Vehicle and Pedestrian Segmentation and Tracking using YOLO-Seg + Labellerr

1. Introduction

This project demonstrates an end-to-end instance segmentation and tracking workflow for vehicles and pedestrians. Unlike bounding-box detection, segmentation requires polygon-based annotations that precisely outline object boundaries.

The workflow:

- Dataset creation & annotation using Labellerr (polygon segmentation).
- Model training with YOLOv8-Seg.
- Integration with ByteTrack for tracking IDs.
- Deployment as a Streamlit web app.

2. Dataset Creation with Labellerr

2.1 Raw Dataset

- I collected road traffic videos including vehicles and pedestrians in urban settings.
- Frames were extracted and uploaded to Labellerr.

2.2 Annotation Process

- First attempt – AI-assisted polygon labeling
    - I tried Labellerr's auto-labeling tool to generate segmentation masks.
    - Problem: Masks were often inaccurate for occluded vehicles and dense pedestrian clusters.
    - Many polygons overlapped or failed to close properly.
- Switch to manual polygon annotation
    - I shifted to drawing polygons manually for vehicles and pedestrians.
    - This produced much cleaner labels, though more time-consuming.
    - Export format: YOLO-Seg format (text files with polygon points).

2.3 Problems Faced

- Export failure:
    - Labellerr Python SDK gave Internal Server Error during export.
    - Even the UI export sometimes showed *failed*.

- Resolution:
    - Exported data in smaller batches.
    - As backup, manually downloaded YOLO-Seg formatted annotations from the Labellerr dashboard.

3. Model Training

3.1 Tools Used

- YOLOv8n-seg (Ultralytics).
- PyTorch 2.2.2 / Torchvision 0.17.2.
- OpenCV for preprocessing.
- Supervision for polygon rendering.

3.2 Training Setup

- Configured data.yaml with classes: vehicle, pedestrian.
- Model: YOLOv8n-seg.
- Input size: 640x640.
- Epochs: 100.

3.3 Observations

- Polygon annotations improved segmentation quality, especially for pedestrians in groups.
- Pretrained YOLO weights accelerated convergence.

4. Tracking Integration

- Used ByteTrack for assigning consistent IDs across frames.
- Combined with YOLO-Seg predictions: each segmented mask retained its unique ID over time.
- Vehicles and pedestrians were highlighted with polygons instead of boxes, with labels like ID 12 (vehicle) or ID 5 (pedestrian).

5. Deployment with Streamlit

- Built a web app where users can upload a video.
- The app outputs a video with polygon masks + tracking IDs.

- Challenges during deployment:
    - Streamlit Cloud defaults to Python 3.13 (Torch incompatible).
    - Resolved by adding runtime.txt → python-3.11.6.
    - Simplified requirements.txt to avoid dependency hangs.

6. Problems Faced & Resolutions

Problem	Impact	Resolution
AI-based polygon labeling inaccurate	Mis-segmented vehicles/pedestrians	Switched to manual polygon annotation
Export failed (Labellerr SDK/UI)	Couldn't retrieve train data	Exported in smaller batches / manual export
Streamlit Cloud Python 3.13	Torch not installable	Forced Python 3.11 via runtime.txt
Build stuck on numpy/pandas	Delayed deployment	Relaxed version pins
Large video blocked git push	Couldn't commit repo	Removed file / updated .gitignore

7. Model Results & Metrics

- mAP@0.5 (vehicles): ~0.82
- mAP@0.5 (pedestrians): ~0.74
- Mask Quality: Clear polygon boundaries preserved.
- Tracking: ~85% ID persistence across frames.
- Example output:
    - Vehicles = blue polygons with IDs.
    - Pedestrians = green polygons with IDs.

8. Guide for Future Fellows

1. Data Preparation
    - Collect videos → extract frames.
    - Upload to Labellerr.

    - Prefer manual polygon annotation for segmentation tasks.
2. Model Training
    - Use YOLOv8n-seg for lightweight deployment.
    - Configure data.yaml with proper classes.
    - Train on a balanced dataset of vehicles & pedestrians.
3. Tracking
    - Combine YOLO-Seg with ByteTrack.
    - Maintain IDs for real-world video streams.
4. Deployment
    - Use Streamlit for user interface.
    - Add runtime.txt with python-3.11.6.
    - Keep requirements.txt minimal and wheel-compatible.
5. Evaluation
    - Track mAP, segmentation mask accuracy, ID switches, FPS.

9. Conclusion

This project demonstrates a complete end-to-end polygon segmentation + tracking workflow:

- Polygon annotation with Labellerr.
- Segmentation model training with YOLO-Seg.
- Tracking with ByteTrack.
- Deployment as a Streamlit app