NTCS
NEXTGEN TRAFFIC CHALLAN SYSTEM

Sidhaarth Shree

INTRODUCTION

Project Overview:

- An AI-based traffic monitoring system that detects and tracks vehicles in real time.
- Uses computer vision and machine learning to estimate vehicle speed and identify violations automatically.
- Combines YOLO detection, tracking, perspective transformation, and LLM-based calibration for accurate, sensor-free speed measurement.

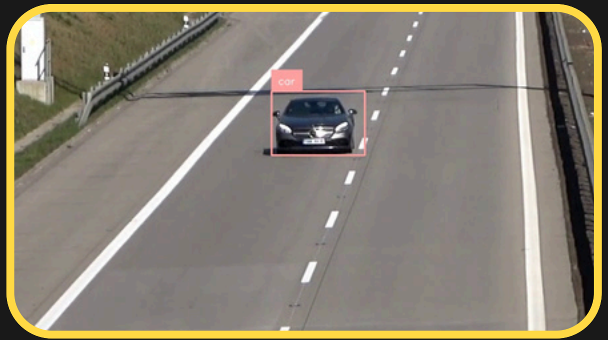
Why this project?

- Traditional speed detection needs manual calibration & sensors.
- We built a fully automated ML pipeline that detects, tracks, and measures vehicle speed using computer vision + LLMs.
- Focus: Convert pixel motion → real-world speed through calibration and regression.

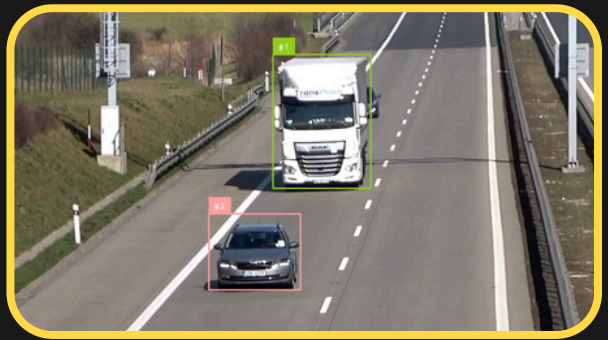
End-to-End ML Pipeline

Road Segmentation

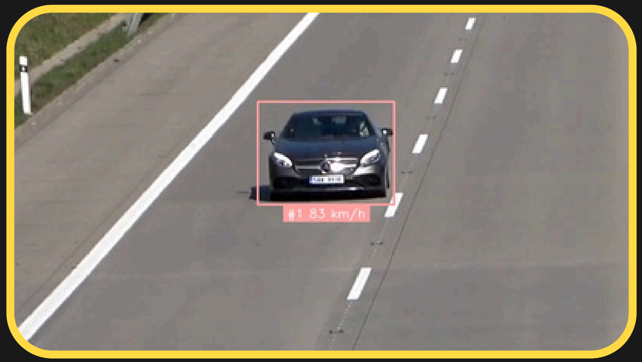
Object Detection



Vehicle Tracking

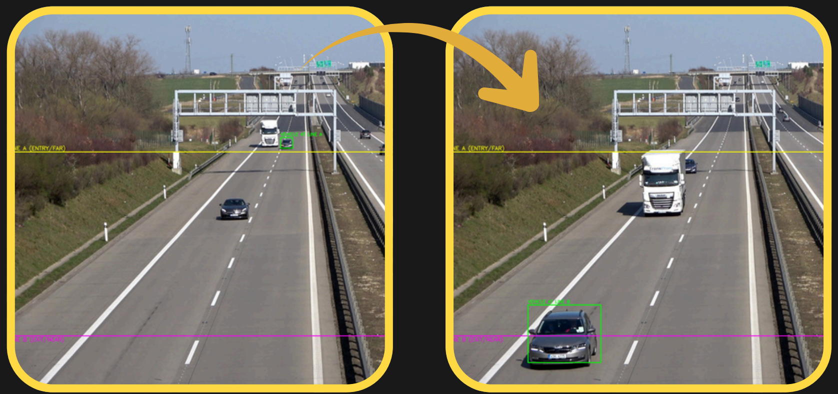


Speed Estimation



Perspective Transform

LLM-Based Calibration



ROAD SEGMENTATION (DEEPLABV3+)

- **Goal:** Isolate the drivable region where speed estimation applies.
- **Model:** DeepLabV3+ (ResNet-50 backbone)
  - Pixel-wise segmentation → “road vs non-road”
  - Pretrained on Cityscapes dataset (urban driving)
- **How it helps:**
  - Defines the area for motion tracking.
  - Removes noise like sidewalks, trees, or background vehicles.
  - Provides geometric boundaries for calibration later.

OBJECT DETECTION (YOLOV8)

GOAL

Detect all vehicles entering the segmented region.

MODEL

YOLOv8l (large variant)

- Detects cars, buses, trucks, motorcycles
- Fast inference (~15 ms/frame on GPU)

OUTPUT

Bounding boxes with class & confidence.

PURPOSE IN PIPELINE

Marks pixel coordinates of vehicles — the basis for tracking and speed computation.

MULTI-OBJECT TRACKING

Goal: Track each vehicle across frames to know how far it moved.

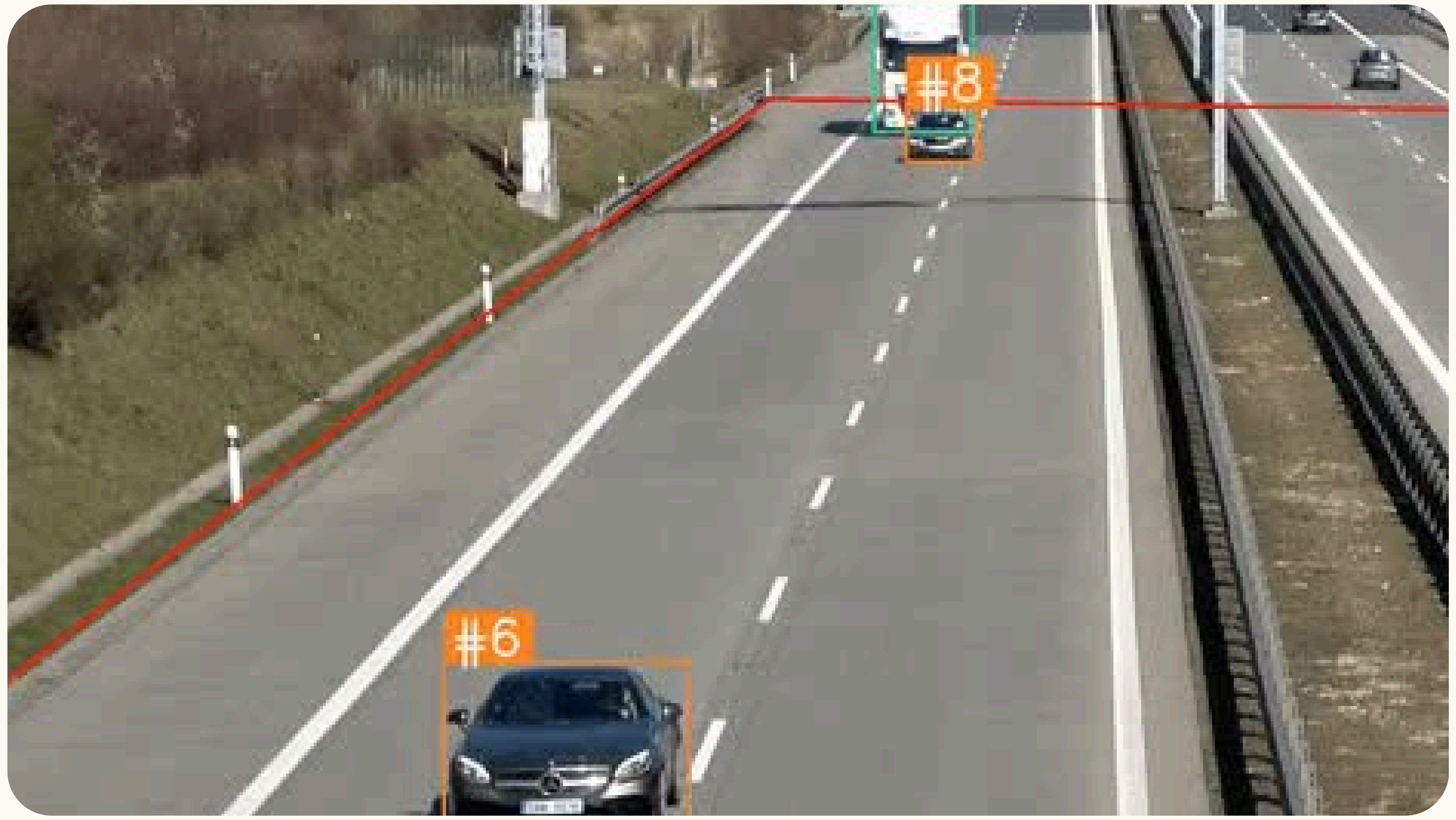
Model: ByteTrack (Kalman Filter + IoU matching)

How it works:

- Predicts vehicle position in next frame.
- Matches detections even under partial occlusion.
- Maintains consistent track IDs for each vehicle.

Result: A trajectory of pixel positions (x, y) over time.
These pixel motions form the raw data for speed estimation.

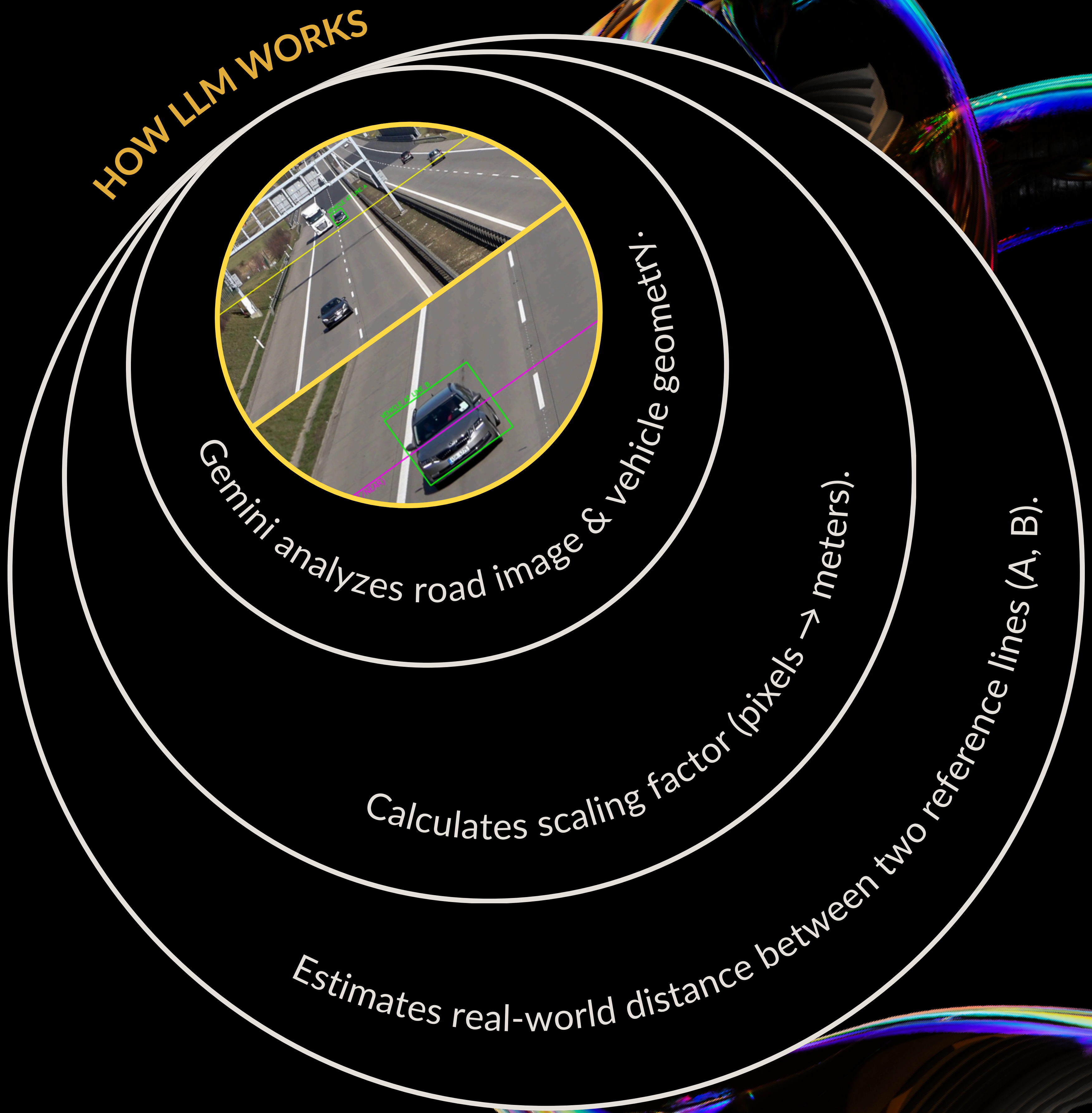
DEMO



LLM-BASED CAMERA CALIBRATION (GEMINI 2.0 FLASH)

Solution: Use LLM reasoning for automated perspective calibration.

Impact:
This calibration converts perspective-distorted pixel distances into accurate meter-scale measurements, enabling real speed estimation.



PERSPECTIVE TRANSFORM

Purpose

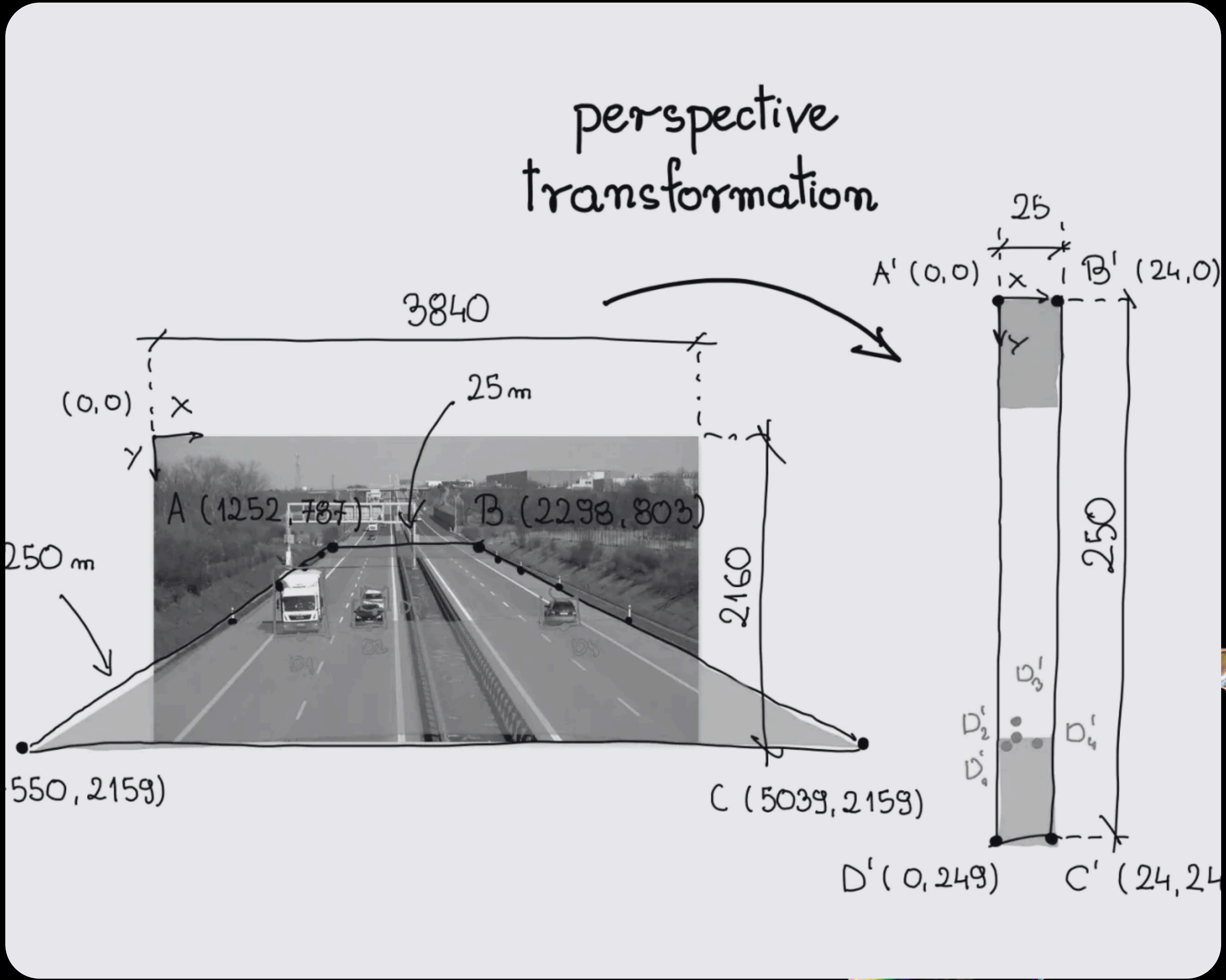
- Convert the angled camera view into a top-down road view
- So pixel distances = real-world distances

Why it matters

- Vehicles farther away now appear at correct size
- Motion becomes straight and measurable
- Enables true speed estimation from video frames

How it works

- Take four reference points from calibration lines (AB & CD)
- Map them to their real-world positions (in meters)
- Warp the image using a homography transform
- Result: The road looks flat, and every movement on it is scale-accurate



SPEED ESTIMATION (REGRESSION + EMA)

Goal: Convert the tracked motion of a vehicle into accurate, real-world speed.

How it works:

- After perspective transform, each vehicle's position is measured in meters.
- The system observes how that position changes frame by frame over time.
- It fits a smooth trend line through all these points to find the average movement rate.
- Short-term fluctuations are then stabilized using an Exponential Moving Average (EMA) filter.

Why this approach?

- Uses data from many frames → reduces random noise.
- Handles detection errors and tracking jumps effectively.
- Produces smooth, realistic speed values instead of frame-to-frame spikes.

Speed accuracy around ±2.8 %

Visual stability improved by ~75 %

THANK YOU

for your time and attention