# Multiple object tracking and re-identification with DeepSORT

In [11]:
```
!pip install ultralytics
!pip install deep-sort-realtime
```

### Class to write videos

In [17]:
```
import os, cv2

class Video:

    def __init__(self, codec:str='mp4v', fps:int=3, shape:tuple=(854, 480), overwrite=False):
        self.codec = codec; self.fps = fps; self.shape = shape
        self.overwrite = overwrite

    def writer(self, path):
        if not self.overwrite and os.path.exists(path):
            print(f'ANNOTATE VIDEO TIMESTAMP FAILED. FILE ALREADY EXISTS · FILE-PATH: {path}')
            return False
        return cv2.VideoWriter(path, cv2.VideoWriter_fourcc(*self.codec), self.fps, self.shape)


#### OPEN VIDEO FILE WRITER · Method #1
# video_path = "output.mp4"
# fps, shape = get_video_metadata(video_path, transform=None)
# shape = (shape[1], shape[0]) # witdth, height
# overwrite = True
# video = Video(fps=fps, shape=shape, overwrite=overwrite)
# writer = video.writer(path=video_path)

# writer.release(); cv2.destroyAllWindows()
```

### Function to get basic metadata from video file:

    fps: frames per second (FPS) of video file
    shape: shape of first frame

In [20]:
```
def get_video_metadata(video_path, transform=None):
    cap = cv2.VideoCapture(video_path)
    fps = cap.get(cv2.CAP_PROP_FPS) # get the fps
    _, frame = cap.read() # read the first frame
    if transform is not None: # custom transformation
        frame = transform(frame)
    shape = frame.shape; # get the shape
    cap.release(); cv2.destroyAllWindows()
    return fps, shape
```

### Function to open video file writer

In [1]:
```
import cv2

def create_video_writer(video_cap, output_filename):

    # grab the width, height, and fps of the frames in the video stream.
    frame_width = int(video_cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    frame_height = int(video_cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
    fps = int(video_cap.get(cv2.CAP_PROP_FPS))

    # initialize the FourCC and a video writer object
    fourcc = cv2.VideoWriter_fourcc(*'MP4V')
    writer = cv2.VideoWriter(output_filename, fourcc, fps,
                             (frame_width, frame_height))

    return writer
```

### Function to write demo video of multiple object tracking and re-identification with DeepSORT

In [42]:
```
# Code from: https://www.thepythoncode.com/article/real-time-object-tracking-with-yolov8-opencv

import datetime
from IPython.display import clear_output as co
from ultralytics import YOLO
import cv2
from deep_sort_realtime.deepsort_tracker import DeepSort

def tracking_reid_demo(video_path, to_video_path, CONFIDENCE_THRESHOLD=0.3, model=YOLO("yolov8n.pt"), tracker=DeepSort(max_

    GREEN = (0, 255, 0)
```

```python
WHITE = (255, 255, 255)

# initialize the video capture object
video_cap = cv2.VideoCapture(video_path)
total_frames = int(video_cap.get(cv2.CAP_PROP_FRAME_COUNT))
# initialize the video writer object
writer = create_video_writer(video_cap, to_video_path) # None


i = 0
while True:
    i += 1
    start = datetime.datetime.now()

    ret, frame = video_cap.read()

    if not ret:
        break

    # run the YOLO model on the frame
    detections = model(frame)[0]

    # initialize the list of bounding boxes and confidences
    results = []

    ######################################
    # DETECTION
    ######################################

    # loop over the detections
    for data in detections.boxes.data.tolist():
        # extract the confidence (i.e., probability) associated with the prediction
        confidence = data[4]

        # filter out weak detections by ensuring the
        # confidence is greater than the minimum confidence
        if float(confidence) < CONFIDENCE_THRESHOLD:
            continue

        # if the confidence is greater than the minimum confidence,
        # get the bounding box and the class id
        xmin, ymin, xmax, ymax = int(data[0]), int(data[1]), int(data[2]), int(data[3])
        class_id = int(data[5])
        # add the bounding box (x, y, w, h), confidence and class id to the results list
        results.append([[xmin, ymin, xmax - xmin, ymax - ymin], confidence, class_id])

    ######################################
    # TRACKING
    ######################################

    # update the tracker with the new detections
    tracks = tracker.update_tracks(results, frame=frame)
    # loop over the tracks
    for track in tracks:
        # if the track is not confirmed, ignore it
        if not track.is_confirmed():
            continue

        # get the track id and the bounding box
        track_id = track.track_id
        ltrb = track.to_ltrb()

        xmin, ymin, xmax, ymax = int(ltrb[0]), int(
            ltrb[1]), int(ltrb[2]), int(ltrb[3])
        # draw the bounding box and the track id
        cv2.rectangle(frame, (xmin, ymin), (xmax, ymax), GREEN, 2)
        cv2.rectangle(frame, (xmin, ymin - 40), (xmin + 40, ymin), GREEN, -1)
        cv2.putText(frame, str(track_id), (xmin + 5, ymin - 8),
                    cv2.FONT_HERSHEY_SIMPLEX, 1, WHITE, 2)

    # end time to compute the fps
    end = datetime.datetime.now()
    # show the time it took to process 1 frame
    co(True); print(f"Time to process frame {i}/{total_frames}: {(end - start).total_seconds() * 1000:.0f} milliseconds
    # calculate the frame per second and draw it on the frame
    fps = f"FPS: {1 / (end - start).total_seconds():.2f}"
    cv2.putText(frame, fps, (50, 50),
                cv2.FONT_HERSHEY_SIMPLEX, 2, (0, 0, 255), 8)

    # show the frame to our screen
    if show:
        cv2.imshow("Frame", frame)
    if writer is not None:
        writer.write(frame)
    if cv2.waitKey(1) == ord("q"):
        break
```

```
    video_cap.release()
    writer.release()
    cv2.destroyAllWindows()
```

## Multiple object tracking and re-identification with DeepSORT

In [ ]:
```
# system paths
folder = '../Dados/Demos/smartphone-video-samples/'
to_folder = '../Dados/Demos/tracking-id/'
file_name = 'VID_20230515_125317.mp4'

video_path = folder + file_name
to_video_path = to_folder + file_name

# load the pre-trained YOLOv8n model
model = YOLO("yolov8n.pt")
tracker = DeepSort(max_age=50)

tracking_reid_demo(
    video_path, to_video_path,
    CONFIDENCE_THRESHOLD=0.3,
    model=YOLO("yolov8n.pt"),
    tracker=DeepSort(max_age=50),
    show=False
)
```

### Write demo video for videos in folder

In [44]:
```
folder = '../Dados/Demos/smartphone-video-samples/'
to_folder = '../Dados/Demos/tracking-id/'

for file_name in os.listdir(data_path):

    video_path = data_path + file_name
    to_video_path = to_data_path + file_name

    tracking_reid_demo(
        video_path, to_video_path,
        CONFIDENCE_THRESHOLD=0.6,
        model=YOLO("yolov8n.pt"),
        tracker=DeepSort(max_age=50),
        show=False
    )
```

```
Time to process frame 293/293: 766 milliseconds
```