Extrinsic Camera Calibration Using Vehicular Interior Features

A Domain-Agnostic and Targetless Pipeline

Master's thesis in Systems, Control and Mechatronics

DAVID MÖRCK
ANDREAS SÄRNHOLM

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

Extrinsic Camera Calibration Using Vehicular Interior Features

A Domain-Agnostic and Targetless Pipeline

David Mörck
Andreas Särnholm

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2026

Extrinsic Camera Calibration Using Vehicular Interior Features
A Domain-Agnostic Pipeline for Targetless Calibration
DAVID MÖRCK
ANDREAS SÄRNHOLM

Academic supervisor: Jennifer Alvén, Department of Electrical Engineering
Industry supervisor: Jonas Ekdahl, Volvo Cars
Examiner: Jennifer Alvén, Department of Electrical Engineering

Master's Thesis 2026
Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Photo depicting an interior of a Volvo positioned on a country road, generated with AI.

Typeset in LaTeX
Printed by Chalmers Digitaltryck
Gothenburg, Sweden 2026

Extrinsic Camera Calibration Using Vehicular Interior Features
A Domain-Agnostic and Targetless Pipeline
David Mörck
Andreas Särnholm
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Advanced driver assistance and safety systems increasingly rely on interior sensing systems, such as Occupant Monitoring Systems (OMS) and Driver Monitoring Systems (DMS). These systems depend on accurately calibrated interior cameras to provide reliable detection of behavior and positioning of the driver and occupants. Traditional camera calibration often relies on dedicated targets such as checkerboards, but such methods can be difficult to integrate into large-scale production environments.

This thesis investigates targetless extrinsic calibration of an OMS camera by estimating its six degree of freedom pose from a single image of the car interior. Instead of using calibration targets, the proposed approach uses interior features as reference points. Two methods are evaluated: a relative method using a nominal reference image, and an absolute method using triangulated 3D points from known established views.

The results show that extrinsic calibration using interior features is possible, with especially strong rotation estimation. Both methods presented here achieve errors as low as 0.09 degrees on synthetic data, while translation proved more challenging within the small mounting tolerances. The Absolute Method performed best for translation on synthetic data, reducing the error from an average of 5 mm to 1 mm, whereas the Relative Method showed no reduction in error. The Relative Method did however show better performance on limited real vehicle data by consistently reducing the reprojection error compared with the uncalibrated baseline.

Overall, the thesis demonstrates the potential of targetless calibration for scalable OMS camera deployment. Although it is demonstrated on car interiors, the proposed pipeline is domain-agnostic and may be applicable to other environments containing repeatable visual features.

Keywords: Extrinsic camera calibration, targetless calibration, 6DOF pose estimation, occupant monitoring system, feature matching.

Acknowledgements

First and foremost, we sincerely thank Volvo Cars and our industry supervisor, Jonas Ekdahl, for the opportunity to conduct our master's thesis within the company, and for the valuable support provided by him and his colleagues. We also wish to express our profound gratitude to our academic supervisor and examiner, Jennifer Alvén, for her invaluable guidance, continuous feedback and help throughout the project and the thesis writing.

David Mörck, *Andreas Särnholm*, Gothenburg, May 2026

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis, presented in alphabetical order:

2D	Two-dimensional
3D	Three-dimensional
6DOF	6 Degrees of Freedom
ADAS	Advanced Driver Assisting System
CDF	Cumulative histogram
DMS	Driver Monitoring System
FoV	Field of View
IR	Infrared
LLM	Large Language Models
LPIPS	Learned Perceptual Image Patch Similarity
MAE	Mean Absolute Error
ML	Machine Learning
OMS	Occupant Monitoring System
PnP	Perspective-n-Point
SfM	Structure from Motion
SSIM	Structural Similarity Index Measure

Contents

1
Introduction

Improving vehicle safety is one of the challenges in automotive engineering. One way of addressing this challenge has been the development of increasingly advanced driver-assistance and perception systems. In recent years, significant advances in computing power have enabled the data processing and split second decision-making required for safe advanced driver-assistance systems (ADAS). Examples of such systems that are included in a new Volvo car are collision avoidance and lane keeping aid. Beyond providing visual and audible warnings of a potential danger, the collision avoidance system can intervene by actively controlling the vehicle's steering and braking to prevent or mitigate collisions [1].

1.1 Background

To support these safety-critical systems, vehicles require a continuous stream of reliable input data. Most modern cars are equipped with a sophisticated suite of sensors such as cameras, radars, ultrasonic sensors and lidars, which provide the vehicle with data depicting its surrounding environment.
A newer class of safety systems is interior sensing systems, which make use of similar sensors inside the vehicle [2]. Two examples are Occupant Monitoring Systems (OMS) and Driver Monitoring Systems (DMS) which primarily rely on cameras for detection [3]. These systems look at the driver and passengers to detect conditions such as driver distraction, incorrect seatbelt positioning and unsafe posture. For these systems to provide reliable estimates, which make the vehicle safer without generating unnecessary warnings, their readings must be consistent. Achieving such consistency requires accurate camera calibration. This thesis was conducted in collaboration with Volvo Cars with the broader aim of contributing to safer vehicles through improved calibration of interior monitoring cameras.

1.1.1 Problem description

Extrinsic camera calibration is a critical prerequisite for most vision-based applications. However, existing calibration methods typically rely on dedicated calibration targets, controlled environments, or manual intervention. This complicates deployment in well established large-scale production-line settings, especially for interior facing vision systems.
This thesis addresses the challenge of estimating a camera's six degrees of freedom (6DOF) pose, i.e., its three dimensional position and orientation, using a single snap-

shot of the vehicle's interior. By exploiting naturally occurring structural features as reference points, the proposed approach aims to eliminate the need for specialized calibration targets. The goal is to enable a faster, more user-friendly, and scalable calibration process that can be deployed in real world environments, both on and beyond the production line.

1.1.2 Research questions

The problem description can be summarized as three concrete research questions:

1. To what extent can state-of-the-art feature extractors and matchers reliably establish correspondences across varying vehicular interior environments?
2. What quantitative accuracy can be achieved using the proposed targetless calibration pipeline, and how does this compare to relying solely on the camera's uncalibrated nominal pose?
3. To what extent is the proposed calibration pipeline suitable for scalable deployment in automotive production environments?

1.2 Scope

The thesis focuses on a monocular camera mounted in a typical position for monitoring occupants of a vehicle. The calibration performed using the estimation method is limited to camera pose rather than intrinsic parameters. Furthermore, it is assumed that the parts of the interior that are not dynamic, such as seats, belts or steering wheels, are fully static, and stay consistent across all data. It is also assumed that the camera is a fish-eye infrared (IR) camera, and that the images are therefore in grayscale and significantly distorted.

The project makes use of synthetic data generated from virtual cameras within a simulation environment. The usage of synthetic images saves time and guarantees access to the ground truth. In this thesis, that means having knowledge of the true pose of the camera, making both development and validation easier. Furthermore, it is assumed that adjustable interior components, such as seats and steering wheels, remain fixed throughout the data collection.

Due to limited amounts of data covering real images and ground truth labels, the project will not pursue an end-to-end ML solution, as these might not adapt well from synthetic data to real data. This also limits the number of unique vehicles tested for the final evaluation of the system.

For this problem, it is assumed that the camera is mounted in a fixed nominal position, with installation errors limited to a certain tolerance range. It is assumed that there is a maximum deviation of $\pm 3°$ in orientation and ± 3 mm in position along each individual axis.

Since the thesis aims to develop a non-intrusive calibration method, the proposed approach is limited to software-based solutions. Additionally, the project excludes tools restricted to non-commercial use and prioritizes software that does not require paid licensing.

1.3 AI usage

The authors of this thesis have had access to and used Large Language Models (LLMs) as part of this project. During development LLMs have been used as code-generators for rapid prototyping covering initial implementation of parts of the presented pipelines and evaluations. Furthermore, they have also been used for troubleshooting of produced code.
For this paper, LLMs have been used iteratively as reviewers, with its feedback and suggestions being taken into account by the authors. With this said, the authors of this report take full responsibility for the entirety of the content produced in this report and the work provided to Volvo Cars.

2

Preliminaries

This chapter gives a brief overview of camera geometry and calibration, together with an introduction to tools either used throughout the thesis, or deemed interesting for the topic.
Throughout the thesis, 2D points in the image are referred to as image points, 3D coordinates in the scene as world points, and the corresponding points in the camera coordinate frame as camera points.

2.1 Camera geometry

Camera geometry explains the mathematical transformation from a world point in 3D space to an image point in 2D space. This transformation can be expressed by Equation 2.1:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = P \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}. \tag{2.1}$$

In this equation, λ is a scaling factor, u and v correspond to the pixel coordinate of the image point, P is the camera matrix of size 3×4 and X_w, Y_w, Z_w make up the 3D coordinate of the world point. The equation uses homogeneous coordinates, which solves the dimensionality discrepancy. Furthermore, the camera matrix P can be formulated using Equation 2.2:

$$P = K \left[\begin{array}{c|c} R & t \end{array} \right], \tag{2.2}$$

where K is a 3×3 matrix containing the intrinsic parameters of the camera, while R, t denote the pose of the camera, that is, the rotation and translation in 3D. In the two upcoming sections 2.1.1, 2.1.2, the parameters are introduced in more detail.

2.1.1 Camera extrinsics

The camera extrinsic matrix represents a rigid transformation in 3D, from a world coordinate system O_w to a camera coordinate system O_c. This transformation is done according to the 3×3 rotation matrix R and translation along the 3×1 vector

t as can be seen in Equation 2.3:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}. \tag{2.3}$$

Here, the subscripts of each axis X, Y, Z describe the coordinate system of which it is expressed.

2.1.2 Camera intrinsics

The intrinsic matrix K is used to project camera points in the camera coordinate frame to image points on the image plane. This projection is done using five intrinsic parameters, focal length f_x, f_y, principal point c_x, c_y and skew s, resulting in the matrix seen in Equation 2.4:

$$K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.4}$$

2.1.3 Distortion

There is an additional factor that affects the internal projection from 3D to 2D in a camera equipped with a lens, namely, lens distortion. Two main types of distortion exists, radial distortion and tangential distortion. Radial distortion is a result of the light rays bending more towards the edges of the lens rather than in the center. This results in straight lines in the real world appearing as curved in an image. Tangential distortion expresses itself by tilting the image and is present if the lens is not mounted perfectly perpendicular to the plane of the image sensor. The two types of distortion mentioned here are visualized separately in Figure 2.1. However, the distortion in a real image can be made up of a combination of both types of distortions.

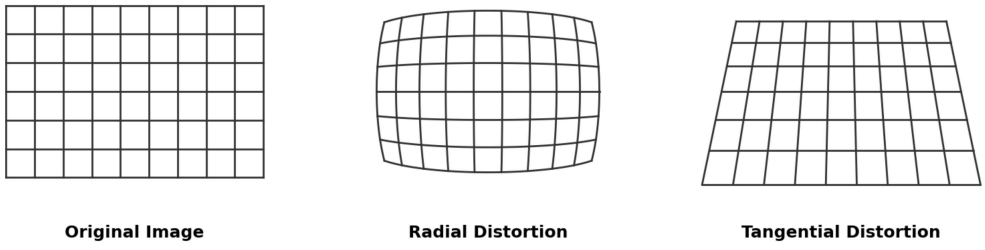


Figure 2.1: A figure depicting the two main types of distortion present in typical camera systems.

There are multiple camera models that incorporate lens distortion into the typical pin-hole camera model [4] [5] [6]. All of these models can accurately simulate distortion to a certain degree, depending on the field of view (FOV) of said camera.

2.1.4 Fish-eye camera model

This thesis makes use of the Kannala-Brandt camera model [5] to govern the transformation of a 3D world point to a 2D image point in the used wide-angle lens camera. Below follows an outline of the mapping from a 3D world point to a 2D image point.
The world point is expressed in the camera coordinate system according to Equation 2.3, where X_c, Y_c and Z_c is the camera point coordinates and X_w, Y_w and Z_w are the world point coordinates. The angle from the sensor plane to these camera coordinates are calculated using the following equation:

$$\theta = \arctan(\sqrt{(\frac{X_c}{Z_c})^2 + (\frac{Y_c}{Z_c})^2}).$$

This angle is then used together with the distortion coefficients to calculate the fisheye distortion angle θ_d given by

$$\theta_d = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8),$$

which is further used to calculate the distorted camera point coordinates x' and y' using

$$x' = \frac{\theta_d}{((\frac{X_c}{Z_c})^2 + (\frac{Y_c}{Z_c})^2)}\frac{X_c}{Z_c},$$
$$y' = \frac{\theta_d}{((\frac{X_c}{Z_c})^2 + (\frac{Y_c}{Z_c})^2)}\frac{Y_c}{Z_c}.$$

These coordinates are finally converted into image point coordinates which lie on the sensor plane, using the following equations:

$$u = f_x(x' + \alpha y') + c_x,$$
$$v = f_y y' + c_y.$$

2.1.5 Reprojection error

Reprojection error is a geometric error corresponding to the image distance between a projected world point and its true, measured 2D projection on the image plane. It is a standard metric used to quantify how accurately an estimated camera model represents the physical environment. To calculate this error, a known 3D world point is projected onto the image plane using the camera's extrinsic and intrinsic parameters. The Euclidian distance between the ground truth image coordinates (u_i and v_i) and their mathematically reprojected counterparts (u_i' and v_i') yields the reprojection error for a single point:

$$E_i = \sqrt{(u_i - u_i')^2 + (v_i - v_i')^2}. \tag{2.5}$$

To evaluate the fit of the camera model across N observed image points, the mean reprojection error is expressed as the average of these individual errors:

$$E = \frac{1}{N} \sum_{i=1}^{N} E_i. \tag{2.6}$$

2.1.6 3D-point triangulation

A world point can be triangulated by back-projecting two image points from two cameras with different poses. This triangulation is visualized in Figure 2.2, and a more in-depth explanation follows below.
The 2D projections of X_1 and X_2 are shown as green circles in the figure and are denoted with x_1, x_1', x_2, x_2'. By back-projecting the coordinates of the image-points of cameras $C1$ and $C2$ through their respective camera matrix P, the blue rays are traced. Each corresponding ray intersects in a 3D location, triangulating the position of the two 3D object points X_1 and X_2 shown as orange crosses.

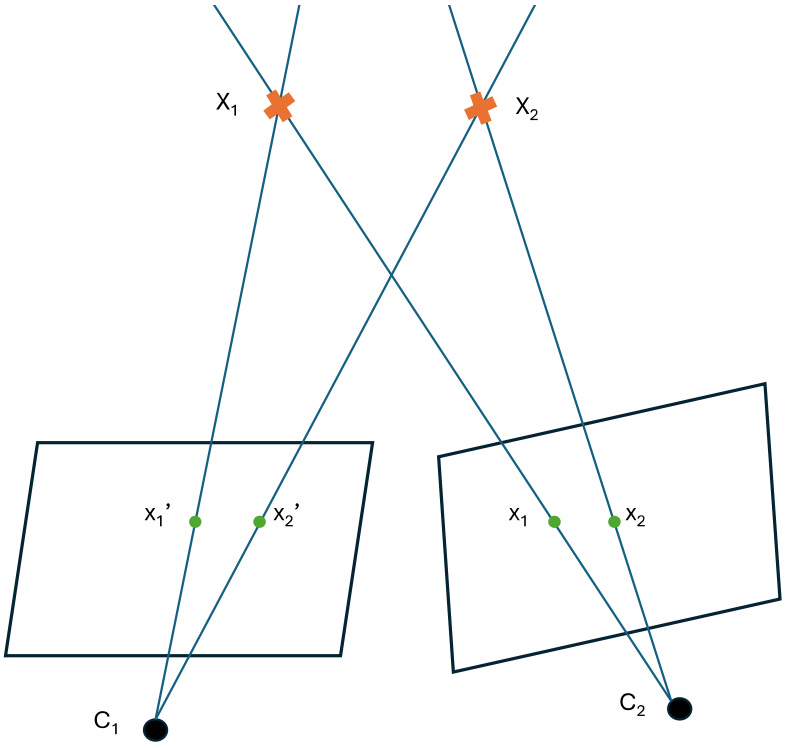


Figure 2.2: Visualization of the triangulation process.

2.1.7 Homography

A homography transformation is not coupled to camera geometry per se, however, it is often used within computer vision. Fundamentally, a homographic transformation is a transformation mapping from one planar surface to another. The result of a homographic transformation is visualized in Figure 2.3. Transformation of a point

(x, y) in the source plane to the corresponding point (x', y') in the target plane is computed for the 3×3 matrix H as

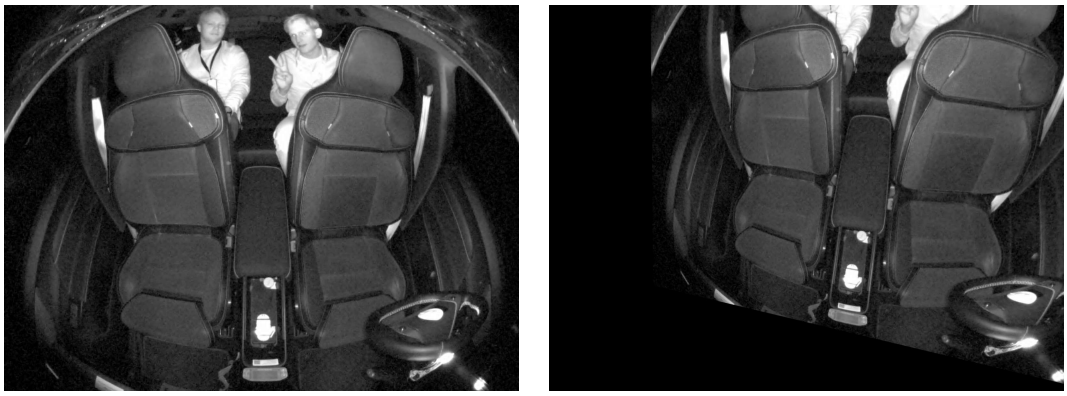
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

A key practical application of this is perspective transformation, which determines how a planar surface appears from different camera perspectives. In that case, the homography matrix H is calculated according to Equation 2.7, where K_1, K_2 are the intrinsic matrices of camera 1 and 2, R, t are the relative rotation and translation between the two cameras, n is a vector orthogonal to the viewed plane and d the distance from said plane to the optical center of camera 1.

$$H = K_2 \cdot \left(R - \frac{t \cdot n^T}{d} \right) \cdot K_1^{-1} \tag{2.7}$$

An important fact for Equation 2.7 to be valid is that the environment viewed by the camera is a 2D plane. This is due to parallax, meaning that a visualized object closer to the camera moves faster relative to the camera than an object far away. This is troublesome since a view of a non-planar environment would require the 3D distance to all corresponding 2D points if the camera experiences a translation for this equation to be valid. This is however not the case for rotation, as a rotation shift all of the pixels equally. For a pure rotation homography transformation ($t = 0$), Equation 2.7 can be rewritten to

$$H = K_2 \cdot R \cdot K_1^{-1}. \tag{2.8}$$



(a) Original image.

(b) Image transformed using a rotation homography matrix.

Figure 2.3: Image transformed using a homography matrix.

2.2 Camera pose estimation

Extrinsic camera calibration or pose estimation is the task of estimating a camera's pose in 3D given a set of image point projections [7]. The derived pose can be

calculated in an absolute or relative fashion. The absolute estimate returns an absolute pose in the world coordinate system, while the relative estimate returns the relative transformation between a reference camera coordinate system to the current one. Below follows a more comprehensive introduction to the two methods.

2.2.1 Absolute pose estimation

This problem is commonly called a Perspective-n-Point-problem (PnP), where n is the number of world points with corresponding image points used. With a requirement of $n \geq 3$ for the solver to yield a finite number of solutions, in the case of $n = 3$, it yields four plausible solutions [8]. The procedure consists of acquiring three or more distinctive world points and their corresponding projection onto the image plane.
Provided the 3D - 2D correspondencies, there exist multiple solvers, some of which date back to 1841, and some that are more recent, such as the method presented by Gao et al. (2003) [9]. The solvers can be roughly grouped into two camps, iterative, and non-iterative solvers. Gao's method is an example of a non-iterative solution where an algebraic expression is used to derive the pose of the camera.
Iterative methods have, however, been proven to be more accurate and flexible [7]. Pan and Wang also argues that the iterative method is generally more robust [10]. A general iterative method works by converting the pose solving problem into a nonlinear least squares problem, where the reprojection error in either image plane (2D) or object space (3D) is minimized. As with any non-linear optimization, the risk of reaching a local minima solution exists, which Pan and Wang mention as one of the drawbacks of an iterative solution together with the higher computational cost [10].

2.2.2 Relative pose estimation

The relative pose estimation relies on epipolar geometry to calculate the transformation between two separate cameras and is widely used as part of the Structure from Motion (SfM) technique. This method works by finding a number of image points, with five as a minimum, corresponding to the same world points in the two camera views. Provided these image points, and a solver such as the five-point algorithm [11], the essential matrix can be retrieved. By applying singular value decomposition on the essential matrix it can be decomposed into its components, a rotation matrix and a normalized translation vector describing the transformation from one camera to the other.
The aforementioned five-point algorithm builds upon a core concept within camera geometry, namely the Epipolar constraint. Below follows a short introduction into this constraint and its governing geometry.

Epipolar geometry

Epipolar geometry is the geometry governing stereo vision, where two cameras view a 3D scene from two separate positions. In Figure 2.4, X is the world point, O_L and O_R are the optical centers of the left and right camera respectively. The plane

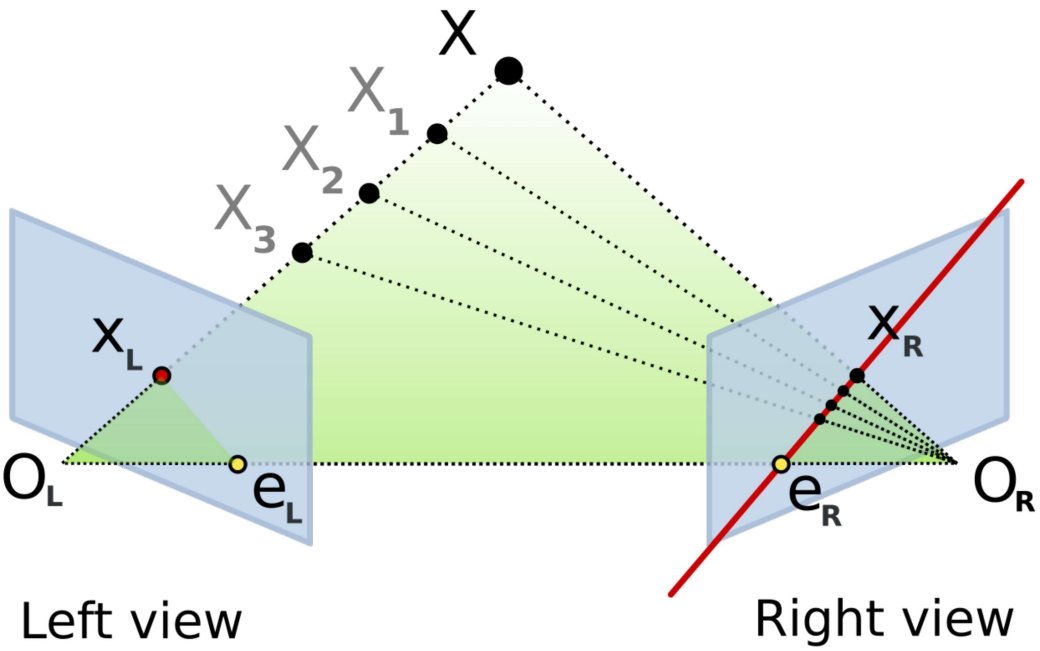


Figure 2.4: A figure showcasing the fundamentals of epipolar geometry [12].

that connects the three points O_R, O_L, X is called the Epipolar plane. The two 2D projections of X are the image points called x_L in the left camera and x_R in the right. The 2D point at which the left camera sees the right camera can be seen as e_L in the aforementioned figure and this point is called the Epipole of the left image.

The line O_L-X is seen as a point x_L by the left camera since it passes through its optical center, whereas the same line is seen as the red line in the right image. This red line is called the Epiline of the point x_L. The same thing can be seen the other way around where the line x_L-e_L is the epiline of x_R.

These epilines create the foundation for a constraint, the so called epipolar constraint. This constraint states that for any X with its corresponding image point x_L in the left camera, you do not have to search the entire view of the right camera to find this X, since this X is constrained to lie on the epiline of x_L. Building upon this constraint, Equation 2.9 can be constructed, where x_1,x_1' are a pair of corresponding projected camera points from cameras one and two of a specific world point, and E is the essential matrix:

$$x_1^T E x_1' = 0. \tag{2.9}$$

Using Singular Value Decomposition (SVD), this E can be further decomposed into a rotation and translation according to Equation 2.10 and 2.11. This rotation matrix R and translation vector t describes the transformation between the two camera poses. From E, this vector t can only be retrieved up to scale, meaning its direction

is known but not its magnitude.

$$[t]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{2.10}$$

$$E = [t]_\times R \tag{2.11}$$

Sampson distance

Sampson distance is a first-order Taylor approximation of the algebraic error of the epipolar constraint in Equation 2.9. This error is present due to imperfect image point matchings, pixel noise, or an inaccurate essential matrix E. The Sampson distance is often used as the objective in iterative solvers used to estimate E. The squared Sampson distance of two image point correspondences x_1, x_1' from two difference views, where the transformation between the two views can be decomposed from the matrix E, is given by

$$d^2 = \frac{({x_1'}^T E x_1)^2}{(Ex_1)_1^2 + (Ex_1)_2^2 + (E^T x_1')_1^2 + (E^T x_1')_2^2}. \tag{2.12}$$

2.3 Feature detection and matching

In this thesis, a feature is defined as consisting of two parts: a keypoint, which specifies its position, and a descriptor, which is a vector that describes its characteristics. Defining sparse local features of an image is a fundamental part of many problems within computer vision. Object detection, 3D-scene reconstruction and pose change estimation rely on scale and rotation invariant features. In these problems, features are used to locate correspondences over different views of the same object. For this, the keypoints provide information of the correspondences position in each view. The descriptors describe the characteristics of each detected keypoint and enable matching between separate views. In an ideal case, the descriptor of two corresponding features from two views, results in the same exact vector, in reality, this is rarely the case making the matching non-trivial. An example of feature detection and matching can be seen in Figure 2.5 where the Chalmers logo is photographed from two different angles.
One of the first solutions to the feature detection problem was Scale Invariant Feature Transform (SIFT), introduced by David G. Lowe in 2004 [13]. Lowe shows that SIFT is scale invariant and also robust against changes in illuminance and addition of noise. This method provides a robust way of identifying the same features across different images, enabling applications like pose estimation. Other papers have shown that different approaches can have other advantages. ORB, presented by Rublee et al. (2011), is another feature detector that is orders of magnitude faster than SIFT [14]. This feature detector is rotation invariant and much more resistant to high levels of noise than SIFT, but in contrast to SIFT, it is not scale invariant.

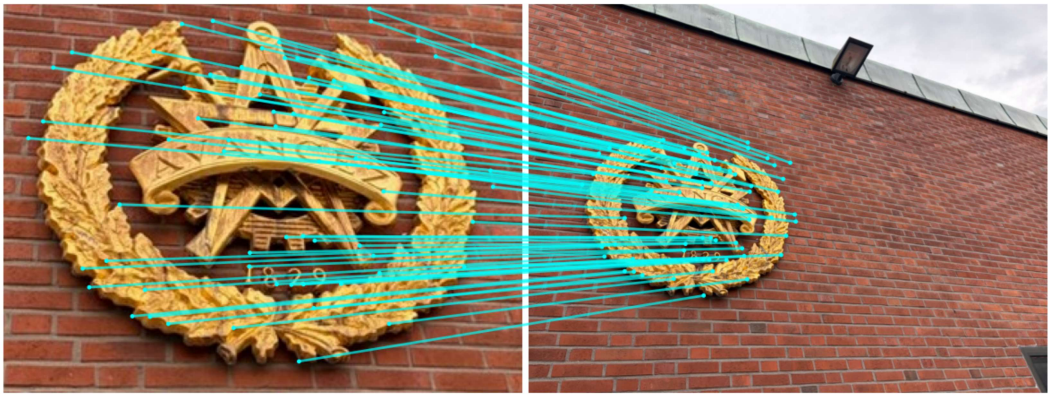


Figure 2.5: Matched features between two different images of the Chalmers logo.

Features are useful on their own, but to solve the problems of image matching and 3D-scene reconstruction, feature matching is needed. The optimal method to match features is to evaluate the descriptors of the features and identify the nearest neighbor among all of the other features [13]. This way of matching is known as brute-force matching. How the nearest neighbor is found depends on what kind of descriptor is used. For vector based features like SIFT, it is based on a linear search of the euclidean distance of the descriptor vectors [13]. Binary features like ORB often use Locality Sensitive Hashing (LSH) to find the nearest neighbor [14]. The features can be matched using these exact methods, but this is not very efficient, and not suited for all applications. To solve this, there are more sophisticated and fast approaches to matching, for example, Fast Approximate NN (FLANN) matching. FLANN presents faster solutions to matching both vector based and binary based feature descriptors with little or no sacrifice on precision [15].

2.3.1 Learning-based methods

Machine learning (ML) and more specifically, neural networks have shown promising results for feature detection and matching [16]. Recent work on dense feature matching have increased the robustness of correspondence estimation under large changes in viewpoint and illumination. In a conference paper, Edstedt et al. (2023) presents RoMa, a state-of-the-art feature detector and matcher built on a neural network [17]. RoMa achieves large gains in difficult benchmarks, showing strong pose estimation performance thanks to its superior robustness compared to previous approaches.

Xfeat is another neural network-based feature detection and matching method that is made to be efficient and fast, enabling it to run smoothly on resource-limited devices [18]. There are also methods that only perform feature matching, like LightGlue. LightGlue is a feature matcher that can use classical detectors like SIFT, or neural network-based solutions, to detect features [19]. SuperPoint is one of these learning-based feature detectors, and is based on a convolutional neural network [20]. SuperPoint and LightGlue are presented more in depth below.

SuperPoint

SuperPoint is a fully-convolutional neural network feature point extractor, presented by Detone et al. [20]. The network can be seen in Figure 2.6 and consists of one encoder and two parallel decoders, with its structure enabling the input to be an entire image rather than a patch of an image. The encoder manages to decrease the spatial dimensionality to one eighth of the input by combining convolutional layers, pooling layers, and non-linear activation functions.

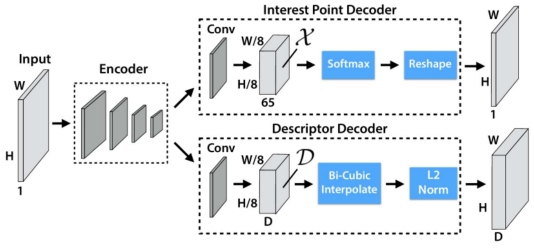


Figure 2.6: The network architecture of SuperPoint visualized [20].

This is done by grouping the pixels in patches of 8×8 and then storing them depth-wise in the resulting output tensor. This results in fewer computations during training while at the same time enabling the retrieval of the original dimensionality by reshaping, avoiding the possible issue of having to upsample on the decoder-side of the network. The same output tensor is forwarded to the two decoders, the keypoint decoder and the descriptor decoder, respectively. The keypoint decoder computes a probability of each pixel containing a keypoint and the descriptor decoder computes fixed length L2-normalized descriptors.
This entire network has then been trained in a self-supervised manner. Detone et al. initially trained the detection part of the network on labeled images containing geometric shapes, the resulting network they call MagicPoint. They further show that MagicPoint greatly outperforms classical corner detection methods such as FAST [21] and Harris corners [22] on a held-out test set of the aforementioned geometric shapes dataset. These differences were especially prominent when noise was introduced to the data.
Detone et al. further uses this MagicPoint together with Homographic adaptation to create pseudo-ground truth labels on previous unlabeled images. This data is then used to train the resulting SuperPoint network, where the prediction of the detection part of the network is compared to the pseudo-ground truth labels. For the descriptor training, two images are fed into two separate SuperPoint networks, network 1 retrieves the original input image, whilst network 2 gets the same image that has been altered with a random homography transform. The descriptor loss is then based on the similarity of the pairs of corresponding descriptors from the two networks. The entire network is trained using the combined loss of the interest point and descriptor encoders.
Detone et al. conclude their report by sharing results on homography estimation on the HPatches dataset using SuperPoint, where they state that SuperPoint shows high repeatability in its detection and a performance that exceeds methods such as LIFT [23] and ORB [14] when used for estimating the homography. The license of

SuperPoint allows for commercial use, which was one of the constraints of the thesis work.

LightGlue

LightGlue is a deep neural network that matches feature points between two images [24]. It is a continuation of the work on SuperGlue, presented by Sarlin et al. in 2020 [25]. Multiple parts of the network have been reworked to make LightGlue faster, more efficient and more accurate than SuperGlue. The network visualized in Figure 2.7 processes feature points from both images jointly through a transformer backbone, using both self-attention and cross-attention units at each layer. This enables the model to capture the internal geometry of each image and the correspondences between them to effectively match feature points and reject outliers. This outlier rejection combined with early stopping also makes LightGlue much more efficient than brute force matching methods. Much like SuperPoint, the LightGlue license allows for commercial use, which was a constraint of the thesis.

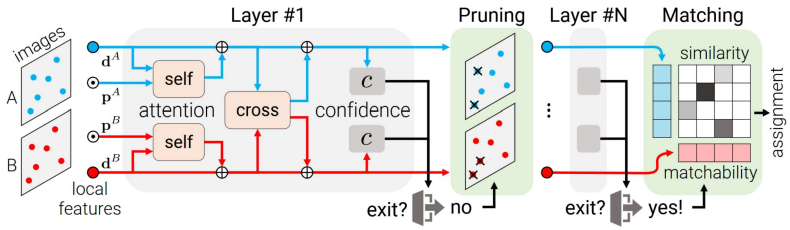


Figure 2.7: The network architecture of LightGlue [24].

2.3.2 End-to-end ML methods

In addition to the aforementioned feature detection and matching methods, there are also end-to-end ML approaches. These perform feature detection and matching, and finally include a pose estimation. Kendall et al. (2015) demonstrated that a convolutional neural network could be trained to calculate the 6DOF pose from a single image [26]. This estimation worked indoors and outdoors and had a respectable error of 0.5m and 5 degrees indoors. Building on this, Chaudhry et al. (2025) showed that deep learning models could be used to calculate both extrinsic and intrinsic camera parameters [27].

2.4 Robust model fitting

A challenge in processing real world data is that it is inherently noisy. The performance of the state estimators mentioned in Section 2.2 is greatly affected by the accuracy of the correspondence between the world points and their corresponding image points or the 2D to 2D image points matches. This is even more pronounced if the correspondences are few, since the addition of one outlier to a small set of

inliers has a greater impact on the ratio between them than it would if the set of inliers was larger.
To solve this problem, Martin A. Fischler and Robert Bolles presented their method, Random Sample Consensus (RANSAC), back in 1981 [28]. The method works by iteratively trying to fit a model to experimental data. Initially, the model is fitted to as few data points as possible, from there, the number of points closely surrounding the model indicating that they are compatible with the model are saved and used to update the parameters of the model. Points that do not align with the intended model, meaning they exceed a deviation threshold τ from the model, are excluded from the model estimation. Martin and Robert conclude their report by stating that with RANSAC they have introduced a new paradigm for fitting a model to experimental data. By interpreting and smoothing data that contain significant portions of gross errors, it is ideally suited for applications in automated image-analysis where interpretation is based on data provided by error-prone feature detectors [28].
For long, RANSAC was the go-to method for robust model fitting, however, the original has some inherent issues related to choosing its tuning parameters such as its deviation threshold. To tackle this problem, more modern adaptations of RANSAC exist, examples of these are, LO-RANSAC , MAGSAC++ , GC-RANSAC and USAC [29][30][31][32]. Among these modern iterations, MAGSAC++ [30] fundamentally redefines how data points are evaluated by eliminating the use of a strict, manually tuned deviation threshold τ.
In classical RANSAC, the selection of τ is critical. With this, a point is classified as either an inlier or an outlier based on whether its residual falls below this hard boundary. MAGSAC++ removes this limitation by introducing a marginalization approach. Instead of choosing a single threshold, the algorithm marginalizes over a continuous spectrum of noise scales σ up to a maximum expected threshold τ_{max}. By treating the noise scale as an unknown variable, every data point is assigned a weight based on its likelihood of being an inlier across all considered noise levels. This allows the algorithm to remain highly robust even when the precise noise characteristics are unknown.
Beyond threshold marginalization, MAGSAC++ introduces major speed improvements over its predecessors. It uses an optimized weight updating strategy that significantly accelerates the model refinement process. Additionally, it integrates an advanced sampling method, rather than picking random points blindly across the entire image as RANSAC does, the sampler leverages that true matches are usually clustered close to one another in physical space. This allows MAGSAC++ to find a set of true inliers in fewer iterations.

2.4.1 Robust relative pose estimation

Algorithm 1 shows the outline of an algorithm exemplifying how RANSAC can be used for essential matrix estimation. The algorithm makes use of two sets of corresponding points and the deviation threshold τ as arguments.

Algorithm 1 Essential Matrix Estimation with RANSAC Outlier Rejection Algorithm

```
bestEssentialMatrix = null
maxInlierCount = 0
iterations = calculateMaxIterations(confidence = 0.95, outlierRatio = 0.5, sampleSize = 5)
for i = 1 to iterations do
    sample1, sample2 = randomlySelectMinimalSamples(points1, points2, sampleSize = 5)
    hypothesesE = solve5PointAlgorithm(sample1, sample2)
    for each E in hypothesesE do
        inliers = []
        for each p1, p2 in points1, points2 do
            error = computeSampsonDistance(p1, p2, E)
            if error < threshold then
                inliers.append(p1, p2)
        if length(inliers) > maxInlierCount then
            maxInlierCount = length(inliers)
            bestEssentialMatrix = E
            iterations = updateIterations(maxInlierCount, totalPoints, confidence)
finalEssentialMatrix = refineModelLeastSquares(bestEssentialMatrix, inliers)
return finalEssentialMatrix
```

3
Methods

This thesis investigates a vision based approach for estimating the pose of a camera on the inside of a car. The method is based on the identification of the interior features of the car, without any modifications. The features of an image are matched with the features of another image to calculate the camera's pose. To solve the actual calibration, two methods are proposed, the Relative Method in Section 3.3 and the Absolute Method in Section 3.4. The Relative Method calculates the relative pose change from a nominal reference camera to the un-calibrated camera by comparing their views. The Absolute Method uses two cameras with known poses to calibrate a third camera.
The methods and tools used within the two suggested pipelines are implemented with the help of OpenCV [33], an open source computer vision library.

3.1 Data collection

To be able to test and validate the pose estimation pipeline, a lot of data is needed. The images have to be representative of real world scenarios, in this case, differing mounting poses. The pose is assumed not to rotate more than $\pm 3°$ and not to translate more than ± 3 mm along each axis of the camera coordinate frame. To test that the pipeline is capable of estimating all real world scenarios, images of random combinations of rotations and translations must be collected.
Validating how well the pipeline performs is done by creating synthetic images with known pose transformations. Thanks to knowing the ground truth pose, the error can easily be calculated, and potential biases become clear. However, this does not ensure that the pipeline works on real world vehicles. Therefore, real images must also be collected. Images from different vehicles of the same model is desirable to evaluate how the model performs in representative real world scenarios.

3.1.1 Synthetic images

Synthetic images are collected using a simulated environment in Unity [34]. The images are generated using a simulated model of a vehicle that contains both a camera and light sources. To emulate the information collected by the real camera, only one of the color channels is used to capture the images. The real camera system captures IR images, and uses IR lamps to illuminate the interior. To mimic this, the simulated camera only uses the red color channel, as this is the closest frequency-wise to IR, and the scene is illuminated with red light. By making the

simulated images as similar as possible to real data, the system can be developed using ground truth pose data while minimizing compatibility issues with the real system. Additionally, because the pose estimator requires a reference image with a known pose during real-world calibration, a simulated image can be used. Feature matching between the simulated and real images is significantly easier when the two datasets are visually coherent.
Since many images with different poses must be captured, the process of capturing images is automated. This automated process captures as many images as needed with randomized poses within the $\pm 3°$ and ± 3 mm range. Each axis can be changed individually resulting in 7^6 generated images, since all six axes are changed in seven integer steps from -3 to 3. Saving all $7^6 = 117,649$ possible images would generate an excessive amount of data. Instead, a random sample of the poses is saved to reduce the size of the dataset while avoiding systematic bias.
To facilitate the process of developing and testing parts of the system, datasets originating from the same environment with different transformations are created. The different transformations are pure translation, pure rotation and transformation. In Figure 3.1, four images are shown, three belonging to each of the aforementioned datasets and the fourth being an image taken with the nominal pose of the camera.
In addition to the synthetic data mentioned above, synthetic data depicting the vehicle in a sunny outdoors environment is gathered. For this, the camera has been shifted and rotated according to the same tolerances as with the synthetic data. This data is purely used to evaluate the robustness to different lighting conditions of the system. An example of an image from the natural lighting dataset can be seen in Figure 3.2.

3.1.2 Real images

Real images were collected from two separate vehicles of the target vehicle model. To test that the pipeline works in real world conditions, images are collected from the two vehicles in varying lighting conditions. The real images are also used to validate the performance on real world data. An image of the interior of a vehicle of the target model is shown in Figure 3.3.

3.2 Intrinsic camera calibration

The cameras are intrinsically calibrated using the methodology described in Section 2.2.1, and implemented using OpenCV functionality, an overview can be seen in Algorithm 2. The algorithm makes use of three arguments, a folder containing images of the calibration board with different poses, the size of the internal checkerboard pattern, and the size of the checkerboard squares in millimeters.
To simplify the calibration, the world points are generated as a list containing the 3D coordinates reflecting where each black square intersects another black square in a nominal chessboard pattern. Furthermore, the board is assumed to be flat, which means $z = 0$ for all intersections, and the x, y coordinates increase according to the pattern and square size. This means that, even though the checkerboard that is present in the images is moved between separate images, this is not apparent to

(a) Nominal pose [0, 0, 0, 0, 0, 0].

(b) Translated pose [-2, 0, 3, 0, 3, 0].

(c) Rotated pose [0, 3, 0, -2, 0, 3].

(d) Transformed pose [-3, 2, 3, 2, 2, 0].

Figure 3.1: Four synthetic images demonstrating different camera poses in $[x, \phi, y, \theta, z, \psi]$ order.

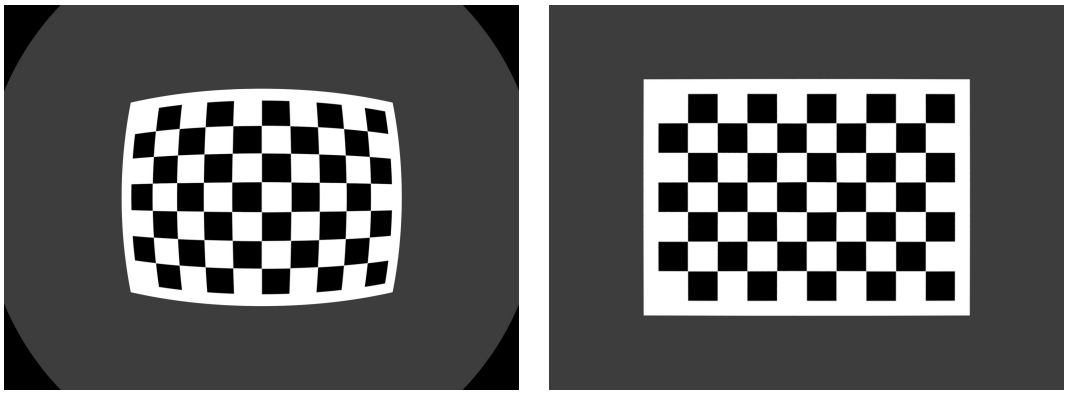
Figure 3.2: Synthetic image with the vehicle in a natural lighting condition.

Algorithm 2 Intrinsic Camera Calibration Algorithm

```
procedure IntrinsicCalibration(img_folder, patternSize, squareSize)
    objPoints ← generateNominal3dChessboardCoordinates(patternSize, squareSize)
    imagePoints ← []
    for each image in imgFolder do
        imgP ← findChessboardCorners(image)
        if imgP is found then
            imagePoints.append(imgP)
        end if
    end for
    cameraMatrix, distCoeffs ← calibrate(objPoints, imagePoints)
    calibrationError ← []
    for each image in imgFolder do
        imgP ← findChessboardCorners(image)
        rotation, translation ← solvePnP(objPoints, imgP, cameraMatrix, distCoeffs)
        imgPoints2 ← projectPoints(objPoints, rotation, translation, cameraMatrix, distCoeffs)
        error ← norm(imgP, imgPoints2, L2)/numberOfPoints
        calibrationError.append(error)
    end for
    return cameraMatrix, distCoeffs, calibrationError
end procedure
```

Figure 3.3: Real image captured from the interior camera of a vehicle in a garage setting.

the algorithm. The algorithm sees the board as static and the camera's position as dynamic between images. This in turn does not affect the intrinsic parameters, and since the extrinsic parameters are not of interest in this calibration, this implementation makes the retrieval of exact 3D coordinates of the board much easier.
By using OpenCV's fisheye.calibrate() on the generated object points and all corresponding image points contained in the files in the calibration folder, the intrinsic parameters and the distortion coefficients of the camera are found.
To validate the calibration-method and its parameters, a reprojection error is calculated. This is done by projecting the nominal world points through the retrieved camera matrix and comparing the resulting 2D projected image points to its corresponding match in the original image. In addition to doing this during calibration, the same reprojection is done post-calibration on a hold-out evaluation dataset.
For the virtual camera, images of a chessboard pattern placed in the Unity environment facing the simulated camera are captured, whereas for the real camera, images of a chessboard pattern are captured using the interior camera of a vehicle.
The retrieved parameters from the real car are assumed to be representative for all cars having the same type of camera, limiting this calibration to be performed only once. In Figure 3.4a, one of the synthetic calibration images can be seen and in Figure 3.4b, that image is undistorted. In Figure 3.5a, one of the real images can be seen, and in Figure 3.5b, that image is undistorted.

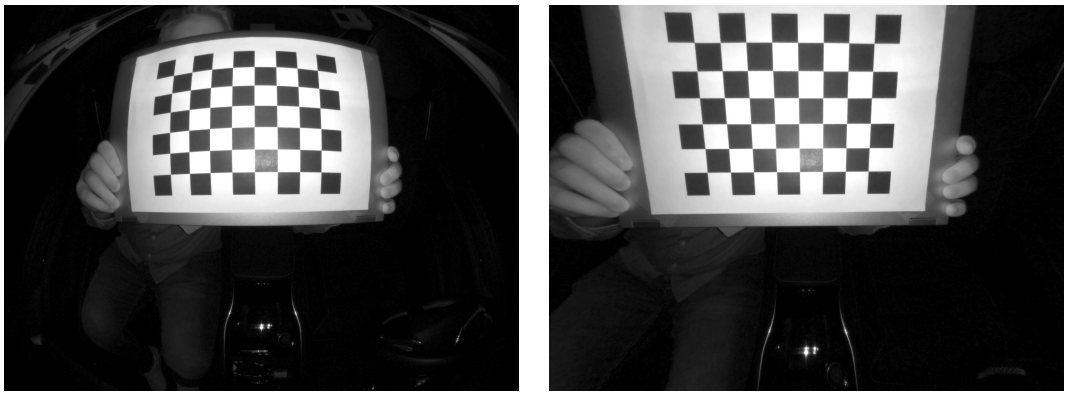


(a) Image of a checkerboard. **(b)** Undistorted image of a checkerboard.

Figure 3.4: Images of a checkerboard from the simulated camera.

3.3 Relative pose estimation pipeline

The implemented relative pose estimation pipeline is shown in Figure 3.6, and can also be seen in a larger format in Figure A.1. It takes two images with their corresponding camera intrinsics matrix and distortion coefficients, and outputs the relative pose change between the two images. One of the images is from a camera that is considered to have a nominal pose, and the second image is taken using a camera with an unknown pose. With this information, the second camera can be calibrated, and its view changed to match the pose of the nominal camera.



(a) Image of a checkerboard. **(b)** Undistorted image of a checkerboard.

Figure 3.5: Images of a checkerboard from the real camera.

In a production setting, the reference image may be either a synthetic nominal image from a virtual model or a real image from a calibrated nominal vehicle. This method requires only one nominal reference view, making either option feasible.
This method is referred to as the 'Relative Method'.

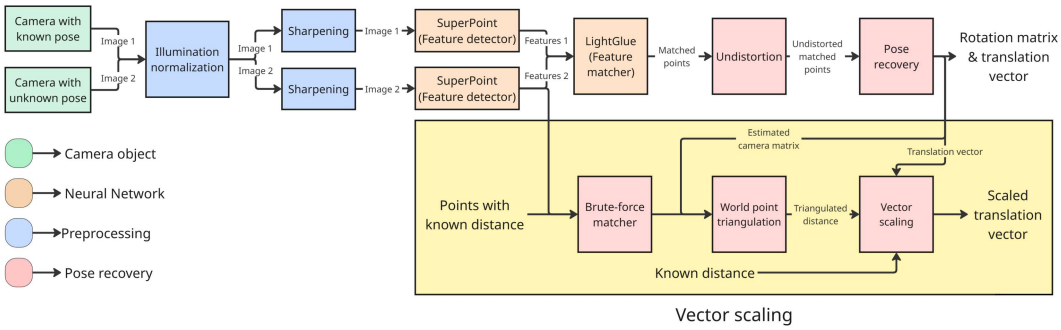


Figure 3.6: Overview of the Relative Method pose estimation pipeline.

3.3.1 Preprocessing

Before feature extraction and matching is performed, both images go through preprocessing in order to improve matching robustness. The first step is to give the images more similar illumination levels. The illumination processing normalizes the illumination between the images by matching their cumulative histograms (CDFs). The CDF for each image is calculated and fused to create a combined target CDF. The pixel intensities of both images are then remapped to match this target distribution. This equalizes the intensity levels of the images while preserving their structural content, allowing better feature matching. This illumination normalization method is heavily inspired by the method proposed by Daud et al (2012) [35]. The images are then passed through an edge sharpening filter to make the image features more distinct. In Figure 3.7, preprocessing has been performed on one sim-

ulated image and one real image, pictured in Figure 3.7a, 3.7b respectively. The results of the preprocessing can be seen in Figure 3.7c,3.7d.

(a) Unprocessed synthetic image.

(b) Unprocessed real image.

(c) Synthetic image after preprocessing.

(d) Real image after preprocessing.

Figure 3.7: Images of vehicle interiors before and after preprocessing.

3.3.2 Feature detection

Features from both images are extracted using SuperPoint, up to 2048 points are collected from both images. For each point, the network outputs a coordinate x, y, a confidence score c and a 256-dimensional descriptor vector d. For this thesis, no adaptation or training was performed on the SuperPoint network. Instead it is applied to the problem as is, with the network and weights as presented in the report by Detone et al. [20]. The points detected by SuperPoint on a simulated image can be seen in Figure 3.8. The feature extraction in the pipeline does not have to be done by SuperPoint, and can easily be exchanged with extractors such as SIFT or ORB.

3.3.3 Feature matching

The features found by SuperPoint are matched using LightGlue. LightGlue establishes correspondences between the extracted feature sets. In Figure 3.9, matches

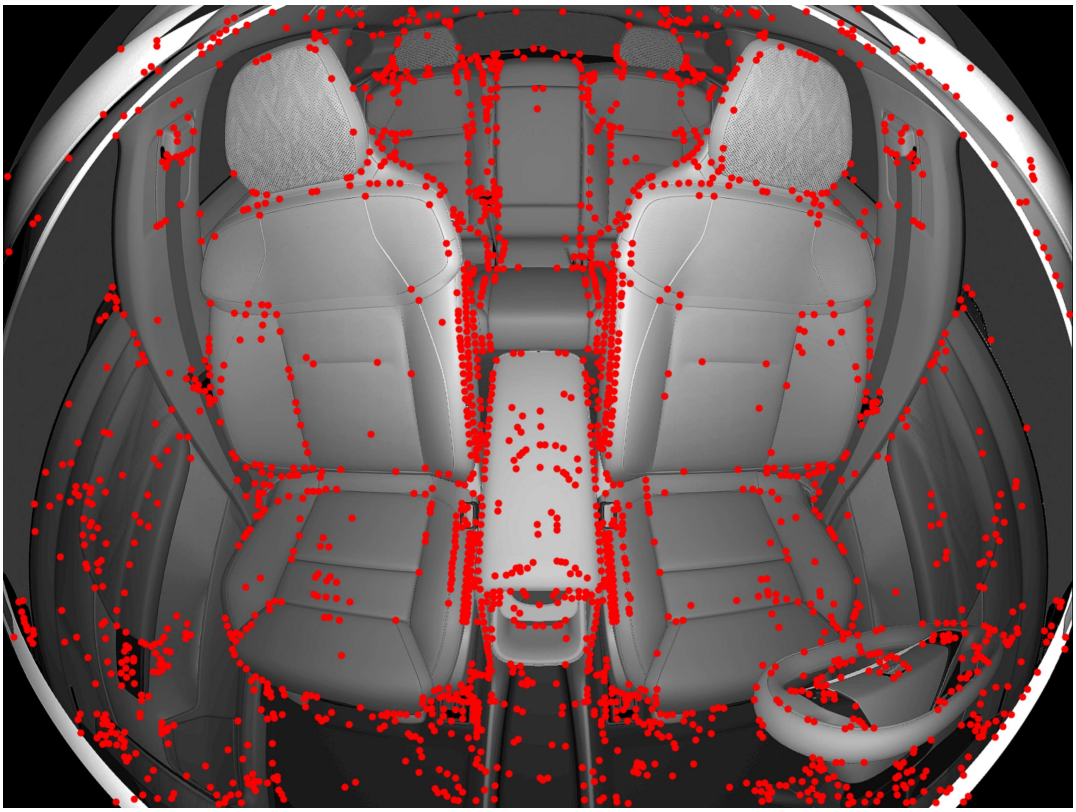


Figure 3.8: SuperPoint features found.

between a simulated image with a nominal pose and an image rotated −3° about x, 3° about y and 3° about z are shown. In Figure 3.10, only 1% of these matches are chosen, making each match more visible. The LightGlue matching in the pipeline can easily be exchanged with one of the more classical matching methods mentioned in Section 2.3.

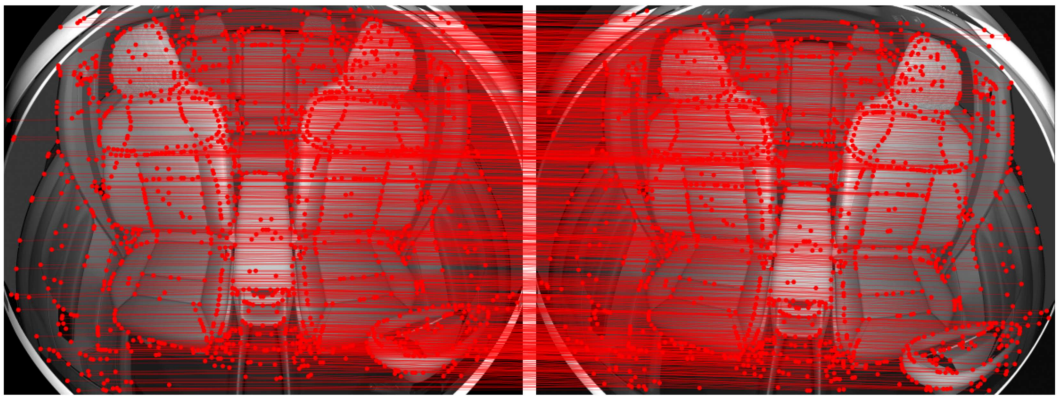


Figure 3.9: Feature matches found by LightGlue.

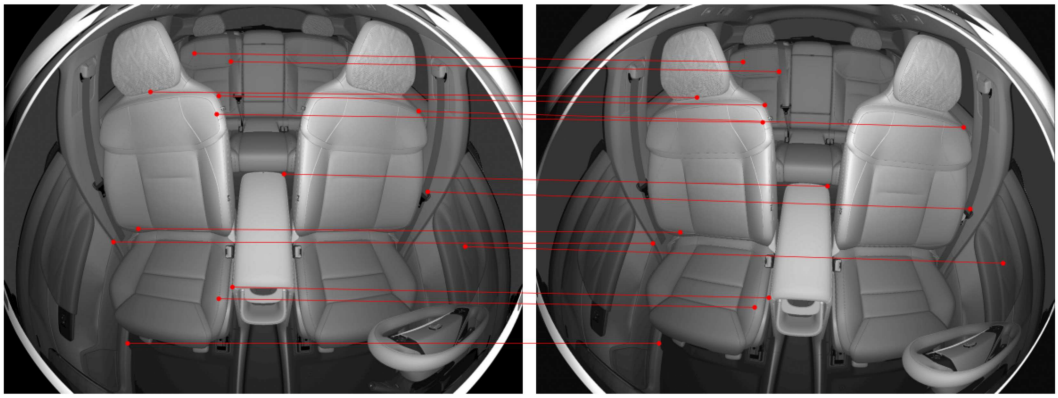
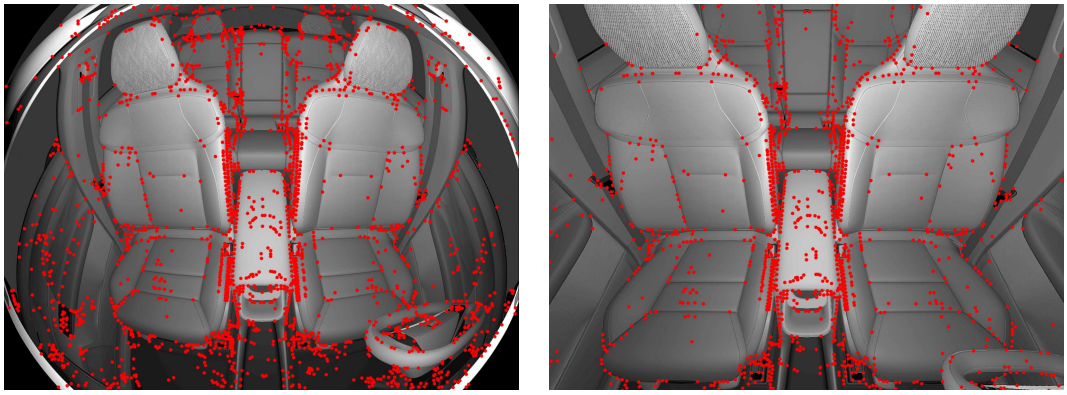


Figure 3.10: 1% of the matches found by LightGlue.

3.3.4 Undistortion

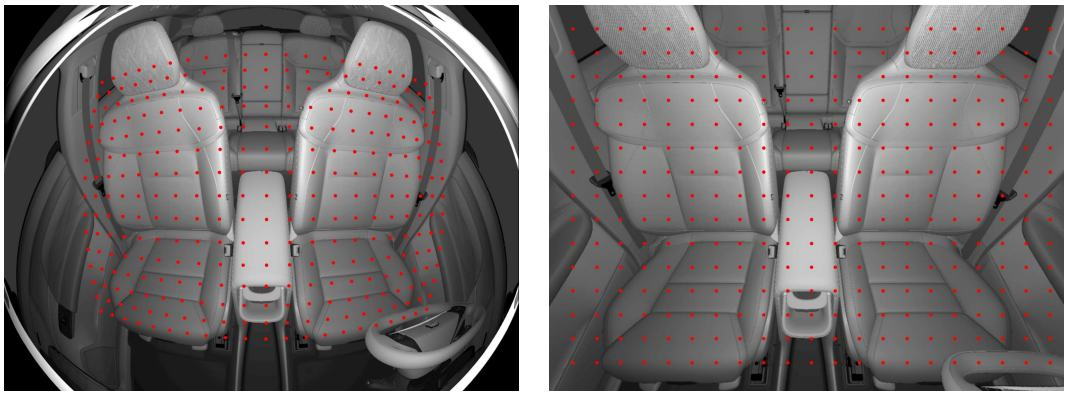
After feature matching, the keypoints of the matched features are undistorted. Undistortion corrects the point coordinates for lens distortion, ensuring real world straight lines appear straight in the projected 2D geometry. This is done using the camera intrinsic matrix and the distortion coefficients retrieved from the calibration mentioned in Section 3.2. In Figure 3.11a, the detected features are shown on top of the image, and in Figure 3.11b, these points are undistorted, as well as the image. The importance of undistorting the points is made clearer in Figure 3.12. In Figure 3.12a, the points have been distorted to match the distortion of the simulated camera. In Figure 3.12b, both the image and the points have been undistorted using the intrinsics of the camera.



(a) Features on top of the original image.

(b) Undistorted features on top of the undistorted image.

Figure 3.11: Simulated images with detected SuperPoint features.



(a) Grid of points on the original image.

(b) Undistorted grid of points on top of the undistorted image.

Figure 3.12: Simulated images with a grid of points.

3.3.5 Pose recovery

The relative pose is then estimated using the undistorted image point correspondences. First, the essential matrix is calculated using the five-point algorithm solver [11] applied to the matched and undistorted image points. The five-point algorithm only considers five image point correspondences. These are initially chosen randomly, but they are iteratively refined using MAGSAC++. The MAGSAC++ threshold is scaled using the focal length, since the undistorted points are expressed in camera coordinates which are normalized to the camera's focal length. This new threshold is calculated using the equation below.

$$\text{scaled threshold} = \frac{\text{outlier threshold}}{\text{focal length}}$$

Finally, the camera rotation matrix R and the normalized translation vector t are extracted from the essential matrix using SVD. With the retrieval of the essential matrix and the intrinsic matrix from Section 3.2, the complete camera matrix P for the previously non-calibrated camera is known up to scale.

3.3.6 Translation vector scaling

The inherent issue with working strictly in 2D is scale ambiguity, meaning that the perceived scale of an object in the 2D projection is highly dependent on its absolute distance from the camera. Due to this problem, the translation vector retrieved in the previous section is normalized and of unit length, having no real world interpretation.

To solve this scale ambiguity issue, a scale factor from unit length to millimeters is calculated. This is done by triangulating two 3D points and comparing the distance between them to the real world distance.

The triangulation is done by iterating through all features found in the non-calibrated camera view and match it to previously stored features from the nominal image using a brute-force matcher. The stored features correspond to 2D projections of easily recognizable 3D world points of which the distance between them have been measured in the physical car. These stored features are called landmarks and their image points are back-projected into 3D, triangulating the two world points corresponding to the features. The euclidean distance between these points is compared to the real world distance and the scale factor is calculated as

$$s = \frac{\text{real world distance}}{\text{triangulated distance}}$$

and this scale factor is multiplied to the normalized translation vector as

$$t = s \cdot t_{norm}$$

where t is the final translation estimate.

3.4 Absolute pose estimation pipeline

This estimation method combines world point triangulation and the absolute pose estimation methodology presented in Section 2.2.1, where the relationship between corresponding image and world points is used to retrieve the camera's parameters. The pipeline is pictured in Figure 3.13, and can also be seen in a larger format in Figure A.2.
The method makes use of three separate images, two images from cameras with known poses, and one image from the camera that should be calibrated.
From this point, this estimation method is referred to as the 'Absolute Method'.

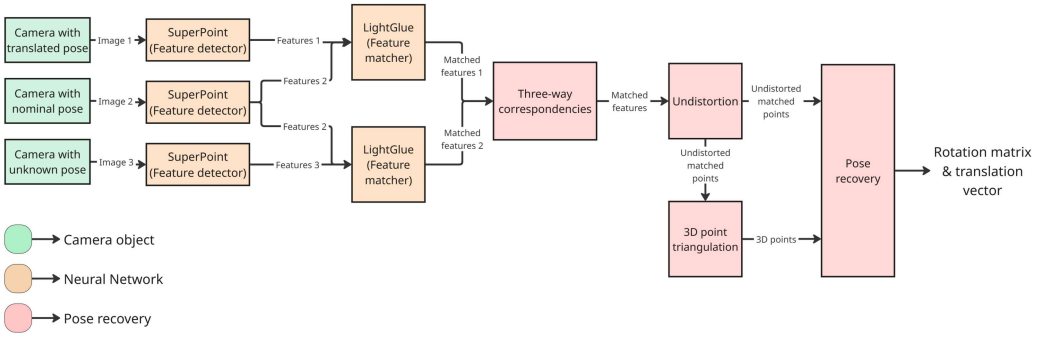


Figure 3.13: Overview of the Absolute Method pose estimation pipeline.

3.4.1 Feature detection & matching

Features are found in the three separate images using SuperPoint and matched with LightGlue. LightGlue can only match two sets of features at once, however, the entire pipeline relies on accurate correspondences over all three images. To solve this, a three-way correspondence is created according to

$$M = (F_1 \cap F_3) \cap (F_3 \cap F_2) = F_1 \cap F_2 \cap F_3$$

where M is the resulting matched set derived from the feature sets F_i originating from images $i = 1, 2, 3$.

3.4.2 Undistortion

The resulting image points from the previous step are undistorted to restore real world geometries. Since the end goal is to estimate the extrinsic parameters, this is done to the image points from all three images. However, it could also only be applied to the sets from the cameras with known poses if the objective was to estimate the entire camera matrix, but this would require small changes to the rest of the pipeline.

3.4.3 World point triangulation

The sets of image points in the matched set M from the two cameras with known poses are together with their camera matrices used to triangulate a set of 3D world points as per Section 2.1.6. These world points are then the 3D counterpart of the image points of M originating from the un-calibrated camera. This results in a set of 2D - 3D correspondencies necessary to perform PNP calculation.

3.4.4 Pose estimation

A PNP solver is applied to the aforementioned set of 2D - 3D correspondencies, for this, the method makes use of OpenCV's solvePnPRansac(). This solver iteratively estimates the camera's pose using RANSAC for outlier rejection. In each iteration, RANSAC includes four of the correspondence points to retrieve an estimate of the pose, using this pose, the world points are projected into 2D and a reprojection error comparing these 2D points to the original image points is calculated. By redoing this for all points, a set of inlier image-world-point pairs is established, and this set is used for the final iterative estimation of the pose.

3.5 Evaluation

The systems are developed in a domain that consists of simulated images but is aimed to solve an issue in the real world domain. This means that it is insufficient to evaluate the performance in only one of these domains. Three evaluation methods are presented. While some of these methods are exclusive to either the simulated or real world domain, combining them provides a comprehensive assessment, ensuring that strong simulated results translate effectively to real world performance.

3.5.1 Ground truth comparison

As mentioned in Section 3.1.1, a lot of data is gathered in a simulated environment where the camera view can be changed freely. By step-wise changing the pose of the camera and saving the image taken together with information about the pose for the said snapshot, data containing the ground truth pose can be gathered. This makes it possible to compare the estimated pose of the camera with its ground truth pose and recover a numerical expression of how well the estimate is. The ground truth pose is stored according to a fixed coordinate frame, originating from an assumed nominal position of the camera. From this position, translations along the fixed coordinate frame in X, Y, Z are described in millimeters, and rotations are described extrinsically using Euler angles expressed in degrees.
The accuracy of the estimation is evaluated using the combined Mean Absolute Error (MAE) rotation and translation-wise over all axes. Strict rotation-errors are also expressed using the MAE of the Geodesic distance, which is the shortest path between two rotation matrices. In addition to this, the distribution of error per axis is fitted with a Gaussian distribution and its mean value μ and standard deviation σ are calculated.

In order to evaluate the robustness to different lighting conditions of the system, the same test will be run on the natural lighting dataset mentioned in Section 3.1.1.

3.5.2 Point mapping error

Reprojection error is commonly used to evaluate both extrinsic and intrinsic camera calibration. This measure is evaluated by calculating the euclidean distance between two corresponding image points, as mentioned in Section 2.1.5. This measure requires a combination of image and world points, which the Relative Method does not inherently have access to, whilst the Absolute Method does and typically makes use of to iteratively improve its estimate.
The Relative Method, on the other hand, typically makes use of the Sampson distance mentioned in Section 2.2.2 to evaluate how well the estimated essential matrix fits the experimental data. This measure is also typically used to iteratively exclude outliers and improve the estimate.
To enable fair cross-method comparisons on the suggested methods, both measures are implemented using a hold-out set called Landmarks. A more in-depth explanation of the implementation is found below.

The evaluation landmarks dataset

As part of the translation vector scaling in the Relative Method, a dataset was created containing fixed recognizable points in the car. Their positions in the nominal frame have been manually annotated and then linked to a SuperPoint feature within 5 pixels of that point. The locations and descriptors of the features matched to the manually marked landmarks are saved in a database together with an ID. The ID, location, and descriptor of a known point in the interior is called a landmark. To make sure that all of these landmarks are found in both synthetic and real images, two separate databases are created by doing the same process for the nominal synthetic image and a real image. These landmark image points are visualized as green dots in Figure 3.14 and Figure 3.15.
Since the positions of these landmarks are known in different views, a second view of these landmarks is used to triangulate the 3D world points of these landmarks. This completes the necessary data to perform both the reprojection error and the sampson distance evaluation. However, as explained in the introduction to this section, the estimates make use of these measures during optimization, which could result in each of the estimation methods having a biased result towards their optimization method if not treated.
To solve this problem, features of the Landmarks dataset are actively excluded from the features used to perform the estimation. This ensures that the features used for the estimation are not the same as those used for the evaluation. However, a downside of this is that the pose estimate might be less accurate due to this exclusion. The landmarks have been chosen in a way that they are positioned on static objects, making them ideal for evaluating reprojection error between environments where dynamic objects, such as seats, have been moved. Similarly, these points would also be ideal to use for the pose estimate in the same scenario, resulting in a worse estimate than if no evaluation and no exclusion were done.

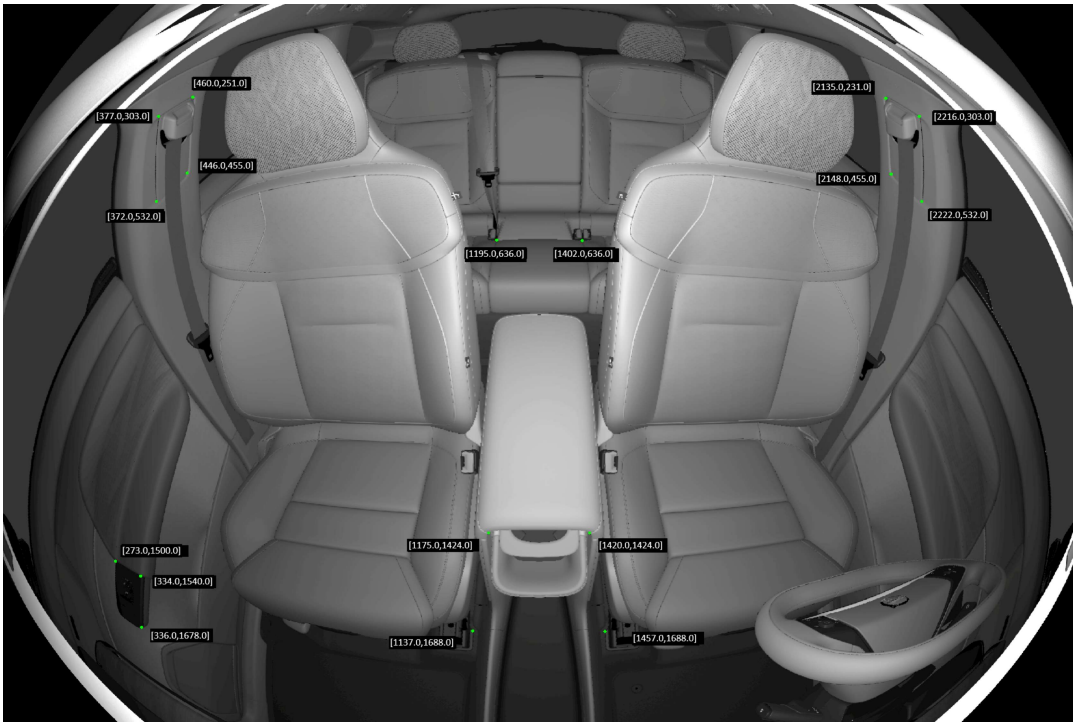


Figure 3.14: The nominal synthetic image with the landmarks highlighted in green with their location in the image.

Figure 3.15: A real image with the landmarks highlighted in green with their location in the image.

Reprojection error

To evaluate the accuracy of the pose estimation pipelines, the theoretical reprojection error is applied to the detected landmarks. First, the landmarks are localized within the target image and associated with their corresponding 3D world point from the database. To establish a baseline metric, these 3D points are projected onto the image plane using the camera's intrinsic parameters, the fisheye distortion model detailed in Section 2.1.4, and an assumed *nominal* pose. The error between these projected image points and the image points of the localized landmarks yields the baseline reprojection error.
Next, this identical projection is repeated using the newly estimated extrinsic pose. By evaluating the difference between the reprojection error of the estimated pose and the nominal baseline, the performance of the pose estimation can be quantified. A decrease in this error indicates that the estimated pose more accurately aligns the known 3D features with their true expected locations on the image sensor.

Sampson distance

The sampson distance is used to evaluate the epipolar consistency of the estimated relative pose. The same landmark correspondences as in the reprojection error evaluation are used. For each landmark, the image point in the nominal image is denoted $\mathbf{x}_i$, and the corresponding image point in the target image is denoted $\mathbf{x}'_i$. The points are first undistorted and expressed in normalized camera coordinates. The Sampson distance is then computed using Equation 2.12 together with the essential matrix obtained from the estimated pose.
This gives an approximation of the geometric error between each point and its epipolar line. A lower value therefore indicates that the estimated pose better satisfies the epipolar constraint for the landmark correspondences. For each evaluated image, the mean, median, and maximum distance over all localized landmarks are reported. As with the reprojection error evaluation, the landmark features are excluded from the pose estimation step to avoid using the same correspondences for both estimation and evaluation.

3.5.3 Image re-alignment

By applying the estimated extrinsic parameters to the images taken using an uncalibrated camera, the images are transformed to match the nominal image. An image re-alignment does not render any numerical expression in the similarity between the original and the re-aligned image. However, it can still provide results in terms of visual feedback of the estimated pose transformation.
This thesis makes use of homography transformation as mentioned in 2.1.7 to perform image re-alignment. Since the interior camera's view is not of a planar surface and instead of a vehicle interior with objects located at different depths, the homography is expressed in its simplified format only incorporating rotation. The homography matrix is calculated according to equation 2.8 with the rotation matrix extracted from the estimated extrinsic parameters.

Image similarity scores

To numerically express how well the resulting image re-alignment aligns with the nominal image, tools frequently used to evaluate image similarity are deployed. These tools are Mean Squared Error (MSE), Structural Similarity Index Measure (SSIM) [36] and Learned Perceptual Image Patch Similarity (LPIPS) [37].
MSE calculates the squared error between each corresponding pixel value in the two images and returns the mean value of this. This is a quick and effective measurement of how similar two images are, however, it is notoriously poor at matching human visual perception. For this, SSIM is commonly used throughout the industry, SSIM computes its similarity-score as a combination of the two images luminance, contrast and structure. LPIPS is a deep learning approach similar to SSIM, where the two images are fed into a pretrained image classification network, and their similarity is scored depending on the similarity of the intermittent output of this network. As part of this report, AlexNet [38] is used as the pretrained image classification network.

4
Results

This chapter presents the results gathered from the experiments mentioned in Section 3.5. The experiments are performed using the data gathered according to Section 3.1 and applied to the two developed pose-estimation-methods. The nominal image used for all experiments is a nominal synthetic image from a simulated vehicle. When applicable, the results of the two methods are compared to a 'baseline' displaying the retrieved error of performing no calibration.
Furthermore, for synthetic data, the ground truth pose is used as a comparator for the reprojection error. The ground truth is the optimal pose that can be retrieved from the pose-estimation-methods. Thus if a projection matrix constructed using this pose, results in a reprojection error, this indicates that this error is a measurement error and not an error due to a bad pose.

4.1 Ground truth comparison

Ground truth comparison was performed using synthetic data generated in a Unity simulation environment as mentioned in Section 3.1.1. The evaluation is divided into three sections: Rotations, Translations, and Transformations, evaluated on their corresponding dataset. The first two sections analyze performance under pure rotation and pure translation, respectively. The transformations section evaluates data subjected to complete rigid body transformations, combining both rotational and translational movements.

4.1.1 Rotations

Both methods were tested on the synthetic dataset mentioned in Section 3.1.1. This dataset contains 343 synthetic images that closely resemble real images of the car positioned in a dark garage setting. An example of such data can be seen in Figure 3.1a, this figure shows the nominal view from which the rotations are calculated.
The test compares the estimated relative rotation from said nominal view with the ground truth, which is shown in Figure 4.1. The discrepancy between the estimate and the ground truth is shown in Figure 4.2, together with the Mean Absolute Error (MAE) of all three axes. Figure 4.3 shows histograms depicting the error distribution of each rotation axis, a Gaussian distribution has been fitted to each histogram and its parameters are displayed. All of this data is summoned and displayed in Table 4.1, where the performance of the Relative and Absolute Method are compared to the Relative's pipeline except it uses other commonly used feature extractors and

matchers. In addition to this data, a baseline is presented as the first result, this baseline present the overall error over the dataset provided no estimation is done. The same test was performed on the Natural lighting dataset where the same nominal view as before was used, see Figure 3.1a, and the pose responsible for each image was calculated. The aforementioned dataset consists of 62 images and the results of the test have been summarized and can be found in Table 4.2.

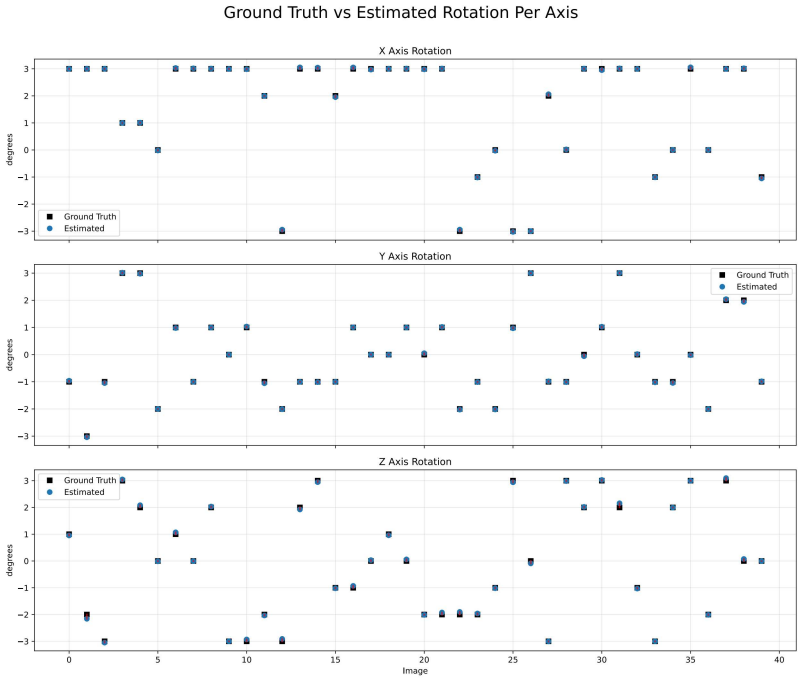


Figure 4.1: Resulting rotation estimate and the ground truth rotation of each axis visualized on a subset of the synthetic rotations dataset. The estimate is calculated using the Relative pipeline using SuperPoint and LightGlue.

Table 4.1: Table showcasing the performance of the rotation estimate on the synthetic dataset. The estimation error is presented with the combined axes and geodesic MAE together with each axis mean error μ and standard deviation σ. The results from the two final implemented versions of the pipelines are shown in bold.

Detector	Matcher	Dataset	Method	Geodesic MAE (°)	Euler XYZ MAE (°)	X μ(°)	X σ(°)	Y μ(°)	Y σ(°)	Z μ(°)	Z σ(°)
None	None	Synthetic	None	3.33	5.14	-	-	-	-	-	-
ORB	Brute-Force	Synthetic	Relative	1.69	7.86	1.06	7.21	-0.47	4.06	2.93	32.45
SIFT	Brute-Force	Synthetic	Relative	0.19	0.68	0.04	0.74	0.00	0.04	-0.53	9.71
SIFT	LightGlue	Synthetic	Relative	0.62	2.45	-0.73	10.24	-0.01	0.69	-0.50	15.83
SuperPoint	**LightGlue**	**Synthetic**	**Relative**	**0.07**	**0.09**	**0.00**	**0.02**	**0.00**	**0.02**	**0.00**	**0.07**
SuperPoint	**LightGlue**	**Synthetic**	**Absolute**	**0.07**	**0.10**	**0.00**	**0.03**	**-0.01**	**0.03**	**-0.01**	**0.07**

From Tables 4.1, 4.2, it can be gathered that the implementations of the Relative Method using SuperPoint and LightGlue are superior to the other combinations of feature detectors and matchers. The combination of ORB and brute-force matcher performs the worst. This is probably due to inclusion of either outliers or incorrect matches which leads to a high MAE and high variance in the error. Both SIFT implementations perform reasonably well with some outlier estimates increasing its

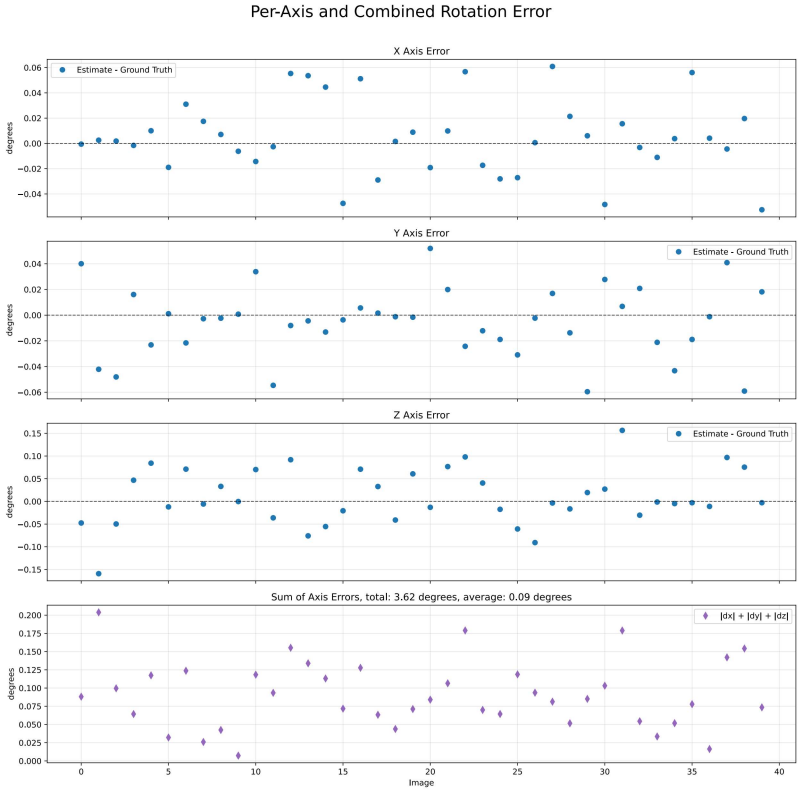


Figure 4.2: Difference between the estimate and the ground truth rotation of each axis visualized on a subset of the synthetic rotations dataset.

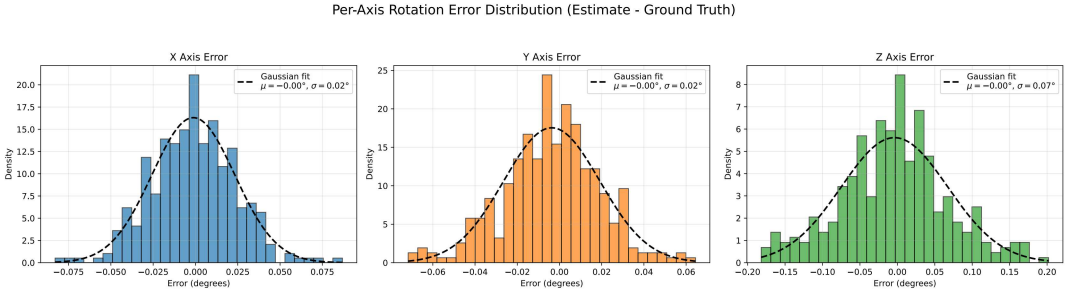


Figure 4.3: Each axis estimation error shown as histograms with a fitted Gaussian distribution. The error is calculated over the entire Synthetic dataset.

Table 4.2: Table showcasing the performance of the rotation estimate on the natural lighting dataset. The estimation error is presented with the combined axes MAE together with each axis mean error μ and standard deviation σ. The results from the two final implemented versions of the pipelines are shown in bold.

Detector	Matcher	Dataset	Method	Geodesic MAE (°)	Euler XYZ MAE (°)	X μ(°)	X σ(°)	Y μ(°)	Y σ(°)	Z μ(°)	Z σ(°)
None	None	Natural	None	3.29	5.06	-	-	-	-	-	-
ORB	Brute-Force	Natural	Relative	91.54	82.54	-2.91	48.95	-3.60	32.78	-4.94	55.95
SIFT	Brute-Force	Natural	Relative	3.53	31.33	2.27	50.97	-1.81	8.58	3.69	23.46
SIFT	LightGlue	Natural	Relative	0.18	0.19	-0.06	0.06	-0.06	0.04	0.00	0.07
SuperPoint	**LightGlue**	**Natural**	**Relative**	**0.09**	**0.12**	**-0.03**	**0.03**	**-0.03**	**0.02**	**0.01**	**0.07**
SuperPoint	**LightGlue**	**Natural**	**Absolute**	**0.09**	**0.14**	**0.05**	**0.04**	**-0.01**	**0.04**	**-0.02**	**0.07**

overall MAE. Table 4.2 showcases the strength of LightGlue in matching features across multi-modal data where the MAE of SIFT with LightGlue is much smaller than SIFT and brute-force. Furthermore, the Relative Method performs better than the Absolute Method in its rotation estimate, the difference is however, not significant.

4.1.2 Translations

Just as with the rotation, both methods ability to estimate pure translation was tested. This test was performed according to the same methodology as the rotation but on a synthetic translation dataset consisting of 38 images. A comparison between the translation estimate of the Relative Method and the ground truth of this dataset is visualized in Figure 4.4. A more comprehensive result of the entire test can be seen in Table 4.3.

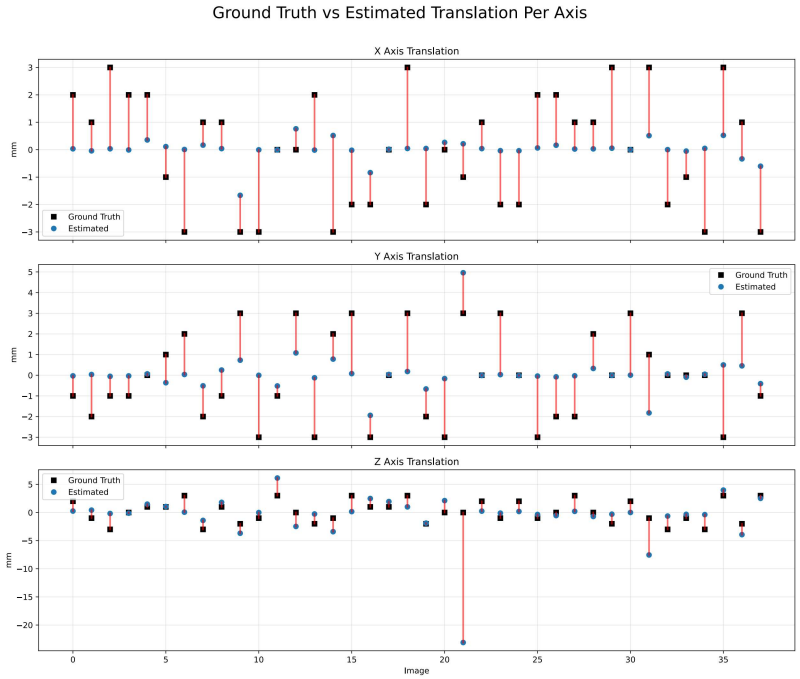


Figure 4.4: Resulting translation estimate of the Relative Method and the ground truth translation of each axis visualized on a synthetic translation dataset. The estimate is calculated using the Relative pipeline using SuperPoint and LightGlue.

This part of the system was also further tested on two different datasets, namely Synthetic_{15} and Synthetic_{50}, consisting of larger translations. Both datasets consist of images originating from the same synthetic environment, however, Synthetic_{15} consists of 77 images with poses changing in the range of ± 15 mm, and Synthetic_{50} consists of 65 images ranging in ± 50 mm.
For this evaluation the size of the estimation error must be put in relation to the size of the translations since they are inherently dependent. For example, an error of 5 mm can be considered rather good if the translation itself is 10 m but not if the

Table 4.3: Table showcasing the performance of the translation estimate on the synthetic translation dataset. The estimation error is presented with the combined axes MAE together with each axis mean error μ and standard deviation σ. The results from the two final implemented versions of the pipelines are shown in bold.

Detector	Matcher	Dataset	Method	XYZ MAE (mm)	X $\mu(mm)$	X $\sigma(mm)$	Y $\mu(mm)$	Y $\sigma(mm)$	Z $\mu(mm)$	Z $\sigma(mm)$
None	None	Synthetic	None	5.16	-	-	-	-	-	-
ORB	Brute-Force	Synthetic	Relative	192.12	40.13	181.29	-37.20	160.35	-7.74	480.53
SIFT	Brute-Force	Synthetic	Relative	7.57	-0.12	2.14	-0.37	2.80	1.24	5.28
SIFT	LightGlue	Synthetic	Relative	35.27	6.25	38.07	-21.79	133.12	1.02	10.34
SuperPoint	**LightGlue**	**Synthetic**	**Relative**	**5.29**	**-0.03**	**1.95**	**0.06**	**2.01**	**-0.29**	**2.31**
SuperPoint	**LightGlue**	**Synthetic**	**Absolute**	**1.04**	**0.16**	**0.58**	**0.27**	**0.47**	**0.13**	**0.30**

translation is 1 mm. To solve this, all of the errors are normalized which enables fair cross-dataset comparison. The errors are expressed as a percentage of the maximum elongation along one axis contained in that dataset and can be found in Table 4.4.

Table 4.4: Table showcasing the performance of the translation estimate on the synthetic dataset with exaggerated translations. The estimation error is presented as a percentage of the maximum translation along one axis within that dataset. This error is further broken down into parts, as the angular and magnitude error between the estimated and ground truth vectors, the magnitude error is expressed as an percentage in the same way as the overall MAE.

Detector	Matcher	Dataset	Method	XYZ MAE (%)	∠ XYZ MAE (°)	|XYZ| MAE (%)
None	None	Synthetic	None	172	-	-
SuperPoint	LightGlue	Synthetic	Relative	176.3	49.0	92.7
SuperPoint	LightGlue	Synthetic	Absolute	34.67	8.92	9.3
None	None	Synthetic_{15}	None	175.7	-	-
SuperPoint	LightGlue	Synthetic_{15}	Relative	57.33	5.92	38.6
SuperPoint	LightGlue	Synthetic_{15}	Absolute	7.87	2.23	2.6
None	None	Synthetic_{50}	None	182.3	-	-
SuperPoint	LightGlue	Synthetic_{50}	Relative	24.48	1.40	15
SuperPoint	LightGlue	Synthetic_{50}	Absolute	3.25	0.861	1.2

Table 4.3 displays a less than satisfying result from the Relative Method concerning the translation estimate. The two main components of this estimate is the up-to-scale translation vector retrieved from the essential matrix and its triangulated scaling factor. During the tests it was clear that the distance between the triangulated world points was fluctuating wildly, varying from $\sim$ 20 to $\sim$ 2000 from one image to the other. This is not a scientific artifact, instead by analyzing Figure 2.2, this result and problem is quite intuitive. Due to the small deviations in position ($\pm$3 mm) the views overlap and the back-projected rays becomes next to parallel, which in combination with noisy feature matching leads to an incredibly sensitive triangulation.
The Absolute Method on the other hand is not dependent on the calibration view to perform its triangulation, instead it uses an image from the Synthetic_{50} dataset, which makes it more robust and accurate. However, its estimate is also not perfect, hinting at the problem of estimating the initial translation vector can be inherently difficult for small translations. The reason for this may be that a small translation does not displace the image points enough to be able to draw strong conclusion regarding the camera's displacement, this is discussed more in depth later in the

reprojection part of the evaluation.
With the feedback of information contained in Table 4.3, a test on larger translations was conducted, resulting in the results found in Table 4.4. This table further confirms the hypothesis of a large portion of the Relative Method's translation error being dependent on the size of the translation, as we can see the MAE drop from 176% of ±3mm to 24.48% of ±50 mm. This hypothesis can be further strenghtened by comparing Figure 4.4 to Figure 4.5, where it in the latter figure can be seen that the estimate generally follows the ground truth much closer than the former, both in scale and direction. Actually, the MAE for Synthetic_{50} would be even lower if it were not for some exaggerated translations leading to a semi-blocked view of the camera, resulting in a failed localization of the triangulation landmarks. This can be seen in Figure 4.5 where the estimate is close to zero for all axes for some samples, an example of this can be seen in the seventh sample.

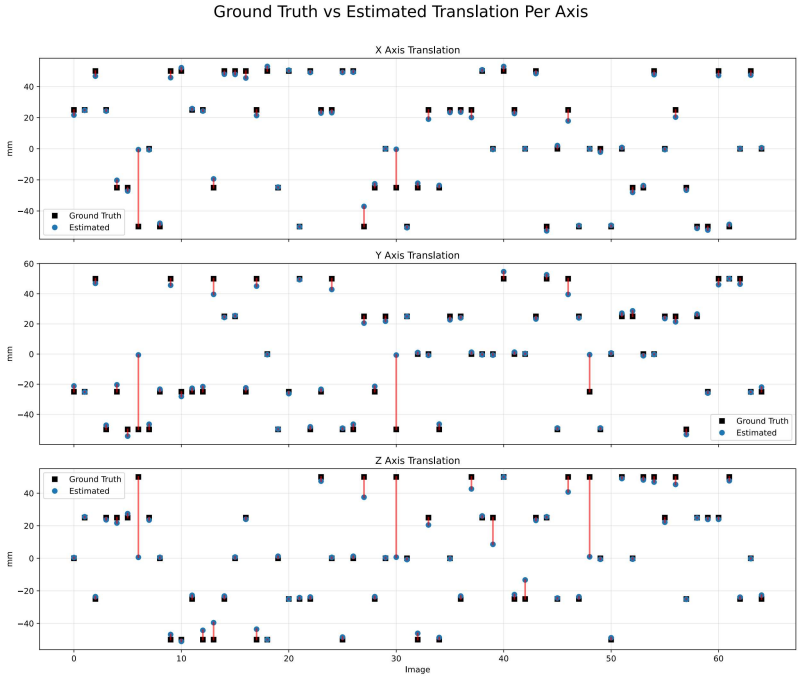


Figure 4.5: Resulting translation estimate of the Relative Method and the ground truth translation of each axis visualized on the Synthetic_{50} translation dataset. The estimate is using SuperPoint and LightGlue.

In addition to the triangulation working better for larger translations, the improved performance of both methods concerning the angular error over larger translations in Table 4.4, suggests that the translation vector itself is easier to estimate accurately for larger translations.

4.1.3 Transformations

Finally, the two methods were evaluated on complete rigid transformations with the same overall procedure as with the rotation and translation evaluations. Both

methods were tested on the synthetic transformation dataset with 125 images and the natural lighting dataset containing 53 images, the results are gathered in Table 4.5 and 4.6 respectively. Table 4.5 shows once again the Relative Method using

Table 4.5: Table showcasing the performance of the transformation estimate on the synthetic dataset. The results from the two final implemented versions of the pipelines are shown in bold.

Detector	Matcher	Dataset	Method	Euler XYZ MAE (°)	X μ(°)	X σ(°)	Y μ(°)	Y σ(°)	Z μ(°)	Z σ(°)
None	None	Synthetic	None	5.30	-	-	-	-	-	-
ORB	Brute-Force	Synthetic	Relative	3.93	-0.20	4.66	-0.30	3.13	0.16	21.65
SIFT	Brute-Force	Synthetic	Relative	0.31	-0.01	0.17	0.01	0.14	0.00	0.09
SIFT	LightGlue	Synthetic	Relative	0.32	-0.02	0.17	0.00	0.14	0.01	0.08
SuperPoint	**LightGlue**	**Synthetic**	**Relative**	**0.35**	**-0.03**	**0.18**	**0.00**	**0.15**	**0.01**	**0.08**
SuperPoint	**LightGlue**	**Synthetic**	**Absolute**	**0.10**	**0.01**	**0.03**	**-0.01**	**0.03**	**-0.01**	**0.07**
Detector	Matcher	Dataset	Method	XYZ MAE (mm)	X $\mu(mm)$	X $\sigma(mm)$	Y $\mu(mm)$	Y $\sigma(mm)$	Z $\mu(mm)$	Z $\sigma(mm)$
None	None	Synthetic	None	5.05	-	-	-	-	-	-
ORB	Brute-Force	Synthetic	Relative	60.98	7.18	73.37	-3.02	86.94	36.91	292.92
SIFT	Brute-Force	Synthetic	Relative	6.93	-0.07	2.27	-0.52	2.80	0.07	4.95
SIFT	LightGlue	Synthetic	Relative	6.61	-0.14	1.93	-0.09	1.93	-0.07	5.49
SuperPoint	**LightGlue**	**Synthetic**	**Relative**	**6.32**	**-0.17**	**1.94**	**-0.32**	**2.12**	**-1.33**	**4.81**
SuperPoint	**LightGlue**	**Synthetic**	**Absolute**	**0.93**	**0.24**	**0.47**	**0.11**	**0.41**	**0.07**	**0.34**

ORB as heavily inferior to its counterparts. However, the performance covering both rotation and translation is quite even when using both SIFT and SuperPoint. SIFT is slightly better at rotation, and SuperPoint slightly more accurate in translation, but there are no significant differences in either of the cases. Further, it is clear that with the introduction of rotations, the translation estimate of the Relative Method on small translations is still bad. The best result achieved with this method is a MAE = 6.32mm which can be compared to what the average error would be if no calibration or adjustment were made, resulting in a MAE of 5.05mm over this dataset.

Furthermore, it is clear that for this test the Absolute Method is superior, having 32.3%, 14.7% of the MAE over both rotation and translation, respectively, compared to the best results using the Relative Method.

Table 4.6: Table showcasing the performance of the transformation estimate on the natural lighting dataset. The results from the two final implemented versions of the pipelines are shown in bold.

Detector	Matcher	Dataset	Method	XYZ MAE (°)	X μ(°)	X σ(°)	Y μ(°)	Y σ(°)	Z μ(°)	Z σ(°)
None	None	Synthetic	None	5.34	-	-	-	-	-	-
SIFT	Brute-Force	Synthetic	Relative	51.34	1.50	48.32	-1.39	17.41	0.52	34.22
SIFT	LightGlue	Synthetic	Relative	0.36	-0.02	0.20	-0.06	0.14	0.02	0.09
SuperPoint	**LightGlue**	**Synthetic**	**Relative**	**0.35**	**0.01**	**0.18**	**-0.01**	**0.15**	**0.02**	**0.09**
SuperPoint	**LightGlue**	**Synthetic**	**Absolute**	**0.14**	**0.04**	**0.04**	**0.00**	**0.04**	**0.00**	**0.08**
Detector	Matcher	Dataset	Method	XYZ MAE (mm)	X $\mu(mm)$	X $\sigma(mm)$	Y $\mu(mm)$	Y $\sigma(mm)$	Z $\mu(mm)$	Z $\sigma(mm)$
None	None	Synthetic	None	4.81	-	-	-	-	-	-
SIFT	Brute-Force	Synthetic	Relative	5.05	0.03	1.87	0.58	2.04	0.35	2.00
SIFT	LightGlue	Synthetic	Relative	4.47	-0.21	1.82	0.49	1.92	-0.21	1.49
SuperPoint	**LightGlue**	**Synthetic**	**Relative**	**4.38**	**-0.15**	**1.81**	**0.47**	**1.90**	**0.33**	**1.47**
SuperPoint	**LightGlue**	**Synthetic**	**Absolute**	**1.76**	**0.16**	**0.58**	**0.95**	**0.58**	**-0.20**	**0.37**

4.2 Point mapping error

The point mapping error evaluation consists of two different metrics, reprojection error and sampson distance. Reprojection error is an evaluation of a mapping from 3D to 2D and sampson distance can be interpreted as an evaluation of a 2D to 2D point mapping. Below follows both of them, evaluated on the landmarks hold out set on synthetic and real images separately.

4.2.1 Synthetic images

Reprojection error

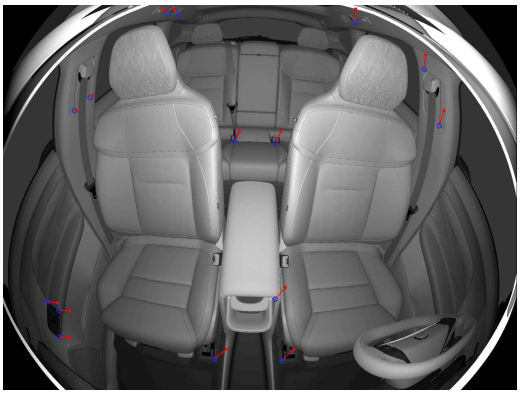
The two methods were evaluated using reprojection error for the three synthetic datasets. The results of this can be seen in Table 4.7. For both the transformation and rotation datasets, both methods reduced the reprojection error greatly, with the Absolute Method proving to be slightly better on both datasets. The difference is bigger for the transformation dataset, where the Absolute Method is 2 pixels better than the relative. This is most likely due to a worse capability to estimate translations, which is shown in the results for the translation dataset. For this dataset, the Relative Method does not improve the reprojection error, whereas the Absolute Method actually manages to reduce the error.
Two specific images are also included in the table. For these specific examples, the Absolute Method performs slightly better, but both greatly reduces the reprojection error. The images are shown in Figure 4.6 together with a visual representation of the reprojection error for the baseline and estimates.
The reprojected world points of the landmarks are shown as red, yellow or green dots, and the corresponding detected landmarks are shown as blue circles. If a reprojection error for a point is less than 2 pixels, the point it shows up as green, it shows up as yellow if that error is less than 5, and red otherwise.

Table 4.7: Table showing the mean reprojection error for the different datasets.

Dataset	Baseline (px)	Ground truth (px)	Relative Method (px)	Absolute Method (px)
Synthetic Transformations	51.21	3.69	5.42	3.46
Synthetic Rotations	49.96	3.73	4.07	3.54
Synthetic Translations	3.92	2.59	4.00	2.56
Image with pose [0,-3,0,-2,0,2]	54.14	-	4.14	2.81
Image with pose [3,3,-3,-2,1,-1]	48.88	-	3.26	1.98

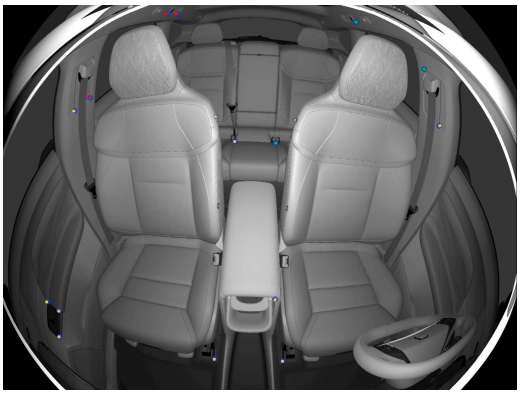
Sampson distance

The sampson error for both methods was evaluated on the datasets containing translations. The mean reprojection error for all points in each image is recorded, and the mean, median and maximum of these means are presented in Table 4.8. The Relative Method performed slightly better than the Absolute Method when it comes to the maximum reprojection error, but they were quite similar when it comes to the mean and median values.



(a) Simulated image 1.

(b) Simulated image 2.



(c) Relative pose estimate for simulated image 1.

(d) Relative pose estimate for simulated image 2.

(e) Absolute pose estimate for simulated image 1.

(f) Absolute pose estimate for simulated image 2.

Figure 4.6: Reprojection error for landmarks for the baseline and the estimates from both methods on two simulated images.

Table 4.8: Table showing the sampson distance for the synthetic dataset.

Dataset	Method	Mean (px)	Median (px)	Max (px)
Synthetic Translations	Relative	1.9	1.95	2.77
Synthetic Translations	Absolute	2.04	1.91	3.93
Synthetic Translations	Ground Truth	1.82	1.69	2.88
Synthetic Transformations	Relative	2.01	1.94	3.35
Synthetic Transformations	Absolute	2.20	2.05	4.45
Synthetic Transformations	Ground Truth	2.4	2.31	4.53

4.2.2 Real images

Reprojection error

Two images from cameras in different vehicles were used to evaluate the performance of the two methods using the reprojection error. In Table 4.9, the baseline reprojection error and the results of the two methods is shown.
The Relative Method performs better for both images, but the difference is larger for image 1. Using this method resulted in a smaller reprojection error than the baseline, which can not be said when applying the Absolute Method, since this method only managed to reduce the error for the second image.
In Figure 4.7, the reprojected world points of the landmarks are shown as red, yellow or green dots, and the corresponding detected landmarks are shown as blue circles. If a reprojection error for a point is less than 2 pixels, the point it shows up as green, it shows up as yellow if that error is less than 5, and red otherwise.

Table 4.9: Table showing the mean reprojection error for the two real images.

Dataset	Baseline (px)	Relative (px)	Absolute (px)
Real image 1	20.31	9.04	21.31
Real image 2	16.42	7.45	12.04

Sampson distance

The sampson error for the same points in the two images, for both methods, is presented in Table 4.10. Here, the error for each point in an image is stored and the mean, median and max is calculated from that. The Relative Method gets a slightly lower mean for both images and a much lower maximum error for image one. In image 2, the Absolute Method instead gets a lower maximum error.

4.3 Image re-alignment

This section covers the results of the image re-alignment evaluation mentioned in Section 3.5.3. The images are transformed with a homography transformation using the estimated rotation and Equation 2.8.

(a) Real image 1.

(b) Real image 2.

(c) Relative pose estimate for image 1.

(d) Relative pose estimate for image 2.

(e) Absolute pose estimate for image 1.

(f) Absolute pose estimate for image 2.

Figure 4.7: Reprojection error for landmarks for the baseline and the estimates from both methods on two real images.

Table 4.10: Table showing the sampson distance for the two real images.

Dataset	Method	Mean (px)	Median (px)	Max (px)
Real image 1	Relative	13.12	14.04	28.57
Real image 1	Absolute	13.55	11.92	39.14
Real image 2	Relative	9.45	10.01	17.72
Real image 2	Absolute	10.23	10.17	16.17

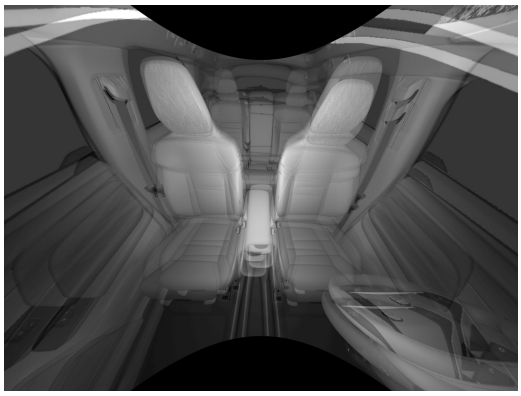
Small deviations in structure and alignment are hard to distinguish from two separate images, these changes are, however, more prominent if the images structural outline are placed on top of each other. This is done with blending, meaning that the resulting image is created pixel-by-pixel by computing a weighted sum of the corresponding pixel intensities in the two source images. Additionally, each image is undistorted and padded with zeros where data is lost as a result of undistortion or sub-optimal pose.
The nominal image in Figure 3.1a is used as the target view, to which the test data should be aligned, both for synthetic and real images.

4.3.1 Synthetic images

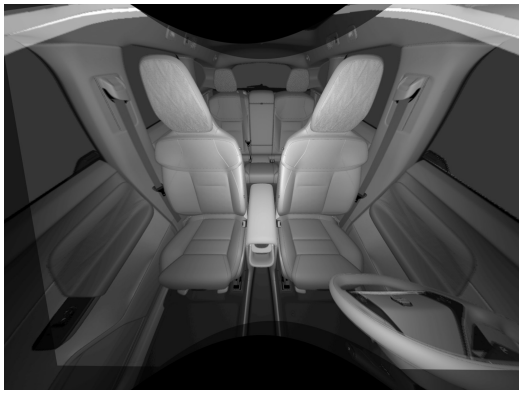
A synthetic image taken with pose $[-1, -3, 2, -2, -3, 3]$ is used as a source view. The resulting blend with the nominal image without transformation can be seen in Figure 4.8a, and the transformed using the relative and Absolute Method are found in Figure 4.8b, 4.8c respectively.
As a result of the image-blending the initial image in Figure 4.8a appears blurry due to an overall misalignment in the visual features in the interior. From viewing this image it is clear that the image consists of two separate images placed on top of each other.
Figure 4.8b and 4.8c shows that both estimates re-aligns the images well. In these two images, the depicted objects from the two views appear to align well. In contrast to Figure 4.8a, in these images it is hard to distinguish one view from the other if not for the edge of the second view appearing as part of the image.

4.3.2 Real images

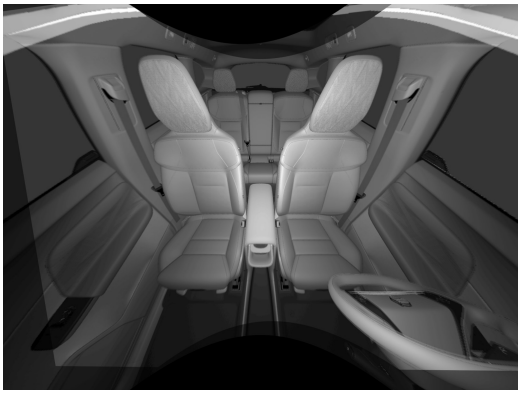
The real image shown in Figure 3.3 is used as the source view for image-alignment. The result of blending this with the nominal synthetic image with and without any transformation can be seen in figures 4.9, 4.10 and 4.11.
At first glance, the results found in figures 4.10 and 4.11 do not show any successful re-alignments. However, this is mostly due to the dynamic objects such as seats and cup-holders not being in the same position in the pictured 3D environments. For a more proper re-alignment result, the focus should instead be on static objects in the view, such as the center console and door-handles. As an example of this, see Figure 4.12 for a zoomed-in view of the center console between the non-aligned and re-aligned blend of the synthetic and real car data.
Figure 4.12b shows that the center console aligns well across the two images after



(a) Undistorted blend of two synthetic images with no transformation.



(b) Undistorted blend with one image transformed with the relative rotation estimate.



(c) Undistorted blend with one image transformed with the absolute rotation estimate.

Figure 4.8: Three figures displaying the blended alignment of two synthetic images with and without transformation.

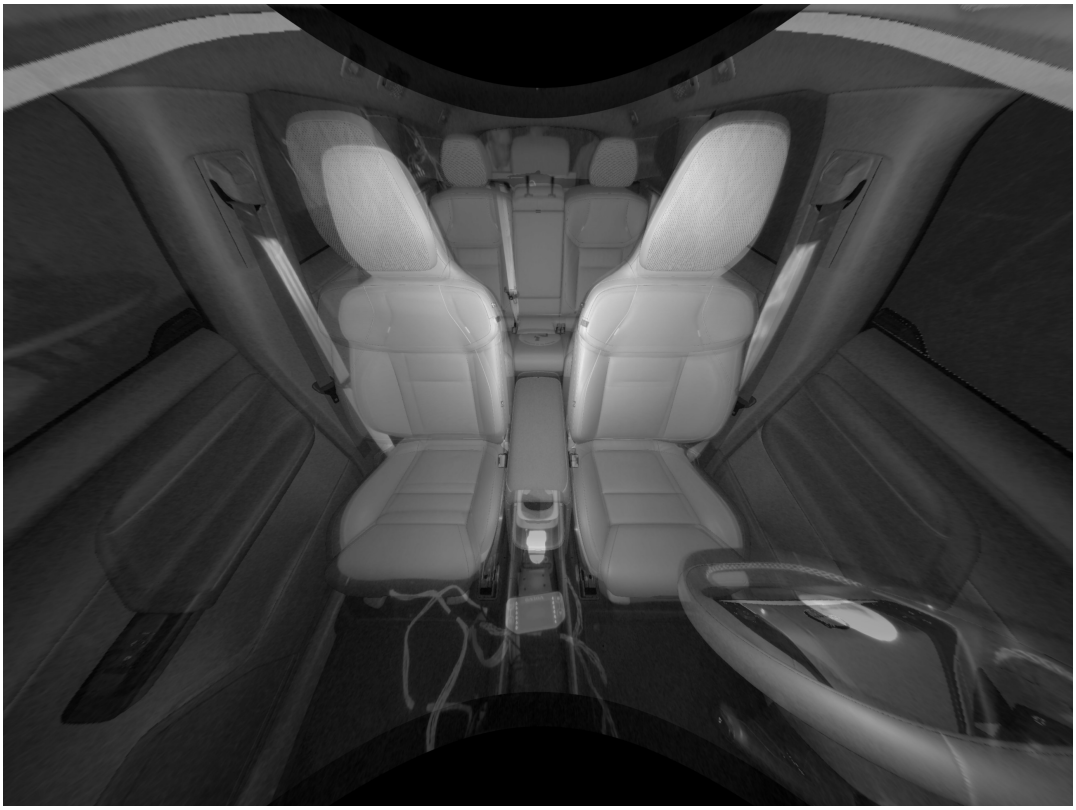


Figure 4.9: Blend with no transformation.

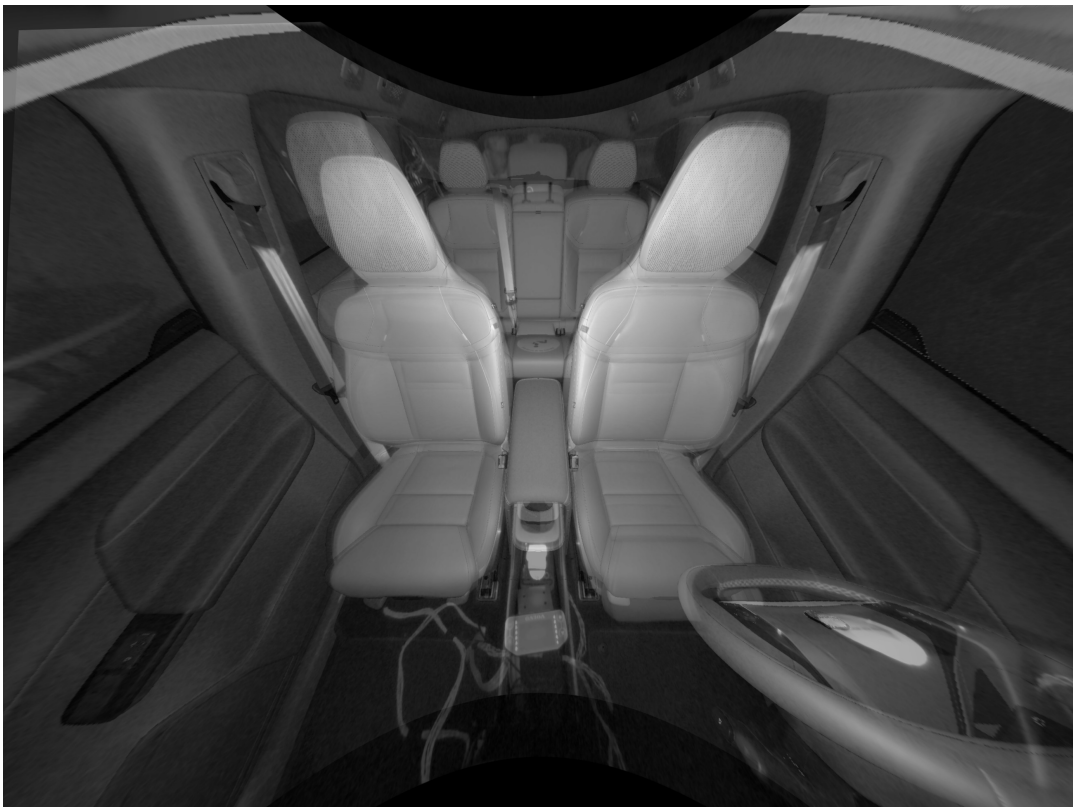


Figure 4.10: Blend with the real image transformed with the relative rotation estimate.

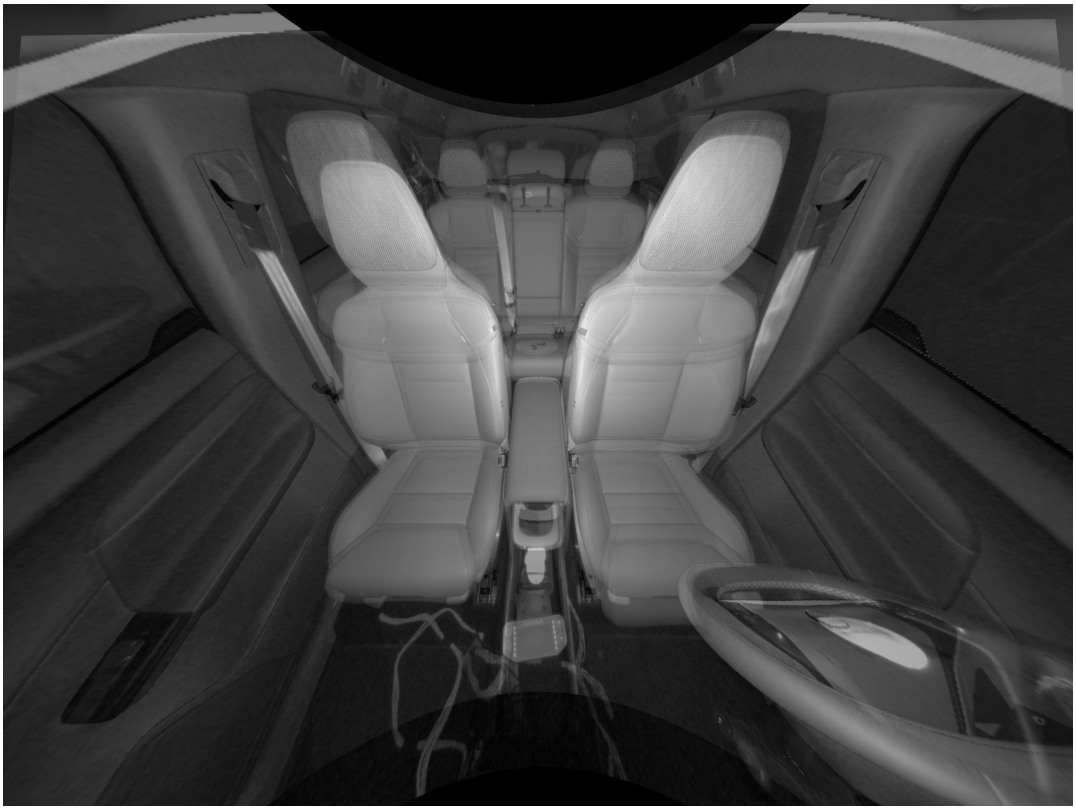


Figure 4.11: Blend with the real image transformed with the absolute rotation estimate.

(a) Without re-alignment.

(b) With Relative Method's re-alignment.

Figure 4.12: Zoomed-in view of the center console with and without re-alignment.

the re-alignment. This is especially prominent in the far horizontal edge, where this edge occurs two times in Figure 4.12a, this is however not the case for the re-aligned version as that has one consistent edge.
This was just one example of several static objects that appear well aligned in figures 4.10 and 4.11, another would be the door-handles. From these two images it can furthermore be gathered that the Relative Method's transformation seems to align the two images better than the Absolute Method.

4.3.3 Image similarity score

In an effort to numerically evaluate how similar the transformed source image is to the target image, three measures are evaluated as mentioned in Section 3.5.3. An undistorted version of the synthetic nominal image is used as a reference image for this evaluation. The same real and synthetic images as per the previous section are reused as source-images for this evaluation, also these undistorted. The result of this evaluation can be seen in Table 4.11 and 4.12.

Table 4.11: Table showing the image similarity scores with the best scores expressed in bold text. An undistorted version of synthetic image with pose [-1,-3,2,-2,-3,3] is used as source image.

Dataset	MSE	LPIPS	SSIM
No transform	0.029	0.316	0.731
Relative transformed	0.021	**0.122**	0.763
Absolute transformed	**0.020**	**0.122**	**0.764**

Table 4.12: Table showing the image similarity scores with the best scores expressed in bold text. An undistorted real image used as source image.

Dataset	MSE	LPIPS	SSIM
No transform	**0.049**	0.378	0.546
Relative transformed	0.050	**0.375**	**0.547**
Absolute transformed	0.051	0.381	0.539

For the transformed synthetic image as source image as part of Table 4.11, all measures indicate that the re-alignment produced by both methods make the image more similar to the nominal counterpart. The only measure that shows a clear indication of this is however LPIPS that decreases from 0.316 to 0.122, whereas the other measures just show a small inclination towards this result.
Coherently the measures struggle to numerically express the differences between non-transformed and transformed real image as shown in Table 4.12. The numerical discrepancy between the best and worst result of each measure in this table is small, making it difficult to draw any definitive conclusions. With the two images depicting interiors with multiple objects having different colors between them and also the dynamic objects having different positions, this result is somewhat expected.

5
Discussion and conclusion

This chapter summarizes the main results of the thesis and discusses their implications for the targetless camera calibration in vehicle interiors. The applicability of the proposed methods in real world scenarios is discussed. Finally, the main limitations of the work are discussed, and suggestions for future research and further development are presented.

5.1 Project summary

This thesis addresses the problem of performing camera calibration without the need for specific calibration targets. The thesis showed how large reprojection errors can be expected if no calibration is performed. These results are based on expected variations in virtual camera pose originating from tolerances in the camera mounting position in the ranges $\pm 3°$ and ± 3 mm in rotation and translation, respectively.
Without any adaptations to the camera projection, i.e. without using any calibration method to estimate and then model the extrinsic parameters , the mounting variations will result in different 3D objects appearing in the same place in the image for different vehicles. This complicates computer vision tasks such as object classification and detection using tools such as bounding boxes in the image. Since the position of the boxes is often fixed in the image, the content of the box may vary due to the variation in mounting position.
This could result in a bounding box that is larger than necessary, modeling the high uncertainty of its position. This could lead to a bounding box that encapsulates multiple positions in the car, resulting in incorrect detections. The bounding box can also be smaller than sufficient, resulting in detections escaping the correct bounding box to another, or detections being left out entirely.
To solve the camera calibration problem, two methods were developed. Both implementations are targetless and rely strictly on software, requiring no manual intervention in the current production line to perform the calibration. What they do require, however, is either one or two reference images, for the relative and Absolute Method respectively.
Both methods require a nominal image. This image can either be a true nominal image, meaning that it is positioned in a nominal known pose. It can also be an image with an assumed nominal image, meaning that its coordinate system and view is the reference for all downstream cameras, etc. The second method requires one additional image, the pose of this second camera should have a substantial change in its position in relation to the nominal camera, while still viewing the

same unobstructed scene. The size of the translation used for this thesis is ≥ 50 mm and is purely a guideline derived from the results found in Table 4.4, which show that triangulation is more accurate for larger translations.
For both development and evaluation of the two methods, synthetic data was used. This synthetic data was collected using a simulated model of the real car in a Unity environment. The synthetic data enabled data collection with known camera poses, which was later used as the aforementioned reference images and for evaluation. In addition to the synthetic data, the two methods were evaluated on real data originating from two unique vehicles of the corresponding make and model.
Three different evaluation strategies were applied to the two methods. The first evaluation method consisted of comparisons between ground truth poses and estimated poses on sets of synthetic data. This was followed by point mapping error, covering two typical metrics used within the field of camera calibration, the reprojection error and sampson distance. The estimates from the two methods were used to realign images, and the result is visualized and evaluated using popular image similarity metrics.

5.2 Main findings

This thesis has demonstrated that camera pose estimation from a single snapshot of a vehicle interior is feasible; however, the robustness of the resulting estimate depends on which DOFs are considered. Both proposed pipelines were able to accurately estimate rotation on synthetic data, but translation was significantly more challenging.
Among the evaluated feature extraction and matching approaches, the combination of SuperPoint and LightGlue was found to give the most accurate and reliable pose estimates.
Both estimation methods showed strong performance on synthetic data. For the dataset of transformed images, they reduced the mean reprojection error from 51.2 pixels to 5.4 pixels and 3.5 pixels for the relative and absolute pose estimation methods, respectively. On real data, the Relative Method achieved superior performance, in terms of reprojection error, on the two available real images. It approximately halved the reprojection error for both images, whereas the Absolute Method produced a noticeable reduction in reprojection error for only one of the images.

5.2.1 Rotation estimation

Rotation estimation was the strongest part of the estimates regarding both methods. On the synthetic dataset, the methods exhibited very similar performance, resulting in an euler MAE of 0.09° for the Relative Method and 0.10° for the Absolute Method. The geodesic error was 0.07° for both methods. The same tendencies were observed on the natural dataset, where the Relative Method achieved marginally superior performance. However, both methods can be concluded to estimate rotational parameters with high accuracy on the synthetic data.
For the synthetic rotation dataset, both methods substantially reduced the reprojection error. Specifically, the reprojection error decreased from 49.96 px to 4.07

px and 3.54 px for the relative and Absolute Methods, respectively. This indicates that, given a set of 2D image points and their corresponding 3D world points, the 2D projections of the 3D points more closely match the observed 2D image locations after calibration. This implies that the estimated camera parameters provide a more accurate explanation of the object's position in the image following the calibration procedure.

The reprojection error of around 4.07 px and 3.54 px is very close to the reprojection error retrieved using the ground truth rotation, which is 3.73 px. This indicates that a significant part of the error from the estimates comes from measurement noise for the reprojection error measurement. The noise most likely stems from the localization of the landmarks. The features detected by SuperPoint are not always detected on the exact same location in the image each time. This localization noise is due to how the network is constructed and is acknowledged by the authors of the SuperPoint paper [20]. Crucially, this magnitude does not imply a systematic bias of 3.73 px across all detections, which would severely degrade the performance of the estimator. Assuming the localization noise is zero mean and isotropic, the law of large numbers dictates that given a sufficiently large set of feature correspondences, the directional errors converge to zero during optimization. This means that a small set of correspondences based on the landmarks dataset, is more sensitive to noise than all correspondences found from two car interiors. Furthermore, to mitigate the impact of statistically significant deviations, both estimation pipelines employ outlier rejection schemes in form of MAGSAC++.

The observation that the ground truth rotation yields a slightly higher reprojection error than the Absolute Method is not necessarily unexpected. The reprojection error is computed using both the camera extrinsics and the intrinsics. Since the same intrinsics are used during pose estimation and evaluation, any inaccuracies in the intrinsic calibration may be partially compensated for by the estimated extrinsics. As a result, an estimated pose that is incorrect may be estimated as it can compensate for errors in the intrinsic model. Conversely, even though the ground truth rotation represents the correct physical orientation of the camera, projecting points with imperfect intrinsic parameters can lead to a higher reprojection error. Therefore, a lower reprojection error does not necessarily imply a more accurate pose in terms of true extrinsics. A reprojection error close to that of the ground truth does, however, indicate that the estimated extrinsics are close to the ground truth extrinsics.

5.2.2 Translation estimation

Estimating the translational components proved challenging for both estimation methods. Specifically, resolving the small translational offsets arising from the inherent mechanical tolerances of production vehicles presented a significant difficulty. For this small translation dataset, the Relative Method resulted in a larger error than the baseline error, i.e. the average error that is caused by the random translations. The Absolute Method performed significantly better and was able to estimate translation with approximately millimeter-level accuracy. This method managed to reduce the mean error from 5 mm to 1 mm.

While this constitutes a substantial improvement over the baseline, the relative gain was less pronounced than that observed for the rotation estimates. The inferior performance of the Relative Method is primarily attributable to the minimal perspective change between the nominal and actual camera poses, since this change is negligibly small in comparison to the distance to the points of interest.
This minimal change in perspective is also reflected in the reprojection error evaluation. For the synthetic translation dataset, the baseline reprojection error was 3.92 px, as opposed to the 49.96 px of the rotation dataset. This demonstrates how much more of a difference the changes in rotations make to the pixels of the images than the changes in translation. This reprojection error was reduced to 2.56 px using the Absolute Method, whereas the Relative Method did not manage to decrease the error, and instead produced an error of 4.00 px. This points in the same direction as the ground truth comparison results, that show that the Relative Method cannot reliably estimate translation, but the Absolute Method can.

5.2.3 Translation vector scaling of the Relative Method

The poor translation performance of the Relative Method was further investigated by isolating the effect of the translation vector and its scaling. Since small translations cause only minor visual changes, the estimated translation vector is both difficult to recover and sensitive to errors in triangulation. To test whether the translation estimate was the main source of error, an additional experiment was performed in which the Relative Method was used with the translation vector fixed to $[0, 0, 0]^\intercal$. This simplified variant reduced the reprojection error from 5.42 to 5.01 px on the synthetic transformation dataset. A similar improvement was observed on real data, where the error decreased from 7.45 to 6.91 px for real image 1, and from 9.04 to 8.91 px for real image 2. These results are summarized in Table 5.1.

Table 5.1: Reprojection error for the Relative Method with translation estimate and with translation set to zero.

Data	Baseline (px)	Relative Method (px)	Relative Method t=0 (px)
Real image 1	20.31	9.04	8.91
Real image 2	16.42	7.45	6.91
Synthetic	51.21	5.42	5.01

By omitting the translation vector, the Relative Method is made much more compact and simple, see Figure 3.6 where the entirety of the yellow box is omitted. Furthermore, this also makes the method more robust, since it is not dependent on the sensitive triangulation. Indeed, it does result in a system with all of the above attributes while also outperforming the original Relative Method in terms of reprojection error.

5.2.4 Comparison of the estimation methods

The translation accuracy is the biggest difference between the methods. The Relative Method estimates the pose difference between two images, given that one of

the image is from a camera with a nominal pose, the other camera can be calibrated to this pose. Its main advantage is that it is simpler and only requires one nominal reference view. Its main weakness is the translation scaling, which does not work well on small scales. This scaling also adds complexity to the method, since it needs a database of points with known distances between them and a way to find two of these points. This requires more manual labor and input.
The Absolute Method uses an additional camera view with known pose to triangulate world points and then estimate the pose of a third camera using 2D-3D correspondences. This results in this method estimating translation much better than the Relative Method. The disadvantage with this method is that it requires more setup. For the Relative Method, a real image could be used as the nominal camera pose. This is much harder for the Absolute Method. For that to be used in the Absolute Method, there would have to be a second camera with a (to a very high accuracy) known pose in relation to the mounted OMS camera.

5.3 Real world applicability

Much effort has been invested both during development and evaluation for the two methods to be robust and invariant to differences in color and lighting. An example of such a measure is the pre-processing step in the Relative Method where a luminance normalization function and sharpener is applied to the image.
The goal of the thesis was to produce a calibration method that was ready for use both in the production line and after production. Some results support the belief that the two suggested methods satisfy this, and some results indicate that they do not. Below follows some aspects and viewpoints concerning the real world applicability given the results.

5.3.1 Real data evaluation

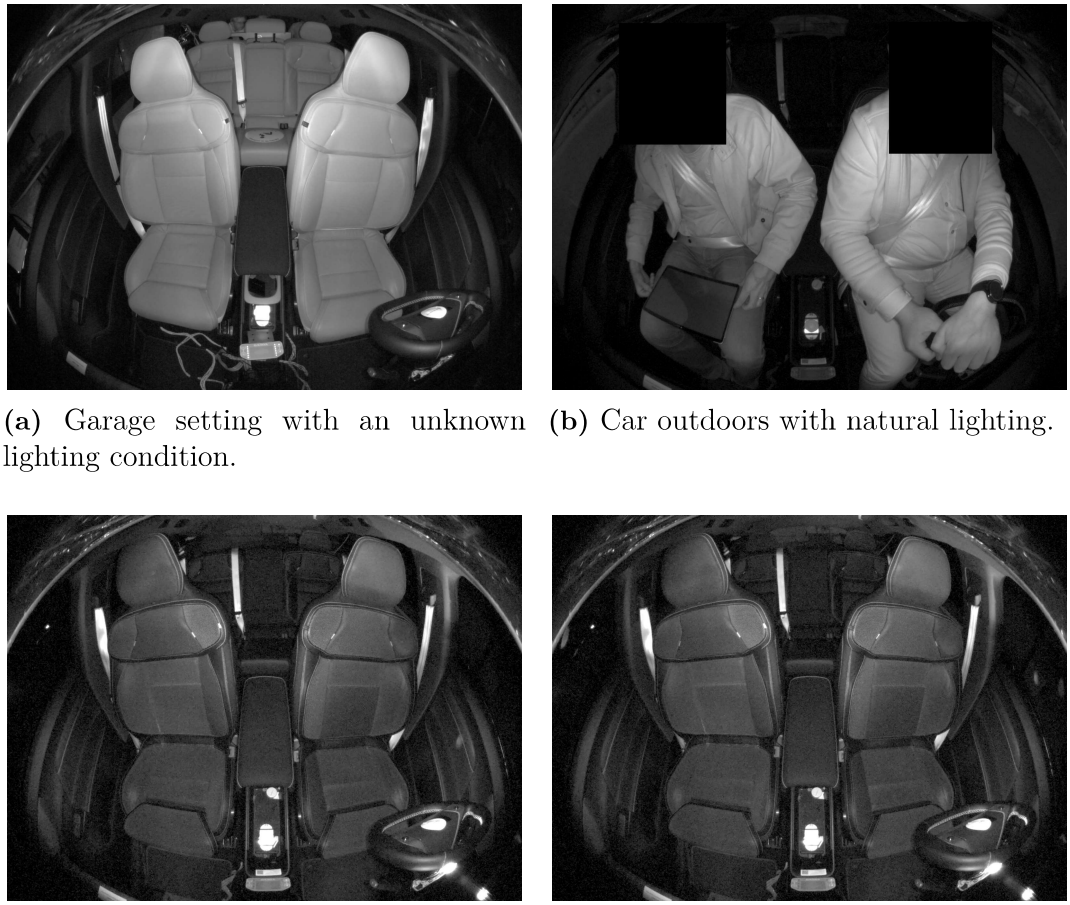
To investigate how well the two methods would function in the real world, tests on real data were conducted as per the last chapter. The real world evaluation was performed on images from two unique vehicles limiting the strength of the possible conclusions that can be drawn from them.
With this mentioned, the Relative Method produced lower point mapping error than the Absolute Method, and a smaller reprojection error than the baseline on both images. This indicates that the Relative Method is superior to the Absolute Method and also the baseline of not performing any calibration on real data. Furthermore, in terms of the Relative Method outperforming the baseline on both synthetic and real data, this shows that the Relative Method is not severely negatively affected by the switch from synthetic to real data.
The point mapping results on real data exhibit a steep decline in performance for the Absolute Method, which on the synthetic data performed comparatively to the ground truth pose and outperformed the baseline and Relative Method in reprojection error. On the real data however, the Absolute Method performs worse than the baseline on one of the images and slightly better on the other.

The image re-alignment test shows a similar result where the visual part of the test shows that the re-alignment using the relative estimate seems to align the two images better than the absolute. This is also shown with the image similarity metrics even though the differences in the result is not significant.
The general suprising theme with the real data evaluation is the steep decline in performance of the Absolute Method. This method either performs comparatively or outperforms all others on all synthetic data tests, but for the real data, it does not. It instead performs significantly worse than the Relative Method. It is important to add, however, that this conclusion is based on two images, which makes it hard to conclude that this would be a general fact.
Both methods rely on SuperPoint and LightGlue for feature extraction and matching. A significant decline in performance for both methods would indicate that these two methods struggle to accurately locate good matches across the different datasets. This is not the case, but the Absolute Method does have a slightly different matching problem than the Relative Method. The Absolute Method has to match the features between three different images, meaning the Absolute Method normally relies on fewer matches.
Furthermore, for evaluation, the features of all landmarks are excluded from the estimation. As previously mentioned, the landmarks are selected on clearly identifiable static features which would also be optimal for estimation. This may result in too few good features for the iterative solver to use in order to accurately estimate the pose, resulting in a inferior estimate.
Additionally, the estimated pose from the real vehicle, results in a large translation ($\geq$ 30 mm) along one of the axes. This could be a bad estimate, but it could also show that there is a discrepancy between the model and the actual car. This discrepancy could be that the camera in the simulated car is positioned differently than the camera in the real car. But it could also be explained with the geometry within the interior being different. None of these two aspects are within the scope of this thesis to evaluate.
In order to further investigate this and draw strong conclusions, more real data would be necessary. However, for the available results and data, we believe that the Relative Method is ready for real world use. Whereas the real world applicability is uncertain for the Absolute Method. Further investigation and troubleshooting on more data would be needed to make any conclusions.

5.3.2 Lighting conditions

The project suffered from a small dataset of real images taken in different environments. As a result of this, it was unclear how the lighting affected the imagery and thus also the features and the estimation results.
To counter this knowledge gap, a worst case-scenario-dataset was created which was called 'Natural lighting' and this was used to evaluate the methods. This evaluation did not expose any direct flaws with any of the methods, although the overall performance on it was slightly worse than the original synthetic dataset.
This evaluation was later proven to be less insightful, since post-development, we received more real data with different lighting conditions, a summary of this data

can be seen in Figure 5.1.
This real data shows that the worst case dataset was beyond worst case, since the real data shows no exaggerated changes in luminance dependent on its surroundings. Instead, the luminance level seems fairly consistent over the four images, even though the exterior light setting changes.
However, the results of applying the methods on the natural lighting dataset could hint at the pipeline also does work for combinations of IR and RGB cameras, in terms of a reference image from an IR camera being used to extrinsically calibrate a RGB camera.



(a) Garage setting with an unknown lighting condition.

(b) Car outdoors with natural lighting.

(c) Garage setting with no lit exterior lighting.

(d) Garage setting with lit exterior lighting.

Figure 5.1: Four images demonstrating different lighting conditions from the real car.

5.3.3 Production-line considerations

Even though this thesis has not investigated or tested a traditional calibration approach of utilizing dedicated calibration targets within the vehicle's interior to perform extrinsic camera calibration, it is deemed reasonable that the result of one of

these methods would outperform or equal that of the two presented methods. The belief is however, that this solution would not be sought after by neither Volvo Cars or the scientific community.
A traditional approach with calibration targets in the interior camera calibration case, would most probably require a specially built station in the production line. A hypothetical implementation of this calibration-station would place the targets in known poses relative the car frame and then perform a traditional PnP calibration routine similar to what was presented in Section 3.2. This would be applicable to future cars, but it could not be implemented in cars that are already in production, if not additional stations external to the production line were built.
With neither of the methods requiring any adaption to the current production line, the two calibration methods functionality can be readily applicable within and beyond a well-established production line. And since both methods solely rely on software, these calibration methods could theoretically be uploaded to the vehicles as a part of an over-the-air update. Furthermore, as shown in Section 5.3.2, different exterior lighting conditions does not seem to affect the imagery significantly, enabling the calibration to be executed from wherever the vehicle is parked. This enables the same calibration technique to be deployed for both future and past cars of the intended model. One limitation that have been recognized with both calibration techniques is that humans seated in the calibrated vehicle may obstruct the view to an extent that not enough features are retrieved.
As mentioned, the two proposed methods does not need any hardware to function. They do, however, require reference images. This thesis has used synthetic imagery as reference, but these could be replaced with real images as long as the pose of the originating camera is either known or assumed to be nominal. This is especially useful for the Relative Method that only relies on one of these images. For the Absolute Method to utilize real imagery, the transformation between the two reference images must be known, making real images slightly harder to implement for this method than its counterpart.
Additionally, since both methods only require reference images, the method is general, allowing the method to be applicable to other possible vehicle models as long as the correct reference imagery is acquired. The method is also not limited to cars or the car's interior, instead it could be used on any scenery that does not suffer from too few identifiable features. This since the two methods rely on a minimum of five and three reference points for the relative and Absolute Method respectively, to compute any estimation.

5.4 Limitations

This paper is written as part of a master's thesis project, and thus inherently experiences limitations in form of resources. The work had a duration of 20 weeks and all work was done by two master's students. With these conditions, it limits the number of solutions and implementations that is possible to experiment with. It has also resulted in a prioritization of immediately promising methods over other that require further work and tuning. Below follows three sections that cover how the aforementioned limitations and others affected the results of this project.

5.4.1 Data deficiency

As part of this thesis, the authors had access to a model of the car within a Unity game engine environment. Using this model, scripts to automatically gather data across varying camera poses, efficiently generating large datasets were developed.
This allowed for ground truth pose evaluation and rigorous testing of the two point mapping errors which are outlier prone.
However, this project had a large deficit on real data from unique vehicles. As a result of this, both the development and evaluation was affected. For the development, this led to precautions and general choices that may not have been necessary as mentioned in Section 5.3.2.
Regarding the evaluation, few unique real data-samples were provided to us. This limited the evaluation covering to what extent the performance of the two methods was transferrable from synthetic to real data, which was mentioned in more depth in Section 5.3.1. In this section we conclude that the Relative Method seems to estimate the pose reasonably well on the available data but the Absolute Method does not. Which is an unintuitive result since the latter method perform better than or equal to the former in all other measurements. With more data this could be investigated further, and both methods estimate could be evaluated on more unique vehicles with different colored interiors, better representing the entire vehicle lineup. This would increase the certainty that the estimates work for most cars of the intended make and model, which cannot be done with the data available.

5.4.2 Landmarks exclusion

For the ground truth comparison in Section 4.1, all detected features were used when calibrating the camera. This was not the case for the later sections in the result where all features matched to a landmark were excluded from the calibration. This ensured that the evaluation was not biased in terms of optimizing and evaluating on the same image points.
However, as previously mentioned, this may have negatively impacted the estimate, especially on real images. When two synthetic images are compared to each other, all dynamic objects such as seats are in the same position. This makes it sustainable to use features on these dynamic objects and also static ones when estimating the pose. When comparing synthetic to real, the dynamic objects are positioned differently, making the outlier rejections exclude them from the estimate correctly. However, this makes the estimate heavily dependent on the features from the static objects. With the implementation of landmark exclusion, it is therefore reasonable to expect a decrease in performance in the estimate. This leads to an indirect measurement error, since the system performs better when the estimate is not evaluated.
How much this landmark exclusion affects the estimate cannot be measured directly. If more time was available, a starting point would be to exclude the landmarks for the ground truth comparison, and compare the results with and without landmark exclusion. However, this may not show any significant decrease in performance since the synthetic data is not as reliant on the static features corresponding to the landmarks as mentioned above.

5.4.3 Intrinsic calibration assumptions

Although a method of performing intrinsic calibration is presented as part of the method chapter, the authors have not intended to investigate such a calibration method in depth. This calibration was performed purely to establish non-distorted geometries from image points which enables the pose estimation calculations.
The calibration was performed on two separate real vehicles, resulting in two results with what was deemed an insignificant discrepancy between each intrinsic measure. From this, the values retrieved from the two calibrations have been used alternatively to evaluate whether one is better than the other in terms of decreasing errors seen in the real car evaluation. No such clear indication has been found, and thus both sets of intrinsic parameters are assumed to be representative for the entire lineup of cars with the same camera.
However, in strict terms, this assumption is probably incorrect, and the extrinsic parameters could be estimated more accurately if the correct intrinsic parameters were known for each camera. How much difference this actually makes spanning a large set of unique vehicles is unknown. The inclusion of intrinsic parameter estimation is mentioned as a part of the future work of this thesis.

5.5 Research questions

The answer to the research questions is presented in this section. The research questions are the following:

1. To what extent can state-of-the-art feature extractors and matcher reliably establish correspondences across varying vehicular interior environments?
2. What quantitative accuracy can be achieved using the proposed targetless calibration pipeline, and how does this compare to relying solely on the camera's uncalibrated nominal pose?
3. To what extent is the proposed calibration pipeline suitable for scalable deployment in automotive production environments?

5.5.1 Question 1

The results show that state-of-the-art learning-based feature extraction and matching methods can reliably establish correspondences across varying vehicular interior environments, both between simulated and real images. In particular, SuperPoint together with LightGlue consistently outperformed traditional approaches in both the quantity and quality of correspondences.
The results of the Absolute Method and the Relative Method in Section 4.1 showed that enough qualitative matches were acquired to establish correspondences between the simulated images using SuperPoint and LightGlue. This can be seen in Table 4.5. What is further prominent in this table is that the same relative pipeline using state-of-the-art feature extractor and matcher succeeds where the traditional approaches fail. This is shown with the difference in MAE of the Relative Method using the combination of SuperPoint and LightGlue versus any other combination of feature detector and matcher.

The reprojection error for the real images further shows that a satisfactory amount of accurate matched features is established for the Relative Method to reduce the reprojection error for both images. Meanwhile, it may also display that a sufficient number of correspondences cannot be established for the Absolute Method to work accurately, at least not after landmarks exclusion.
The amount of correspondences found with the different detectors and matchers has been evaluated and the results can be found in Appendix B. From this evaluation, it is clear that SuperPoint together with LightGlue can reliably establish correspondences across varying vehicle interiors. For the simulated image, this method produces the highest number of matches, but the other methods also manage to produce a lot of matches. The SuperPoint + LightGlue method is alone, however, in reliably being able to produce matches for the real images. This method produces on average 1413.38 and 671 matches for the synthetic transformation images and real images respectively.

5.5.2 Question 2

The proposed targetless pipeline clearly improves the camera pose estimate compared to relying on the uncalibrated nominal pose, but the improvement depends strongly on whether rotation or translation is considered. On synthetic pure rotation data, the baseline has a geodesic MAE of 3.33°, whereas the Relative Method achieves a geodesic MAE of 0.07°, and the Absolute Method a geodesic MAE of 0.07°.
For the synthetic translation dataset, the baseline error is 5.16 mm. Using the Relative Method, this is increased to 5.29 mm, which is likely due to the translation vector scaling problem mentioned earlier. The Absolute Method did however decrease this error to 1.04 mm.
The reprojection error for the synthetic transformations dataset has a baseline pixel error of 51.21 pixels. This is reduced to 5.42 pixels using the Relative Method and 3.46 pixels using the Absolute Method. This demonstrates that the targetless calibration greatly improves the reprojection error for synthetic data over the baseline.
For the real data, the baseline is on average 18.4 pixels for the two images. This is reduced to an average of 8.2 pixels using the Relative Method and 16.7 pixels using the Absolute Method. The Absolute Method does not manage to reduce the error for both images however, producing a larger error for image 1. This shows that the Relative Method manages to significantly reduce the reprojection error over the baseline. This improvement is important to enable accurate detection using the OMS camera.

5.5.3 Question 3

The proposed methods show strong potential for scalable deployment in automotive production environments. In contrast to traditional calibration approaches relying on physical calibration targets, both presented methods are fully targetless and software-based. This removes the need for additional calibration hardware and dedicated calibration stations. This enables the same calibration method to be inte-

grated both within and beyond the production line. This software implementation further enables the possibility of recalibration after production, for example through service procedures or over-the-air software updates.
The Relative Method has on available real data showed that it more than halves the reprojection error over the baseline of assuming the camera is positioned according to a nominal pose. On the same data, the Absolute Method has shown mixed results. To prove either of these two methods operational enough to be implemented in a production line, further evaluation has to be performed on a larger dataset of images from real cars.

5.6 Future work

This section covers topics and problems that are deemed interesting by the authors to further investigate provided more available resources.

5.6.1 Real data collection

Before implementing any of the two suggested pipelines for production use, they need to be evaluated on more real data. This data will allow for deeper insight into many vital parts of the final results. These being, the lack of performance of the Absolute Method on real imagery, the robustness and performance of the two methods, and how well the tolerances in the synthetic data are represented in the real world.
Suggested future work would be to perform the point mapping error evaluation on a larger real world dataset. This could also be done with and without landmark exclusion to see how this affects the result.

5.6.2 The translation estimation of the Relative Method

Due to limitations in time and resources, the implementation of the translation vector scaling was slightly lackluster. The original results from the non-scaled translation estimate implementation on the synthetic translation data showed that performing this estimate is not trivial. With the estimate being nowhere near the ground truth in many samples, not only in magnitude, but also in direction. These results have affected the amount of resources later invested in the vector scaling.
To perform the vector scaling, a dataset called Landmarks was created. From these landmarks, any two points where the real world distance between them is known could have been used. However, the final implementation made use of two specific landmarks, these being the two positioned in the rear seat in Figure 3.15. These were chosen since they showed high repeatability in terms of feature matching. With this said, they were not found at all times, which led to a non-scaled translation vector for those samples. Furthermore, the landmarks matching was performed with a brute-force matcher, which struggled to match features over synthetic to real and also between the two real images.
The matching from synthetic to real was solved by the authors creating two sets of landmarks, one for the synthetic dataset and one for one of the real images.

The matching between the two real images was never solved, leading to insufficient matches and no scaling for the second real image. With the current setup of brute-force matcher, a set of landmarks would most certainly have to be created for numerous types of different colored interiors.
The suggested future work would be to implement the LightGlue matcher for the landmarks matching, since this can mitigate the problem of matching features originating from different colored interiors. It would also be beneficial to implement more landmarks in the matching process, this would allow for an iterative process choosing which two landmarks were found and the appropriate real world distance for the scaling factor to be computed. This would render a more robust translation vector scaling, and a more accurate translation vector in terms of magnitude than was originally shown in this thesis.

5.6.3 Intrinsic camera calibration

As mentioned in limitations, the intrinsic parameters gathered from the calibration in Section 3.2, were assumed to be representative for all imagery. Since the main objective of the thesis was to determine the extrinsic parameters, this was a natural limitation.
The intrinsic calibration has been performed on two unique real vehicles, where the end result displayed a difference in the parameters. This difference did not generate any significant differences in terms of the reprojection error on real data, and thus one set of parameters were used for all other evaluations. However, it can still be considered to be of interest. In Chapter 2.2.1, it was mentioned that there may be a possibility of extending the Absolute Method to also estimate the intrinsic parameters. This is because the method provides all information required for this calibration, namely the 2D–3D point correspondences.
The idea is to follow the matching procedure used in the current Absolute Method. Using the image points originating from the two reference images, these points would be undistorted and used to triangulate 3D object points. Further, by entering these object points and the distorted image points from the un-calibrated camera into an entire camera matrix estimator, the extrinsic and intrinsic parameters should be possible to estimate.
Estimating the entire P matrix, enables more degree of freedom in terms of possible configurations responsible for such a projection, which could result in a less accurate estimate for the extrinsics. The future work could investigate this idea further, evaluate how well this works, and if this results in a trade-off between accurate estimation of one set of parameters or an approximate of both.

5.7 Final conclusion

This thesis demonstrates that targetless extrinsic calibration of an interior OMS camera from a single snapshot is feasible using repeatable interior features.
Across experiments on synthetic data, both proposed pipelines achieve a very high rotation accuracy, while translation estimation is substantially more challenging at

production scale tolerances. The Absolute Method provides the strongest translation performance on synthetic data, whereas the Relative Method shows the most consistent improvement over the nominal baseline on the limited real vehicle evaluation.
Overall, the results support software only, targetless calibration as a scalable approach for interior sensing, with further real world validation identified as the primary remaining step.
As a final suggestion from the authors to Volvo Cars, the most promising pipeline to implement with minimal additional development, is the Relative Method assuming zero translation. Within the expected mounting tolerances, translation induces comparatively small image displacement while being difficult to estimate robustly. By excluding the translation vector scaling of the Relative Method, the pipeline becomes simpler and more robust, while maintaining comparable results in terms of reprojection error.

Bibliography

[1] Volvo Car Corporation. (2026) Volvos säkerhetssystem: Innovation för trygg körning. [Online]. Available: https://www.volvocars.com/se/safety/technology/

[2] Smart Eye. (2026, Jan.) Interior sensing. [Online]. Available: https://smarteye.se/solutions/automotive/interior-sensing/

[3] Vietsol. (2025, Oct.) What are driver and occupant monitoring systems (dms/oms)? [Online]. Available: https://vietsol.com.vn/driver-and-occupant-monitoring-system-dms-oms/

[4] D. Brown, "Decentering distortion of lenses," 1966. [Online]. Available: https://api.semanticscholar.org/CorpusID:117271607

[5] J. Kannala and S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, pp. 1335–40, 09 2006.

[6] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easily calibrating omnidirectional cameras," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 5695–5701.

[7] R. Szeliski, *Computer Vision: Algorithms and Applications*, 2nd ed. Cham: Springer, 2022. [Online]. Available: http://szeliski.org/Book/

[8] X. Lu, "A review of solutions for perspective-n-point problem in camera pose estimation," *Journal of Physics: Conference Series*, vol. 1087, p. 052009, 09 2018.

[9] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 930–943, 2003.

[10] S. Pan and X. Wang, "A survey on perspective-n-point problem," in *2021 40th Chinese Control Conference (CCC)*, 2021, pp. 2396–2401.

[11] D. Nistér, "Nist??r, d.: An efficient solution to the five-point relative pose problem. ieee-tpami 26(6), 756-770," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, pp. 756–77, 07 2004.

[12] A. Nordmann, "Epipolar geometry," 2007, file: https://commons.wikimedia.org/wiki/File:Epipolar_geometry.svg. License: CC BY-SA 3.0.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, p. 91 – 110, 2004, cited by: 52330. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-3042535216&doi=10.1023%2fB%3aVISI.0000029664.99615.94&partnerID=40&md5=41ad3cf67358797730a0a2f02284a746

[14] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," 2011, Conference paper, p. 2564 – 2571, cited by: 10680. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84856627527&doi=10.1109%2fICCV.2011.6126544&partnerID=40&md5=dbc566b76d19c879142bd4e2b622e4e6

[15] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, p. 2227 – 2240, 2014, cited by: 1240; All Open Access, Bronze Open Access. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84906339516&doi=10.1109%2fTPAMI.2014.2321376&partnerID=40&md5=3f8a750d66e1c74fcda88acc25eb747e

[16] J. Revaud, P. Weinzaepfel, C. D. Souza, N. Pion, G. Csurka, Y. Cabon, and M. Humenberger, "R2d2: Repeatable and reliable detector and descriptor," 2019. [Online]. Available: https://arxiv.org/abs/1906.06195

[17] J. Edstedt, Q. Sun, G. Bökman, M. Wadenbäck, and M. Felsberg, "Roma: Robust dense feature matching," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 19 790–19 800.

[18] G. Potje, F. Cadar, A. Araujo, R. Martins, and E. R. Nascimento, "Xfeat: Accelerated features for lightweight image matching," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 2682–2691.

[19] P. Lindenberger, P.-E. Sarlin, and M. Pollefeys, "Lightglue: Local feature matching at light speed," 2023, Conference paper, p. 17581 – 17592, cited by: 555. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85168832147&doi=10.1109%2fICCV51070.2023.01616&partnerID=40&md5=3cf75222e2a51ea0c749e604ad7534be

[20] D. Detone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," vol. 2018-June, 2018, Conference paper, p. 337 – 349, cited by: 2428; All Open Access, Green Open Access. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85060857830&doi=10.1109%2fCVPRW.2018.00060&partnerID=40&md5=a74aa85211ffe1a2bf48e90881c177f0

[21] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, 2010.

[22] C. G. Harris and M. J. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, 1988. [Online]. Available: https://api.semanticscholar.org/CorpusID:1694378

[23] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: Learned invariant feature transform," 2016. [Online]. Available: https://arxiv.org/abs/1603.09114

[24] P. Lindenberger, P.-E. Sarlin, and M. Pollefeys, "Lightglue: Local feature matching at light speed," 2023, Conference paper, p. 17581 – 17592, cited by: 568. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85168832147&doi=10.1109%2fICCV51070.2023.01616&partnerID=40&md5=3cf75222e2a51ea0c749e604ad7534be

[25] P.-E. Sarlin, D. Detone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," 2020, Conference paper, p. 4937 – 4946, cited by: 2760; All Open Access, Green Open Access. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85089135983&doi=10.1109%2fCVPR42600.2020.00499&partnerID=40&md5=1e593715e4cb52c50a049d377969df69

[26] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2938–2946.

[27] F. M. Chaudhry, J. Ralli, J. Leudet, F. Sohrab, F. Pakdaman, P. Corbani, and M. Gabbouj, "Deep-brownconrady: Prediction of camera calibration and distortion parameters using deep learning and synthetic data," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 18 481–18 492, 2025.

[28] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, p. 381–395, Jun. 1981. [Online]. Available: https://doi.org/10.1145/358669.358692

[29] O. Chum, J. Matas, and J. Kittler, "Locally optimized ransac," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 2781, p. 236 – 243, 2003, cited by: 727. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-33745175816&doi=10.1007%2f978-3-540-45243-0_31&partnerID=40&md5=8c2e4ae2838d693bc7a70e4448a2ce15

[30] D. Barath, J. Noskova, M. Ivashechkin, and J. Matas, "Magsac++, a fast, reliable and accurate robust estimator," 2020, Conference paper, p. 1301 – 1309, cited by: 298; All Open Access, Green Open Access. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85089155154&doi=10.1109%2fCVPR42600.2020.00138&partnerID=40&md5=8c8acef38b2fed7c1f9de2ad3dc7cdae

[31] D. Barath and J. Matas, "Graph-cut ransac," 2018, Conference paper, p. 6733 – 6741, cited by: 354. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85051226271&doi=10.1109%2fCVPR.2018.00704&partnerID=40&md5=118ccf4dd6235980190e8daad41d1a33

[32] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J.-M. Frahm, "Usac: A universal framework for random sample consensus," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, p. 2022 – 2038, 2013, cited by: 641. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84879855860&doi=10.1109%2fTPAMI.2012.257&partnerID=40&md5=7c30f2ff0ac9c247f7e6974e0a1ba3b4

[33] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[34] Unity Technologies. (2026) Unity real-time development platform | 3d, 2d, vr & ar engine. [Online]. Available: https://unity.com/

[35] M. Mat Daud, Z. Kadim, S. L. Yuen, H. H. Woon, I. Faye, and A. S. Malik, "A pre-processing approach for efficient feature matching process in extreme illu-

mination scenario," in *2012 3rd International Conference on Image Processing Theory, Tools and Applications (IPTA)*, 2012, pp. 275–279.

[36] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[37] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," 2018. [Online]. Available: https://arxiv.org/abs/1801.03924

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

A

Method pipelines

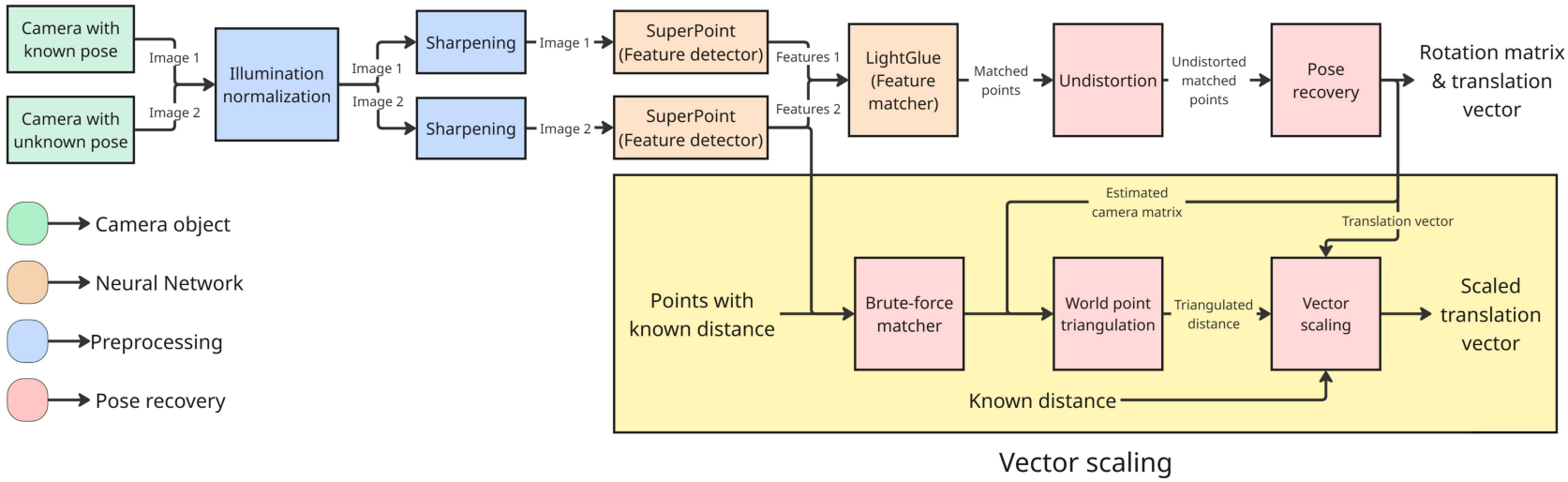


Figure A.1: An image of the pipeline for the relative method.

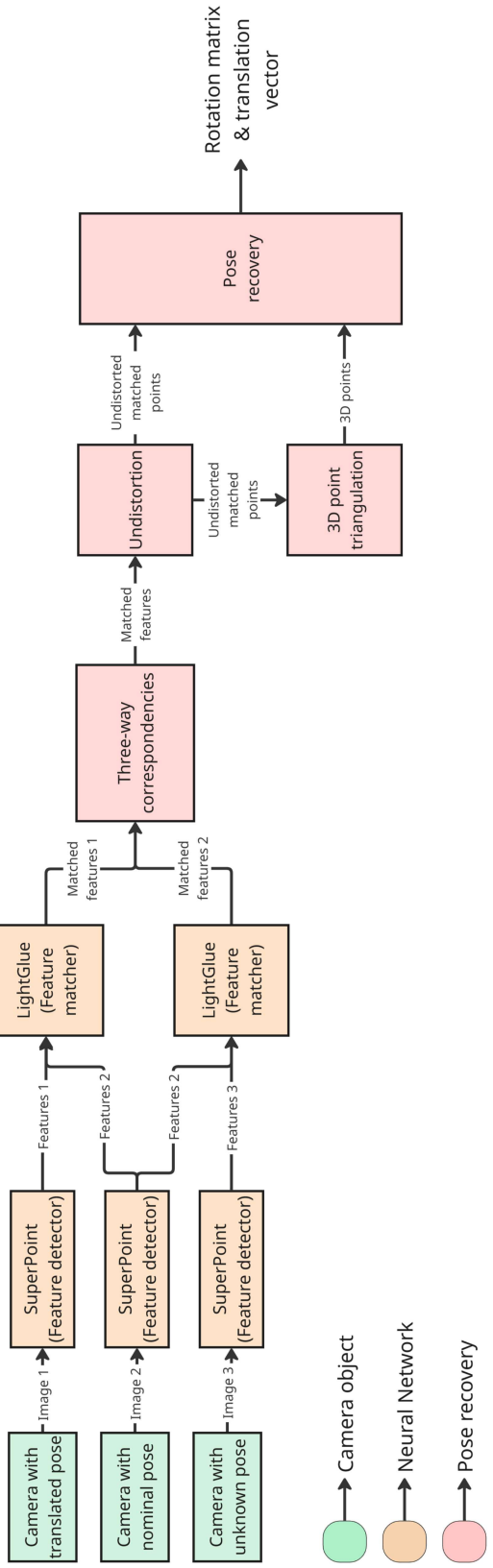


Figure A.2: An image of the pipeline for the absolute method.

B

Comparison of feature matchers and extractors

Three different combinations of feature extractors and matchers were compared using differences between the nominal image and both real and simulated images. The methods compared were ORB together with a Brute Force (BF) matcher, SIFT together with a BF matcher and SuperPoint together with LightGlue. The number of matches and the number of MACSAC inliers for estimating the essential matrix were recorded. The results for a simulated image with the transformation [2,3,-2,-1,2,3] are presented in Table B.1. It is clear that ORB + BF performs much worse than SIFT + BF and SuperPoint + LightGlue. Both of these methods perform similarly in terms of matches but Superpoint + LightGlue produces twice as many MAGSAC inliers. The 50 first matches from the SIFT + BF method and SuperPoint + LightGlue method for the same simulated image is shown in Figure B.1.

Detector	**Raw**	**Inliers**
ORB + BF	193	83
SIFT + BF	1180	212
SuperPoint + LightGlue	1442	429

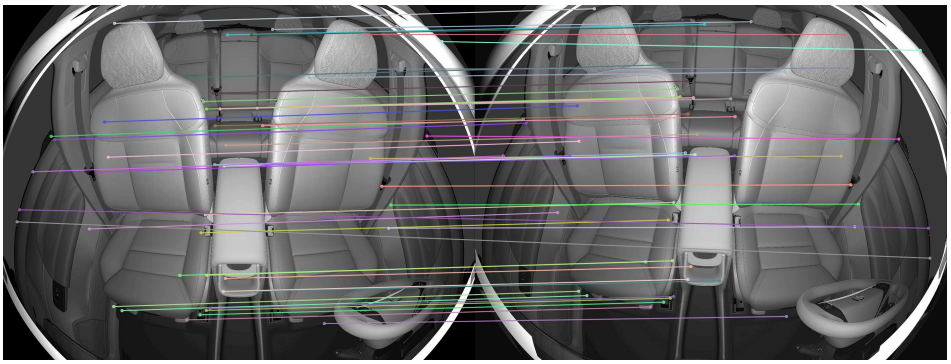
Table B.1: Comparison of detector matching results for a simulated image.

The same results are also gathered for a real image. These results are presented in Table B.2. This matching scenario is much more challenging and therefore produces much fewer matches. ORB + BF produces too few matches that are MAGSAC inliers for this to be a reliable method. Both ORB + BF and SIFT + BF creates a lot of faulty matches, while SuperPoint + LightGlue produces much more consistent matches. This is illustrated with the drawn matches in Figure B.2.

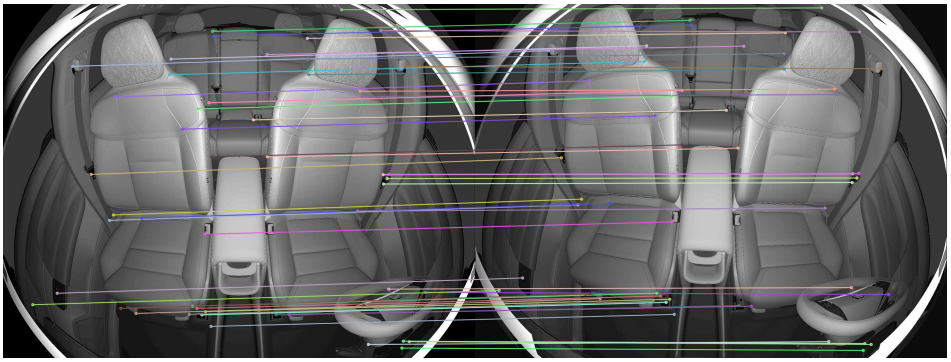
Detector	**Raw**	**Inliers**
ORB + BF	119	8
SIFT + BF	953	46
SuperPoint + LightGlue	651	69

Table B.2: Comparison of detector matching results for a real image.

The SuperPoint + LightGlue method produces on average 1413.38 and 671 matches for the synthetic transformation images and real images respectively, it also produces 417.75 and 75.5 MAGSAC inliers for the synthetic transformation images and real images respectively.

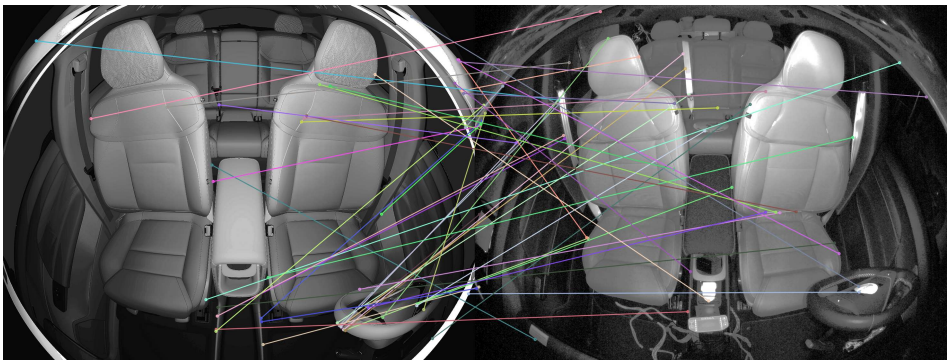


(a) Drawn matches for SIFT + BF.

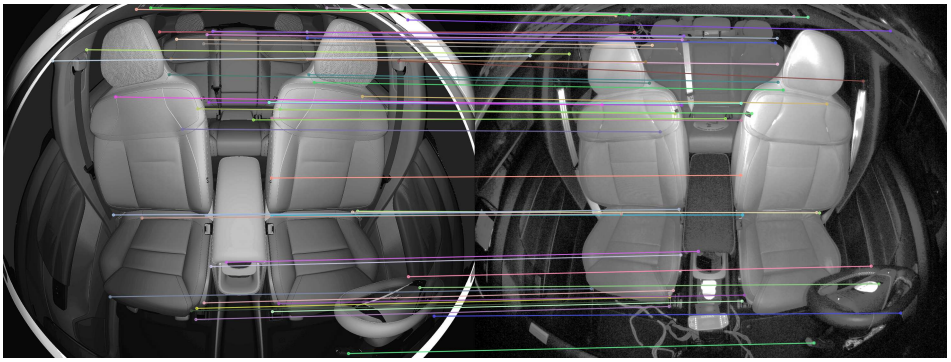


(b) Drawn matches for SuperPoint + LightGlue.

Figure B.1: Drawn matches between the first 50 matched points.



(a) Drawn matches for SIFT + BF.



(b) Drawn matches for SuperPoint + LightGlue.

Figure B.2: Drawn matches between the first 50 matched points.

DEPARTMENT OF Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY