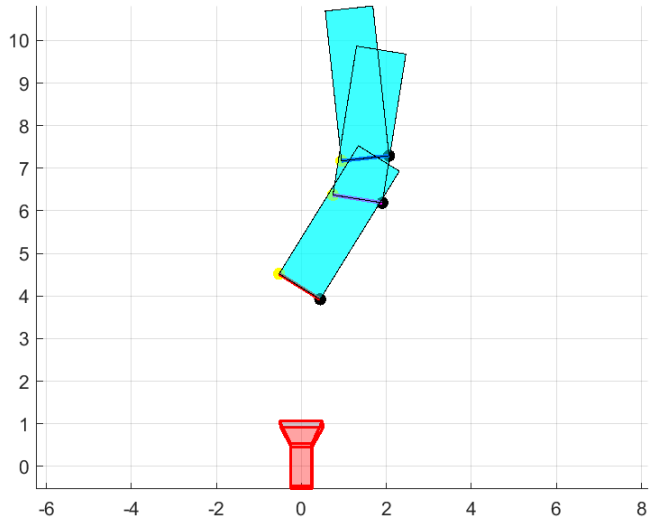
Vehicle Space Occupancy

IACV Project Report

Ermelinda Giulivo
Codice Persona: 10790499
Matricola: 246877
E-mail:
ermelinda.giulivo@mail.polimi.it

Luca Masiero
Codice Persona: 10683660
Matricola: 245458
E-mail:
luca3.masiero@mail.polimi.it



Politecnico di Milano
Academic Year 2024-2025
July 8, 2025

1 Introduction

This document presents the report for the 2024/2025 project of the Image Analysis and Computer Vision course. It is structured to provide a comprehensive overview of the entire project development process.
The report begins with a definition of the problem addressed, followed by a description of the setup phase, which outlines the tools, technologies, and development environment configured to support the subsequent work. It then details the reasoning and methodology that led to the proposed solution. The results obtained during the implementation and testing phases are subsequently presented and analyzed. Finally, the report concludes with a discussion of the outcomes and the key insights gained throughout the project.
In this GitHub repository can be all the code developed for the project.

2 Project Overview and Problem Definition

The goal of this project is to develop an algorithm capable of estimating the 3D position of a vehicle from images captured by a fixed camera. The project is divided into two main objectives. The first focuses on localizing the vehicle using a single image by leveraging geometric constraints and known calibration parameters. The second, instead, uses three consecutive frames in order to estimate the trajectory of the vehicle over time. Both tasks assume the availability of prior knowledge, such as the camera's intrinsic matrix and the fixed real, world distance between two identified feature points on the vehicle—to constrain and guide the reconstruction process.

Specifically, the following information are assumed to be known:

- The intrinsic calibration parameters of the camera, represented by the K matrix.
- A model of the vehicle, including its length, width, and the distance between the rear lights.

The analysis is based on the following assumptions:

- The vehicle exhibits a vertical plane of symmetry, which simplifies geometric reasoning.
- The two rear lights of the vehicle are visible and symmetrically placed, enabling reliable reference points for estimation.
- The camera is positioned behind the vehicle, observing it from the rear, and it is fixed.c
- Between consecutive frames, the vehicle either translates forward or follows a path with constant steering curvature.

- The road surface is locally planar, allowing for simplified projection and transformation calculations.

These assumptions provide the foundational constraints needed to infer the vehicle's position and orientation over time and to estimate the area it occupies in each frame of the video.

3 Project Setup

The first step of the project was setting up the necessary components. We chose an iPhone 13 as our camera and selected a Fiat Panda as the vehicle for our analysis. The process then followed these main steps:

- **Camera Calibration**: To obtain the intrinsic parameters of the iPhone 13 camera, we used the Camera Calibrator app in MATLAB. This tool processes multiple images of a checkerboard pattern taken with the camera and computes the camera's intrinsic calibration matrix (K matrix).
- **Car Model Acquisition**: We searched online for a CAD model of the Fiat Panda and supplemented it with some manual measurements when necessary to ensure accuracy.
- **Software Choice**: Finally, we chose to develop the project in MATLAB, not only because it had been consistently used throughout the course and was therefore a familiar environment, but also because it is widely adopted in real-world image analysis applications. Its extensive documentation and robust set of built-in tools make it a reliable choice.

4 Single Frame Vehicle Localization via Back-Projection Ray Geometry

The first objective of this project was to determine the position of a car with respect to a fixed camera using a single frame. The approach employed to localize the vehicle on the road leverages the concept of the **back projection ray**. A back projection ray is defined as the light ray that reflects from a 3D point U, passes through its corresponding 2D image point u, and reaches the optical center of the camera.

Assuming that the feature points have already been extracted from the image, as described in Section 4, we consider the four vertices of a quadrilateral defined by the two bottom corners of the license plate and two symmetric points on the taillights.
The 2D image points are represented in homogeneous coordinates as:

$$u_1, u_2, u_3, u_4 = (x, y, 1) \tag{1}$$

Figure 1: In red the feature points and the quadrilateral to be localized; in green the principal point of the camera.

As mentioned above, the objective is to localize this quadrilateral with respect to a reference frame centered at the camera. From this, the spatial volume occupied by the entire vehicle, approximated as a parallelepiped, can be inferred.

This geometric procedure begins with the computation of the **vanishing point**, which is obtained by determining the intersection of the lines formed by connecting the symmetric feature points in pairs.

$$l1 = u1 \times u2 \; ; \qquad l2 = u3 \times u4 \tag{2}$$

$$p_{inf} = l1 \times l2 \tag{3}$$

This yields the homogeneous 2D coordinates of the vanishing point in the image plane coordinate system. Each 2D image point is then multiplied by inverse of the camera calibration matrix K to obtain the corresponding back-projection rays. These are then normalized to obtain unit vectors representing the directions.

$$u_{dir} = K^{-1}u \tag{4}$$

$$r = u_{dir} \; / \; ||u_{dir}|| \tag{5}$$

The rays formed by back-projecting 2D image points create triangular shapes that extend from the camera center to the corresponding 3D points on the

vehicle. If enough geometric information about these triangles is available, we can apply the Law of Cosines to compute the distances from the camera to the 3D points; these distances correspond to the sides of the triangles.

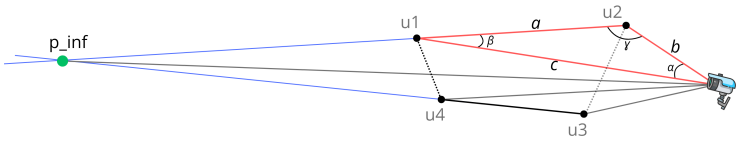


Figure 2: This simplified visualization gives an idea of what is the structure obtained by back-projecting the feature points of the trapezoid. In red, the triangle used for computing the distances of the first two features (i.e., the taillights) from the camera.

From the assumptions and the CAD model of the vehicle, we already know the length of one side of the triangle, denoted as segment a.

A key observation allows us to formulate a system of three equations with three unknowns: the back-projection ray corresponding to the vanishing point represents the direction of the real-world segments that were used to compute it, segments that are parallel in the physical world. This directional information makes it possible to compute the angles (β and γ) between the back-projection rays and the real-world segments connecting symmetric feature points.

The resulting system of equations, based on the Law of Cosines, is as follows:

$$\begin{cases} c^2 = a^2 + b^2 - 2ab\cos\gamma \\ a^2 = b^2 + c^2 - 2bc\cos\alpha \\ b^2 = a^2 + c^2 - 2ac\cos\beta \end{cases} \tag{6}$$

Once the distances (b and c) from the camera to each feature point are computed, we can recover the 3D coordinates of each point in the camera reference frame. This is done by scaling the unit direction vector of the corresponding back-projection ray:

$$\mathbf{u}_{cam}^{(1)} = c \cdot d_{u1} \tag{7}$$

$$\mathbf{u}_{cam}^{(2)} = b \cdot d_{u2} \tag{8}$$

Here, b and c are the distances, along the back-projection rays, obtained from the system of equations above, d_{u1} and d_{u2} are the unit vectors along the back-projection rays corresponding to the feature points.

This system (6) and the equations (7) and (8) are used twice: once for the upper feature points (taillights), and once for the lower feature points (the corners of the license plate).

While this would suffice for a basic localization, we acknowledge that the camera may not be oriented parallel to the road. Typically, it is directed slightly towards the center of the road, making its reference frame non-intuitive. In the standard camera frame, the z-axis extends along the optical axis, which is not aligned with the direction of the road.

To address this, we propose expressing the results in a new reference frame that better aligns with the road. In this **world reference frame**:

- The x-axis, which is the back projection of the vanishing point, points to the left,
- The y-axis points forward along the road,
- The z-axis points downward,
- The xy-plane is parallel to the ground.

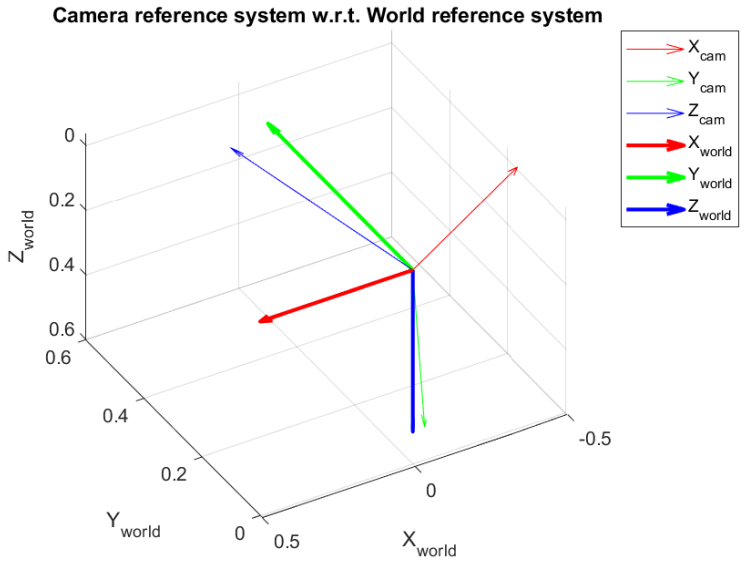


Figure 3

We define this new frame by using the back-projection ray of the vanishing point as the x-axis, and then constructing the remaining axes accordingly.

$$X = inf\_dir; \tag{9}$$

$$Y = X \times [0,1,0]' \quad or \quad Y = X \times [0,0,1]' \tag{10}$$

$$Z = X \times Y \tag{11}$$

where inf_dir is the back projection ray of the point at infinity. The Y axis is computed using vector $[0, 1, 0]$ if it sufficiently perpendicular to X, otherwise vector $[0, 0, 1]$ is used. This is done to obtain a reference frame that is as precise as possible.

This world reference frame shares its origin with the camera frame but is oriented differently. The rotation between the two frames can be expressed through a rotation matrix R, where each column represents the coordinates of a world frame axis with respect to the camera frame. By multiplying the 3D points in the camera frame by the R, we transform them into the world frame:

$$R = \begin{bmatrix} | & | & | \\ X_{world} & Y_{world} & Z_{world} \\ | & | & | \end{bmatrix} \tag{12}$$

$$World\_Coords = R \cdot Camera\_Coords \tag{13}$$

Figures 3 to 5 illustrate the results of the proposed algorithm, visualizing the localization: the vehicle modeled as a parallelepiped, alongside the original image and the quadrilateral defined by the extracted feature points.

This method demonstrates a reasonable degree of accuracy, even considering potential errors in the feature extraction process. In practice, slight asymmetries or misplacement of the feature points can lead to minor deviations in localization, but overall, the approach remains sufficiently robust and effective.

Figure 4

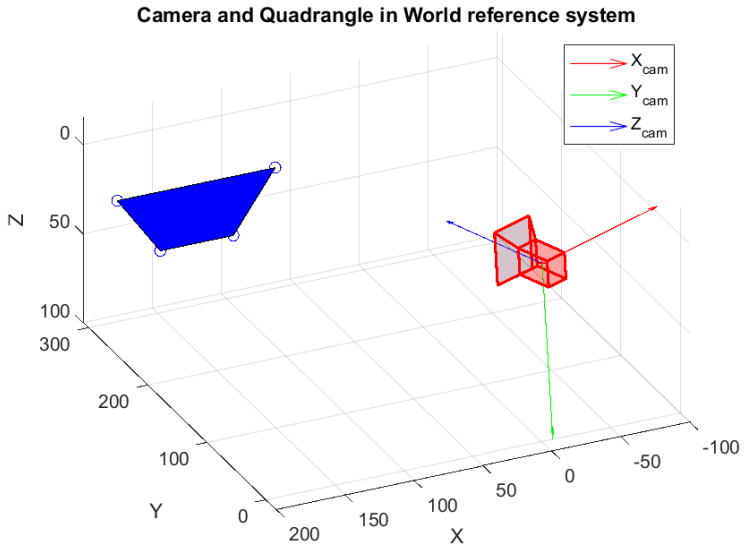


Figure 5

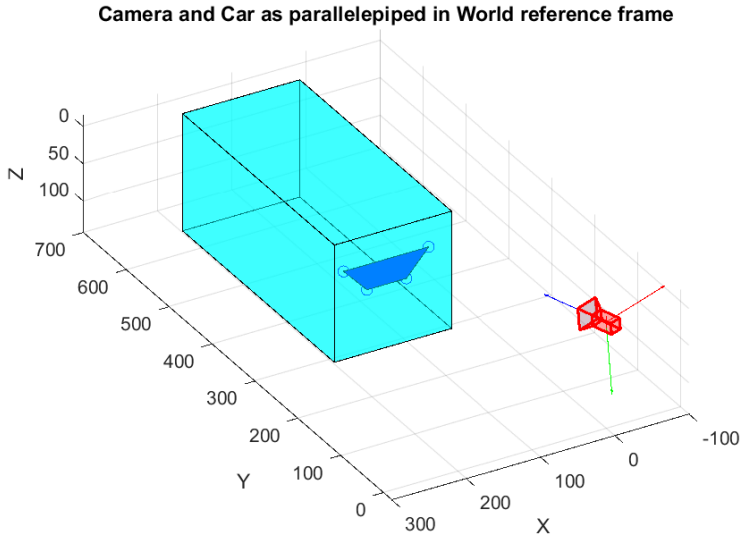


Figure 6

5 Multi-Frame Vehicle Trajectory Estimation

The second objective of this project was to estimate the trajectory of a vehicle using only three consecutive video frames and two feature points. The method is based on the assumption that the two selected points, chosen to be symmetric, maintain a constant distance from each other across all three frames. Furthermore, these points are assumed to lie on a horizontal plane, denoted as π, which is ideally parallel to the road surface.

The proposed method can be conceptually divided into two main steps:

1. **Plane Definition**: estimating the normal vector of the plane that contains the three line segments connecting the selected feature points in each frame. This normal is obtained by enforcing the physical constraint that the real-world distance between the taillights remains constant across all frames.

2. **Distance Estimation**: once the normal vector is known, each image point is back-projected into 3D space, and its intersection with the estimated plane is computed to localize the car with respect to the camera.

The image below illustrates the selected feature points, which we refer to consistently throughout this chapter. In our case, these points correspond to the vehicle's taillights, chosen for their symmetry. While our implementation uses these specific features, the method is general and can accommodate other known-distance points if available.

Next, we provide a detailed explanation of the adopted methodology.

Figure 7: undistorted image with extracted features p and q (highlighted in red) and camera principal point (highlighted in green).

5.1 Reference Frame Definition and Optimization Procedure

To determine a vector $\mathbf{n}$ that defines the plane π containing all three taillights segments we set up minimization problem, defined as follows:

$$\min \quad \sum_{i=1}^{m} (\|\mathbf{p}_i - \mathbf{q}_i\| - d)^2 + \alpha(1 - \mathbf{n}^T \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix})^2 \tag{14}$$

$$\text{s.t.} \; \|\mathbf{n}\| = 1 \tag{15}$$

where m is the number of frames, which is three in our case, d is the real length of the segment pq, *alpha* is a weight, set to 2 in our case, but whose value can be changed to fine-tune the optimization function. At each iteration of the optimization procedure, the vector $\mathbf{n}$ is updated. The process is initialized by setting $\mathbf{n} = [0, 1, 1]^T$.

The first part of the *loss function* requires that the segments $p_i q_i$ are all equal to d. At the same time, the part outside of the summation pushes for finding a normal vector $\mathbf{n}$ that is close to the y-axis of the camera frame.

To compute the distance between the two feature points in each frame, we first reconstruct their position in the word by intersecting their back-projection ray with plane π. To do this the following formulae were used:

$$\begin{aligned} \mathbf{dp}_i &= \frac{K^{-1}\mathbf{p}'_i}{\|K^{-1}\mathbf{p}'_i\|} & \mathbf{dq}_i &= \frac{K^{-1}\mathbf{q}'_i}{\|K^{-1}\mathbf{q}'_i\|} \\ \lambda_i^p &= \frac{1}{n^T \cdot \mathbf{dp}_i} & \lambda_i^q &= \frac{1}{n^T \cdot \mathbf{dq}_i} \\ \mathbf{p_i} &= \lambda_i^p \cdot \mathbf{dp}_i & \mathbf{q_i} &= \lambda_i^q \cdot \mathbf{dq}_i \end{aligned} \tag{16}$$

where $\mathbf{p}'_i$ and $\mathbf{q}'_i$ represent the homogeneous pixel coordinates of the feature points in the i^{th} frame, K is the intrinsic camera matrix and $\mathbf{n}$ is the normal vector to plane π. Notice that λ_i^p represents the distance along the back-projection ray $\mathbf{dp}_i$, and the same holds for $\mathbf{q}$.

Once the optimization converges and the minimum of the error function is found, the final value of $\mathbf{n}$ can be used to computed the actual position of the features points in the world, with respect to the camera reference frame.

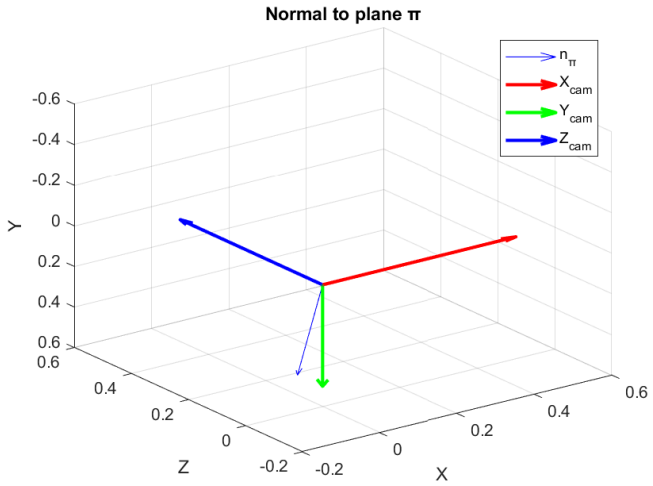


Figure 8: The thin blue vector represent the estimated normal vector to the plane π, in this case $n = [-0.0126, \quad 0.9872, \quad 0.1591]^T$. It can be noticed that the representation is coherent with the fact the camera was looking slightly downward towards the road, as it can be seen also in Figure 8.

5.2 Distance and Trajectory Estimation

Now that the horizontal plane π has been defined, we can proceed to compute the absolute distance of the vehicle from the camera in each of the three frames.

To compute the 3D coordinates of the feature points, we can simply apply the formulae (15) displayed above, this time using the *optimal* value of vector **n**.
This allows us to localize the vehicle in 3D space relative to the camera for each frame, thus enabling a reconstruction of its trajectory in absolute terms.

Given the normal vector **n** to plane π we can also compute the rotation matrix R of the camera with respect to the plane.
What the procedure does is computing the rotation matrix R such that:

$$R \cdot \mathbf{n} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}.$$

The method used is based on the *Rodrigues rotation formula* which, given an axis of rotation $\mathbf{v} = [v_x, v_y, v_z]^T$ and an angle of rotation θ, allows to compute the rotation matrix with a simple formula:

$$R = I + \sin(\theta)H + (1 - \cos(\theta))H^2 \tag{17}$$

where H is a skew-symmetric matrix defined as

$$H = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix}. \tag{18}$$

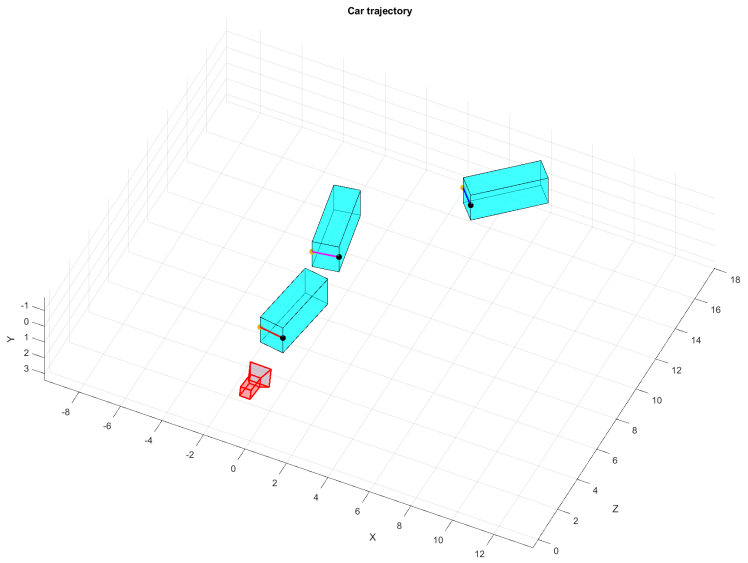
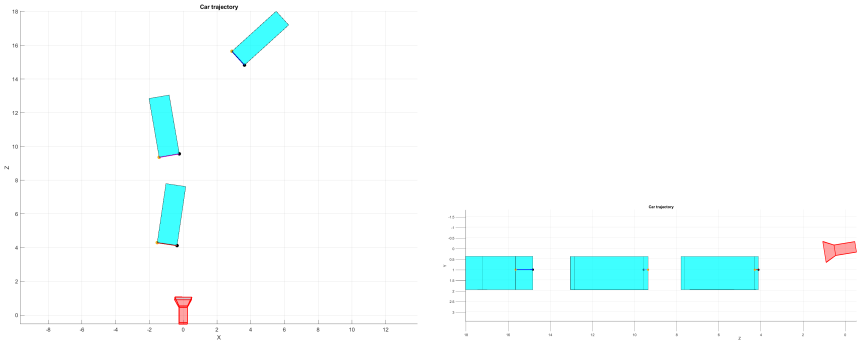


Figure 9: The car is represented as a light-blue parallelepiped at the positions corresponding to the three frames. Feature point **p** (the left taillight) is shown in yellow, while feature point **q** (the right taillight) is shown in black. The reference frame is that of the camera.



5.3 DLT for frames alignment

To increase the robustness of the method, we introduce a preprocessing step to be performed after feature point extraction and before estimating the normal vector to the plane π. This additional step accounts for small camera movements that may occur between consecutive frames. It is based on homography estimation using the Direct Linear Transformation (DLT) algorithm.

Instead of using only the two feature points **p** and **q**, we also select four additional background points, assuming that the background remains static across frames. With these four points per frame, we estimate a homography that aligns each subsequent frame to the first one. This alignment compensates for minor camera shifts, reducing potential errors caused by unintended motion.

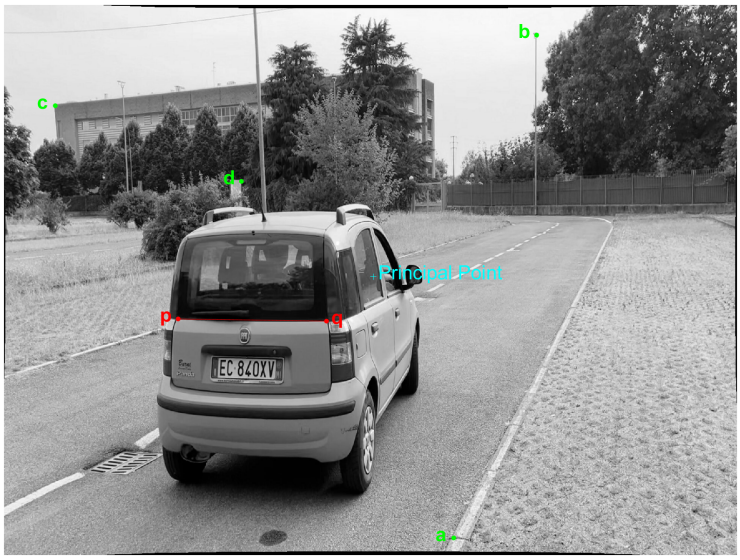


Figure 10: An example of feature extraction when using DLT for frames alignment. In red the taillights feature points of the car, in green the feature points needed to execute DLT and in cyan the camera principal point.

It is important to note that, for this project, all feature points were manually selected. Although great care was taken to ensure consistency across frames, small inaccuracies may still have been introduced. Nonetheless, the results obtained were satisfactory and sufficiently accurate for our purposes. Even though, in most cases, the homography-based alignment did not yield a noticeable improvement, likely because the original images already exhibited minimal displacement between frames, it remains an important preprocessing step that should always be run to provide greater guarantees on the correctness of the results.
In the following image, successive feature pairs (**p** and **q**) and their displace-

ments are visualized together on the first frame, showing the result of such alignment step.

Figure 11: Unfilled points indicate the manually extracted feature locations, while filled points show the corresponding positions after applying the DLT-based alignment. Feature points from each frame are color-coded as red, magenta, and blue, respectively. All points are overlaid on the first frame of the sequence.

5.4 Testing process

To verify the correctness and robustness of our algorithm, we first tested it in a simplified, controlled scenario. Instead of using images of a car, we simulated the taillight feature points using a pen, which represented the segment connecting the two points. The pen was moved across a table to mimic the motion of a car on a road, while a ruler was placed nearby to provide a reference for the displacement between frames. The camera was positioned either directly behind the pen (simulating a rear-facing view) or slightly offset to the right, simulating a side-of-the-road perspective.

This testing procedure was crucial for evaluating the algorithm's behavior in a manageable setting, where we could select the feature points more easily and accurately. It revealed some key issues, which were subsequently addressed and resolved. As a result, the algorithm became more accurate and robust when applied to the actual car frames.

One specific issue encountered during testing was that, due to the small displacement between successive pen positions, the algorithm tended to estimate the road plane π as nearly vertical, rather than horizontal as it should be (since

the table represents the road surface). This was resolved by incorporating the second term of the error function (Formula 13), which encourages the estimated normal vector $\mathbf{n}$ to align with the camera's y-axis. This constraint effectively guides the plane π to adopt a more horizontal orientation, consistent with the real-world setup.

The following images illustrate one of these test cases and the corresponding results.

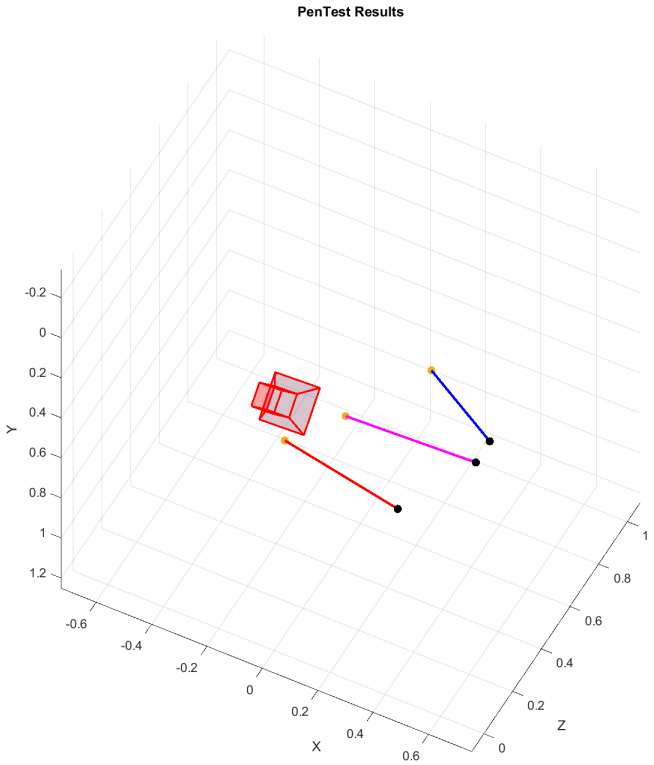
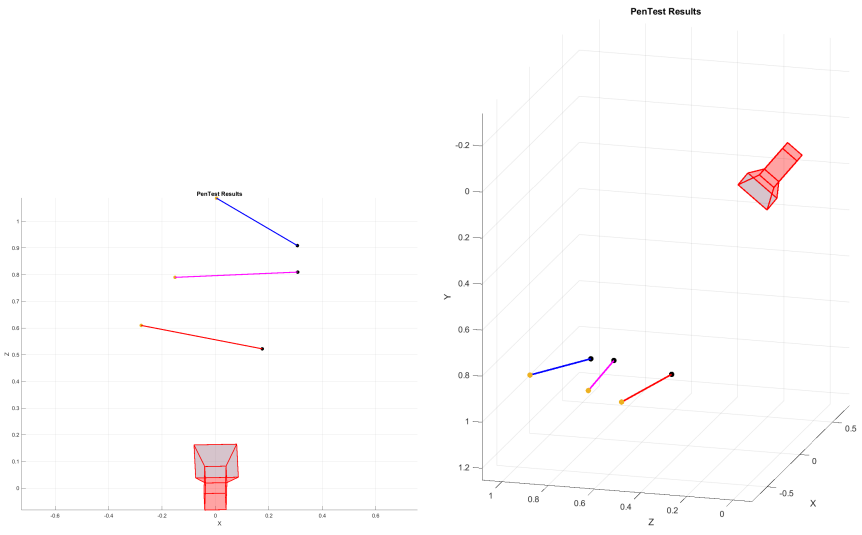


Figure 12: The camera looks at the seen from the right, as if it was on the side of the road. The reference frame is that of the camera.



6 Conclusion

In recent years, image analysis and computer vision have significantly transformed traffic management systems, harnessing the power of AI and machine learning to improve road safety and efficiency. By analyzing data from traffic cameras, these technologies enable real-time monitoring and automated control, addressing critical urban challenges such as congestion, accidents, and pollution.

The goal of our project was to explore traditional image analysis techniques for localizing a vehicle on the road using minimal information and to estimate its trajectory across multiple frames. This approach has potential applications in the monitoring and detection of hazardous situations. Thanks to the lightweight computational demands typical of many image analysis methods, such a system could be deployed in real-time traffic monitoring solutions, capable of instantly alerting authorities in the event of an accident.

Our work was intended as a proof of concept, focusing primarily on the methods and algorithms rather than on real-world deployment challenges. As outlined in Section 2, we made several simplifying assumptions to make the problem tractable. For instance, we assumed a known and precise 3D model of the vehicle for some parts of the method (specifically in Approach 1), which in a real-world system would require identifying the vehicle and retrieving its dimensions from an external database.

Another key challenge is feature extraction. While not the central focus of our project, in a practical application, this step would need to be fully auto-

mated and highly accurate, an inherently complex and non-trivial task.

Despite these limitations, the developed algorithms showed promising results and achieved sufficient accuracy for the scope of our project. With further development and refinement, such methods could contribute meaningfully to real-time, camera-based traffic monitoring systems.