CHALMERS
UNIVERSITY OF TECHNOLOGY

High-Precision Positioning of Known Objects Using a Static Monocular Camera

Master's Thesis in Electrical Engineering

Ariel Naderi - Mahin Garg

MASTER'S THESIS 2025

High-Precision Positioning of Known Objects Using a Static Monocular Camera

Assessment and Development of Image-Based Object Detection Methods

Ariel Naderi - Mahin Garg

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
Gothenburg, Sweden, 2025

High-Precision Positioning of Known Objects Using a Static Monocular Camera
Assessment and Development of Image-Based Object Detection Methods
Ariel Naderi - Mahin Garg

Supervisor: Jonas Ekdahl, Mattias Morichetto, Volvo Cars
Examiner: Jonas Sjöberg, Department of Electrical Engineering

Master's Thesis 2025
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Göteborg,
Sweden
Telephone +46(0)31 772 1000

Cover: Illustration of in-cabin environment of Volvo cars model S90

Typeset in LaTeX
Printed by Chalmers Digitaltryck,
Gothenburg, Sweden 2025.

High-Precision Positioning of Known Objects Using a Static Monocular Camera
Assessment and Development of Image-Based Object Detection Methods
Ariel Naderi - Mahin Garg
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Accurate perception of the internal vehicle environment is essential for occupant safety, enabling timely and coordinated system responses in critical situations. While advanced perception systems based on deep learning have shown strong performance, they often require high computational and financial resources. This thesis explores classical, image-based object detection methods using a static monocular camera as a lightweight and cost-effective alternative. The focus is on precise localization of the steering wheel as a dynamic object in a static in-cabin environment.

Although the camera is fixed, the setup is interpreted as a reverse scenario of moving camera and static object, meaning the object motion simulates camera movement. By tracking keypoint displacement across frames, the corresponding camera motion is inferred.

The methodology includes image preprocessing for distortion correction and enhancement, evaluation of classical methods for feature extraction and matching, camera motion estimation using a visual odometry pipeline, and triangulation-based 3D reconstruction of 2D image points.

Among the tested methods, ORB-BF and Lucas-Kanade (LK) show superior performance in keypoint detection, matching, and motion vector extraction. ORB-BF is used for consistent keypoint matching, while LK provides displacement vectors, both essential for pose estimation and 3D localization. The resulting pipeline is interpretable, computationally efficient, and well-suited for low-power, cost-sensitive in-cabin monitoring systems.

Overall, the results confirm that reliable object localization is achievable using only a single monocular camera. However, the absence of real-world measurements, such as known object dimensions and camera intrinsics, limits the full validation of displacement and depth estimations.

Keywords: High precision positioning, object tracking, static sensors, computer vision, visual odometry

Acknowledgment

For the EENX30 Master's thesis in Electrical engineering course, we the thesis project members would like to express our deepest gratitude to our supervisor and examiner Jonas Sjöberg from Electrical Engineering Division at Chalmers University. He has helped us by constantly supervising, assisting and helping us to analyze this project from various angles. He also helped us with various other aspects of the project.

Additionally, we would also like to express our gratitude to Jonas Ekdahl, our supervisor, and Erik Liedholm as team manager, both from the Internal Perception team at Volvo Cars, for providing us with the opportunity to work on this project and their technical support throughout various aspect of it.

It is through consistent support, guidance, and dedication of all these people which has played a crucial role in the successful development and advancement of the drawbar eye identification and guiding project.

Mahin Garg, Gothenburg, June 2025
Ariel Naderi, Gothenburg, June 2025

List of Acronyms

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

ADAS	Advanced Driver Assistance System
AKAZE	Accelerated KAZE
BF	Brute-Force
BRIEF	Binary Robust Independent Elementary Features
DoG	Difference of Gaussians
FN	False Negative
FP	False Positive
FAST	Features from accelerated segment test
IR	Infrared
KNN	K-nearest neighbors
LLM	Large Language Machine learning models
L2	Least-squares
ML	Machine Learning
RGB	Red, Green, Blue
ORB	Oriented FAST and rotated BRIEF
SIFT	Scale-invariant feature transform
SLAM	Simultaneous Localization and Mapping
SURF	Speeded up robust features
TP	True Positive
VSLAM	Visual Simultaneous Localization and Mapping
V2X	Vehicle to X e.g. Vehicle, infrastructure, etc.
VO	Visual Odometry

List of Figures

List of Tables

Contents

1

Introduction

1.1 Problem background

Humanity has set an ambitious goal to enhance the safety in transportation systems while advancing sustainable mobility. Vision Zero [1] shifts this responsibility from drivers to system designers, which has led to the advancement of intelligent systems such as ADAS, V2X communication and machine learning. In recent years, the tendency toward machine learning-based approaches has grown significantly, contributing to major advancements in active safety systems. However, these methods are often computationally and financially demanding, as they are data-intensive and require large datasets to achieve high accuracy. One way to mitigate these costs is to reduce the reliance on AI models by leveraging classical techniques (which do not require AI models) while optimizing safety.

A common approach to enhance safety is to improve a system's perception and understanding of its environment. This capability has rapidly evolved from reliance on satellite-based systems, which was limited by cost and signal issues, to Simultaneous Localization And Mapping (SLAM). SLAM processes real-time sensor data to perceive and navigate environments. Depending on the sensors used, SLAM is categorized into various types, including LiDAR SLAM (Light Detection and Ranging SLAM) and VSLAM (Visual SLAM). VSLAM relies on visual odometry (VO), which refers to the process of tracking objects and estimating camera's movement over the time and frames.

The workflow steps for VO involves feature detection, extraction and matching, camera motion estimation and pose estimation. The common work flow of VO is presented in figure 1.1

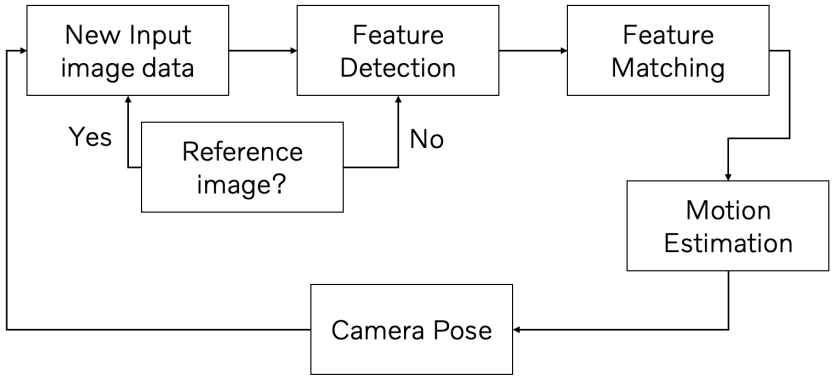

Figure 1.1: Visual Odometry Work Flow

Another challenge in vehicle safety is that collisions involve both external and internal factors which affect occupants inside the vehicle as well as those outside. While modern SLAM systems primarily focus on external perception, the in-cabin environment is often overlooked. Volvo Cars, a leader in automotive safety, actively addresses this gap by focusing on both external sensing and in-cabin occupant safety, reinforcing its commitment to comprehensive safety solutions. One such example is the activation of airbags, which can use additional data from accurate steering wheel location for minimizing post-crash injuries to the driver via an enhanced assessment of driver location, thus leading to a more controlled activation of airbags.

This thesis focuses on finding the 3-D position of the known dynamic objects such as steering wheel using a static monocular (single lens) camera, via the assessment of existing VO-methods, and 3D reconstruction of detected 2D keypoints in the 2D images.

1.2 Contributions

The main contributions of provided by the thesis are as follows:

- Assessment of classical computer vision methods for feature extraction, matching, object detection, and classification, with the goal of reducing reliance on AI-based approaches and minimizing overall computational and financial costs.
- Development of an efficient image-based method for detecting and localizing known dynamic objects such as steering wheel.

2
Related Work

2.1 Related Work

In visual odometry, feature extraction and matching are considered fundamental steps for detecting and localizing objects. Feature extraction [2] is typically performed by identifying unique patterns in images.

Methods like SIFT [3], SURF [4], KAZE [5], AKAZE [6], and ORB [7] have been widely used for extracting features and patterns in stereo vision (images taken from cameras with two lenses) and RGB image contexts where depth information is available. While these methods show good performance in scenarios where the camera is moving, their adaptability to static setups and IR images remains under-explored.

Traditional object detection approaches are broadly classified into pixel-based and keypoint-based categories. Pixel-based methods, such as frame differencing (FD) [8], Horn-Schunck optical flow [9], and Farnebäck optical flow [10], detect motion by analyzing pixel intensity changes across frames. While simple and effective in RGB scenes, they are computationally expensive due to processing of higher number of pixels (due to addition of pixels in RBG images compared to IR images) and their accuracy tends to degrade near object boundaries due to smoothness assumptions in the algorithms.

Keypoint-based methods rely on local feature extractors such as SIFT and ORB. These approaches are robust in cluttered environments but are sensitive to low-light conditions and geometric distortions. In these methods, RANSAC [11] is commonly used to remove outliers and improve tracking, especially when a static camera is involved. For scenes with multiple moving objects, motion segmentation [12] and background subtraction (BgS) [13] are often used for real-time motion detection.

Template matching [14] is also evaluated as a classical, interpretable approach for static object detection and internal calibration. Its sensitivity to IR-specific

conditions, such as illumination changes and limited contrast, is assessed to determine its suitability for supporting calibration and tracking tasks. While this method demonstrates good capability in identifying and locating stationary objects, its performance on IR images has not been widely studied.

However, the majority of these methods are developed for stereo vision systems or scenarios involving moving cameras. This work, by contrast, uses a static monocular camera in a constrained environment with a single dynamic object.

3

Theory

This chapter presents the theoretical background of the image-based motion estimation and object localization methods, which form the basis of the research done and are applied in the Methodology chapter.

3.1 3D Reconstruction

In computer vision, 3D reconstruction refers to the process of recovering the three-dimensional structure of objects from 2D images shown in figure 3.1.

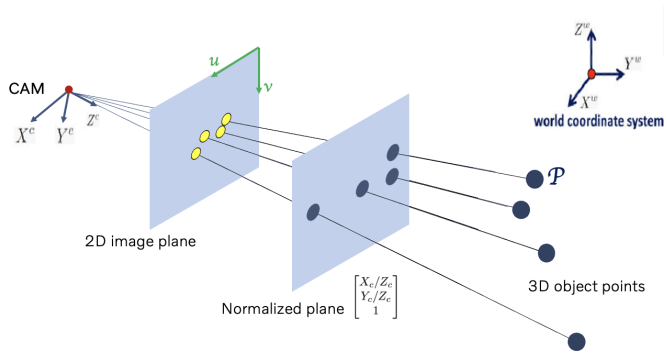

Figure 3.1: 3D projection of the 2D points

This process relies on two fundamental equations: equation 3.1 and equation 3.2.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Intrinsic Matrix } K} \cdot \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \\ 1 \end{bmatrix} \tag{3.1}$$

$$\begin{bmatrix} X_C \\ Y_C \\ Z_C \end{bmatrix} = \underbrace{R}_{\text{Rotation Matrix}} \cdot \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} + \underbrace{t}_{\text{Translation Matrix}} \tag{3.2}$$

Here, X_c, Y_c, Z_c denotes the camera coordinates, while X_W, Y_W, Z_W represents the real-world coordinates. The value Z indicates the depth, i.e., the distance from the camera to the object in the scene.

Estimating the 3-D position of an object in the real world using a uncalibrated monocular camera in a static setup, requires recovering the camera motion (R, t), estimating depth, and utilizing the intrinsic matrix (K) along with motion vectors derived from keypoint correspondences.

3.2 Triangulation

Triangulation is a method that can be used to estimate depth (Z) of a 3D point from two 2D image views. In moving camera scenarios, based on trigonometric principles, if the baseline between cameras is d, and viewing angles α and β, the depth z of a point P can be expressed in equation 3.3 below:

$$z = \frac{d \cdot \sin\alpha \cdot \sin\beta}{\sin(\alpha + \beta)} \tag{3.3}$$

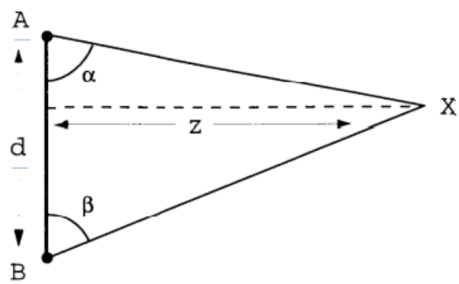

Figure 3.2: Visual description of Triangulation

In homogeneous coordinates, the triangulated 3D point is returned as $[X'Y'Z'W]^T$, where W stands for homogenous scale factor. To convert this point to standard 3D coordinates and handling the projective scaling the X, Y, Z must be divided by scale factor W, which provides real-world location.

$$X = \frac{X'}{W}, Y = \frac{Y'}{W}, Z = \frac{Z'}{W} \tag{3.4}$$

Known object size is one approach to estimate the scale factor W when using monocular cameras.

$$\text{scale factor} = \frac{\text{real world size}}{\text{pixel size}} \tag{3.5}$$

In this equation, pixel size refers to the size of the object in the 2D image plane (in pixels), while real-world size represents the actual size of the object in the real-world coordinate system, typically measured in meters or centimeters.

3.3 Camera Calibration

Camera calibration refers to estimating the internal and external parameters which defines how a camera captures 3-D scenes and projects them onto a 2D image plane.

Intrinsic parameters includes focal lengths (f_x, f_y), the principal point (c_x, c_y), skew, and lens distortion coefficients. These parameters are captured by the intrinsic matrix K:

$$K = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

In contrast, extrinsic parameters define the camera's position and orientation in the world coordinate system. They consist of a 3×3 rotation matrix R and a 3×1 translation vector t. These parameters transform 3D world coordinates (X_w, Y_w, Z_w) into camera coordinates. These matrices are essential for accurate motion and pose estimation, and correcting lens distortion.

3.4 Camera Distortion

There are two main types of lens distortion that are observed during camera calibration - radial distortion and tangential distortion. Analyses and calibration of both these type of distortions is important since it helps in providing accurate point-to-point distance values in an image.

In radial distortion, straight lines tend to appear curved, either outward or inward. Radial distortion becomes larger the farther points are from the center of the image. This type of distortion is mainly due to curvature of lens that capture the light, and trying to use a wide-angle lens to capture details often leads to radial distortions, especially near the edges of an image. Similarly, tangential distortion occurs because the image-taking lens is not aligned perfectly parallel to the imaging plane. So, some areas in the image may look nearer than expected, and others may look further away.

Figure 3.3: Radial and tangential distortion

3.5 Epipolar Geometry

Epipolar geometry is an essential concept in computer vision that describes the geometric relationship between two camera views of the same scene. It enables motion estimation, 3D reconstruction by searching for corresponding points across images.

Given a point p in one image, has an corresponding point p' in the second image, which must lies on a line called the epipolar line (l', l') as mentioned in the figure 3.4.

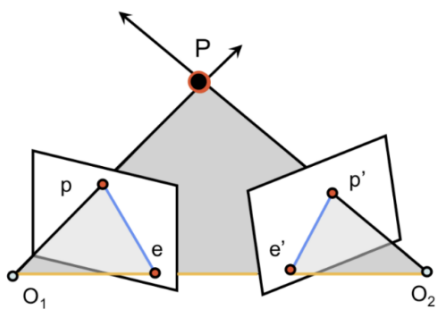

Figure 3.4: Camera epipolar geometry

Epipolar line equation can be expressed as below equation 3.7

$$l = F^T p' l' = Fp \tag{3.7}$$

Where F represents Fundamental Matrix $F \in \mathbf{R}^{3\times3}$ that relates pixel coordinates p and p' as the following equation 3.8:

$$p'^{T} F p = 0 \tag{3.8}$$

F is driven from Essential Matrix $F \in \mathbf{R}^{3\times3}$ that encodes the relative rotation R and translation t between two cameras as mentioned in equation 3.10.

$$\mathbf{F} = K'^{-\top} E^{-1} \tag{3.9}$$

$$E = [t]_{\times} R \tag{3.10}$$

Where $[t]_{\times}$ is the skew-symmetric matrix:

$$[\mathbf{t}]_{\times} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{3.11}$$

Epipolar geometry can be used to identify the dynamic and static objects in scenarios with moving camera. Point that lies on the epipolar lines counts as static point, which is called inliers. While dynamic points has their own movements therefore can be outliers.

3.6 Motion Estimation

The 3D position of an object in the real world coordinate system is directly effected by the camera location in the space. In scenarios with moving camera, camera motion can be estimated using Singular Value Decomposition (SVD) of Essential matrix. Let I_t denote the current frame at time t, and I_{t+1} the next frame, then camera motion between consecutive frames can be determined from equation 3.12

$$R_{t+1} = UWV^{\top}, \quad t_{t+1} = \pm U[:, 2] \tag{3.12}$$

Where R and t represents Rotation (Yaw, Pitch, Roll) and Translation (x, y, z) respectively. For further explanation about SVD method and U, W, V matrices check [15].

3.7 Uncertainty Metrics

The performance of algorithms can be evaluated using different metrics such as precision, Recall, and F1 score.

Precision refers to the algorithm's ability to correctly identify relevant results out of all the results it predicted as correct. In the case of feature extraction and matching, precision checks out of all good matches how many are correct using following equation:

$$\text{precision} = \frac{\text{Inliers}}{\text{good matches}}$$

Good matches represents the matches that pass initial filtering like ratio test or distance threshold. Inlier refers to subset of good matches that are geometrically consistent through frames.

Recall measures how many of the actual correct results the algorithm have been able to detect. In the case of feature extraction and matching, recall checks out of all actual correct correspondences, how many are detected.

$$\text{recall} = \frac{\text{Inliers}}{\min(\text{KP1, KP2})}$$

KP stand for keypoints detected by algorithm in the reference image (1) and following image (2). The term $min(\text{KP}_1, \text{KP}_2)$ represents the maximum number of potential correct matches, as it is limited by the smaller number of keypoints detected in either of the two frames.

F1 score balances Precision and Recall to give a single performance metric. This metric can be used to check if an algorithm is good at finding correct correspondences without missing ones.

$$\text{F1 Score} = 2 * \frac{\text{Precision * Recall}}{\text{Precision + Recall}}$$

4

Methodology and Algorithm Evaluation

This chapter consists of the methodology pipeline followed for data collection, pre-processing of images and the three phases for evaluation of different algorithms. These steps are based on related works and theoretical backgrounds covered in the previous chapters.

4.1 Methodology Pipeline

The proposed methodology pipeline to estimate the 3D position of the steering wheel is illustrated in figure 4.1.

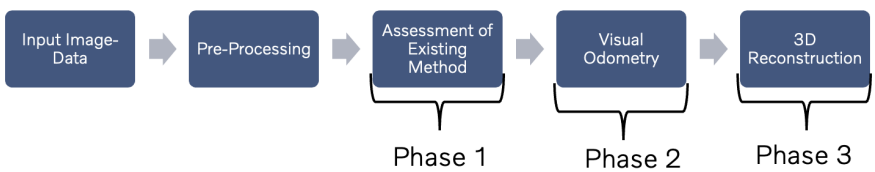

Figure 4.1: Methodology Pipeline

The first step involves input image data collection and basic pre-processing, as the data is collected from an uncalibrated camera. Phase 1 of evaluation focuses on the assessment of classical vision methods for feature detection, matching, and object detection. In Phase 2, visual odometry is applied to estimate the camera's motion and pose. Finally, Phase 3 performs 3D reconstruction by estimating depth to localize the position of steering wheel. Each phase is described in detail in the following sections.

4.2 Experimental Setup and Data collection

An uncalibrated monocular IR camera mounted in the test vehicle is used to collect the data images. This is done to test different algorithms in a way that is verifiable and repeatable for future testing and validation.

A Volvo XC 90 test vehicle is used in the experiment. It is a production vehicle with adjustable steering wheel and a monocular IR camera. The steering wheel has 2 types of movement - telescopic and tilt. For experimental purposes, the minimum, maximum and the middle positions of the steering wheel along both of the movement axes are marked for references due to lack of CAD model measurements.

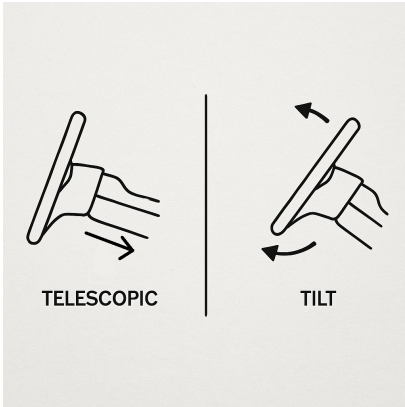

Figure 4.2: Telescopic and tilt movement in steering wheel [16]

The ground truth distance value refers to the absolute distance between middle position (marked with yellow dot) and extreme values of steering motion. In the telescopic direction (OUT-IN), this value is 2.75 cm, while in tilt direction (UP-DOWN), it is 1.25 cm.

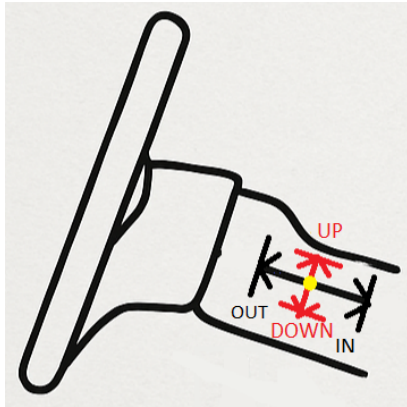

Figure 4.3: Ground truth different positions for data collection

The image data covers all the extreme positions in which the steering wheel can be adjusted by the driver. This includes the motion of the steering wheel along the telescopic and tilt direction in different positions and combinations as mentioned in the table below:

Table 4.1: Data collection for algorithm testing

Frame Number	Steering Position (Telescopic)	Steering Position (Tilt)
1	Middle	Middle
2	In	Down
3	In	Up
4	Out	Down
5	Out	Up

All these 5 images are collected with two different sizes- low resolution setting and high resolution setting. This is done to investigate the possibility of any major differences in algorithm performance and accuracy based on their resolution, and if the difference is enough to potentially offset the higher space utilization for higher resolution images.

4.3 Pre-processing and Optimization of Data

The data is collected from an uncalibrated camera which includes lens distortion, therefore a custom enhancement pipeline is applied. This custom enhancement pipeline, called “preprocessing function” builds a virtual camera matrix using image size (length and height), and applies distortion coefficient to fix barrel and tangential distortion, and crop out the black borders. Since verification methods such as checkerboard pattern aren’t available, distortion coefficient is manually defined to correct the radial and tangential distortions.

Next, the function applies Histogram equalization and adaptive enhancement to improve local contrast, remove noises while preserving edges, and enhance edges in the image.

Furthermore, a custom function called "distortion" is implemented to estimate the camera intrinsic parameters (K) based on an initial guessed K. The intrinsic matrix K is frequently used in computations involving intrinsic-based models such as the Essential and Fundamental matrices.

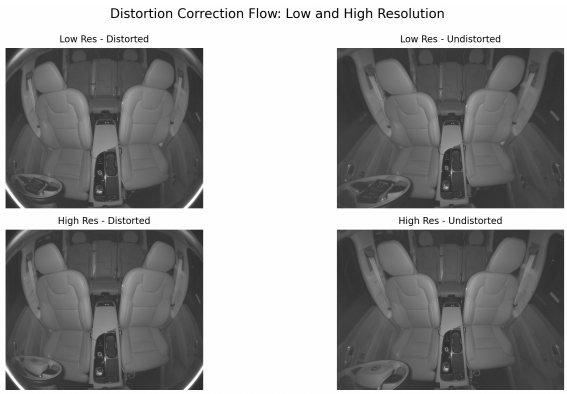

Figure 4.4: Distortion correction of low resolution and high resolution image

The figure 4.4 shows that the barrel lens distortion (visible via bright curving lines at the extreme edges of the image in the distorted images) is removed. The figure also appears flat instead of looking like bulging outwards. However it still has some noise and low contrast, which makes it hard to distinguish certain key features. These problems are corrected next, with the final results shown below -

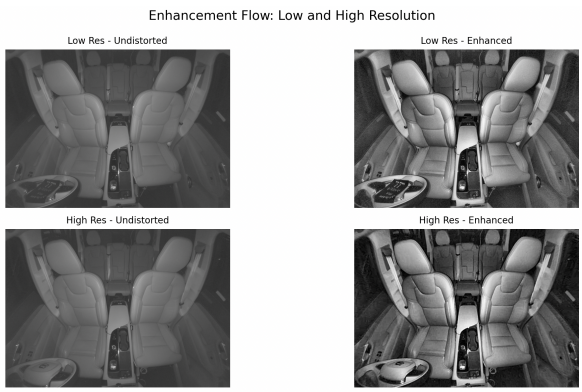

Figure 4.5: Image enhancement of low and high resolution image

The figure 4.5 shows there are noticeable enhancements, with contrast becoming larger between the darker and lighter areas, thus making it easier to identify features like edges and corners. Adaptive enhancement also leads to highlighting of certain new features, particularly on steering wheel (like car logo), which was initially not identifiable clearly.

4.4 Assessment of Existing Methods

4.4.1 Evaluation of Feature Extraction and Matching Methods

In the first phase of methodology pipeline, multiple feature extraction methods are implemented and evaluated to assess their effectiveness in extraction and matching of keypoints. Methods like SIFT, ORB, KAZE and AKAZE are implemented on the raw, undistorted and enhanced data images. Maximum number of features that can be detected is set to 2000 keypoints to keep comparison fair. These methods extracts features and their descriptors following matcher as discussed in related work:

Table 4.2: Feature descriptors and matching strategies

Descriptor	Brute-Force (BF)	K-Nearest Neighbors (KNN)
SIFT	L2	L2
ORB	Hamming	Hamming
KAZE	L2	L2
AKAZE	Hamming	Hamming

RANSAC is used to identify the inliers and outliers from detected and matched keypoints. The matches are filtered further by distance and ratio threshold, which is set at 0.75 pixel value.

Each method is tested on these frame pairs to extract and match the features. The results are evaluated using metrics like precision, recall, F1, and total time. The performance is quantified via factors like the number of detected keypoints, good matches, and inlier. Here, good matches refer to the filtered matches with low descriptor distance, and inlier refers to the matches that agree with the motion between two images.

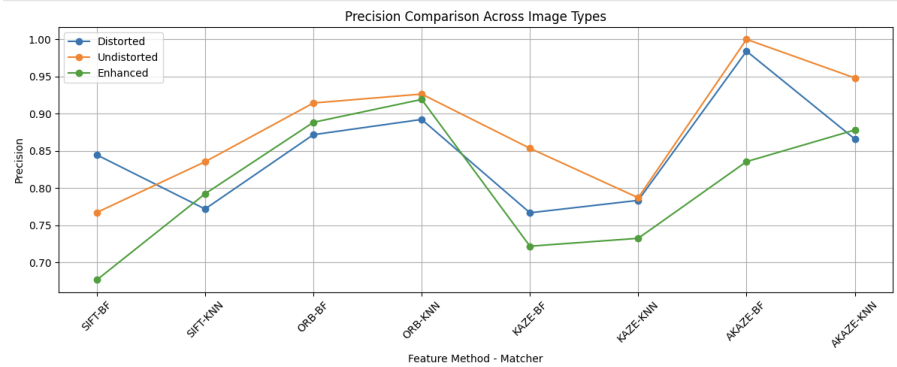

Figure 4.6: Comparison of Precision score over different methods

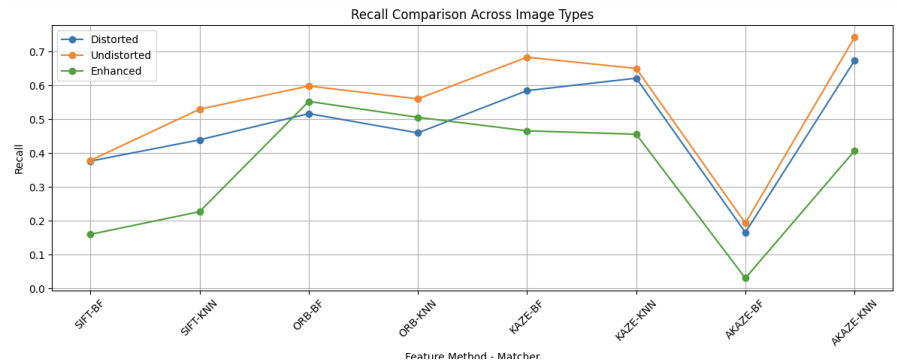

Figure 4.7: Comparison of Recall score over different methods

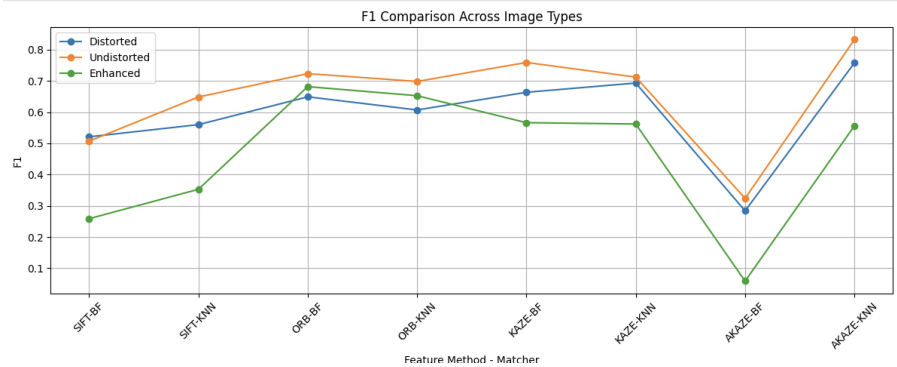

Figure 4.8: Comparison of F1 score over different methods

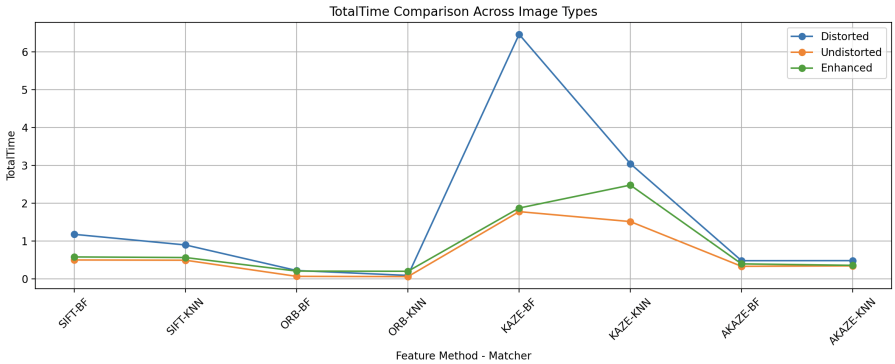

Figure 4.9: Comparison of total process time over different methods

Results from figure 4.6 shows how many matched keypoints are correct out of all the good matches. On comparing different types of frames, AKAZE-BF has the highest precision for undistorted images, and SIFT-BF has the lowest precision for enhanced images. However, ORB is the most balanced and reliable out of all, with better precision compared to other methods.

Results from figure 4.7 indicates that SIFT struggles with distortion and enhanced image matching, while KAZE showes best recall for undistorted images. ORB again is balanced and reliable and has a good recall score compared to other methods.

Results from figure 4.8 shows a higher F1 score for the undistorted images when using the AKAZE-KNN method, indicating a good balance between precision and recall. While SIFT performs weak under all the conditions compared to other methods, ORB is very consistent across the same.

When comparing the total time to process 2000 requested keypoints, ORB is the fastest method, while KAZE-BF is the slowest.
The outcomes from evaluation of feature extraction and matching method indicates that :

- Undistorted images are the best option for evaluation, while the results on enhanced images are inconsistent
- ORB-KNN and BF have good accuracy and are very fast
- AKAZE-KNN has the best F1 score but is slightly slow
- KAZE-KNN and BF have good accuracy but are very slow
- SIFT has bad performance and is unreliable

Based on these outcomes, ORB-KNN and BF are good for real time vision with undistorted images and AKAZE-KNN is good for high accuracy demanding and offline processes. ORB-BF is selected as best performing method since

it shows better results in F1 score compared to ORB-KNN, while being very fast in execution. Furthermore, the enhancement of images is not necessary in real-time processes.

4.4.2 Optimization of Selected Feature Extraction and Matching Method

From feature extraction and matching Mmethods step, ORB-BFF method remains as the most accurate and suitable method. To further optimize its accuracy, an enhanced ORB pipeline is developed. This ORB detector is configured with optimized parameters such as increased patch size, lower FAST threshold, and larger feature count for enhanced detection. Moreover, to ensure spatial consistency, the fundamental matrix (F) is used to reject outliers, which overcomes the Epipolar geometry constraint. For further validation, LK optical flow is computed from frame 1 to 2. Matches are retained only if the flow-track positions are within 5 pixels of the matched keypoints to ensure temporal consistency.

Finally, the enhanced pipeline is compared against a standard ORB configuration based on precision, recall, F1 score, inliers, and processing time.

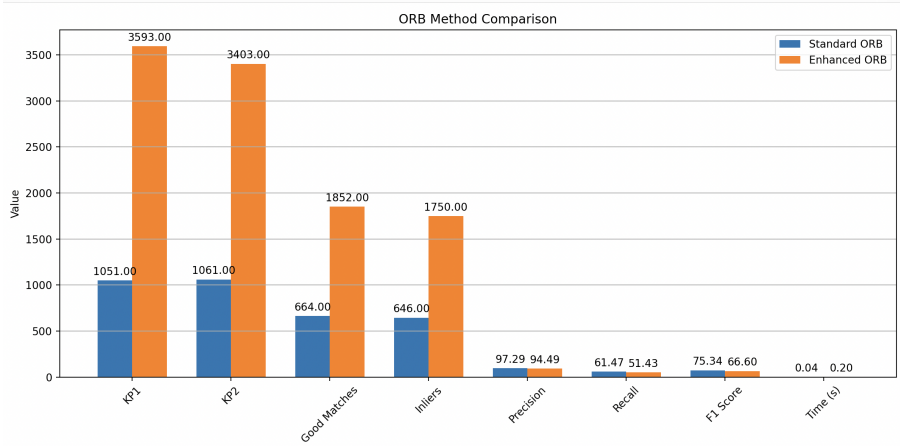

Figure 4.10: Comparison of Standard and Enhanced ORB Pipelines

The figure 4.10 shows the comparison between standard and enhanced ORB methods. Both methods detects a high number of keypoints, with an increase in maximum number of feature detection from 2000 to 10000. The enhanced ORB consistency reaches the 3 times more detected keypoints which represents better feature representation using undistorted frames.

The enhanced ORB provides more good matches and inliers, which indicates higher robustness and geometrically valid correspondences.
However, performance metrics shows better speed in the standard pipeline.

The enhanced pipeline is 5 times slower than the standard one, which is expected since it provides more information like computation of Essential and Fundamental matrices, estimation of keypoints properties (position, scale) and detection of more keypoints.

The outcomes from this optimization shows that the standard ORB is better and more suitable for both speed and accuracy if only keypoints extraction and matching is important. The enhanced ORB is useful for 3D reconstruction since it can detect more features and provide more good matches. Thus the enhanced version of ORB is used for camera motion estimation and 3D reconstruction.

4.4.3 Evaluation of Object Detection Methods

Multiple classical methods are implemented and evaluated to assess their suitability and efficiency in detecting objects between consecutive frames. These include ORB-based keypoint tracking, frame differencing, background subtraction, optical flow methods (LK and Dense), RANSAC and motion segmentation. The focus of this step is to narrow down from the variety of methods to ones which are lightweight, cost effective and efficient in execution. Implemented methods are weighted and eliminated based on the quality of object detection, bounding boxes accuracy, time and memory footprint.

Table 4.3: Comparison of feature tracking and object detection methods

Method	Time (s)	CPU (%)	Memory (MB)
frame_difference	47.11	5.30	49.11
background_sub	0.05	112.80	29.31
ransac_motion_segmentation	22.3494	7.90	84.39
farnebäck_dense	20.49	8.20	38.11
LK_sparse	0.28	114.40	103.16
LK_sparse_kp	20.3175	6.40	23.72
Orb_detection	3.2485	33.90	15.55

Based on the table 4.3, Background Subtraction is the fastest method with a total 0.05 seconds as its processing time, but it has a high CPU usage (112.8 %). Therefore, this method is more suitable for real-time applications in static scenario, but less ideal under scenarios with high computational power usage or camera motion. After Background Subtraction, Sparse Lucas-Kanade (LK) is the next fastest method with processing time of 0.28 seconds. However, it also consumes high memory and has high CPU usage, likely due to its pyramid-level processing.

Farnebäck dense optical flow has a good balance of processing time (20.49 s), has moderate memory usage and low CPU usage, making it effective for detecting dynamic motion with smooth optical flow around object contours.

However, the ORB-based version of LK was a better choice, with processing time of 20.32 s and a moderate memory and CPU usage. Also, ORB-based

detection became significantly faster (3.25 s), while having low memory usage and moderate CPU usage, making it a more feasible option.

Figure 4.11: Object detection: LK-ORB-RANSAC

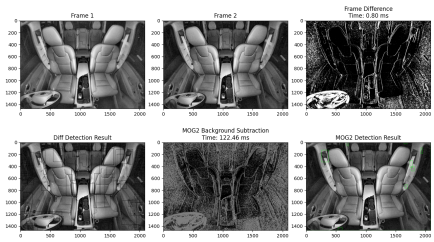

Figure 4.12: Object detection: Background Subtraction method

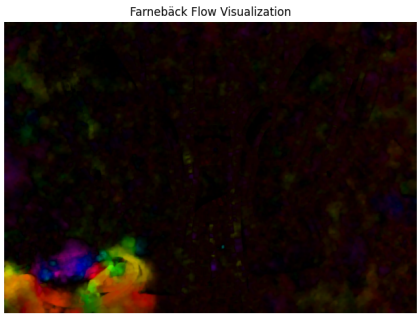

Figure 4.13: Object detection: Farnebäck Flow

Figure 4.14: Object detection: LK

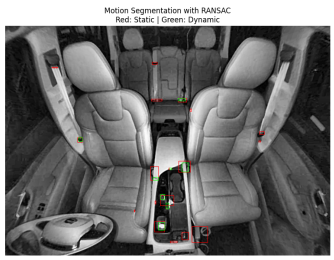

Figure 4.15: Object detection: Motion Segmentation

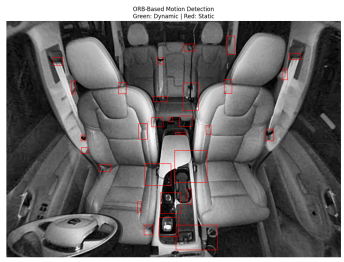

Figure 4.16: Object detection: ORB-Geometry

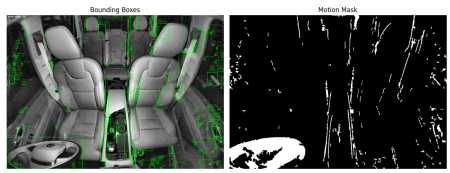

Figure 4.17: Object detection: Frame Difference - Motion mask

In terms of detection accuracy from the above figures, ORB-based methods (figures 4.11 and 4.16) can detect static objects well, but struggles with dynamic ones like the moving points, which often fall outside the inlier set. As shown in the figure 4.11, the combination of ORB, RANSAC and DBSCAN is able to detect dynamic objects but with error.

Farnebäck dense flow consistently identified dynamic objects with smooth and accurate motion fields. This method is based on pixel motions and can only provide dense motion. In contrast, LK keypoint tracking features is able to detect static and minor dynamic features with reasonable localization accuracy and lower noises compared to other methods. However, this method requires higher feature number that 2000 to detect features.

Frame differencing produced many false positives, but after static object filtering, it effectively isolated dynamic motion using mask method. Similar to FD, motion segmentation also uses mask to isolate dynamic object and everything static assumes as background. However, this method is dependent on the FD method.

Given that the final goal is detection and localization of dynamic object, methods such as e.g., background subtraction and dense flow that lack trackable keypoints and includes errors are not considered for further steps. For depth estimation and motion analysis, keypoint tracking is essential, therefore, the selected methods for continued use include FD, Sparse LK and ORB-Geometry, as they provide high accurate bounding box, reliable feature tracking and geometric consistency necessary for downstream localization tasks.

From these results, Farnebäck dense flow, background subtraction and motion segmentation are excluded from further analyses.

4.4.4 Optimization of Selected Object Detection Methods

The remaining methods, ORB-Geometry, FD, and LK have been optimized and combined in various ways to detect both static and dynamic objects. This step aims to find a combination that can detect, bound and classify both dynamic and static objects.

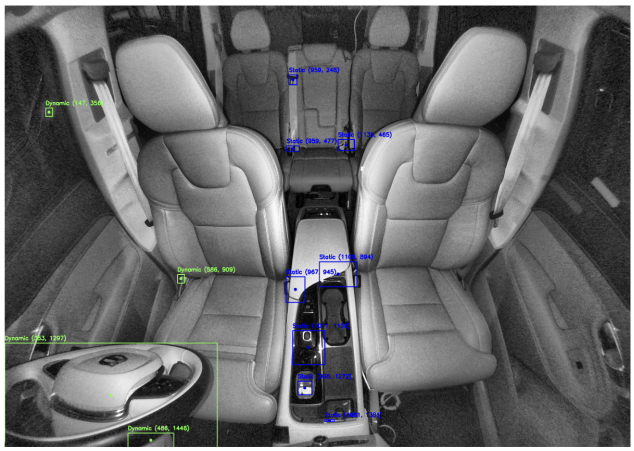

Figure 4.18: Combined FD and ORB methods

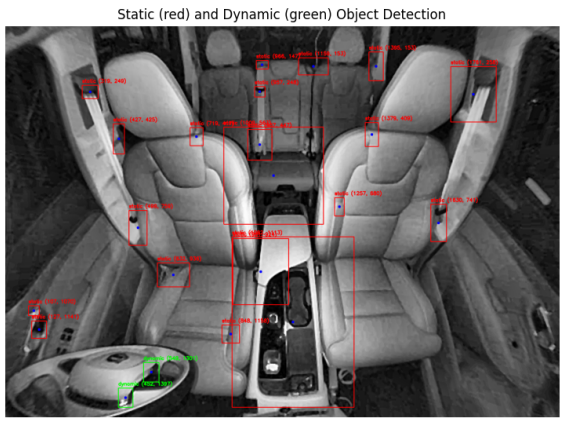

Figure 4.19: Combined LK and ORB methods

Figures 4.18 and 4.19 shows the combinations of FD-ORB and LK-ORB to detect both static and dynamic objects. Results show that ORB-geometry provides accurate detection of static objects, while FD performs better on dynamic objects. The combination of FD and ORB yields accurate object detection and bounding, whereas LK-ORB detects only parts of dynamic objects.

However, after a deeper analysis of the next steps, it is shown that bounding of objects does not effect the final localization of objects. Thus LK and ORB are used for displacement estimation and evaluation, with LK being the only

method suitable for 3D reconstruction, provided it yields at least four valid keypoints.

4.5 Visual Odometry

4.5.1 Scale Factor Estimation

Scale factor is one of the crucial and pre-requisite parameter when converting the measured distances or positions from pixel coordinates to their actual physical distances in the real world. The primary scale factor is depth, but in scenarios with a monocular camera, it can only be estimated from known object sizes and camera calibration.

Line detection methods and event methods are the two methods used to estimate the scale factor. The width of seat belt is used as known object size in real-world as reference. The line detection method uses edge detection to detect horizontal lines of the seat belt in the frame. Based on the detected seat belts, it estimates the factor of centimeter per pixel to convert the pixel values to real world values.

In event method the scale is obtained by measuring the pixel size of the seat belt in the image using mouse clicks method in OpenCV. Blue dots in the figure 4.20 are defined through manually clicking on the image.

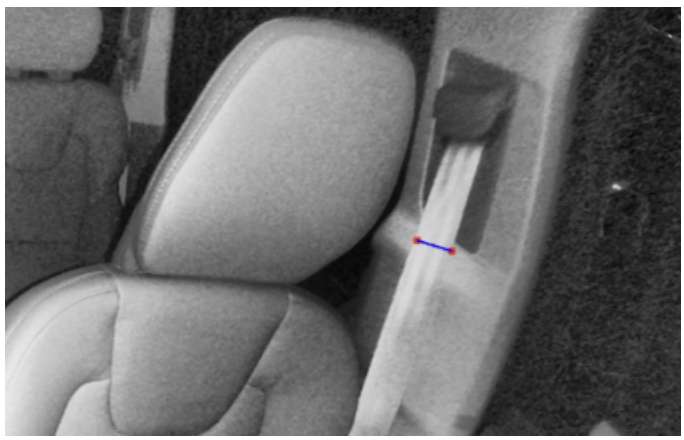

Figure 4.20: Event method - blue dots are defined through manual clicking

Table 4.4: Scale factor estimation based on seat belt measurement

Method	Object	Pixel Length (px)	Real Length (cm)	Factor(cm/px)
Line Detection	Seat Belt	137	4.8	0.0350
Event Method	Seat Belt	100	4.8	0.05

Both Event and line detecting method gave a factor between 0.035-0.05 after several attempts. Therefore the factor of 0.04 cm/pixel is selected to convert pixel displacements to real-life physical displacements.

4.5.2 Camera Motion Estimation

In the second phase of pipeline, visual odometry, the relative motion of camera is analyzed. Although the camera remains fixed, the setup can be interpreted as the reverse of a typical visual odometry scenario, where the camera moves and the scene remains static. By tracking the displacement of keypoints across frames, the equivalent camera motion can be estimated.

Therefore, the relative camera motion is estimated over five frames using ORB keypoint matching and pose recovery along the x and y axes, comparing each frame to the initial reference frame (frame 1). Thereafter, this estimated displacements are converted to centimeters using the estimated scale factor of 0.04 centimeter/pixel.

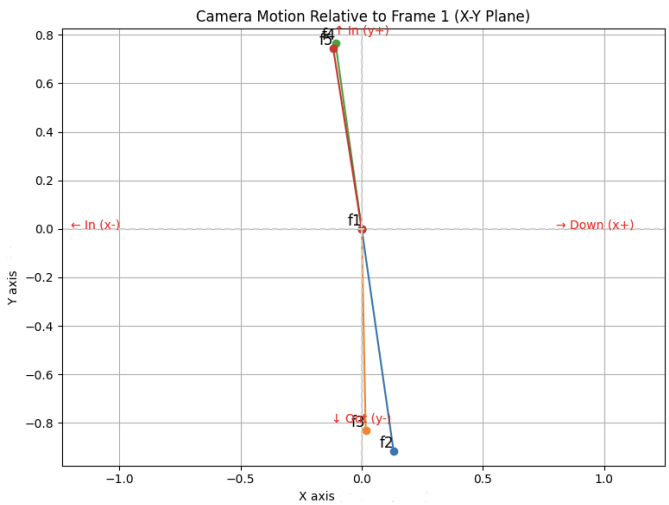

Figure 4.21: Relative camera motion from frame 1

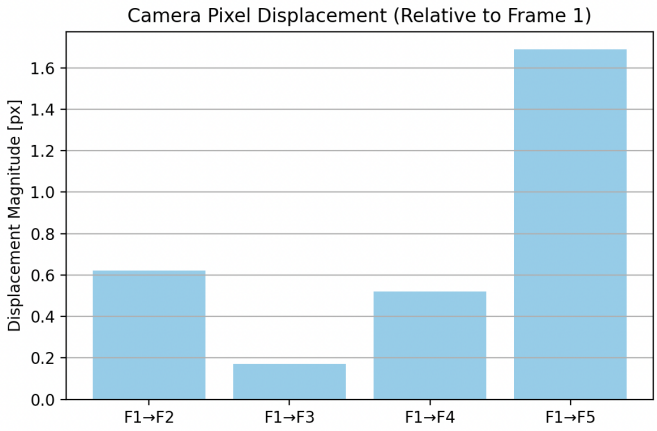

Figure 4.22: Estimated camera displacements in the pixel coordinates

From figure 4.21, it can be inferred that the relative motion of different frames compared to the reference frame aka frame 1 align with their movement direction in the test vehicle presented in table 4.1).

When analyzing the results from figure 4.22, a significantly large displacement is observed between frame 1 and frame 5. This violates the uniformity of displacement measurements, as all values are within [-1.25, 1.25] cm on the x-axis and [-2.5, 2.5] cm on the y-axis from the midpoint. In contrast, frame 1 to 3 shows the smallest displacement when steering wheel is in down-in position which indicates less information captured by the camera. In frame 5, the steering wheel moves upward and outward, allowing the camera to capture more of it. However, frame 1-2 and 1-4 shows almost equal displacements.

The results of converting the pixel displacements to cm are as following:

- Frame 1-2: 0.62 pixels = 0.025 cm
- Frame 1-3: 0.17 pixels = 0.007 cm
- Frame 1-4: 0.52 pixels = 0.021 cm
- Frame 1-5: 1.69 pixels = 0.068 cm

These values indicate relatively small displacements across frames, with the largest movement observed between frames 1 and 5.

4.6 3D Reconstruction

4.6.1 Depth Estimation

Based on the rotation (R) and translation (t) matrices deduced from Camera Motion Estimation depth is estimated using triangulation method. Triangulation provides the 3D points in camera coordinates based on keypoints detected using the ORB method.

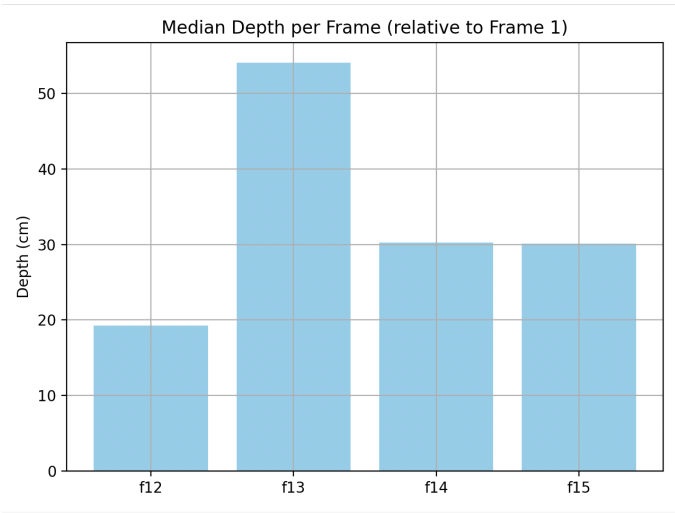

Figure 4.23: Estimated depth in cm using scale factor 0.04 cm/px

The results from depth estimation for each frame pairs are presented in the figure 4.23 as median depth relative to frame 1. The results in this figure shows that:

- Frame 1 to 2 has the lowest depth (19.28 cm) indicating that the camera sees steering wheel closest when it is down in x and y axis.
- Frame 1 to 3 has the highest depth (54.06 cm) when it is in the 'in-up' configuration in x and y axis. This can be due to less features captured by camera compared to other frames, but there can also be some outliers in this frame.
- Frame 1 to 4 and frame 1 to 4 shows almost equal values for depth, which are 30.26 cm and 30.10 cm respectively. As in these frames the steering wheel is out in the the y axis, which means it is further away from camera and thus more features captured by the same.

As there is no manual measurement of the distance between the camera and the steering wheel, an assumed distance of 50 cm at the midpoint (frame 1) can be used for validation. The values follows the telescopic motion pattern. It seems that the tilting motion of steering wheel affects the measurements to a

lesser extent than linear motion in the y axis. However, actual measurements are required to verify the results.

4.6.2 3D Reprojection into Camera Coordinates

Since each camera views the world from its own local frame, the pixel coordinates are converted into camera coordinates. To achieve this, pixel coordinate of each keypoint (u, v) and their estimated pixel depth are used to compute the 3D camera coordinates using equation 3.1.

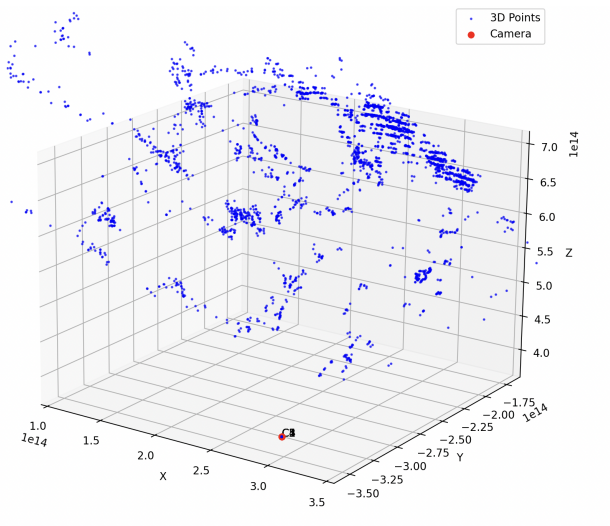

Figure 4.24: 3-D reconstruction of depicted features across frames 1-5

Multiple attempts of 3D reconstruction of steering wheel using pixel/centimeter factor failed, therefore a 3D reconstruction in the camera coordinate is proposed. Figure 4.24 shows the 3D points of detected features on steering wheel over 5 frames. The 3-D coordinate represents the camera coordinate, meaning how camera views steering wheel's motion over different frames, from its position in the frame 1.

5
Project Discussion

The developed methodology successfully demonstrates a structured pipeline for estimating the 3D position of the steering wheel using static monocular camera. The pre-processing step to correct distortions and enhance image quality is a crucial pre-step for accurate vision-based analysis.

From the evaluation of classical methods (ORB, AKAZE, SIFT, KAZE), ORB-BF is the most balanced in terms of detection ability, speed, and consistency. While AKAZE-KNN gives the best F1 score on undistorted images, its higher computational cost makes it less suitable for real-time use.
Results from the assessment of existing methods show that even though image enhancement helps with contrast and edge clarity, it has little effect on ORB's performance. Meaning, enhancement of images is not critical for feature extraction and matching as it is for bounding the steering wheel as a dynamic object. However, bounding the objects doesn't effect the 3D positioning of the objects, at least in the visual odometry phase.
Further, the enhanced ORB pipeline increases the number of detected keypoints up to 3 times and the quality of matched keypoints. The increased processing time about 5 times is expected as this pipeline provides additional information useful for camera motion estimation and 3D reconstruction.

Different object detection methods are tested, including pixel-based (FD, background subtraction) and keypoint-based methods (ORB, LK). ORB-RANSAC-DBSCAN shows good results for both static and dynamic object detection but has struggles with bounding dynamic objects. Farnebäck dense flow provides smooth motion but lacks information of keypoints for triangulation. Sparse LK optical flow combined with ORB gives good motion tracking and vector calculation, which is important for camera motion estimation and 3D reconstruction. ORB provide the keypoints position while sparse LK provides the vector motions (u,v) for pixel coordinates.

In the second phase, visual odometry, camera motion is calculated from ORB keypoints using Essential matrix. The estimated motion from frame 1 to frames

2–5 matches the relative known movement of the steering wheel in x- and y-axis. The estimated scale factor of 0.04 cm/pixel based on seat belt width, is used to estimate the camera displacement in centimeters. However, this scale factor is affected by perspective distortion, which introduces a high risk of inaccuracy and may impact both displacement and depth estimations.

In the final phase, 3D reconstruction, depth is estimated by triangulation from matched keypoints. The results match the expected steering wheel movement: lowest depth (∼19.3 cm) when the wheel is down and inside; highest depth (∼54 cm) when the wheel moves up and inside. However, these expectation is based on an assumption of 50 cm distance between the center of steering wheel and camera in the midpoint position. These results includes high risk due to scale factor.

The estimated depth and detected points using sparse LK are used to project the 2D points into camera coordinates and then into real-world coordinates. Multiple attempts of 3D reconstruction of steering wheel using 0.04 cm/px scale factor failed, therefore only a 3D reconstruction in the camera coordinate is proposed. Unfortunately, the results cannot be verified as real-world measurements, such as the steering wheel's position and its distance to the camera, are not available.

Although template matching is not the main part of the 3D estimation process, the experiments on static objects show that it can be used for internal calibration and extra tracking help. Accuracy is highest on enhanced high-resolution images (∼99%) and drops if the image is rotated more than ±5°, showing that it is sensitive to rotation. This method can support keypoint-based visual odometry by helping with scale and reliability checks.

6
Conclusion

In conclusion, this thesis contributes a motion estimation pipeline based on visual odometry for estimating camera displacement and reconstructing 3D object positions in camera coordinates using monocular image data. The findings confirm that ORB-BF matching and the Sparse Lucas-Kanade method are the most effective for providing keypoint positions and motion vectors, which are essential for camera pose estimation and 3D reconstruction from 2D image points. The proposed classical approach is lightweight, interpretable, and well-suited for low-power or cost-sensitive applications in in-cabin monitoring and safety systems.

However, monocular cameras lack depth information, which leads to complex computations for estimating object scale. External measurements such as the known size of a static object and intrinsic parameters from camera calibration, can reduce this complexity. These can be obtained through checkerboard methods to estimate both object position and camera intrinsics.

Furthermore, these findings apply to ideal static environments where objects are fully visible and occupants are excluded. In more dynamic scenarios involving occupant movement, sensor fusion with radar is suggested.

Bibliography

[1] C. Tingvall and N. Haworth, "Vision zero – an ethical approach to safety and mobility," in *6th ITE International Conference Road Safety & Traffic Enforcement: Beyond 2000*, Melbourne, Australia, 1999. [Online]. Available: `https://www.monash.edu/muarc/archive/our-publications/papers/visionzero` (cit. on p. 1).

[2] I. Guyon and A. Elisseeff, "An introduction to feature extraction," in *Feature extraction: foundations and applications*, Springer, 2006, pp. 1–25 (cit. on p. 3).

[3] J. Wu, Z. Cui, V. S. Sheng, P. Zhao, D. Su and S. Gong, "A comparative study of sift and its variants," *Measurement science review*, vol. 13, no. 3, p. 122, 2013 (cit. on p. 3).

[4] H. Bay, A. Ess, T. Tuytelaars and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008 (cit. on p. 3).

[5] D. Li, Q. Xu, W. Yu and B. Wang, "Srp-akaze: An improved accelerated kaze algorithm based on sparse random projection," *IET Computer Vision*, vol. 14, no. 4, pp. 131–137, 2020 (cit. on p. 3).

[6] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, IEEE, 2018, pp. 1–10 (cit. on p. 3).

[7] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*, Ieee, 2011, pp. 2564–2571 (cit. on p. 3).

[8] R. J. Radke, S. Andra, O. Al-Kofahi and B. Roysam, "Image change detection algorithms: A systematic survey," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 294–307, 2005 (cit. on p. 3).

[9] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1–3, pp. 185–203, 1981 (cit. on p. 3).

[10] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Scandinavian Conference on Image Analysis*, Springer, 2003, pp. 363–370 (cit. on p. 3).

[11] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981 (cit. on p. 3).

[12] P. Ochs, J. Malik and T. Brox, "Segmentation of moving objects by long term video analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 6, pp. 1187–1200, 2014 (cit. on p. 3).

[13] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proceedings CVPR*, IEEE, 1999, pp. 246–252 (cit. on p. 3).

[14] A. Goshtasby, "Template matching in rotated images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 3, pp. 338–344, 1985 (cit. on pp. 3, 33).

[15] GeeksforGeeks. "Singular value decomposition (svd)." Accessed: 2025-05-10. (2025), [Online]. Available: `https://www.geeksforgeeks.org/singular-value-decomposition-svd/` (cit. on p. 9).

[16] O. ChatGPT, *Illustration of steering wheel tilt and telescopic motion*, `https://chat.openai.com/`, Generated with ChatGPT and DALL·E, OpenAI, 2025 (cit. on p. 12).

[17] T. N. Duc, P. Ogunbona and W. Li, "Human detection based on weighted template matching," in *2009 IEEE International Conference on Multimedia and Expo*, IEEE, 2009, pp. 634–637 (cit. on p. 33).

[18] I. Pham, R. Jalovecky and M. Polasek, "Using template matching for object recognition in infrared video sequences," in *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, IEEE, 2015, pp. 8C–13 (cit. on p. 33).

7

Appendix - Template Matching for static object detection

7.1 Working of template matching and possible improvement of its execution

As discussed before, template matching is a type of pixel-base method, which scans an image to find features corresponding most closely to a predefined template. A template is a rectangular image that is given as an output to the algorithm to try and find in any image.

This method is mostly applicable for detection of static objects, but has limitation in the form of invariance to rotation, orientations and skewness changes. Some of these shortcoming can be overcome via usage of putting different wights on different parts of a template [17], using templates of different shapes other than rectangular for higher accuracy of detection in rotated images [14], or by modifying certain intrinsic parameters of its functioning in certain cases, like usage in infrared video sequences [18].

While these methods may overcome the limitations of template matching in many scenarios, they have not been explored in the research done in this thesis.

7.2 Procedure of template matching testing

This method involves predefining what objects can be classified as static objects in a vehicle. These objects had to be visible in the images, have minimal obstructions and be easily detectable in an image. A total of 6 objects were chosen as templates. These included - center console, AC vent, cup-holder, start button, parking button, and belt holder on the driver side of the vehicle. This

was done with the intention of providing redundancy, as multiple templates could be to verify the accuracy of each others measurement. There was also a case that the system may not be able detect one or more objects, thus multiple templates were also chosen to try to offset that possibility.

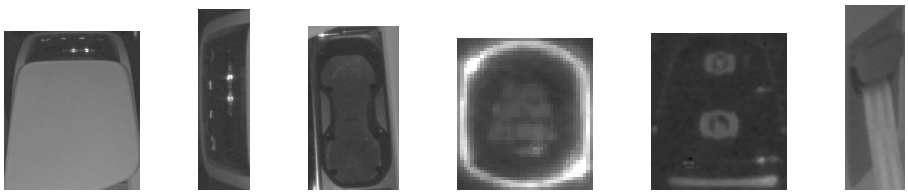

Figure 7.1: Static objects chosen for template matching

The first part of developing this method was to detect the objects from the reference image, and from there analyzing the effectiveness of detection under various scenarios in the dataset. The initial threshold of 90 percent accuracy was set in detecting the static objects, and all the static objects were detected successfully.

Next step was ensuring the positional accuracy of the static objects, since they were supposed to be used as references for detection of dynamic objects, and therefore, have to be invariant to positional changes. In order to do that, a code was written to calculate the distance different static objects, and ensure that the positional changes are minimal.

7.3 Effect of image distortion and enhancement in detection of static objects

In order to measure the accuracy of detection of templates and to compare changes in the same between images of different resolutions and distortions, three cases of images were chosen for testing - low resolution distorted image, high resolution distorted image and high resolution enhanced image (refer to table 3.1). The frame number 1 was chosen for all the 3 cases, and templates of the same 6 selected objects were selected in each of the three images. The matching accuracy of templates was defined as the minimum accuracy level in which all the templates were detected in an image. First the low resolution and the high resolution distorted images were tested, which yielded matching accuracies of 93% and 96% respectively.

Figure 7.2: Template matching comparison between low resolution distorted image (left) and high resolution distorted image (right)

After this, accuracy for the high resolution enhanced image was tested, which gave an accuracy of 99%.

Figure 7.3: Template matching in high resolution enhanced image

Despite the obvious increase in accuracy as we went from low resolution images to higher ones, the change itself was not much, meaning that the method could itself be used in images with lower resolution or distortion. However, the combining this with other methods, which maybe dependent on images of higher resolution should also be considered.

7.4 Rotation variance testing

One of the main assumptions in the thesis was the use of a perfectly calibrated and mounted camera in the vehicle. However, mounting accuracies in order of a degrees can often happen in a vehicle, and in order to test that, the matching accuracies of the templates under various conditions of rotation were observed.

For this part, the high resolution enhanced image was chosen, and since the camera was mounted perfectly, the images were artificially rotated to the angles of 5 degrees and 10 degrees, both clockwise (mentioned as 5 degrees and 10 degrees subsequently) and counter-clockwise (mentioned as -5 degrees and -10 degrees subsequently). These values were chosen to represent the most extreme

edges of mounting errors possible, however in normal situations the mounting errors mostly lie within 1 or 2 degrees on either side.

For the case of 5 and -5 degrees, the The accuracy came out at 62% and 65% respectively.

Figure 7.4: Template matching comparison between 5 degree tilt (left) and -5 degree tilt)

The case of 10 and -10 degrees had few major differences from the previous ones. Namely the detection of all the templates became much more harder in these cases, so for the case of 10 degree rotation, only 3 templates were detected properly out of 6 with an accuracy of 62%. Similarly for -10 degrees, only 4 templates were detected with an accuracy of 63%.

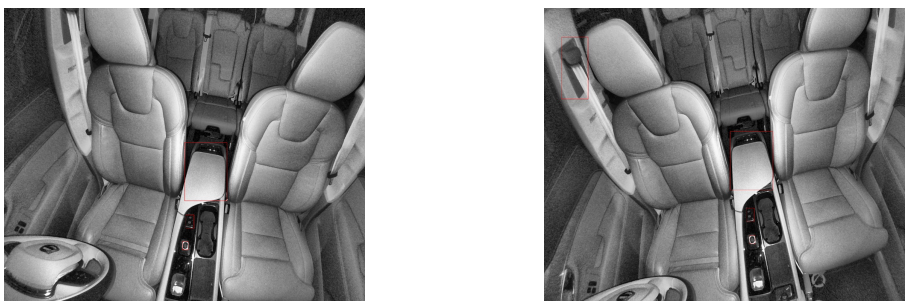

Figure 7.5: Template matching comparison between 10 degree tilt (left) and -10 degree tilt

An issue with this approach was the inability to deduce why the difference in accuracy was large between clockwise and counter-clockwise movement.Also the mentioned cases could differ from real life scenario as the rotation of images was done artificially, so it might not be representative of the true situation. Lastly, while these cases represent the most extreme scenarios, they still highlight the major limitation of this method, namely invariance to changes in rotation.