Car space occupancy

Niccolò Salvi, Alessio Villa, Beatrice Zani

Why?

- **Crucial for autonomous vehicles:** collision avoidance, path planning, parking assistance, traffic analysis.
- Need for robust, real-time vehicle **space estimation** where conventional methods fail (**nighttime**, tunnels, poorly lit areas).
- **Final goal**: use main extracted features and geometric properties to build a bounding box around the vehicle.

Our work

1. Data collection
2. Camera calibration
3. Feature extraction
4. Standard localization via homography
5. Nighttime localization from pair of images
6. Poor perspective localization using out-of-plane features
7. PnP-based vehicle pose estimation from key points

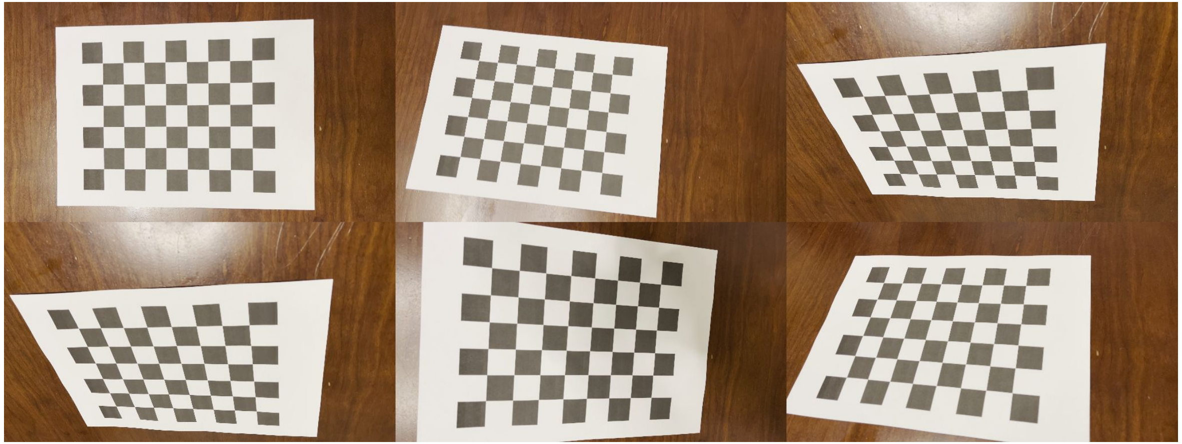
Data Collection

- **Recording device:** iPhone 13 (main wide-angle lens).
- **Camera setup:** static tripod, 2.5m high, 10m from road.
- **Video settings:** 4K resolution, 30 FPS.
- **Locked auto-exposure (AE) & auto-focus (AF):** to ensure consistent camera properties for accurate geometry

Camera calibration & CAD

Data collection & K: recording **99 diverse frames of a standard checkerboard pattern** and processing them with **OpenCV in Python** to determine the camera's unique **intrinsic parameters (K matrix)** and distortion coefficients.

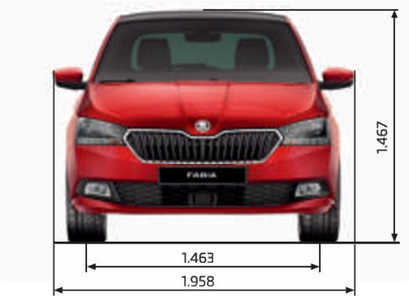
$$K = \begin{bmatrix} 3316 & 0 & 1912 \\ 0 & 3320 & 1072 \\ 0 & 0 & 1 \end{bmatrix}$$

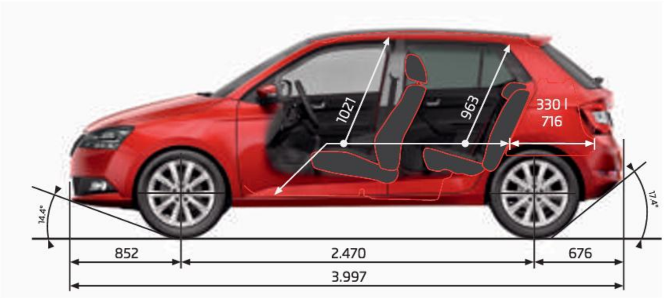


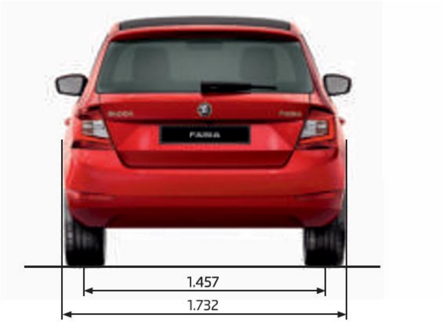
$$\begin{bmatrix} 2.43773846e-01 \\ -1.59544680e+00 \\ -1.15284213e-03 \\ 4.19886247e-04 \\ 3.56681588e+00 \end{bmatrix}$$

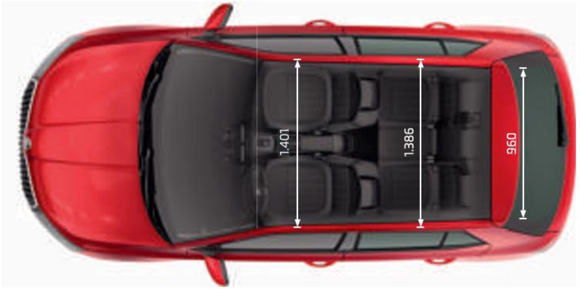
CAD Model

Reference vehicle model: we found a detailed **3D model of a Škoda Fabia**. We exploited this model and extracted exact physical dimensions to better estimate and visualize the 3D bounding box around the car.



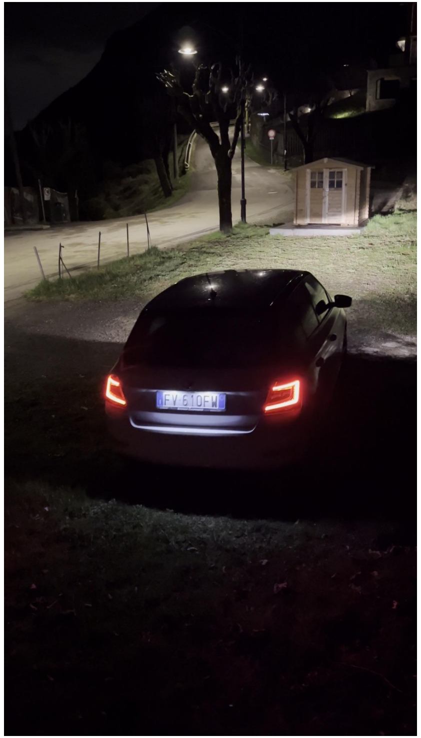


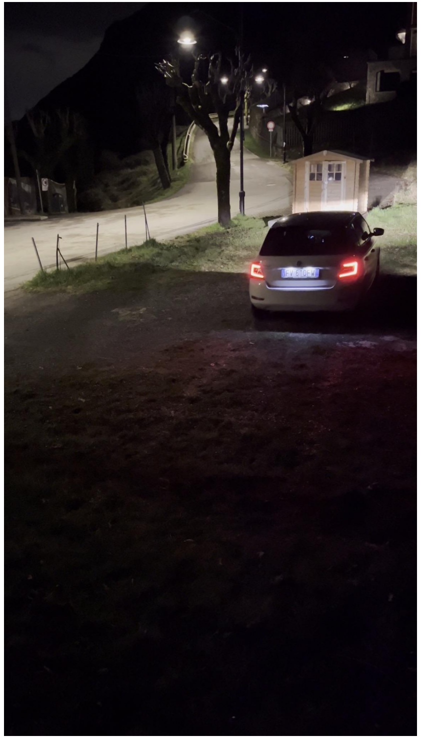




Frames extraction

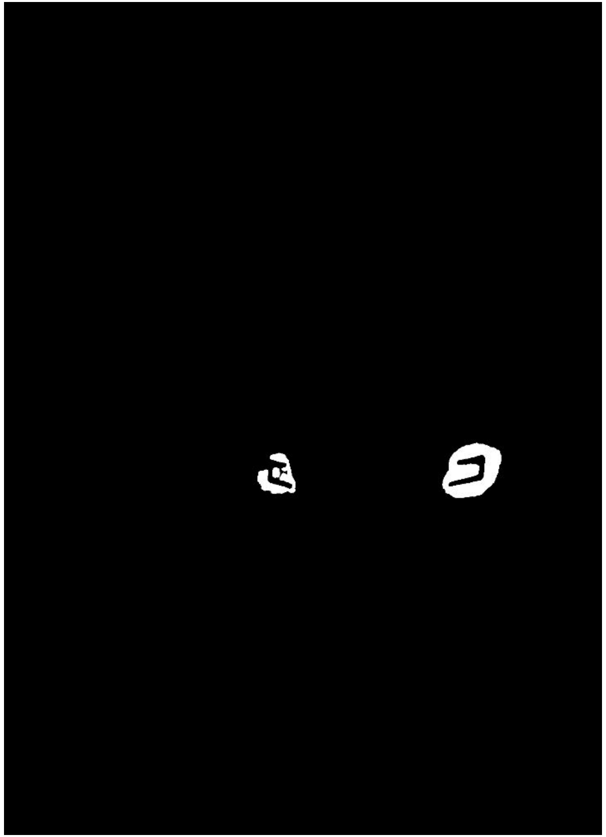
Counted frames present in our video and extracted a fixed number of frames.





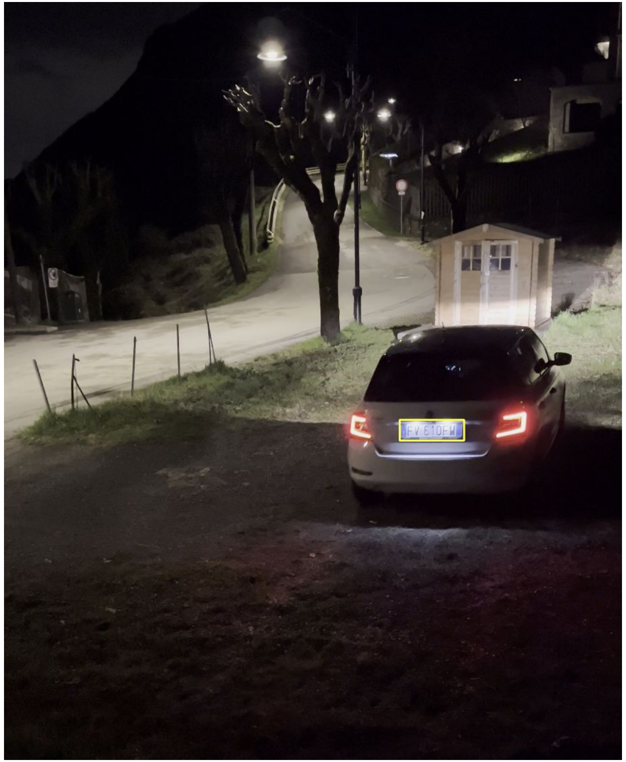
Taillights extraction

- **Initial Vehicle Localization:** we exploited YOLO to find the vehicle and crop the image, allowing us to focus specifically on the vehicle's rear and reduce background noise
- **Taillight isolation:** convert the image from RGB to **HSV color space** and defined dual hue ranges (near 0° and 180°) to capture the full spectrum of red light emitted by taillights.
- Morphological operations and contour filtering to clean up the isolated red regions, ensuring we identify the **two largest and most prominent red areas as the taillights**.



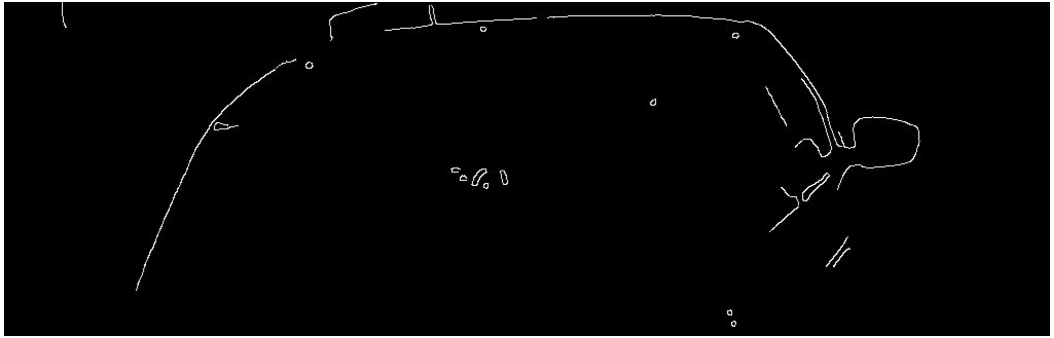
Licence plate extraction

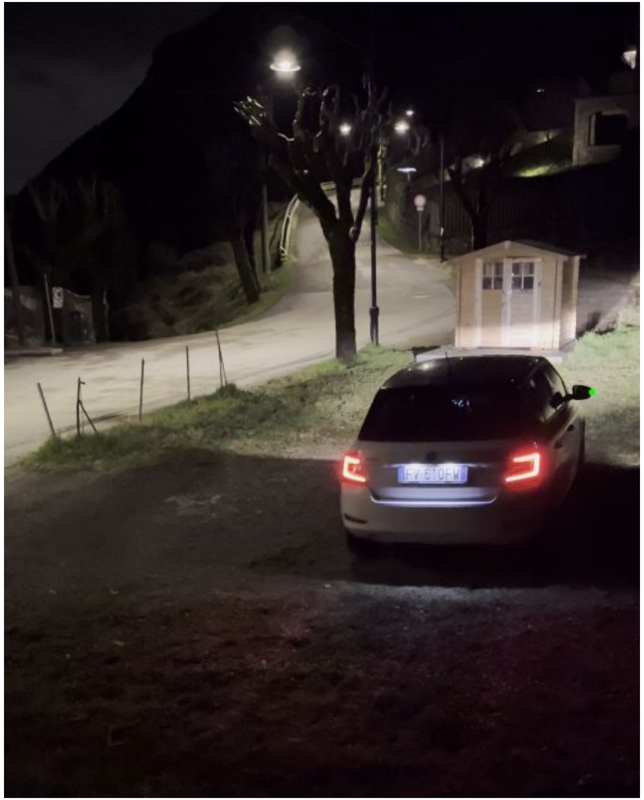
- Defined a **ROI** around the previously located lights.
- We masked the ROI based on **license plate colors** (e.g., white plate body, blue strip)
- Extracted contours from the largest found region.



Side mirror extraction

- **Vehicle cropping:** expand the YOLO vehicle cropped image to ensure the full side mirror is included.
- Apply **Canny edge detection**.
- **Mirror localization:** filter contours, keeping only the most prominent shapes. The side mirror is then identified by locating the **contour point with the highest (X-axis) position**





1. Localization via homography

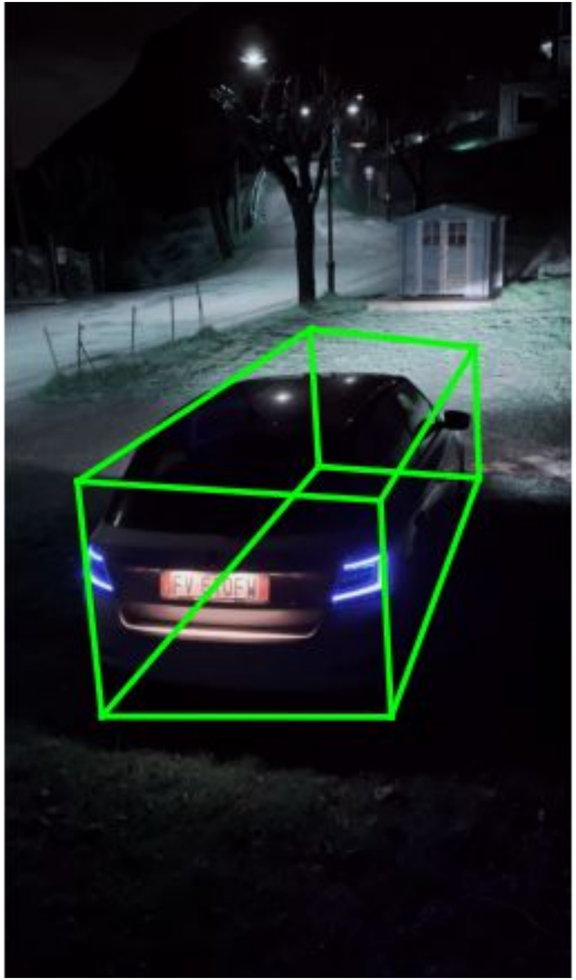
- Select a single image where both taillights and license plate corners are visible
- Identify four key features:
    - TL and TR = top corners of the license plate
    - L2 and R2 = taillights points
- Compute the direction of rays:
    - Use camera intrinsics **K** to back-project TL and TR to 3D rays.
- Estimate the 3D position of TL and TR by:
    - Measuring angle between rays
    - Applying known real-world plate width constraint

- Estimate vehicle orientation from vanishing point:
  - Compute intersection of two parallel edges (e.g., license plate top and taillights)
  - Back-project VP to get 3D direction (vehicle's forward axis)
- Build a local coordinate system:
  - $\boldsymbol{X}$: from TL $\rightarrow$ TR
  - $\mathbf{Y}$: from vanishing direction, orthogonalized to x
  - $\boldsymbol{Z} = \boldsymbol{X} \, x \, \mathbf{Y}$
- Position the vehicle origin at plate center, offset vertically
- Generate 3D bounding box using known vehicle dimensions

2. Nighttime localization from pair of images

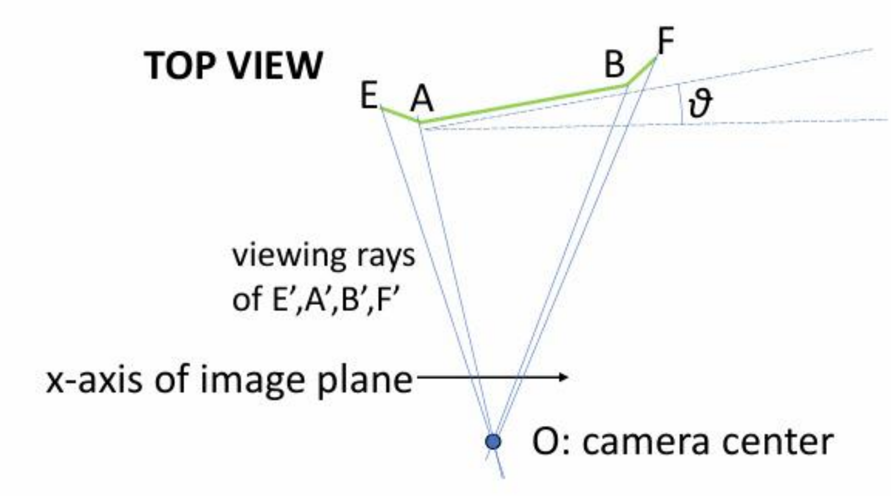
- Select two non consecutive frames.
- Track the two taillights (L1,R1 and L2,R2) and calculate two critical vanishing points:
    - ***Vx***: from the intersection of taillights segments within each frame (horizontal motion).
    - ***Vy***: from the intersection of corresponding taillights across the two frames (depth-wise motion).
- Find **vanishing line** by crossing the two vanishing points found.

- Find the the **normal vector to the vehicle's rear plane** using the vanishing line. This is achieved by multiplying ℓ with the transpose of the camera's intrinsic matrix**.**
- Compute back-projected rays of the two taillights using K−1. The **angle** between these two rays is found using geometry rules.
- Use the angle and the known width of the lights to find **distance** of rear plane from lights.
- Exploit known CAD dimension **to draw the bounding box.**



3. Poor perspective localization using out-of-plane features

- Aims at solving the localization problem when perspective is lacking.
- reduce the 3D-to-2D projection to a **2D-to-1D problem** by projecting model points onto a **horizontal plane through the camera center**, simplifying the unknown to a single **rotation angle (θ)**.
- Use an **iterative method** to solve the problem.

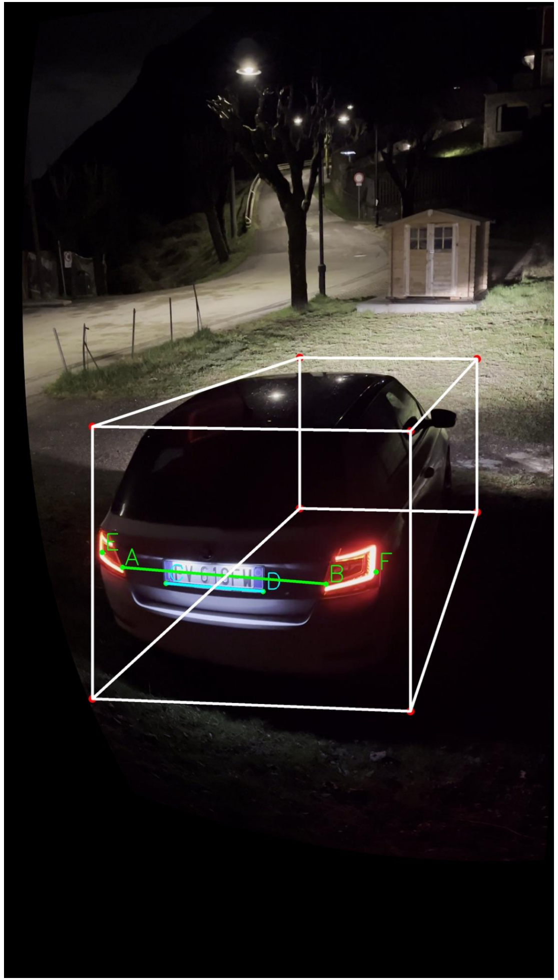


Iterative method

- **Initial estimate:** start with pose of two rear lights and two symmetric corners of the licence plate.
- **Yaw angle:** yaw angle (θ) is found by computing **vertical vanishing point** and **horizontal vanishing line**. Symmetric image points are projected onto the vanishing line to precisely calculate θ.
- **Loop:**
  - update vehicle lateral direction using the current θ
  - perform triangulation using updated rays and known distances
  - refine the back-plane normal via cross product or SVD
  - rotate the back-plane normal to compute the new vertical direction
  - recompute vanishing geometry and find new estimate of θ

- Iteration 1: $\theta = 6.27°,\ \Delta\theta = -2.34°$
- Iteration 2: $\theta = 5.89°,\ \Delta\theta = -1.88°$
- Iteration 3: $\theta = 5.58°,\ \Delta\theta = -1.55°$
- Iteration 4: $\theta = 4.37°,\ \Delta\theta = -1.28°$
- Iteration 5: $\theta = 4.17°,\ \Delta\theta = -0.21°$
- Iteration 6: $\theta = 4.14°,\ \Delta\theta = -0.04°$

Result



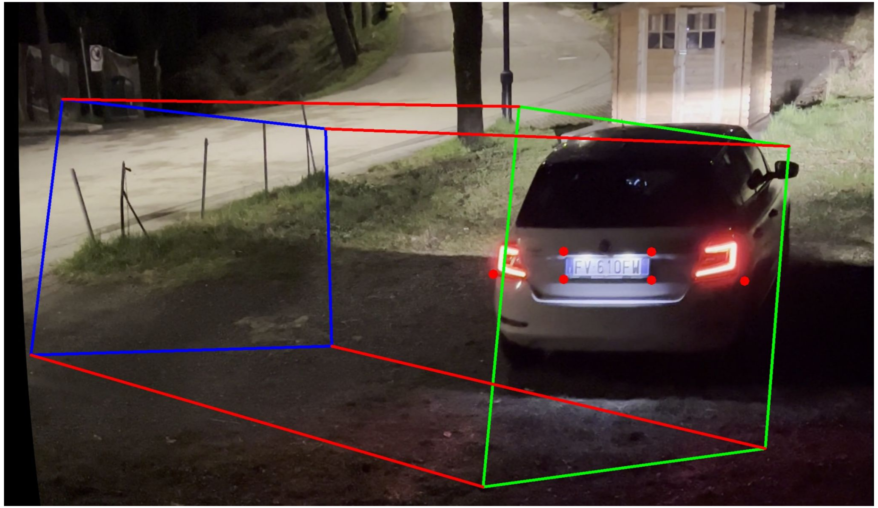
4. PnP-based vehicle pose estimation from key points

- **PnP:** Perspective-n-Point (PnP) aims at estimating the **3D position and orientation** of a camera relative to a known 3D object.
- Achieved using two rear lights points and two licence plate points.
- advantage of PnP over homography methods is its ability to handle **non-coplanar 3D points**.
- **Variants:**
  - **Iterative PnP:** refines an initial guess for higher accuracy.
  - **EPnP (Efficient PnP)**: provides a computationally efficient and stable non-iterative solution, ideal for real-time.
  - **PnPRANSAC:** combines PnP with outlier detection, robustly handling noisy or incorrect point matches.

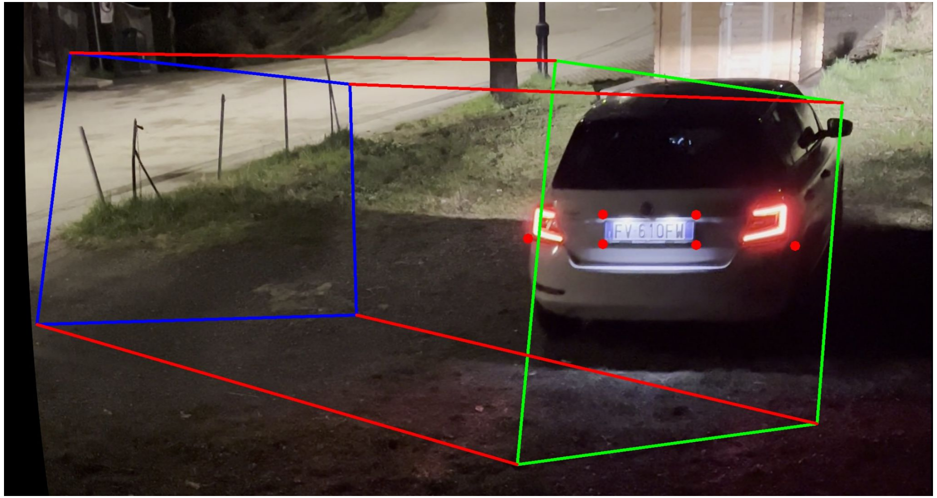
4 points approach

We evaluated both an iterative PnP solver (with an initial pose estimate) and a direct PnP solution (like EPnP).

EPnP



Iterative



5 points approach

We incorporated a fifth point to find the estimate, to make it more robust The side mirror incorporates crucial **depth variation**, which enables more accurate estimation of the car’s lateral extent.

Comparison of results

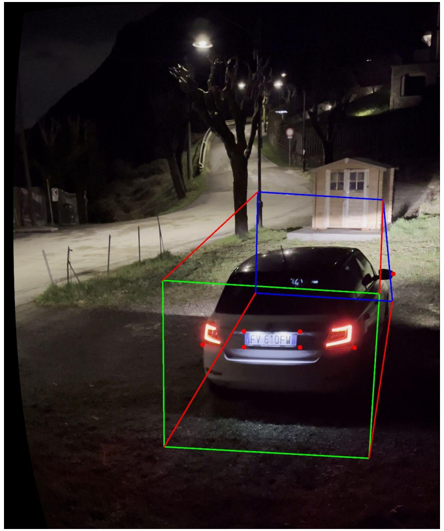
Methods 1 and 4 are based on basic geometric transformations and image matching techniques, making them computationally efficient and easy to implement.
However, they face significant limitations in terms of generalizability and reliability.

Method 1

- Simple and fast to implement
- Highly dependent on planar assumptions
- Performs poorly under perspective distortion or non-flat surfaces

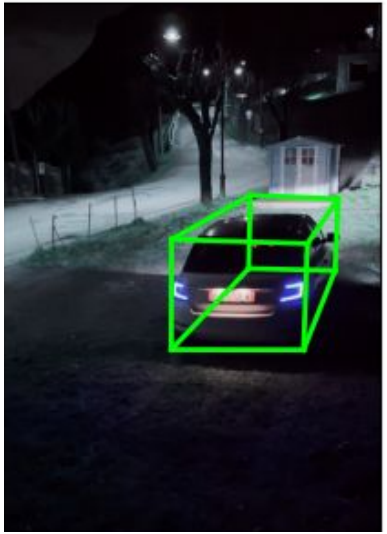
Method 4

- Easy to implement
- Fast runtime
- Very limited robustness to pose and scale variation
- Not reliable under real-world conditions



Comparison of results

Methods 2 and 3 leverage more advanced geometric reasoning and perspective cues to estimate the orientation of objects with higher robustness.

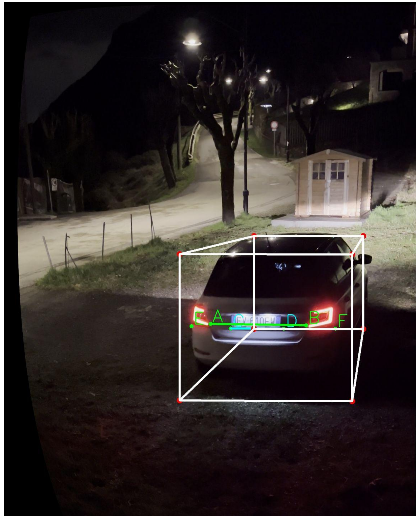


Method 2

- High accuracy with well-detected vanishing points
- Robust to perspective changes
- Depends on clear line features and image resolution

Method 3

- Handles diverse and noisy scenes
- Learns robust features, works well across diverse conditions
- Computationally demanding



Future work

- Automate vehicle data retrieval using a neural network for license plate recognition.
- Query public databases to identify car model.
- Access 3D CAD repositories for accurate geometry.
- Enhance robustness of plate and light detection under challenging conditions with advanced object detection algorithms.