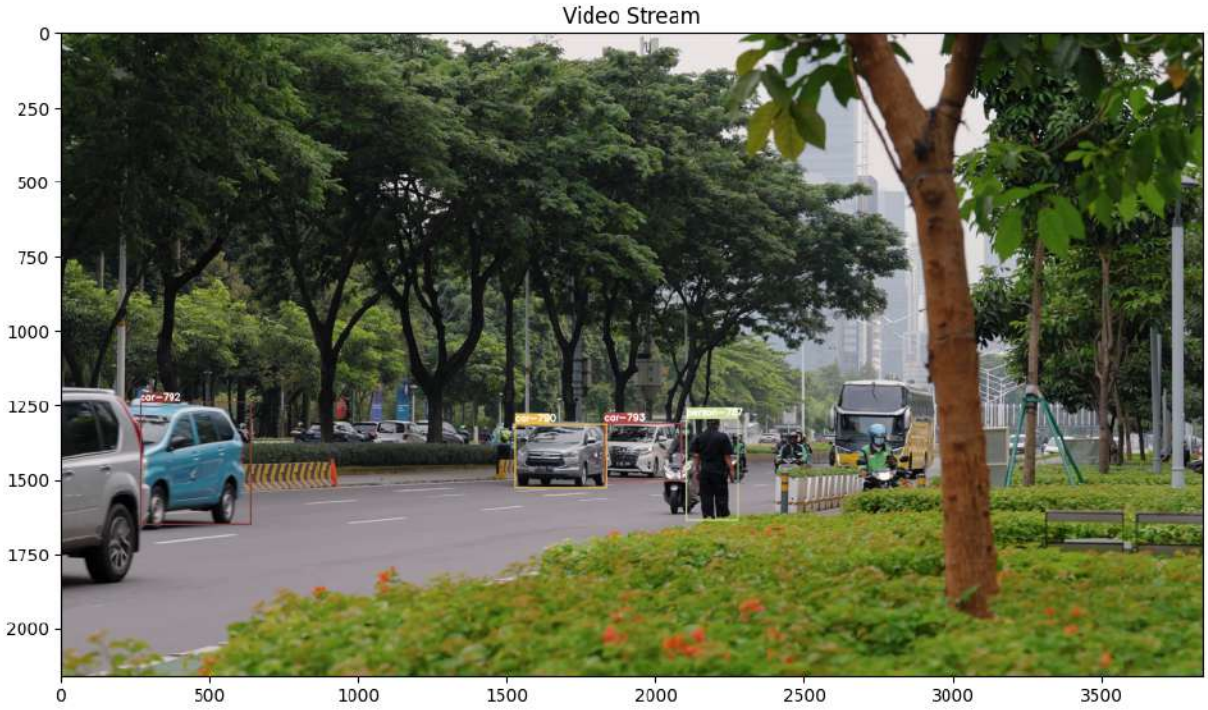



```
ii 8
torch.Size([6, 7])
(5, 6)
```




```
ii 9
torch.Size([5, 7])
(4, 6)
```


```
ii 10
torch.Size([5, 7])
(4, 6)
```




```
ii 11
torch.Size([5, 7])
(5, 6)
```


```
ii 12
torch.Size([7, 7])
(5, 6)
```


```
ii 13
torch.Size([7, 7])
(5, 6)
```


```
ii 14
torch.Size([6, 7])
(4, 6)
```


```
ii 15
torch.Size([6, 7])
(6, 6)
```


```
ii 16
torch.Size([6, 7])
(6, 6)
```


```
ii 17
torch.Size([6, 7])
(6, 6)
```


```
ii 18
torch.Size([6, 7])
(6, 6)
```


```
ii 19
torch.Size([7, 7])
(6, 6)
```


```
ii 20
torch.Size([7, 7])
(6, 6)
```


```
ii 21
torch.Size([7, 7])
(5, 6)
```


```
ii 22
torch.Size([6, 7])
(5, 6)
```


```
ii 23
torch.Size([6, 7])
(5, 6)
```


```
ii 24
torch.Size([5, 7])
(5, 6)
```


```
ii 25
torch.Size([6, 7])
(4, 6)
```


```
ii 26
torch.Size([7, 7])
(4, 6)
```


```
ii 27
torch.Size([7, 7])
(4, 6)
```


```
ii 28
torch.Size([7, 7])
(5, 6)
```


```
ii 29
torch.Size([6, 7])
(4, 6)
```


```
ii 30
torch.Size([9, 7])
(5, 6)
```


```
ii 31
torch.Size([9, 7])
(5, 6)
```


```
ii 32
torch.Size([7, 7])
(6, 6)
```


```
ii 33
torch.Size([8, 7])
(7, 6)
```

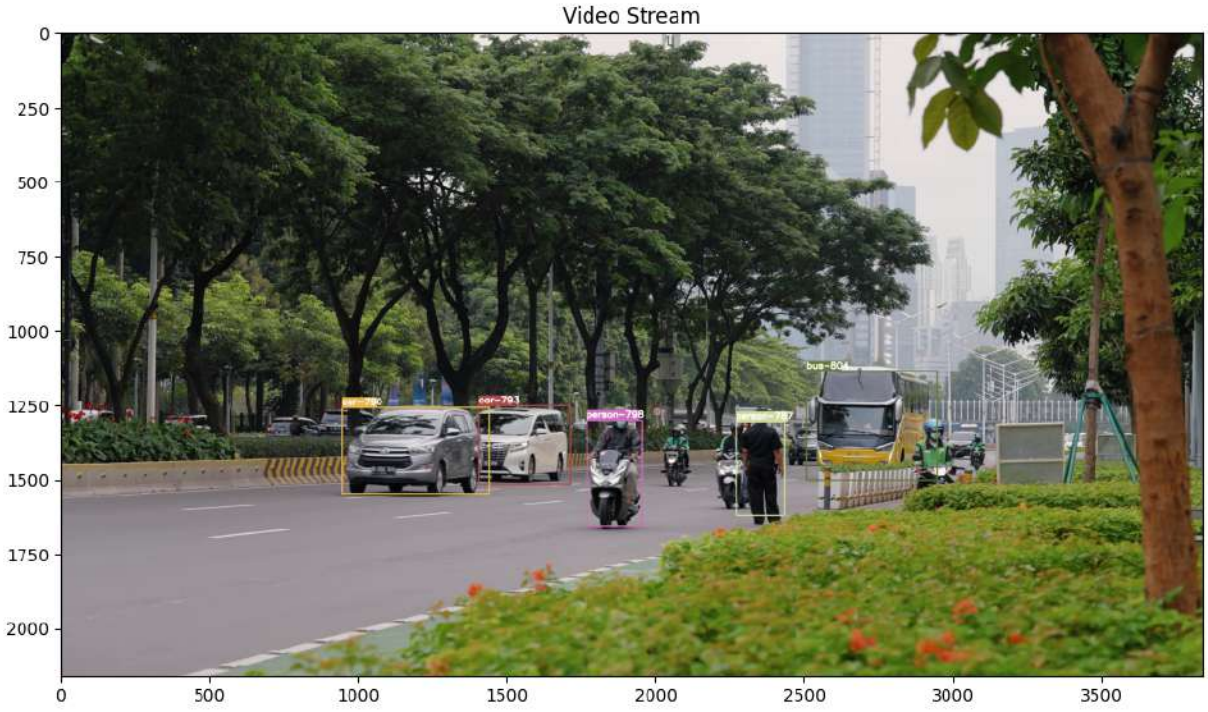

```
ii 34
torch.Size([8, 7])
(7, 6)
```


```
ii 35
torch.Size([7, 7])
(7, 6)
```


```
ii 36
torch.Size([7, 7])
(7, 6)
```


```
ii 37
torch.Size([8, 7])
(6, 6)
```


```
ii 38
torch.Size([8, 7])
(5, 6)
```

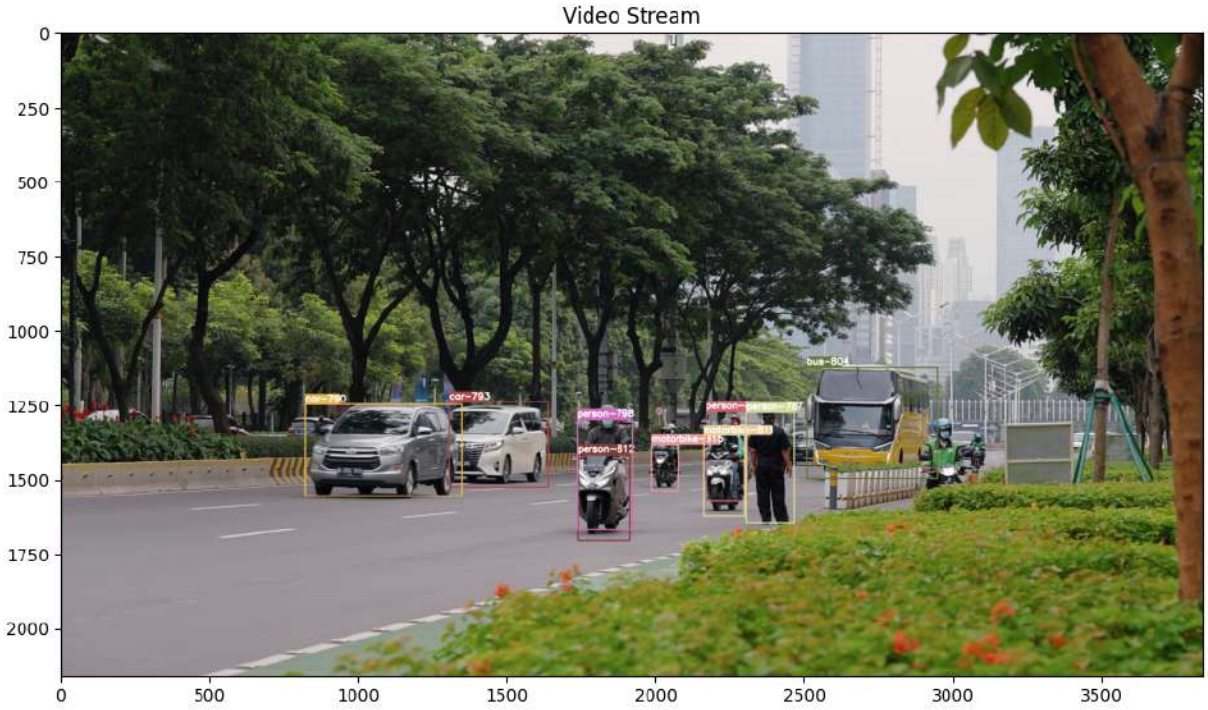



```
ii 39
torch.Size([11, 7])
(5, 6)
```


```
ii 40
torch.Size([10, 7])
(7, 6)
```


```
ii 41
torch.Size([10, 7])
(7, 6)
```


```
ii 42
torch.Size([10, 7])
(9, 6)
```




```
ii 43
torch.Size([9, 7])
(8, 6)
```

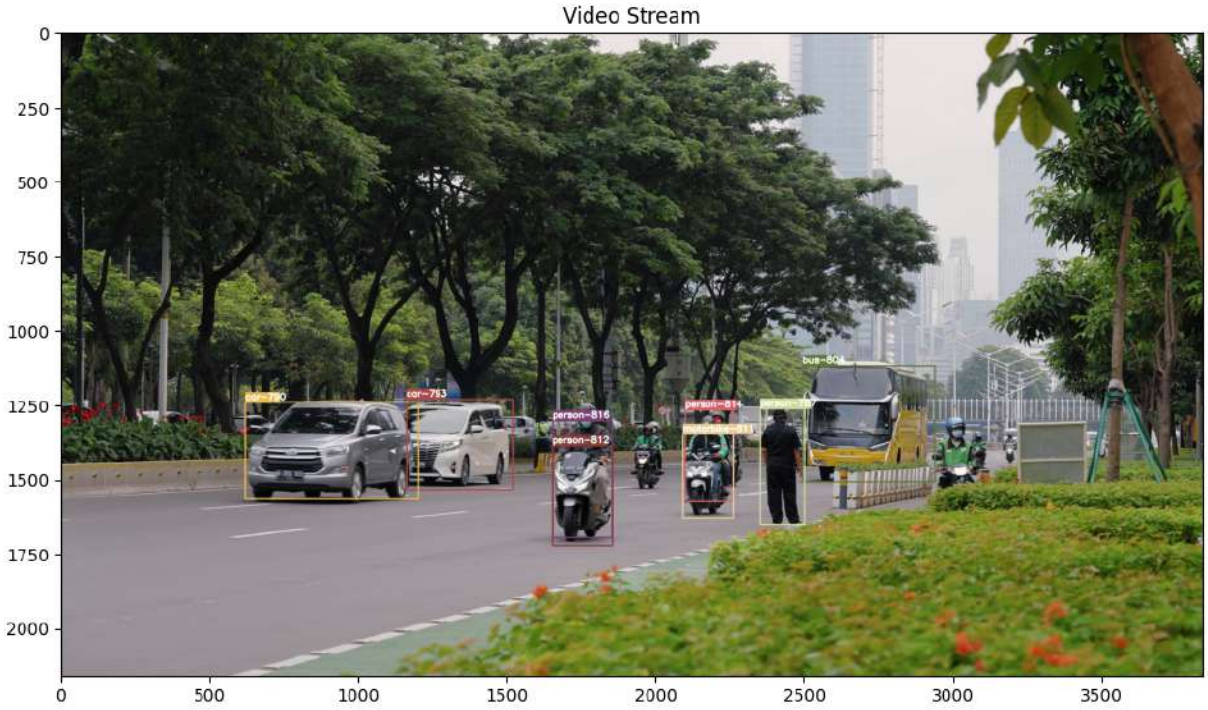

```
ii 44
torch.Size([9, 7])
(8, 6)
```


```
ii 45
torch.Size([9, 7])
(7, 6)
```


```
ii 46
torch.Size([10, 7])
(8, 6)
```


```
ii 47
torch.Size([9, 7])
(8, 6)
```


```
ii 48
torch.Size([9, 7])
(8, 6)
```




```
ii 49
torch.Size([9, 7])
(9, 6)
```

Ο αλγόριθμος σε αυτή την περίπτωση, κρατά πολύ λίγα bounding boxes κατά το non-max suppression. Συγκεκριμένα κρατάει τα bounding boxes με IOU<=0.2. Πάλι κάνει λίγες προβλέψεις, αλλά φαίνεται να μην αναγνωρίζει τα μηχανάκια στο δρόμο.

Για ίδιο confidence threshold, και μεγαλύτερο nms threshold προβλέπουμε πως θα έχουμε περισσότερες προβλέψεις, αφού δεν θα σκαρτάρονται τόσα bounding boxes κατά το non-max suppression.

Μια μεγάλη τιμή του nms_thres, κρατώνταςτην τιμή για conf_trhes=0.8

In [ ]:
```
conf_thres=0.8
nms_thres=0.8
```

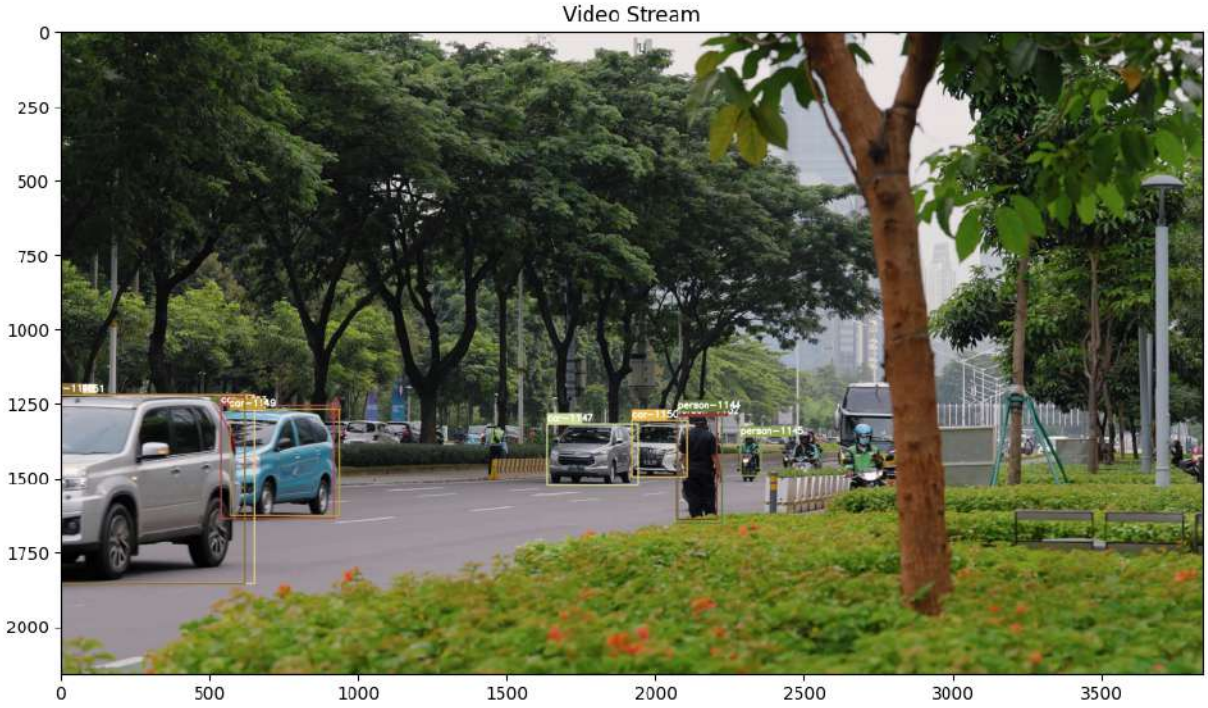
In [ ]:
```
%pylab inline
import cv2
from IPython.display import clear_output

videopath = './drive/My Drive/20.mp4'  # Εδώ θα αλλάξετε το path για να πάρετε το v
                                       # Το video θα

cmap = plt.get_cmap('tab20b')
colors = [cmap(i)[:3] for i in np.linspace(0, 1, 20)]

# initialize Sort object and video capture
from sort import *
vid = cv2.VideoCapture(videopath)
mot_tracker = Sort()
frames = vid.get(cv2.CAP_PROP_FRAME_COUNT)
#while(True):
for ii in range(50):
    ret, frame = vid.read()
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

```
    pilimg = Image.fromarray(frame)
    detections = detect_image(pilimg)

    img = np.array(pilimg)
    pad_x = max(img.shape[0] - img.shape[1], 0) * (img_size / max(img.shape))
    pad_y = max(img.shape[1] - img.shape[0], 0) * (img_size / max(img.shape))
    unpad_h = img_size - pad_y
    unpad_w = img_size - pad_x
    if detections is not None:
        print("ii", ii)
        print(detections.shape)
        tracked_objects = mot_tracker.update(detections.cpu())
        print(tracked_objects.shape)

        unique_labels = detections[:, -1].cpu().unique()
        n_cls_preds = len(unique_labels)
        for x1, y1, x2, y2, obj_id, cls_pred in tracked_objects:
            box_h = int(((y2 - y1) / unpad_h) * img.shape[0])
            box_w = int(((x2 - x1) / unpad_w) * img.shape[1])
            y1 = int(((y1 - pad_y // 2) / unpad_h) * img.shape[0])
            x1 = int(((x1 - pad_x // 2) / unpad_w) * img.shape[1])

            color = colors[int(obj_id) % len(colors)]
            color = [i * 255 for i in color]
            cls = classes[int(cls_pred)]
            cv2.rectangle(frame, (x1, y1), (x1+box_w, y1+box_h), color, 4)
            cv2.rectangle(frame, (x1, y1-35), (x1+len(cls)*19+60, y1), color, -1)
            cv2.putText(frame, cls + "-" + str(int(obj_id)), (x1, y1 - 10), cv2.FON

    fig=figure(figsize=(12, 8))
    title("Video Stream")
    imshow(frame)
    show()
    #clear_output(wait=True)
```

```
Populating the interactive namespace from numpy and matplotlib
ii 0
torch.Size([8, 7])
(8, 6)
```


```
ii 1
torch.Size([8, 7])
(8, 6)
```

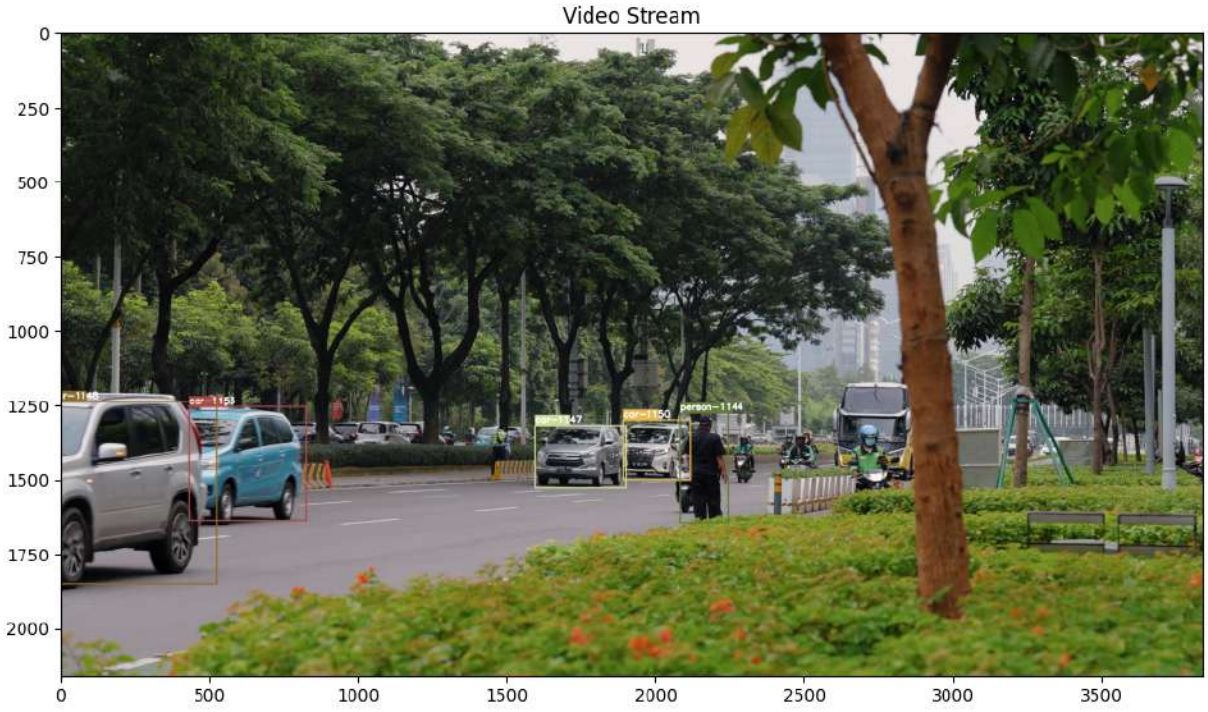

```
ii 2
torch.Size([9, 7])
(9, 6)
```




```
ii 3
torch.Size([6, 7])
(5, 6)
```

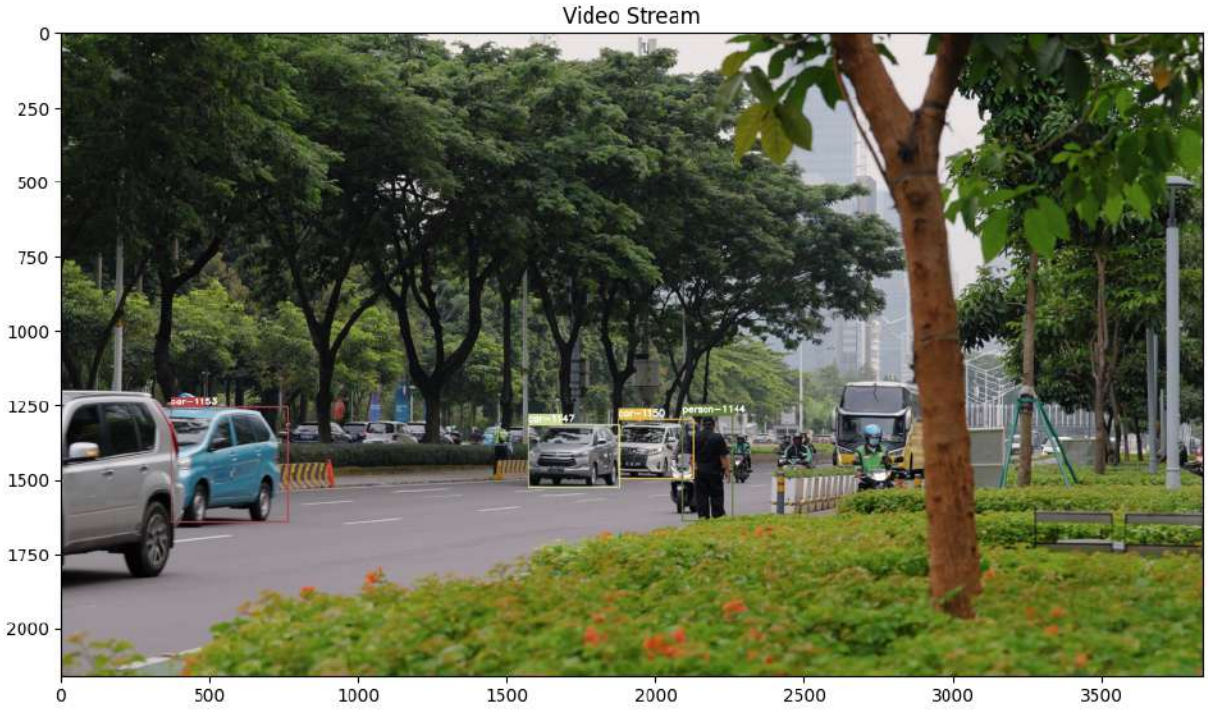

```
ii 4
torch.Size([6, 7])
(5, 6)
```


```
ii 5
torch.Size([7, 7])
(5, 6)
```




```
ii 6
torch.Size([6, 7])
(4, 6)
```


```
ii 7
torch.Size([7, 7])
(4, 6)
```




```
ii 8
torch.Size([7, 7])
(5, 6)
```


```
ii 9
torch.Size([5, 7])
(4, 6)
```


```
ii 10
torch.Size([5, 7])
(4, 6)
```


```
ii 11
torch.Size([5, 7])
(5, 6)
```


```
ii 12
torch.Size([8, 7])
(5, 6)
```


```
ii 13
torch.Size([10, 7])
(5, 6)
```


```
ii 14
torch.Size([7, 7])
(3, 6)
```


```
ii 15
torch.Size([9, 7])
(5, 6)
```


```
ii 16
torch.Size([9, 7])
(6, 6)
```


```
ii 17
torch.Size([6, 7])
(4, 6)
```


```
ii 18
torch.Size([6, 7])
(6, 6)
```

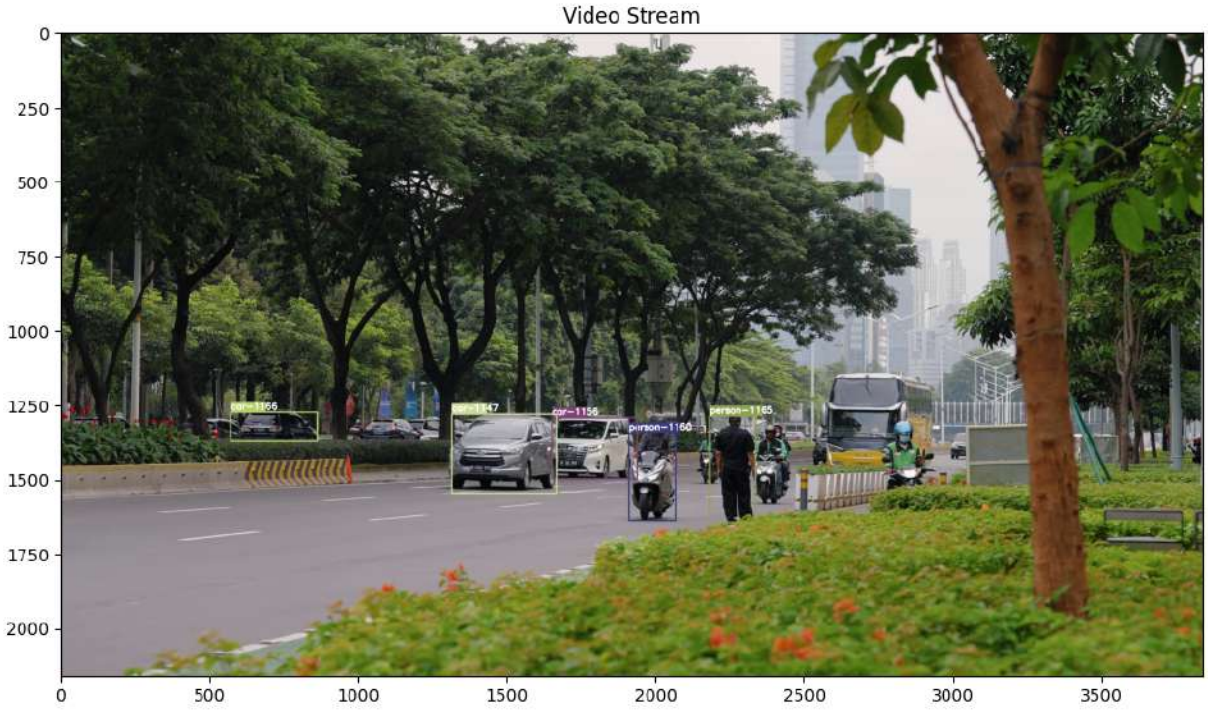

```
ii 19
torch.Size([9, 7])
(6, 6)
```


```
ii 20
torch.Size([9, 7])
(5, 6)
```


```
ii 21
torch.Size([10, 7])
(4, 6)
```


```
ii 22
torch.Size([7, 7])
(4, 6)
```


```
ii 23
torch.Size([7, 7])
(5, 6)
```




```
ii 24
torch.Size([7, 7])
(5, 6)
```


```
ii 25
torch.Size([8, 7])
(4, 6)
```


```
ii 26
torch.Size([11, 7])
(5, 6)
```

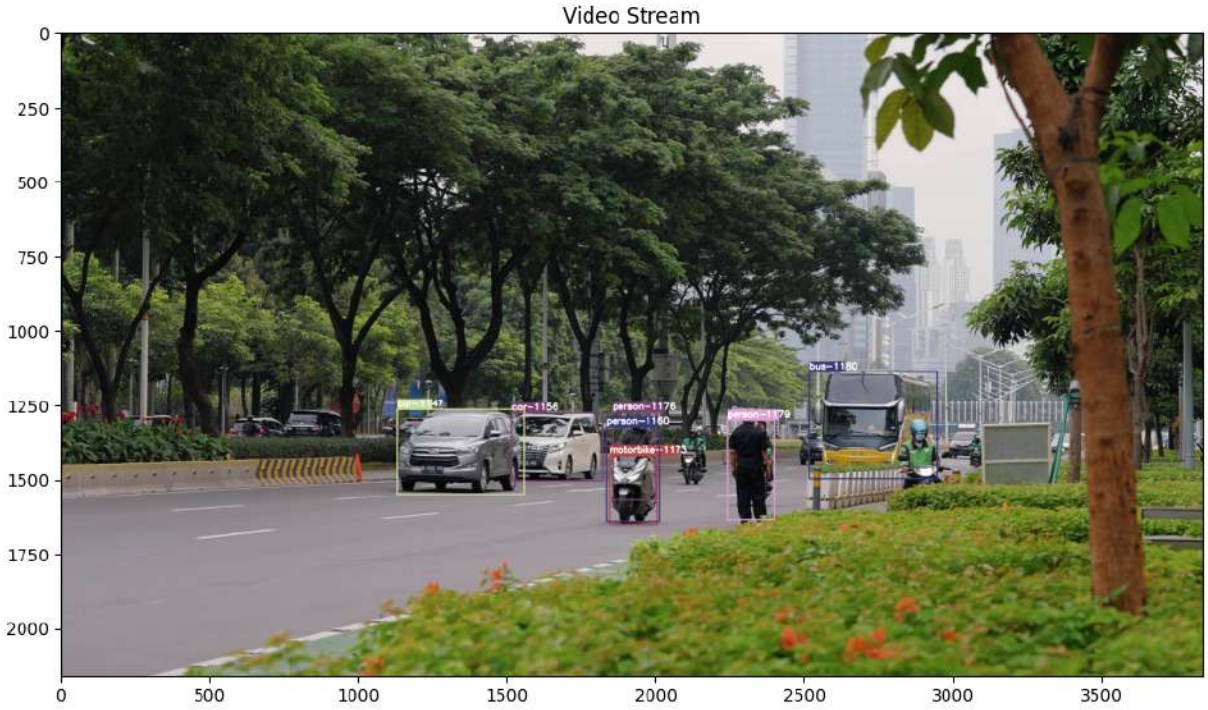

```
ii 27
torch.Size([10, 7])
(6, 6)
```


```
ii 28
torch.Size([10, 7])
(7, 6)
```


```
ii 29
torch.Size([7, 7])
(6, 6)
```


```
ii 30
torch.Size([12, 7])
(7, 6)
```


```
ii 31
torch.Size([10, 7])
(7, 6)
```




```
ii 32
torch.Size([11, 7])
(7, 6)
```


```
ii 33
torch.Size([11, 7])
(8, 6)
```


```
ii 34
torch.Size([11, 7])
(9, 6)
```

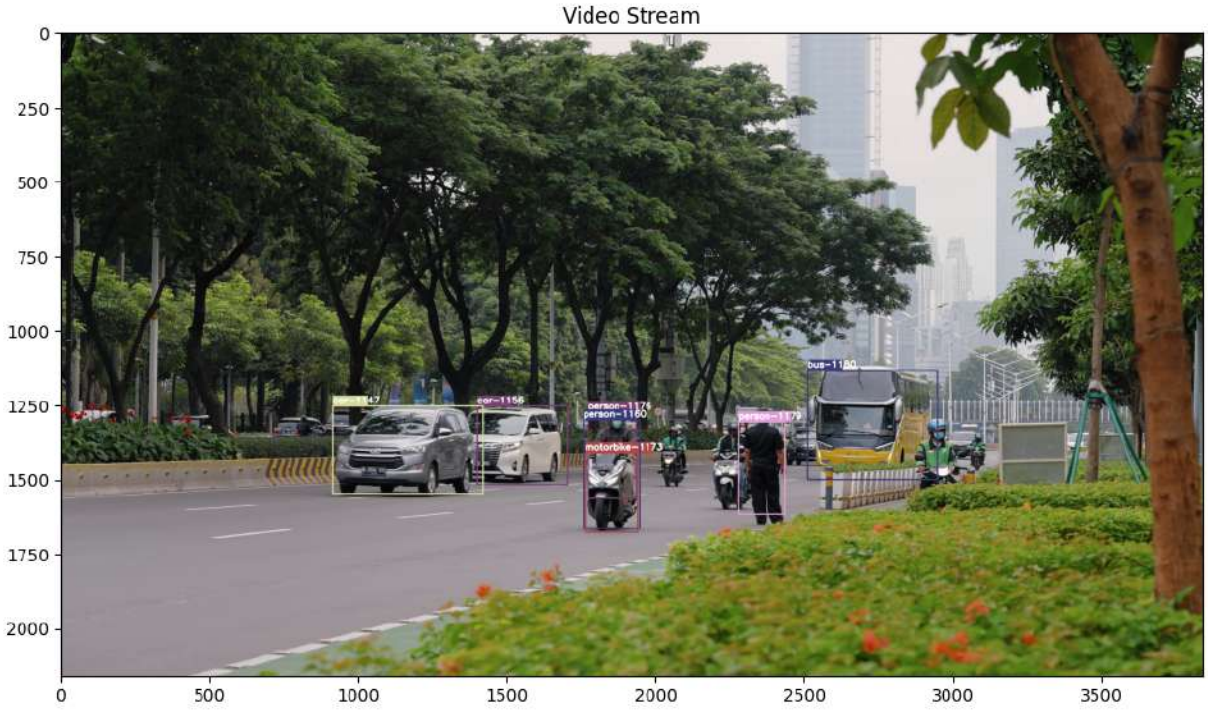

```
ii 35
torch.Size([11, 7])
(9, 6)
```


```
ii 36
torch.Size([10, 7])
(10, 6)
```


```
ii 37
torch.Size([10, 7])
(9, 6)
```

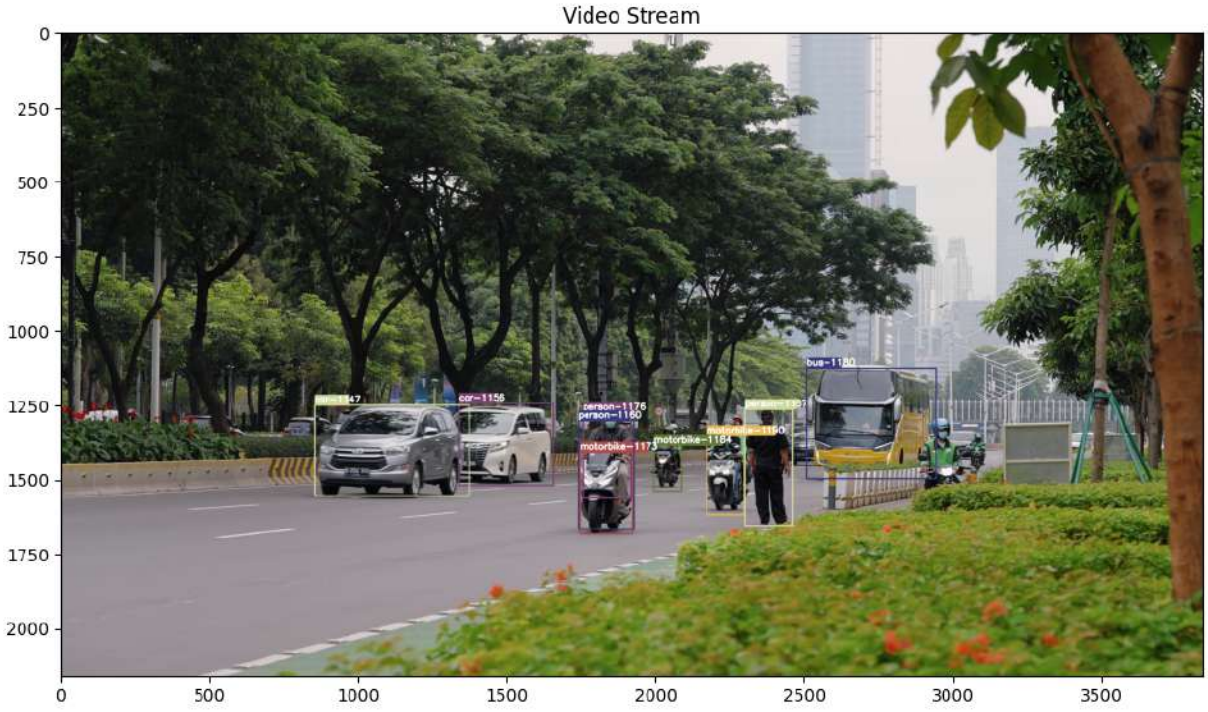

```
ii 38
torch.Size([10, 7])
(7, 6)
```


```
ii 39
torch.Size([14, 7])
(7, 6)
```



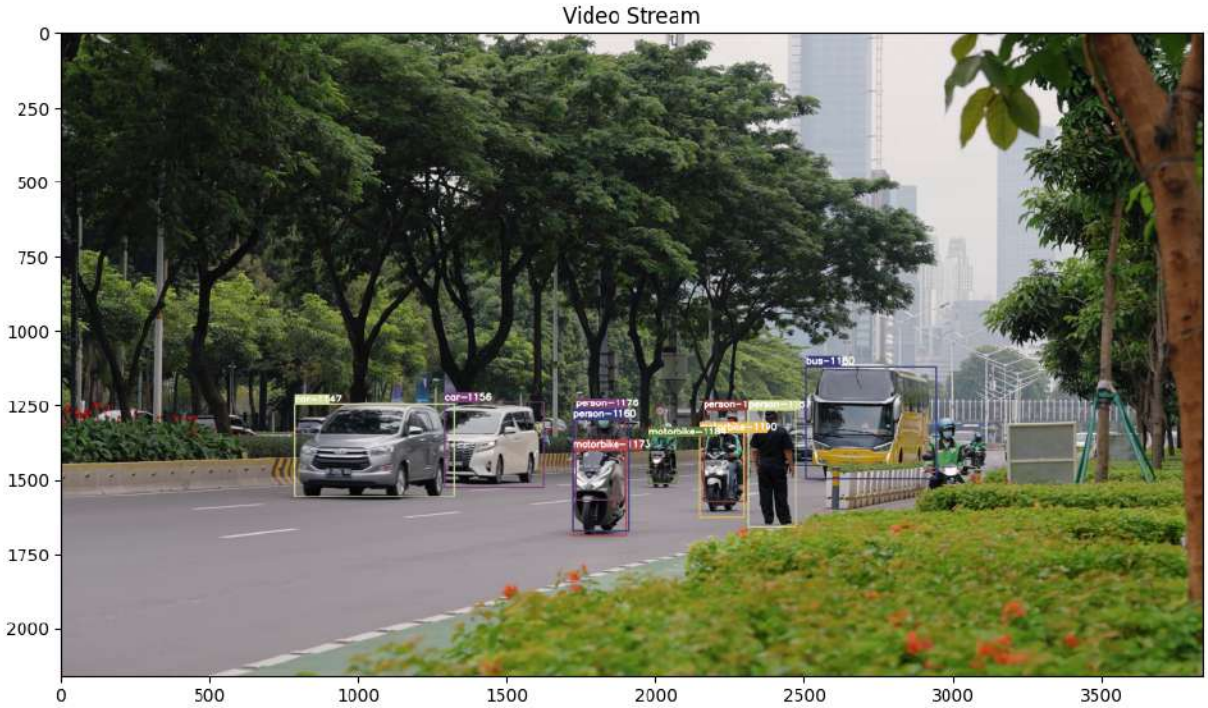

```
ii 40
torch.Size([11, 7])
(8, 6)
```


```
ii 41
torch.Size([11, 7])
(9, 6)
```




```
ii 42
torch.Size([11, 7])
(10, 6)
```


```
ii 43
torch.Size([12, 7])
(10, 6)
```




```
ii 44
torch.Size([14, 7])
(10, 6)
```


```
ii 45
torch.Size([11, 7])
(8, 6)
```


```
ii 46
torch.Size([11, 7])
(8, 6)
```


```
ii 47
torch.Size([11, 7])
(9, 6)
```


```
ii 48
torch.Size([10, 7])
(9, 6)
```


```
ii 49
torch.Size([11, 7])
(10, 6)
```

Εδώ παρατηρούμε κάτι ενδιαφέρον. Ναι μεν ο αλγόριθμος μας επιστρέφει περισσότερες προβλέψεις (bounding boxes), αλλά αρκετές φορές είναι για το ίδιο αντικείμενο (πχ τα αυτοκίνητα στα αριστερά αναγνωρίζονται 2 φορές, και ο άνθρωπος στη μέση της εικόνας το ίδιο). Αυτό γίνεται διότι το threshold του IOU των bounding boxes για να αποφασίσει ο αλγόριθμος ότι είναι το ίδιο αντικείμενο είναι πολύ μεγάλο, και αρα αρκετά bounding boxes τα οποία έχουν σημαντικό ποσοστό IOU αποφαίνονται ως διαφορετικό αντικείμενο, ενώ προκειται για το ίδιο.