# Polygon Annotation Automation on Driving Video Dataset

## Dataset Used & Reasoning

The dataset used is the "Driving Video with Object Tracking" from Kaggle, which is listed as the preferred dataset for this task.

- It includes real-world, high-resolution driving footage (1280×720).  
- Contains >10 objects across multiple classes in urban road conditions.  
- The object classes are visually distinct, diverse, and relevant for computer vision use cases like autonomous driving and smart city applications.

This makes it an ideal candidate for testing polygon annotation automation pipelines.

## Chosen Objects & Why

The following object classes were selected from the video:

- Car  
- Motorcycle  
- Truck  
- Bus  
- Person (re-labeled as Pedestrian for clarity)

These classes were chosen based on:  
- Frequency and visibility across frames.  
- Importance in safety-critical scenarios like road navigation.  
- Clear visual distinction for polygon-level segmentation.

This variety ensures a rich annotation challenge and meaningful insights when evaluating detection + segmentation quality.

## Technical Approach

The pipeline automates polygon annotation in 5 main stages:

1. Preprocessing:  
- Automatically locates the video file in the Kaggle dataset directory.  
- Corrects video orientation using `cv2.rotate()` (important due to sideways camera feeds).  
- Samples every 5th frame for resource efficiency.

2. Object Detection:  
- Uses `YOLOv8n` from Ultralytics to detect bounding boxes for the selected classes.  
- Applies confidence thresholding and class filtering.

3. Polygon Segmentation:  
- Each detected bounding box is passed to Meta AI’s `Segment Anything Model (SAM)` to generate precise masks.  
- These masks are converted into vectorized polygons using `supervision`.

4. Annotation & Output:  
- Annotated frames are saved and stitched into an output `.mp4` video.  
- Polygon data is saved frame-wise in a structured `output\_polygons.json`.

5. Post-Processing:  
- JSON includes object class, index, and polygon coordinates.  
- Temporary frames are cleaned up to optimize space.

## Trade-offs Considered

Trade-off | Decision  
  
Speed vs. Accuracy: Used `stride=5` to reduce processing time while maintaining visual consistency.  
Model Size vs. Flexibility: Selected lightweight `YOLOv8n` to ensure compatibility with both CPU and GPU environments.  
Segmentation Quality vs. Simplicity: Used SAM without user intervention to maximize automation.  
Rotation Fix vs. Manual Review: Added auto-rotation correction to eliminate the need for human adjustment.

## How This Helps Non-Expert Users

This pipeline is designed for full automation — ideal for non-technical users or annotation ops teams:

- No manual frame extraction or video editing required.  
- No prior ML setup — dependencies are auto-installed.  
- Auto-detection of video files and rotation correction simplify input handling.  
- Outputs include an annotated video and ready-to-use JSON format for integration into any downstream ML pipeline.

With minimal effort, users get rich polygon annotations suitable for model training, QA, or dataset curation.

