In [13]:
```
# Cell 0
# Install YOLOv8 (first run only) and import libs
try:
    from ultralytics import YOLO
except ModuleNotFoundError:
    !pip install -q ultralytics
    from ultralytics import YOLO

import cv2, itertools, pathlib, numpy as np
```

In [14]:
```
# Cell 1 – config
VIDEO    = pathlib.Path("assets/Traffic_Laramie_1.mp4")     # swap to _2 later
OUTPATH = VIDEO.with_name(VIDEO.stem + "_yolo_detect.mp4")
SAVE          = True
CAR_CLASSES  = {2, 3, 5, 7}     # COCO ids → car, motorcycle, bus, truck
CONF_THR      = 0.40            # YOLO confidence threshold
MAX_DIST      = 60              # tracker matching radius (px)
TTL_FRAMES    = 20              # frames to keep a lost track
```

In [15]:
```
# Cell 2
# 5 MB Nano weights (downloads once)
model = YOLO("yolov8n.pt")
```

In [16]:
```
# Cell 3
cap   = cv2.VideoCapture(str(VIDEO))
fps   = cap.get(cv2.CAP_PROP_FPS) or 25
W     = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
H     = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

four  = cv2.VideoWriter_fourcc(*"mp4v")
vw    = cv2.VideoWriter(str(OUTPATH), four, fps, (W, H)) if SAVE else None
```

In [17]:
```
# Cell 4
nextID   = itertools.count()          # id generator
tracks   = {}                         # id → (centroid, ttl)
```

In [18]:
```
# Cell 5
while True:
    ok, frame = cap.read()
    if not ok:
        break

    # ---------------------------------------------------------------
    # ---  A) foreground mask ------------------------------
    if 'bg' not in globals():
        bg = cv2.createBackgroundSubtractorMOG2(
                history=500, varThreshold=16, detectShadows=False)
    mask = bg.apply(frame, learningRate=0)          # 255 = motion, 0 = static
    fg   = cv2.medianBlur(mask, 5)                  # quick speckle cleanup

    # 1. YOLO inference   +  filter on motion ratio ------------------
    detections = []
    MOTION_FRAC = 0.03          # 3 % pixels inside the box must be "moving"
    res = model(frame, verbose=False)[0]
    for box, cls, conf in zip(res.boxes.xyxy.cpu().numpy(),
                              res.boxes.cls.cpu().numpy(),
                              res.boxes.conf.cpu().numpy()):
        if int(cls) not in CAR_CLASSES or conf < CONF_THR:
            continue

        x1,y1,x2,y2 = box.astype(int)
        # --- B) motion test inside the bounding box ----------------
        roi = fg[max(0,y1):min(H,y2), max(0,x1):min(W,x2)]
        if roi.size == 0:               # sanity
            continue
        moving_frac = (roi > 0).mean()  # ratio 0-1
        if moving_frac < MOTION_FRAC:   # parked → skip
            continue

        cx, cy = (x1+x2)//2, (y1+y2)//2
        detections.append(((cx,cy), (x1,y1,x2-x1,y2-y1)))


    # -- 2. Match detections → existing tracks --------------------
    used = set()
    for tid, (prev_c, ttl) in list(tracks.items()):
        if detections:
            dists = [np.hypot(cx-prev_c[0], cy-prev_c[1])
                     for (cx,cy),_ in detections]
            idx, dist = int(np.argmin(dists)), min(dists)
            if dist < MAX_DIST:
                (cx,cy), bbox = detections[idx]
                tracks[tid]   = ((cx,cy), TTL_FRAMES)
                used.add(idx)
                x,y,w,h = bbox
                cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),2)
                cv2.putText(frame,f"#{tid}",(x,y-6),
                            cv2.FONT_HERSHEY_SIMPLEX,0.5,(0,255,0),1)
                continue
        # decay TTL if unmatched
        ttl -= 1
        if ttl <= 0:
            tracks.pop(tid)
        else:
            tracks[tid] = (prev_c, ttl)

    # -- 3. New tracks for unmatched detections --------------------
    for i,(centroid,bbox) in enumerate(detections):
        if i in used: continue
        tid = next(nextID)
        tracks[tid] = (centroid, TTL_FRAMES)
        x,y,w,h = bbox
        cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),2)
        cv2.putText(frame,f"#{tid}",(x,y-6),
                    cv2.FONT_HERSHEY_SIMPLEX,0.5,(0,255,0),1)

    # -- 4. Display / write --------------------------------
    if SAVE: vw.write(frame)
    cv2.imshow("YOLO detect + track", frame)
    if cv2.waitKey(1) & 0xFF == 27:   # ESC
        break

cap.release()
if SAVE: vw.release()
cv2.destroyAllWindows()
print("Output saved to:", OUTPATH if SAVE else "<not saved>")
```

```
Output saved to: assets\Traffic_Laramie_1_yolo_detect.mp4
```