SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Robust State Estimation via Covariance Estimation for ICP-based Scan Registration

Berkay Gümüş

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Robust State Estimation via Covariance Estimation for ICP-based Scan Registration

Robuste Zustandsschätzung mittels Kovarianzschätzung für ICP-basierte Scan-Registrierung

Author:	Berkay Gümüş
Examiner:	Prof. Dr.-Ing. Markus Lienkamp
Supervisor:	Dominik Kulmer, M.Sc. Maximilian Leitenstern, M.Sc.
Submission Date:	08.10.2024

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Munich, 08.10.2024 Berkay Gümüş

Abstract

Accurate localization is a key challenge in the development of high-speed autonomous vehicles, especially in environments where GNSS signals may be degraded or unavailable. In such conditions, multi-modal localization systems are crucial to maintain consistent performance. Iterative Closest Point (ICP) localization, using Light Detection and Ranging (LiDAR) sensors, is commonly employed as an additional localization source. However, this introduces the problem of uncertainty estimation, as reliable uncertainty values are essential for robust multi-modal localization.

This thesis investigates covariance estimation for ICP. It begins by explaining the sources of uncertainty in ICP and reviewing existing methods in the literature. New methods for ICP covariance estimation are then proposed. Finally, both the existing and proposed methods are evaluated using real-world data from high-speed autonomous vehicles, focusing on uncertainty estimation accuracy and their impact on the overall multi-modal localization system.

Contents

1. Introduction and Motivation

1.1. Motivation

For several decades, autonomous driving has been a major technological trend, drawing significant attention and investment from industries and academic institutions worldwide [1] [2]. It promises not only more efficient and comfortable transportation but also enhanced road safety [3]. While autonomous vehicles are currently operating in certain areas, such as San Francisco, Phoenix, Wuhan, and Beijing, their autonomy level is still limited. Despite 60% of today's vehicles having driver assistance features, only 1-2% of global vehicle sales in 2026 are projected to include Level 3 automation, according to Goldman Sachs [4]. Furthermore, a 2023 survey by McKinsey forecasts that Level 4 autonomous vehicles in urban areas may be on the roads in approximately 2029 due to technical challenges, safety concerns, and evolving regulatory frameworks [5].

1.2. Problem Statement

To address real-world challenges and improve safety features at high speeds by pushing the boundaries of autonomous driving, autonomous racing competitions are organized by the Indy Autonomous Challenge (IAC) [6] (Figure 1.1) and the Abu Dhabi Autonomous Racing League (A2RL) [7] [8]. The TUM Autonomous Motorsport team has participated in these races since 2022 [9] [10]. In April 2024, the first multi-vehicle race took place at Abu Dhabi's Yas Marina Circuit, whereas other competitions focus on two-car overtaking scenarios and the fastest lap times. The team aims to develop different software stacks of autonomous race car, such as trajectory planning for multi-vehicle scenarios, environment perception, and reliable localization at high speeds [11].

Figure 1.1.: Driverless race cars of the Indy Autonomous Challenge

One of the critical challenges in autonomous driving is achieving precise and reliable vehicle pose estimation, which becomes increasingly critical at high speeds. The performance of the vehicle's controller and trajectory planning is heavily dependent on the accuracy of localization. Global Navigation Satellite Systems (GNSS) provide accurate real-time localization; however, their performance depends on signal quality from satellites. In environments with tall buildings, bridges, or tunnels - as seen in Figure 1.2 - GNSS accuracy diminishes, and uncertainty increases. In such scenarios, additional localization sources must be employed to ensure consistent performance.

Figure 1.2.: Abu Dhabi Yas Marina Circuit

Apart from the GNSS, the race cars are equipped with Light Detection and Ranging (LiDAR), Inertial Measurement Unit (IMU), cameras, and wheel odometry sensors. In the team, Sauerbeck et al. [12] [13] developed a LiDAR localization system based on scan registration, and Wischnewski et al. [14] and Goblirsch et al. [15] implemented state estimation systems for multi-model localization. In such systems, the uncertainty associated with each localization source is critical, as the contribution of each source to the final pose estimation is directly influenced by its level of uncertainty. However,

uncertainty estimation for scan registration remains an open problem.

1.3. Task Definition

In the LiDAR-based localization system of the race car, the Iterative Closest Point (ICP) algorithm is used for scan registration, with KISS-ICP [16] repository adapted for this purpose. The main goal of this thesis is to develop a more robust state estimation through reliable ICP pose covariance estimation (Figure 1.3). Currently, a nearly constant ICP covariance is used, which becomes unreliable when incorrect ICP pose estimates occur. As a result, the ICP pose is only utilized when accurate GNSS pose data is unavailable.

First, existing ICP pose covariance estimation methods are evaluated using real-world data. Second, the sources of ICP uncertainty are investigated, and new approaches are proposed. Finally, these methods are assessed in terms of both ICP covariance estimation and their effect on state estimation.

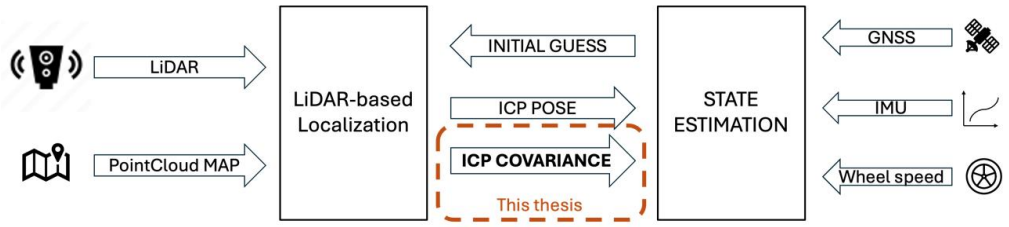


Figure 1.3.: State estimation diagram for localization

1.4. Structure of the Thesis

Chapter 2 provides an brief explanation of the ICP algorithm, detailing the sources of ICP uncertainty and reviewing existing covariance estimation methods. Additionally, some preliminary results are presented. Chapter 3 introduces the multi-model localization system, and a description of the dataset from the Yas Marina Circuit. The chapter further explores the sources of uncertainty within this dataset. Chapter 4 proposes new covariance estimation methods. Chapter 5 presents both quantitative and qualitative results for each of the proposed methods for a comprehensive evaluation. Chapter 6 summarizes and discusses the methods and results briefly.

2. State of the Art

In this chapter, firstly, the ICP algorithm and the sources of uncertainty in this algorithm are described. Then, the methods used to calculate these uncertainties are explained.

2.1. Iterative Closest Point

The Iterative Closest Point (ICP) is a widely-used point cloud registration algorithm designed to align two sets of points by iteratively minimizing the difference between corresponding points [17]–[20]. In Figure 2.1, points on the red and the blue lines are aligned iteratively.

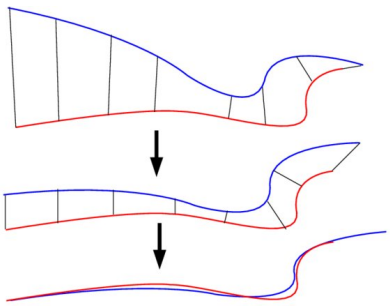


Figure 2.1.: Registration with ICP [21]

In mobile robotics, ICP is widely employed in tasks such as 3D reconstruction and Simultaneous Localization and Mapping (SLAM) where multiple scans are registered [22], [23]. Another usage area is localization, where the algorithm aligns a scan with a known 3D map to estimate the robot's position [16], [24]–[26].

The ICP algorithm has the following essential steps [17]:

1. **Input:**
    - Source point cloud S with points s_i and reference point cloud M with points m_i

- Initial guess of the alignment $T_0 \in \mathrm{SE}(3)$
- Function to find closest points between the point clouds
- Residual function $r(s_i, m_i)$ (e.g., point-to-point or point-to-plane distance)
- Criteria for stopping and convergence, e.g. maximum iterations or threshold error

2. **Initialization:**
   - Final transformation $T \in SE(3)$, initialized as $T = T_0$.
3. **Iterative Process (until convergence or stopping criteria):**
   - **Point Association:** For each point in S, find the closest point in M and generate a set of corresponding point pairs between the two point clouds: S_c and M_c.
   - **Transformation Estimation:** Estimate the rigid transformation $T_j \in SE(3)$ that minimizes the residuals between the corresponding points S_c and M_c for iteration j. The optimization problem can be written as:

$$T_j = \arg\min_T \sum_{i=1} r_i^{\mathrm{T}} r_i \quad \text{where} \quad r_i = r(Ts_i, m_i) \tag{2.1}$$

     where $s_i \in S_c$ and $m_i \in M_c$ are paired points, and $r(Ts_i, m_i) \in \mathbb{R}^m$ is the residual between the transformed source point and the reference point.
   - **Update:** Apply the estimated transformation T_j to align S with M, and update the transformation T accordingly.

$$S \leftarrow T_j \cdot S \tag{2.2}$$

$$T \leftarrow T_j \cdot T \tag{2.3}$$

4. **Output:**
   - Return final transformation $T \in SE(3)$.

Although the fundamental steps of the ICP algorithm remain the same, there are numerous variations in its implementation, utilizing different methods for point association, transformation estimation and additional weighting and filtering features [27] [28], [22].

2.1.1. Data Association

The most critical step of the ICP algorithm is data association, where corresponding point pairs between the source and reference point clouds are established [29]. In this process, each point in the source is matched to its nearest neighbor in the reference, visualized in Figure 2.2. However, it is not guaranteed that these pairs are always correct [30], and this step is also the most computationally expensive part of the algorithm [31]. To address this, several efficient data structures have been developed for fast nearest-neighbor search, including KD-Trees [32], Octrees [33], [34], and voxel grids with hash tables [16], [35],[36].

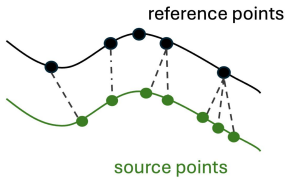


Figure 2.2.: Data association

2.1.2. Transformation Estimation

After establishing the point pairs, a 3D rigid body transformation is estimated based on these pairs. The transformation estimation is formulated as a least squares optimization problem, where the objective is to minimize the sum of squared residuals between corresponding point pairs [37] [38].

To rewrite the Equation 2.1 explicitly, $X \in \mathbb{R}^6$ (with X in capital to avoid confusion with the position variable x) is defined to represent the transformation parameters for 3D translation and 3D rotation, denoted as $x, y, z, \phi, \theta, \psi$, respectively. Throughout this report, X will be used to refer to the transformation parameters.

$$\min_X E(X, S_c, M_c) = \sum_{i=1} r(X, s_i, m_i)^{\mathrm{T}} r(X, s_i, m_i) \tag{2.4}$$

The Jacobian matrix J_i of the residual function is defined as:

$$\mathrm{J}_i = \frac{\partial r(X, s_i, m_i)}{\partial X} \tag{2.5}$$

Thus, the normal equations can be expressed as:

$$\mathrm{J}_i \Delta X = -r_i \tag{2.6}$$

and

$$\left(\sum_{i=1} \mathrm{J}_i^{\mathrm{T}} \mathrm{J}_i\right) \Delta X = -\left(\sum_{i=1} \mathrm{J}_i^{\mathrm{T}} r_i\right) \tag{2.7}$$

Then, the optimal solution is given by:

$$\hat{X} = -\left(\sum_{i=1} \mathrm{J}_i^{\mathrm{T}} \mathrm{J}_i\right)^{-1} \left(\sum_{i=1} \mathrm{J}_i^{\mathrm{T}} r_i\right) \tag{2.8}$$

Furthermore, a weighting kernel w_i can be applied to reduce the influence of outlier pairs. This results in the following formulation:

$$\hat{X} = -\left(\sum_{i=1} w_i \mathrm{J}_i^{\mathrm{T}} \mathrm{J}_i\right)^{-1} \left(\sum_{i=1} w_i \mathrm{J}_i^{\mathrm{T}} r_i\right) \tag{2.9}$$

Two common residual functions in ICP implementations are the point-to-point and point-to-plane error metrics, as shown in Figure 2.3.

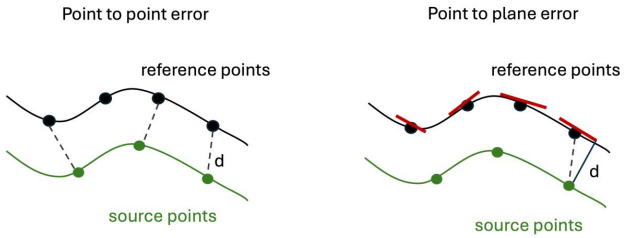


Figure 2.3.: Error metrics

Point-to-Point Error

The point-to-point error is defined as the Euclidean distance between two corresponding points [17], and the residual function is [16]:

$$r_i(\mathrm{R}, \mathrm{t}, s_i, m_i) = \mathrm{R}s_i + \mathrm{t} - \mathrm{m}_i \in \mathrm{R}^3 \tag{2.10}$$

Thus, the squared error function becomes:

$$E(\mathrm{R}, \mathrm{t}, S_c, M_c) = \sum_{i=1} \|\mathrm{R}s_i + \mathrm{t} - \mathrm{m}_i\|_2^2 \in \mathrm{R} \tag{2.11}$$

where:

- $R \in SO(3)$ is the rotation matrix.
- $t \in R^3$ is the translation vector.

Since the residual is in $\mathbb{R}^3$, the Jacobian matrix is in $\mathbb{R}^{3x6}$ [16]:

$$J_{3x6} = \begin{bmatrix} \frac{\partial r_i}{\partial x,y,z} & \frac{\partial r_i}{\partial a,b,c} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & s_{iz} & -s_{iy} \\ 0 & 1 & 0 & -s_{iz} & 0 & s_{ix} \\ 0 & 0 & 1 & s_{iy} & -s_{ix} & 0 \end{bmatrix} = \begin{bmatrix} I_{3x3} & -\widehat{(s_i)} \end{bmatrix}_{3x6} \tag{2.12}$$

where $\widehat{(.)}$ represents the hat operator, which denotes the cross-product operation.

Then, the optimal transformation can be calculated using Equations 2.8 or 2.9.

Point-to-Plane Error

The point-to-plane error is defined as the perpendicular distance between source point and the tangent plane of the reference point [39], and the residual function is [40]:

$$r_i(R, t, s_i, m_i, n_i) = (Rs_i + t - m_i)^T n_i \in R \tag{2.13}$$

The squared error function for point-to-plane becomes:

$$E(R, t, S_c, M_c) = \sum_{i=1} \left((Rs_i + t - m_i)^T n_i \right)^2 \in R \tag{2.14}$$

where $n_i \in \mathbb{R}^3$ is the normal vector of the reference point in the i-th pair.

Since the residual is a scalar, the Jacobian matrix is a row vector, $\mathbb{R}^{1x6}$. It can be derived as [40]:

$$J_{1x6} = \begin{bmatrix} \frac{\partial r_i}{\partial x,y,z} & \frac{\partial r_i}{\partial a,b,c} \end{bmatrix} = \begin{bmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \\ -s_{iz} * n_{iy} + s_{iy} * n_{iz} \\ s_{iz} * n_{ix} - s_{ix} * n_{iz} \\ -s_{iy} * n_{ix} + s_{ix} * n_{iy} \end{bmatrix}^T = \begin{bmatrix} n_i \\ s_i \times n_i \end{bmatrix}^T \tag{2.15}$$

Then, the optimal transformation can be calculated using Equations 2.8 or 2.9.

KISS-ICP

There are various open-source ICP implementations commonly used with different variations, MeshLab [41], libpointmatcher [42], simpleICP [43], PCL [44].

Vizzo et al. [16] proposed an efficient point-to-point ICP implementation called KISS-ICP (Keep It Small and Simple). The method focuses on the efficiency of ICP's core components, avoiding the complexity of ego-motion estimation through external sensors or sophisticated motion models.

In this approach, a constant motion model is employed for the simplicity, making it applicable to any system. Translational and angular velocities, v_t and w_t, are calculated using the last two poses T_{t-2} and T_{t-1}. However, the author notes that this simple motion model can be substituted with a more complex one, while the rest of the model remains directly usable.

Later, because the points in a point cloud P are recorded at different times during a single LiDAR sweep within the acquisition time Δt, they are deskewed using their relative timestamps $s_i \in [0, \Delta t]$ and the vehicle's velocity, resulting in a new deskewed point cloud P^*.

$$p_i^* = \text{Exp}(s_i \omega_t) p_i + s_i v_t \tag{2.16}$$

where $\text{Exp} : \mathbb{R}^3 \rightarrow SO(3)$ computes a rotation matrix from an axis-angle representation

The next step is point cloud subsampling. The deskewed scan P^* is spatially downsampled to $\hat{P}^*$ using 3D voxel grids. Additionally, the local map, which is created and updated after point cloud registrations, is represented by a voxel grid with cells of size v x v x v, each containing a maximum number of points, N_{max}.

For the scan, a smaller voxel size av where $a \in (0.0, 1.0]$ is chosen, and only a single representative point is retained per voxel, resulting in P^*_{merge}. The author suggests that an even lower resolution scan is beneficial for faster and more efficient registration. Thus, a new subsampled scan, $\hat{P}^*$, is created using a voxel size βv where $\beta \in [1.0, 2.0]$, retaining only a single representative point per voxel.

Later, the author explains that the maximum allowable distance between pairs for the data association process depends on the error in the initial pose guess which is

influenced primarily by the motion model and, to a lesser extent, by sensor noise. The deviation σ_t in the motion model is estimated, and an adaptive threshold, τ_t, is set as $3\sigma_t$.

In the final step, the pose estimation is refined iteratively, similar to the iterative process described in Step 3 in Section 2.1 with Equation 2.4. The initial guess T_0 is provided by the motion model, the source point cloud S is the subsampled scan $\hat{P}^*$ and the reference point cloud M is the closest map points found using a nearest neighbor search within the voxel grid map.

In Equation 2.4, instead of using the point-to-point residual directly (Equation 2.11), it is weighted by a kernel for a robust optimization. The weighted squared error function is defined as:

$$E(\mathrm{R}, \mathrm{t}, S_c, M_c) = \sum_{i=1} \rho \left(\|\mathrm{R}\mathrm{s}_i + \mathrm{t} - \mathrm{m}_i\|_2^2 \right) \in \mathrm{R} \tag{2.17}$$

where ρ is a robust kernel:

$$\rho(e) = \frac{e^2/2}{\sigma_t/3 + e^2} \tag{2.18}$$

2.2. ICP Uncertainty Sources

In the novel paper [45], Censi introduced three main sources of ICP uncertainty: wrong convergence, under-constrained situations, and sensor noise. Later, Brossard et al. [46] expanded on these by adding two additional sources: sensor bias and intrinsic ICP randomness.

2.2.1. Wrong Convergence

This category is also called "initial transformation" in [46] and "convergence and initialization" in [47]. ICP is a powerful and efficient method, but the final pose result mostly depends on the initial guess [46]. Because it works iteratively and, in each iteration, the closest points are found in another point cloud set (data association), it is assumed that these pairs are correct, and the transformation is calculated using these pairs. Due to this heuristic strategy, if the initial guess is not close enough, the data association will be wrong and the final pose result will be dramatically inaccurate [48]. Mathematically, ICP is an optimization technique, but it does not guarantee convergence to the global minimum. It can get stuck in a wrong local minimum. Therefore, it is crucial to have an initial guess within the attraction basin of the true solution visualized in Figure 2.4

[46].

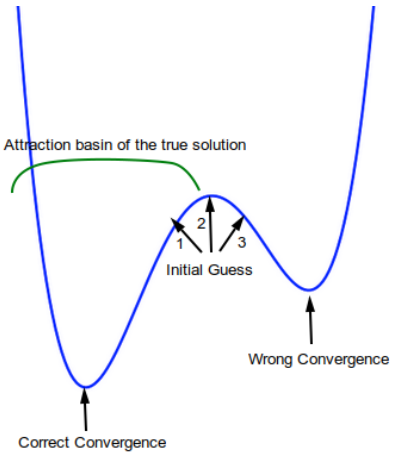


Figure 2.4.: Local minima problem in 1D case

Furthermore, if a wrong initialization/convergence exists, it is the dominant source of the ICP error in practice [46]. Censi notes that this type of error is very difficult to model [45].

2.2.2. Under-Constrained Situations

In certain environments, there may not be enough information to estimate the transformation between point clouds. In such cases, even if ICP works properly and the point clouds are noise-free, the result may be inaccurate due to the lack of information. This is referred to as epistemic uncertainty [47] [49].

A common example of an under-constrained environment is a long, featureless corridor. In these cases, there is ambiguity in the position along the corridor, whereas lateral position and rotation can be estimated accurately. Landry et al [50] sampled the ICP results in an under-constrained environment in Figure 2.5 by registering source points (red) to reference points (blue). The black regions within the green area depict the density of the samples. Due to the presence of regular pillars, the ICP poses predominantly converge to three local minima, indicated by these black regions.

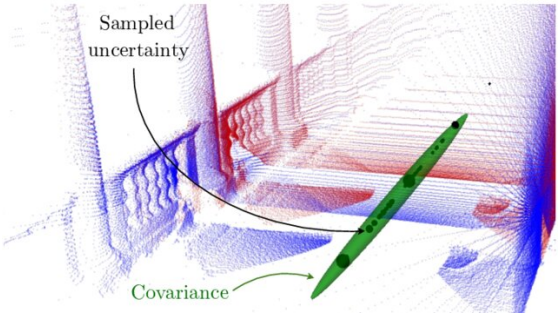


Figure 2.5.: Under-constrained environment

2.2.3. Sensor Noise and Bias

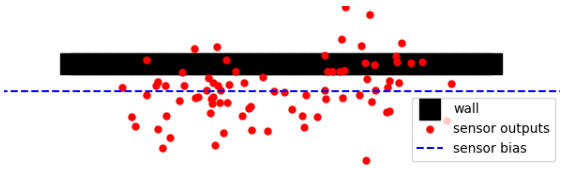


Figure 2.6.: Sensor noise and bias

Sensor noise is a random deviation in the signal values. An offset between correct values and signal values is referred to as sensor bias. Even if the convergence is within the attraction basin of the true solution and the point clouds are fully constrained, sensor noise inherently introduces errors. Due to the nature of the sensors and various environmental conditions, these errors persist and affect the accuracy of the ICP results in practice [47]. Pomerleau et al. [51] listed factors such as internal temperature, target distance, incidence angle, brightness/reflectivity, heat level, ambient light/sunlight, air conditions (e.g dust, rain, smoke), and observed material in their work on noise characterization of 2D depth sensors for 3D applications like ICP. Wang et al. [52] investigated the stability of low cost LiDAR sensors for 3D perception and showed that internal temperature increases over time and it causes drift effect and roll effect. Furthermore, Laconte et al. [53] demonstrated that high incidence angles lead to significant sensor bias. Another critical factor is sensor calibration, wrong calibration can introduce distorted point clouds and a bias in the transformation estimate [54].

2.2.4. Intrinsic ICP randomness

Although ICP is a deterministic method, Martin Brossard [46] claims that the ICP configuration, random filtering processes, and implementations may introduce randomness to the transformation estimation. For example, after outlier rejection and sub-sampling for the same point cloud sets, ICP results may be different [47].

2.3. ICP Covariance Calculation Methods

Various approaches have been proposed to calculate uncertainty in pose estimation using ICP. These methods are typically categorized as pessimistic, optimistic, or accurate after being evaluated on a range of datasets, including 2D, 3D, real-world, and synthetic data, across different applications such as scan matching and localization. An approach is deemed pessimistic when the estimated uncertainty exceeds the true error, and optimistic when it falls below the true error. If the estimated uncertainty closely matches the true error, the result is considered accurate in the literature.

These methods are basically classified as

- Monte-Carlo (also called brute force) Algorithms
- Closed-Form Methods
- Data-Driven Methods

Monte-Carlo algorithms [55], [56], also known as brute force algorithms, perform multiple ICP registrations using noisy scans and varying initial guesses, and the covariance is derived based on the resulting ICP poses. On the other hand, Closed-Form methods [45] [57] analyze the linearization of the objective function around the convergence point. However, these methods assume that the ICP result converges within the attraction basin of the true solution, and do not account for the uncertainty arising from the wrong convergence [45], [46]. Finally, Data-Driven methods [50] [58] [47] utilize learning-based approaches trained on data and deep-learning methods. Each of these methods has its own advantages and drawbacks in terms of accuracy, execution time or data preparation [46].

2.3.1. Least-Squares Uncertainty

The first closed form solution is introduced by Lu [59]. The idea is that ICP is a least squares problem and least squares uncertainty formula can be used for the ICP

uncertainty. Uncertainty formula for the optimal estimate in Equation 2.8 [60, Eq. 4.10], [56]:

$$\text{cov}(\hat{X}) = \left(\sum_{i=1} \mathrm{J}_i^{\mathrm{T}} \mathrm{J}_i \right)^{-1} \sigma^2 \tag{2.19}$$

where:

$$\sigma^2 = \frac{E(\hat{X})}{n-k}. \tag{2.20}$$

- $E(\hat{X})$ is the squared error function defined in Equation 2.4
- n is the number of pairs.
- k is the number of parameters to be estimated (3 for 2D cases, 6 for 3D cases).

The main criticism of this method is that it assumes pairs in the actual scan and reference scan are found correctly, which is often not the case in practice, especially in scenarios such as long corridors [56]. This method does not account for the uncertainty arising from under-constrained environments. For example, for two different parts of a corridor, their alignment may be perfect, and the estimated uncertainty may be zero. This method evaluates how well the scans overlap but does not estimate the actual ICP transformation uncertainty, leading to overly optimistic results [56].

Another drawback is that this method cannot capture the shape of the covariance matrix, resulting in a nearly constant covariance across different environments [56]. The reason of the constant covariance is that the error metric used by Lu [59] is point-to-point and the first part of the Jacobian is the identity matrix, shown in Equation 2.12.

Additionally, Bengtsson [56] presented test results in Table 2.1 for 2D cases across various environments - corridor, circular, square and irregular (constrained) environments. The results indicate that the uncertainty results are highly optimistic and the standard deviations for the x and y directions are nearly identical in almost all environments [56].

2.3.2. Hessian Method

Bengtsson et al. [56] [61] proposed a new method to overcome the problem of Lu's method above. Their idea is to investigate the actual error function by estimating its Hessian matrix.

Hessian matrix of the error function in 2.4 [56]:

$$H = \frac{dE^2(X)}{dX^2} = 2\sum_{i=1} \mathrm{J}_i^{\mathrm{T}} \mathrm{J}_i \implies \sum_{i=1} \mathrm{J}_i^{\mathrm{T}} \mathrm{J}_i = \frac{1}{2} H \tag{2.21}$$

The covariance formula in Equation 2.19 becomes:

$$\mathrm{cov}(\hat{X}) = \left(\frac{1}{2} H\right)^{-1} \sigma^2 \tag{2.22}$$

The Hessian matrix is computed numerically by calculating the alignment errors for slightly different poses in each single direction and in each combined direction. In 2D cases, where $X = \left[x, y, \theta\right]^{\mathrm{T}}$ is the transformation parameters, $\left[x, y\right]^{\mathrm{T}}$ is the position, and θ is the rotation, the Hessian matrix is defined as:

$$H_{3x3} = \begin{bmatrix} \frac{\partial^2 E}{\partial x^2} & \frac{\partial^2 E}{\partial x \partial y} & \frac{\partial^2 E}{\partial x \partial \theta} \\ \frac{\partial^2 E}{\partial y \partial x} & \frac{\partial^2 E}{\partial y^2} & \frac{\partial^2 E}{\partial y \partial \theta} \\ \frac{\partial^2 E}{\partial \theta \partial x} & \frac{\partial^2 E}{\partial \theta \partial y} & \frac{\partial^2 E}{\partial \theta^2} \end{bmatrix} \tag{2.23}$$

After the optimal transformation $\hat{X} = \left[\hat{x}, \hat{y}, \hat{\theta}\right]^{\mathrm{T}}$ is estimated, the algorithm works as follow [61]:

1. Translate/Rotate the source scan by $\hat{X} + (\Delta x, \Delta y, \Delta\theta)$
2. Find the corresponding pairs (new data association) and calculate the new error $E(\hat{X} + (\Delta x, \Delta y, \Delta\theta))$ using Equation 2.11
3. Repeat step 1 and 2 until all elements of Hessian matrix are calculated.

One example is provided for combined elements x-y [61]:

$$\begin{aligned} \frac{\partial^2 E}{\partial x \partial y} &\approx \frac{\partial}{\partial y}\left(\frac{E(\hat{X} + \Delta x) - E(\hat{X} - \Delta x)}{2\Delta x}\right) \\ &\approx \frac{E(\hat{X} + \Delta x + \Delta y) - E(\hat{X} - \Delta x + \Delta y)}{4\Delta x \Delta y} - \frac{E(\hat{X} + \Delta x - \Delta y) - E(\hat{X} - \Delta x - \Delta y)}{4\Delta x \Delta y} \\ &\approx \frac{E(\hat{X} + \Delta x + \Delta y) - E(\hat{X} - \Delta x + \Delta y) - E(\hat{X} + \Delta x - \Delta y) + E(\hat{X} - \Delta x - \Delta y)}{4\Delta x \Delta y} \end{aligned} \tag{2.24}$$

Step 1 and 2 are repeated 4 times for each combined elements (e.g. x and y in the example above), and 3 times for single elements [61]. For all elements in the Hessian

matrix, this requires 21 data association and error calculation processes for different alignments

Bengtsson [56] conducted the same tests mentioned in Subsection 2.3.1 for this method and provided quantitative results in Table 2.1. In contrast to the Least-Squares uncertainty method, which fails to capture the shape of the covariance matrix, the Hessian method effectively represents this structure [56]. For instance, in the 2D corridor scenario along the y-axis in Figure 2.7, the standard deviation in the y-axis is significantly higher than in the x-axis, reflecting the actual geometric constraints. However, the results tend to be somewhat pessimistic [56].

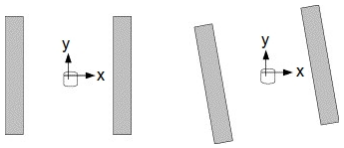


Figure 2.7.: 2D corridor scenarios for two different poses by Bengtsson [56]

Table 2.1.: Uncertainty estimation results for corridor environments by Bengtsson [56]

Corridor environments	σ_x (mm)	σ_y (mm)	σ_θ (°)
Figure 2.7, left side			
Least Squares method	1.1	1.1	0.006
Hessian method	12.9	428.6	0.109
Offline method	2.4	309.2	0.011
True error	2.3	338.2	0.005
Figure 2.7, right side			
Least Squares method	1.2	1.2	0.007
Hessian method	87.8	492.5	0.113
Offline method	62.3	351.5	0.012
True error	61.2	347.2	0.004

Furthermore, although this method involves multiple point rematchings and error calculations, Bengtsson claims that it can be used in real-time systems. However, his tests are conducted in 2D, and it may not be suitable for the 3D real-world applications [56].

2.3.3. Censi Method

Censi [45] critiqued the Hessian method, noting that it assumes the error function is quadratic and estimates the curvature of the error function by sampling it around the optimal transformation vector $\hat{X}$. However, the ICP problem involves not only solving a least squares problem for the transformation parameters but also a data association process, where the error function depends on the point pairs themselves. This data association process is influenced by the measurements. Censi [45] argued that sensor noise alters the measurements, thereby affecting the curvature of the error function, and sensor noise must be taken into account in the covariance calculation.

Censi [45] introduced a novel method considering the sensor noise and ICP process itself. He employed the implicit function theorem to map sensor noise to final pose uncertainty.

Firstly, similar to Equation 2.1, he assumed there is an optimization algorithm A to represent ICP, depending on measurements $\check{Z}$ (corresponding source and reference point pairs S_c, M_c) and optimizing transformation parameters X.

$$\hat{X} = A(\check{Z}) = \arg\min_{X} E(\check{Z}, X). \tag{2.25}$$

Then, the idea is that the sensor noise can be mapped to transformation (ICP) uncertainty using the first order approximation of algorithm A.

$$\mathrm{cov}(\hat{X}) \simeq \frac{\partial A}{\partial Z}\mathrm{cov}(Z)\frac{\partial A^\top}{\partial Z} = \frac{\partial X}{\partial Z}\mathrm{cov}(Z)\frac{\partial X^\top}{\partial Z} \tag{2.26}$$

However, A is not a closed-form function and not differentiable easily [45]. To solve this problem, the author claimed that $A(Z)$ and Z are bound by an implicit function. Moreover, $\hat{X}$ is optimal pose estimated by ICP, and consequently, the necessary condition that the gradient is null at $\hat{X} : \partial E(\check{Z}, \hat{X})/\partial X = \mathbf{0}^\top$ for the implicit function is hold.

Implicit function theorem [62] states that if $f(X, Z) = 0$ and $g(Z) = X$ are continuously differentiable functions on a manifold and $\partial f(X, Z)/\partial X$ is invertible, then

$$\frac{\partial X}{\partial Z} = -\frac{\frac{\partial f(X,Z)}{\partial Z}}{\frac{\partial f(X,Z)}{\partial X}} = -\left[\frac{\partial f(X,Z)}{\partial X}\right]^{-1}\left[\frac{\partial f(X,Z)}{\partial Z}\right] \tag{2.27}$$

For the ICP case,

$$\text{let } f(Z, X) \text{ be } \frac{\partial E(Z,X)}{\partial X} = 0 \quad \text{by the optimization} \tag{2.28}$$

then, equation 2.27 becomes

$$\frac{\partial X}{\partial Z} = -\left[\frac{\partial}{\partial X}\frac{\partial E(Z,X)}{\partial X}\right]^{-1}\left[\frac{\partial}{\partial Z}\frac{\partial E(Z,X)}{\partial X}\right] = -\left[\frac{\partial E^2(Z,X)}{\partial X^2}\right]^{-1}\left[\frac{\partial E^2(Z,X)}{\partial Z \partial X}\right] \tag{2.29}$$

and, equation 2.26 becomes

$$\mathrm{cov}(\hat{X}) \simeq \left(\frac{\partial^2 E}{\partial X^2}\right)^{-1} \frac{\partial^2 E}{\partial Z \partial X}\,\mathrm{cov}(Z)\,\frac{\partial^2 E}{\partial Z \partial X}^{\top}\left(\frac{\partial^2 E}{\partial X^2}\right)^{-1} \tag{2.30}$$

After the ICP error function is explicitly written with each measurement and each parameter, the derivatives in Equation 2.30 for the estimated $\hat{X}$ can be calculated. The derivatives for point-to-plane error are provided in the appendix.

Due to the inverse operation in Equation 2.30, $\partial^2 E/\partial X^2$ must be non-singular. Censi [45] stated that the derivative matrix may be singular only if the environment is under-constrained. For example, in a perfect corridor, the walls are parallel to each other, creating an under-constrained scenario. He [45] also noted that it is not possible to get a singular $\partial^2 E/\partial X^2$ matrix in real-world applications because of the sensor noise, but in that case, ICP fits to the noise. His solution [45] is to use Fisher information matrix $I(S_c, M_c)$ [63].

$$I(S_c, M_c) = \sum_{i=1} \mathrm{J}_i^{\mathrm{T}} \mathrm{J}_i \tag{2.31}$$

Firstly, under-constrained directions are identified using the kernel (null space) of the Fisher information matrix. If there is at least one eigenvalue close to zero, the corresponding eigenvector p_u defines the unobservable (under-constrained) direction while other eigenvectors p_o define observable directions.

$$\text{Unobservable manifold:} \quad \mathcal{U} = \mathrm{span}\{p_{u1}, ...\} \tag{2.32}$$

$$\text{Observable manifold:} \quad \mathcal{O} = \mathrm{span}\{p_{o1}, p_{o2}, ...\} \tag{2.33}$$

Subsequently, the final ICP uncertainty is transformed into observable manifold, shown in Figure 2.8 [45]. On the left side, the corridor direction is unobservable, while on the right side, the rotation is unobservable.[45]

$$\mathrm{cov}_{obs}(\hat{X}) = \mathrm{T}_o\,\mathrm{cov}(\hat{X})\,\mathrm{T}_o^{\mathrm{T}} \quad \text{where} \quad T_o \begin{bmatrix} p_{o1} & p_{o2} & ... \end{bmatrix}^{\top} \tag{2.34}$$

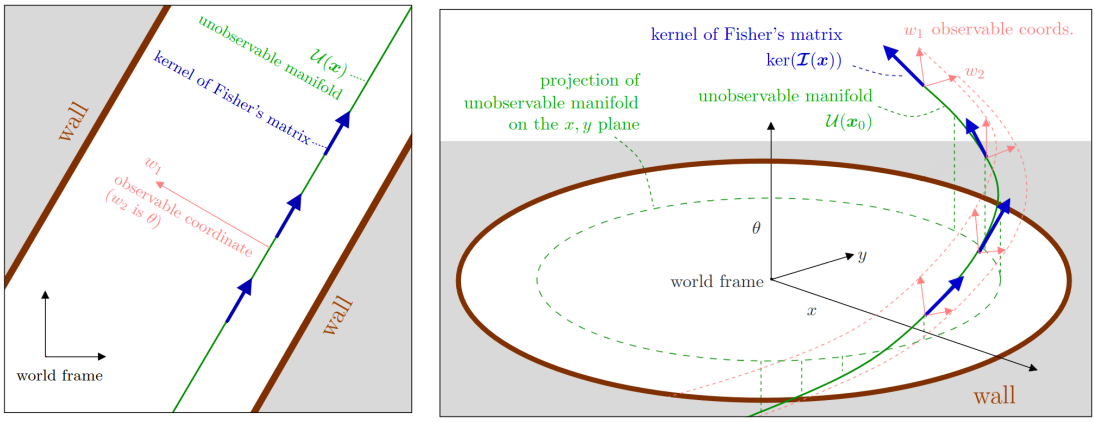
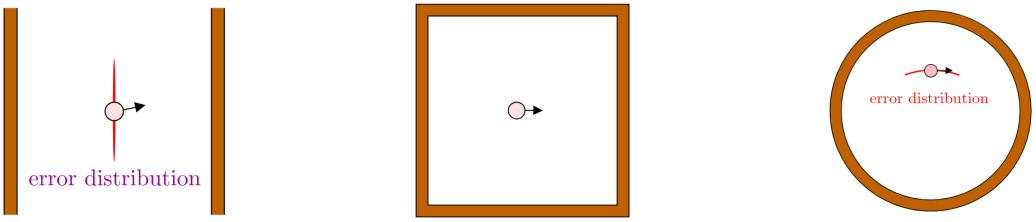


Figure 2.8.: Observable and unobservable manifolds in 2D under-constrained environments with pos parameters x, y and rotation parameter θ. [45]

Furthermore, since Censi's method operates under the assumption that the initial guess lies within the attraction basin of the true solution, it does not consider the uncertainty arising from wrong convergence.

Similar to Bengtsson's test in Subsection 2.3.2, Censi [45] conducted analogous 2D tests across various environments (Figure 2.9), presenting qualitative results in Table 2.2 and quantitative results in Table 2.3. The results for both methods, Hessian and Censi, are promising in these 2D scenarios, especially in corridor scenario [45]. However, Brossard [46] demonstrated that Censi's method tends to be overly optimistic in real-world 3D scenarios in Table 2.5.



(a) 2D Corridor Scenario (b) 2D Square Scenario (c) 2D Circle Scenario

Figure 2.9.: 2D test cases by Censi [45]

*: projection on the observable manifold

Table 2.2.: Qualitative uncertainty estimation results by Censi [45]

	Square	Corridor*	Circle*
Hessian Method	pessimistic	good	moderately optimistic
Censi Method	very good	good	moderately optimistic

Table 2.3.: Quantitative uncertainty estimation results for 2D corridor scenario in Figure 2.9a by Censi [45]

Uncertainty estimations (mm, mm, °)				Projected on $\mathcal{O}$	
	$\sigma(x)$	$\sigma(y)$	$\sigma(\theta)$	$\sigma(w_1)$	$\sigma(w_2)$
True Error	38	219	0.22	0.84	0.38
Hessian Method	33	186	0.24	0.94	0.42
Censi Method	40	257	0.19	0.84	0.33

Prakhya Sai [57] later extended Censi's method for 2D ICP cases with point-to-plane error to 3D ICP cases with both point-to-point and point-to-plane errors without making any assumption about the sensor noise and constraint on the transformation.

Bonnabel [38] demonstrated that Censi's closed-form covariance formula (Equation 2.30) is valid only for point-to-plane error metric. This is because only the Hessian matrix of point-to-plane accurately reflects the characteristics of the actual ICP cost function.

2.3.4. Monte Carlo Simulation

Monte Carlo simulation is a widely used technique to calculate ICP uncertainty [64] [55] [56]. This technique involves running multiple ICP instances with varying initial guesses and calculating the covariance based on the distribution of the resulting ICP poses. While this approach provides accurate uncertainty estimation, it is computationally expensive due to the need for multiple ICP runs, making it impractical for real-time applications [50].

In addition to the Hessian method, Bengtsson [56] proposed using Monte Carlo simulation as an offline approach, as described in Algorithm 1. His idea is to compute covariance matrix for specific scans and then use these results online for all future matches involving the same scans in unchanged or slightly altered environments, shown in Figure 2.10. He [56] provided quantitative results in Table 2.1 for 2D environment.

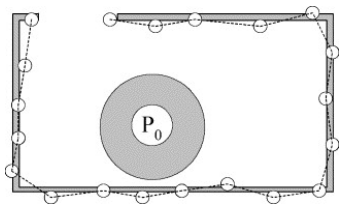


Figure 2.10.: 2D environment, scan and local map created from scan, by Bengtsson [56]

Algorithm 1 Offline ICP Uncertainty Calculation with Monte Carlo by Bengtsson [56]

Input: Initial scan S_0 with N points from environment E, threshold d, initial pose uncertainty
Output: Covariance matrix

Create a geometric map G from S_0 by connecting consecutive points with lines if distance is less than threshold d
for each Monte Carlo iteration (100) **do**
    Randomly choose position P_1 in G based on the initial pose uncertainty
    Create new scan S_1 from P_1 for G
    Match S_1 to S_0 with ICP
    Store true ICP error
end for
Compute the covariance matrix based true ICP errors

Iversen et. al [55] criticised Bengtsson [56] for not validating his offline method with real-world data, despite claiming that the predictions of his offline method are accurate in 2D simulations.

Iversen et. al [55] used this method to predict ICP uncertainties for synthetic depth images, as described in Algorithm 2. In this approach, initial guesses are generated by randomly perturbing the ground truth pose, ideally based on the initial pose uncertainty if available, and noise is added to the depth image (scan points) [55].

Iversen et. al [55] compared his method with Censi's method [45] quantitatively, focusing on translational and rotational true error and uncertainty estimation. To evaluate the performance, he computed the mean and standard deviation of the differences (explained in Subsection 5.1.2) between true error and estimated uncertainty

Algorithm 2 ICP Uncertainty Calculation with Monte Carlo for Depth Images by Iversen [55]

Input: Sampled object model M, generated ideal depth image I
Output: Covariance matrix

for each Monte Carlo iteration k **do**
  Degrade I with noise and generate I_k
  Perturb the ground truth pose randomly to get T_k
  Estimate pose of M in I_k with ICP and initial guess T_k
  Calculate the pose error
end for
Compute the covariance matrix based ICP errors.

values, shown in Table 2.4.

Table 2.4.: Mean and standard deviation of the differences between true error and estimated uncertainty values by Iversen [55]

	Translation [mm]				Rotation [deg]			
	Monte Carlo		Censi		Monte Carlo		Censi	
Object	mean	std	mean	std	mean	std	mean	std
detergent	0.058	0.24	-0.18	0.47	0.038	0.29	0.71	1.17
ice cream bucket	-0.14	0.32	-0.54	0.37	0.037	0.19	-2.73	1.07
cereal box	-0.44	0.73	-1.21	1.15	-0.085	0.42	-0.55	1.19
table leg	0.43	1.67	-1.26	0.76	0.96	1.75	-4.53	5.61

2.3.5. Brossard's Method (12 ICP method)

Brossard et al. [46] claimed that ICP pose and its uncertainty heavily depend on the initial guess, shown in Figure 2.11, and they investigated the relationship between the initial guess and the final pose estimate. Although their study focuses on the effects of the initial guess, their method considers three main sources of uncertainty : wrong convergence, under-constrained situations, and sensor noise.

Initial Guess Investigation

Firstly, the author investigated the initial guess and its effects on the ICP pose and error analytically. Let T_{true} be the correct transformation and T_{ini} be the initial guess with

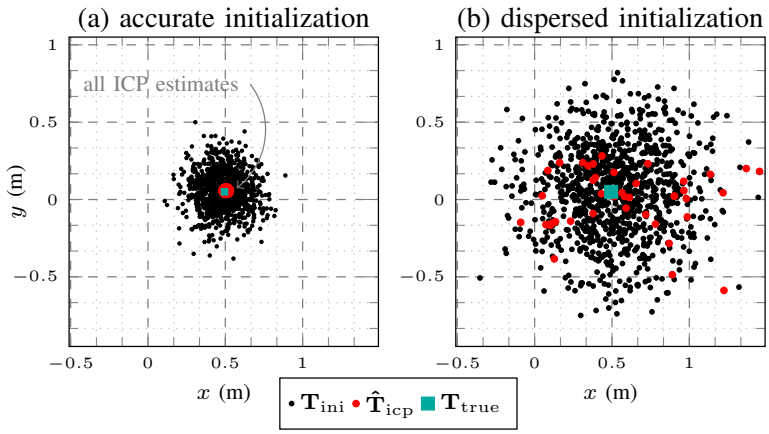


Figure 2.11.: ICP pose results for accurate and dispersed initial estimates [46]

error ξ_{ini} from the covariance matrix Q_{ini} of the initial pose, visualized in Figure 2.12.

$$T_{\text{ini}} = T_{\text{true}} \exp(\xi_{\text{ini}}), \quad \xi_{\text{ini}} \sim \mathcal{N}(0, Q_{\text{ini}}) \tag{2.35}$$

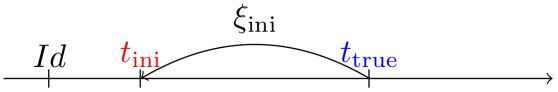


Figure 2.12.: True pose and initial guess for 1D translation example [46]

The goal of ICP is to estimate the correct relative transformation T_{rel}:

$$T_{\text{rel}} := T_{\text{ini}}^{-1} T_{\text{true}} \tag{2.36}$$

$$\begin{aligned} T_{\text{rel}} &= T_{\text{ini}}^{-1} T_{\text{true}} = \exp(-\xi_{\text{ini}}) T_{\text{true}}^{-1} T_{\text{true}} \\ &= \exp(-\xi_{\text{ini}}) \simeq I_4 - \xi_{\text{ini}}^{\wedge} \end{aligned} \tag{2.37}$$

Estimated relative transformation by ICP is:

$$\hat{T}_{\text{rel}} := \text{ICP}\left(\text{M}, T_{\text{ini}}^{-1}\text{S}\right) \quad \text{where M and S are reference and source points} \tag{2.38}$$

Brossard [46] stated that $\hat{T}_{\text{rel}}$ depends on the true relative transformation T_{rel} and that $\hat{T}_{\text{rel}}$ can be represented by a function $f(T_{\text{rel}})$. He noted that correct or wrong convergence depends on this function $f(T_{\text{rel}})$ and that the estimated pose is additionally

affected by sensor noise [46].

Equation 2.38 becomes:

$$\hat{T}_{\text{rel}} = f(T_{\text{rel}}) \exp(Gw) = f\left(\exp(-\xi_{\text{ini}})\right) \exp(Gw) \tag{2.39}$$

where $w \in \mathbb{R}^{6K}$ represents the sensor noise for K pairs, and $G \in \mathbb{R}^{6K \times 6}$ maps this sensor noise to $\mathbb{R}^6$.

Then, the author claimed that since $f\left(\exp(-\xi_{\text{ini}})\right)$ is close to the identity matrix I_4, the function may be linearized around $\xi_{\text{ini}} = 0$, $w = 0$ (Figure 2.13) [46]:

$$\begin{aligned} f\left(\exp(-\xi_{\text{ini}})\right) \exp(Gw) &\simeq f\left(I_4 - \xi_{\text{ini}}^{\wedge}\right) \exp(Gw) \\ &\simeq I_4 + \left(-J\xi_{\text{ini}} + Gw\right)^{\wedge} \end{aligned} \tag{2.40}$$

where matrix J is the linear approximation of $f(\cdot)$.

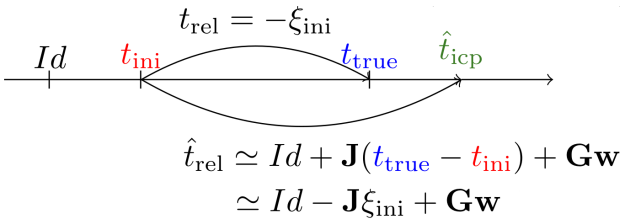


Figure 2.13.: True pose and estimated pose for 1D translation example [46]

ICP final pose estimate:

$$\hat{T}_{\text{icp}} := T_{\text{ini}} \hat{T}_{\text{rel}} = T_{\text{ini}} \text{ICP}(\text{M}, T_{\text{ini}}^{-1}\text{S}) \tag{2.41}$$

With Equations 2.35 and 2.39:

$$\hat{T}_{\text{icp}} = T_{\text{true}} \exp(\xi_{\text{ini}}) f\left(\exp(-\xi_{\text{ini}})\right) \exp(Gw) \tag{2.42}$$

The linearized version using Equation 2.40 (Figure 2.14):

$$\hat{T}_{\text{icp}} \simeq T_{\text{true}} \exp\left((I_6 - J)\xi_{\text{ini}} + Gw\right) \in \text{SE}(3) \tag{2.43}$$

$$\xi_{\text{icp}} = (I_6 - J)\xi_{\text{ini}} + Gw \in \mathbb{R}^6 \tag{2.44}$$

ICP covariance equation is derived as [46]:

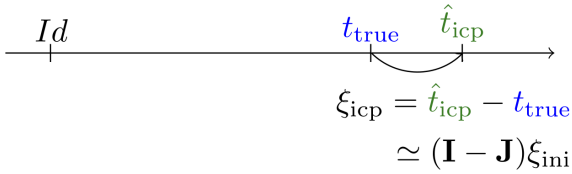


Figure 2.14.: Difference between true pose and estimated pose for 1D translation example [46]

$$Q_{\text{icp}} = (I_6 - J)Q_{\text{ini}}(I_6 - J)^T + GQ_{\text{sensor}}G^T \tag{2.45}$$

While the first term is related to wrong convergence due to initialization and under-constrained environments (if unobservable directions exist), the second term is related to the sensor noise [46].

Unscented Transform

The author [46] stated that, in practice, the first term in Equation 2.45 is significantly more dominant than the second term, making it crucial to calculate the first term accurately.

$$Q_{\text{icp}} \simeq (I_6 - J)Q_{\text{ini}}(I_6 - J)^T \tag{2.46}$$

However, the challenge with the first term lies in the calculation of the matrix J, which is the linear approximation of $f(\cdot)$. This is because both the ICP process and matrix J depend on the initial guess [46]. When the initial guess has sufficiently low uncertainty and is within the attraction basin of the true transformation, the ICP pose converges to the true transformation. Mathematically, a small T_{rel} - a low initial guess error - leads to $f(T_{\text{rel}}) = T_{\text{rel}}$ and $J = I_6$ [46]. However, when the initial guess error is large enough, the ICP may get stuck in a wrong local minimum. In such cases, J is not the analytical Jacobian matrix of $f(\cdot)$ and $J \neq I_6$ [46].

To address this issue, the author [46] employed the unscented transform, as defined in Algorithm 3. For the 3D case, 12 initial guess points are generated using the initialization uncertainty Q_{ini} and propagated through the ICP. Finally, the ICP covariance in Equation 2.46 is calculated using the ICP transformations as shown in Algorithm 3.

Algorithm 3 Computation of J and ICP covariance in Eq.2.46 by Brossard [46]

Require: $\mathcal{M}, \mathcal{S}, T_{\text{ini}}, Q_{\text{ini}}, \hat{T}_{\text{icp}} = T_{\text{ini}}\text{icp}(\mathcal{M}, T_{\text{ini}}^{-1}\mathcal{S})$
  // set sigma points with columns of Cholesky decomposition for Q_{ini}
  $\xi_{\text{ini}}^{j} = \text{col}(\sqrt{6Q_{\text{ini}}})_j,\ j = 1, \ldots, 6$
  $\xi_{\text{ini}}^{j} = -\text{col}(\sqrt{6Q_{\text{ini}}})_{j-6},\ j = 7, \ldots, 12$
  // ICP for each sigma points through Equation 2.38
  for $j = 1, \ldots, 12$ **do**
    $T_{\text{ini}}^{j} = T_{\text{ini}} \exp(\xi_{\text{ini}}^{j})$
    $\hat{T}_{\text{icp}}^{j} = T_{\text{ini}}\text{icp}(\mathcal{M}, (T_{\text{ini}}^{j})^{-1}\mathcal{S})$
    $\xi_{\text{icp}}^{j} = \exp^{-1}(\hat{T}_{\text{icp}}^{-1}\hat{T}_{\text{icp}}^{j})$
  end for
  // compute covariance and J
  $(I_6 - J)Q_{\text{ini}}(I_6 - J)^T = \frac{1}{12}\sum_{j=1}^{12} \xi_{\text{icp}}^{j}(\xi_{\text{icp}}^{j})^T$
  $\hat{\xi}_{\text{icp}} = \frac{1}{12}\sum_{j=1}^{12} \xi_{\text{icp}}^{j}$
  $J = -\left(\sum_{j=1}^{12} \xi_{\text{icp}}(\xi_{\text{icp}}^{j})^T\right) Q_{\text{ini}}^{-1} + I_6$
Ensure: $J, (I_6 - J)Q_{\text{ini}}(I_6 - J)^T$

This method has advantages over the Monte Carlo approach. While initial poses in Monte Carlo are chosen randomly, in this method, they are systematically determined using the Unscented Transform based on the initial pose uncertainty, resulting in a deterministic covariance estimation [46]. Another key advantage is that this method requires only 12 ICP registrations, significantly reducing the computational load [46].

Brossard [46] compared his method to Monte Carlo and Censi's methods using challenging data sets for point cloud registration algorithms [65] and results are shown in Table 2.5 (Normalized Norm Error (NNE) metric explained in Subsection 5.1.1).

Table 2.5.: ICP uncertainty estimation results in challenging data sets [65] by Brossard [46]

	NNE		NNE*	
	trans.	rot.	trans.	rot.
Censi	10^3	10^3	38	10^2
Monte Carlo	10^3	10^2	22	20
12 ICP	4.2	34	0.8	3.8

*: Robust NNE after removing both the more and less accurate quantiles of each

registration

2.3.6. Summary

Least-Square uncertainty method 2.3.1 is classified as too optimistic in Table 2.1, and the method evaluates the alignment, not considering under-constrained environment and wrong convergence situations.

Another closed-form solution, Censi method 2.3.3, is classified as optimistic in Table 2.4 and too optimistic in Table 2.5, although Censi claimed estimations are promising in Tables 2.2 and 2.3. This can be explained by the fact that the method performs well when the true error is low and there is no incorrect convergence or under-constrained environments. However, it may become too optimistic when the ICP pose errors are high.

The Hessian method 2.3.2 uses the Hessian matrix of the error function to calculate uncertainty, similar to closed-form methods. However, it computes the Hessian matrix numerically by performing new data associations and error calculations for slightly different ICP poses (with only one ICP iteration, not multiple iterations until convergence). In Tables 2.2 and 2.3, the uncertainty estimations are accurate for 2D scenarios. However, this method has not been evaluated with 3D real-world data and not mentioned in recent papers, although his offline method, explained in Section 2.3.4, in the same paper [56] has been referenced.

While Monte Carlo method 2.3.4 runs multiple ICP processes with a set of N random initial guess poses, Brossard method 2.3.5 runs only 12 ICP processes for 3D pose estimation and 6 ICP processes for 2D pose estimation using deterministic initial guess poses derived from the initial guess uncertainty. Both methods provide accurate uncertainty estimations, as shown in Tables 2.4 and 2.5. However, they are computationally expensive due to multiple ICP runs and not feasible for real-time applications.

Table 2.6.: Comparison of different methods based on [56], [45], [46], [55]

Method	**Covariance Estimation**	**Sensor Noise**	**Captures Under Constrained Env.**	**Captures Wrong Convergence**	**Covariance Matrix Shape (Captures the Env.)**	**Runtime**
Least-Square	too optimistic	+	-	-	-	low
Hessian	moderate	+	+	+	+	high[1]
Censi	(too) optimistic	+	+[2]	-	+[2]	low
Monte Carlo	moderate	+	+	+	+	very high
12 ICP	moderate	+	+	+	+	high

1: Not specified, but HIGH due to multiple data associations.
2: Excludes purely under-constrained directions in the covariance estimation.

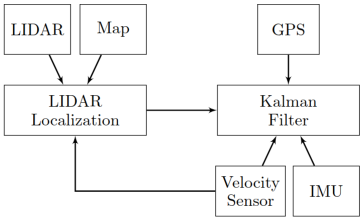
3. Research Project and Dataset

In this chapter, the LiDAR-based localization and state estimation processes used in the vehicle and the dataset from the Yas Marina Circuit are first explained. Next, the convention for the data visualization is described. Finally, the causes of ICP (LiDAR-based localization) uncertainty within this dataset are investigated.

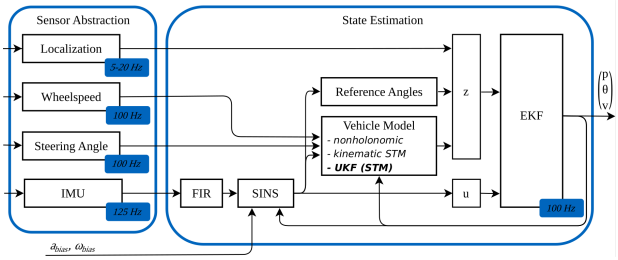
3.1. State Estimation for Localization

For the localization in high-speed autonomous car races, Wischnewski et al. [14] first proposed an Extended Kalman Filter (EKF) based on a kinematic point-mass motion model, supplemented by a Particle Filter for LiDAR-based localization, shown in Figure 3.1a. While this approach demonstrated superior performance under realistic race conditions, it was designed only for 2D scenarios and is not reliable in cases of sensor failures, such as GPS dropouts [14].

Subsequently, Goblirsch et al. [15] proposed a more advanced method. In their approach, a Point-Mass-based EKF is utilized for 3D vehicle dynamics state estimation, independent of vehicle and tire parameters, shown in Figure 3.1b. To improve state estimation, virtual velocity, and reference angle measurements are introduced through a Single-Track-Model (STM)-based Unscented Kalman Filter (UKF), while position and orientation measurements are provided by other localization sources such as GNSS and LiDAR-based localization. Additionally, IMU measurements are filtered using a Finite Impulse Response (FIR) and integrated via a Strapdown Inertial Navigation System (SINS) [15].



(a) State estimation diagram of Wischnewski et al. [14]



(b) State estimation diagram of Goblirsch et al. [15]

Figure 3.1.: State estimation diagrams for the localizaiton

Goblirsch et al. [15] introduced an important feature, covariance adaptation, to prevent sudden jumps in vehicle positions. When covariance values decrease sharply after at least one high uncertainty, rather than immediately using the new low uncertainty values, the uncertainty value is reduced linearly from the previous high to the new lower level. In Figure 3.2, the reported standard deviations(σ_{GNSS}) for the uncertainty and adapted standard deviations (σ_{adapt}) are visualized.

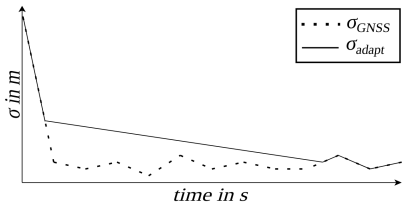


Figure 3.2.: Covariance adaptation [15]

The state estimation system proposed by Goblirsch et al. [15] can be adapted to use any arbitrary number of proprioceptive sensors (e.g., IMU, wheel speeds, steering angle) and exteroceptive sensors (e.g., GNSS, LiDAR, RADAR). The vehicle in this project is equipped with the following sensors and their respective data are listed:

- Kistler SF-Motion Correvit Sensor
  - IMU measurements
  - Longitudinal and lateral velocity
- VectorNav Sensor
  - IMU measurements

  - RTK-GNSS poses
- Steering Angle Sensor
  - Steering angle measurements
- Wheel Speed Encoder
  - Wheel Speed measurements
  - wheel speeds
- 3 X Seyond Falcon Kinetic FK1 LiDAR
  - 3D Point Cloud
- 4 X ZF ProWave RADAR
  - 3D Point Cloud

In the state estimation process, IMU, wheel speed, and steering angle measurements are used for both velocity estimation in the STM and state prediction in the EKF. GNSS sensor provides pose measurements, along with the associated uncertainty values. LiDAR and RADAR are also used as localization sources. However, their sensor data must be processed to estimate the vehicle's pose and the associated uncertainty values.

3.1.1. LiDAR-based Localization

Sauerbeck et al. [13] [12] proposed offline mapping and online localization methods for LiDAR-based localization, shown in Figure 3.3.

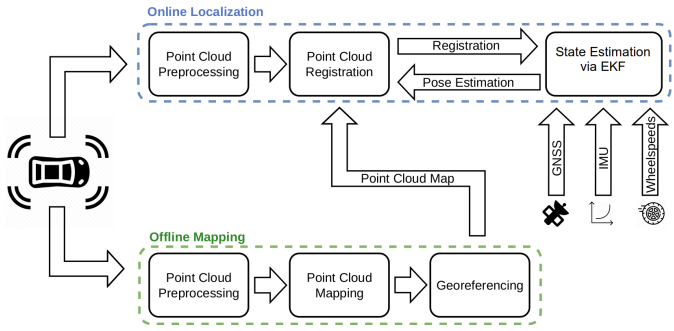


Figure 3.3.: LiDAR-based mapping and localization pipeline [13]

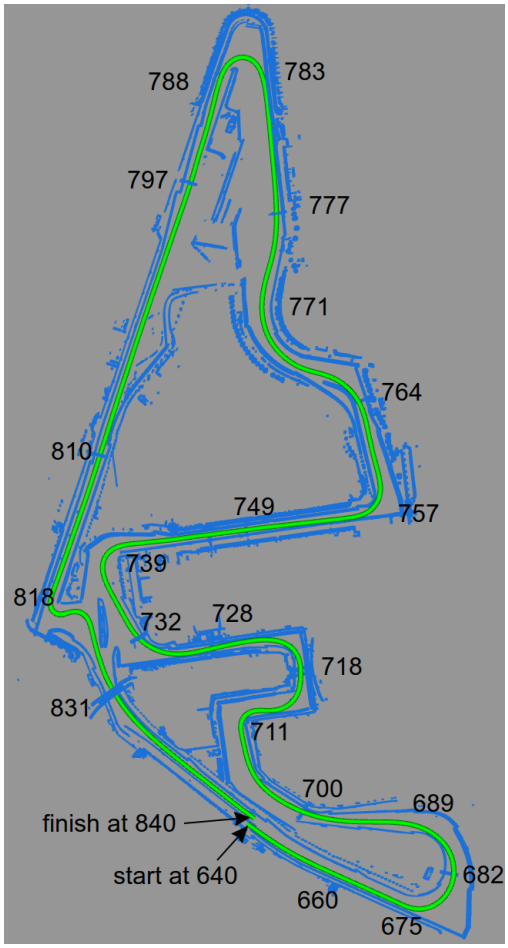
In the mapping process, the LiDAR scans are first registered using KISS-ICP [16] to create an initial 3d point cloud map. Then, this map is post-processed with the Interactive SLAM [66], which performs corrections through loop closures and graph-based optimization utilizing individual poses throughout the trajectory. The corrected 3D point cloud map is georeferenced as a final step by estimating the transformation to global coordinates and applying it to the map [13].

KISS-ICP [16] is adapted to register 3D LiDAR scans into the georeferenced 3D point cloud map in the localization process. Different from the standard KISS-ICP, the initial pose guess for ICP is provided by the state estimation result from the EKF [13] and the georeferenced 3D point cloud map is utilized instead of a dynamic map.

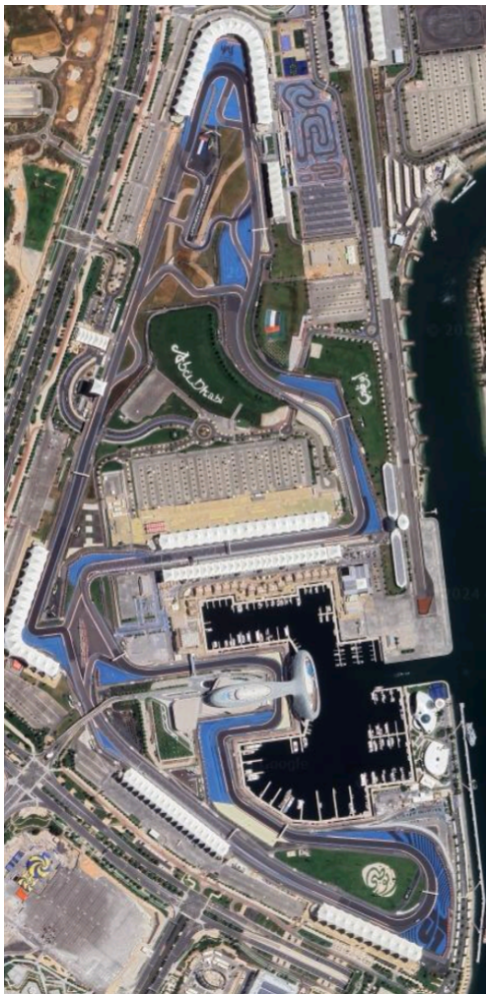
The current implementation calculates the ICP covariance using a method referred to as the "Base method" throughout this report. This method performs an additional data association and transformation after completing the ICP process. The squared values in this transformation vector, $\delta X \in R^6$, are then scaled by constant parameters, and the larger of these results or predefined minimum variance values is selected as the final variance for each direction. Finally, these variance values are used to set the diagonal covariance matrix, used as the covariance estimation.

3.2. Dataset

The primary objective of this thesis is to estimate ICP pose errors for more robust state estimation for the vehicles used in autonomous racing. To validate the proposed methods, the dataset (perception_bag_202404191210) recorded during the Abu Dhabi Autonomous Racing League (A2RL) in April 2024 has been selected for evaluation. In this dataset, the vehicle autonomously completes multiple laps in the Abu Dhabi Race Track (Yas Marina Circuit, shown in Figure 3.4b) using its integrated software stack, including the state estimation implementation for the localization described in Section 3.1. During the run, raw input data from various sensors supporting the perception and localization systems, along with the pose outputs from the sensor data pre-process and localization process, were recorded. The first lap is illustrated in Figure 3.4a, with timestamps relative to the offset value of 1713521000 seconds. For simplicity, all timestamps throughout this report are defined relative to this offset. Furthermore, the 3D point cloud map, generated with the offline mapping process described in Section 3.1.1, of the Yas Marina Circuit is provided.



(a) LiDAR-based point cloud map (blue) and lap 1 (yellow) with relative timestamps



(b) Satellite view of Yas Marina Circuit

Figure 3.4.: Yas Marina Circuit

In the localization, the state estimation pose of EKF is used as the initial ICP pose guess, and later, the ICP pose result is used in the state estimation. This situation creates a circular dependency between the ICP-based LiDAR localization and the state estimation process (Registration and Pose Estimation in Figure 3.3). To evaluate the performance of different ICP covariance estimation methods, two test setups have been defined: closed-loop and open-loop.

3.2.1. Closed-Loop Test and Dataset

In the closed-loop test, only raw sensor topics (highlighted by green in Figure 3.5) are published from the rosbag dataset during the first lap, between 640 and 840 seconds. Other pre-recorded messages, such as the state estimation pose (EKF pose), ICP pose, (highlighted by blue in Figure 3.5), are suppressed. The entire localization pipeline, including LiDAR-based localization (ICP) and the state estimation process (EKF), is re-executed during the test. As a result, the ICP covariance method influences the state estimation outputs, affecting the initial pose guess for the next cycle of the ICP process. Over multiple cycles, this creates a feedback loop where different ICP covariance methods can lead to divergent trajectories, as the prior state estimation result influences each subsequent ICP initial guess. This feedback allows for an evaluation of how different ICP covariance methods impact the overall state estimation and, ultimately, the vehicle's trajectory and robustness in localization. Furthermore, these initial pose guesses can be perturbed to evaluate the performance with wrong initial guesses. For evaluation purposes, in some tests, GNSS data is excluded from the state estimation or its message frequency is decreased. Figure 3.5 visualizes the closed-loop test diagram.

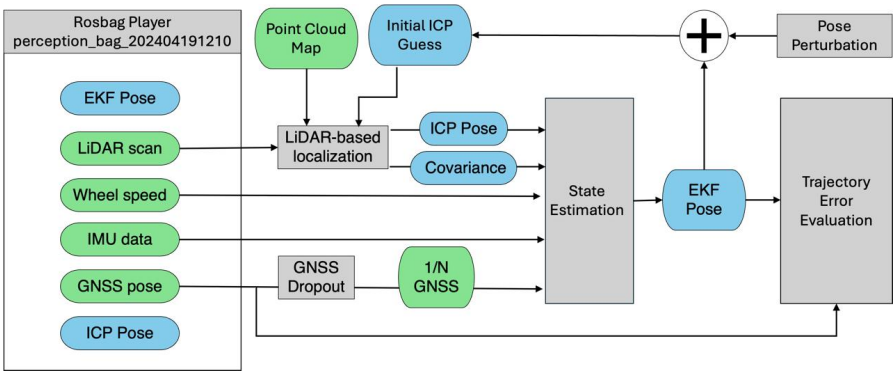


Figure 3.5.: Closed-loop test diagram

3.2.2. Open-Loop Test and Dataset with Different Initial Guess Sets

In the open-loop test, the focus is on the LiDAR-based localization process. The dataset's pre-recorded state estimation poses (EKF pose, highlighted by blue in Figure 3.6) is used directly as the initial guess for each ICP process. Only the LiDAR scans, state estimation poses (EKF pose), and GNSS poses (for comparison purposes) are replayed from the rosbag dataset (Figure 3.6), while the rest of the localization stack is not re-executed. Since the initial guess (EKF pose) is fixed and remains unchanged

across runs, the ICP pose results are independent of the specific covariance estimation method. Therefore, for all ICP covariance methods, the resulting ICP poses and pose errors are expected to remain the same across different open-loop runs. This allows for a direct comparison of the covariance methods under identical conditions, enabling evaluation of how well the covariance estimates align with the fixed errors in the tests. In other words, this setup provides an opportunity to compare the uncertainty estimates of different methods for similar error values, without influencing the trajectory or pose results. Figure 3.6 visualizes the open-loop test diagram.

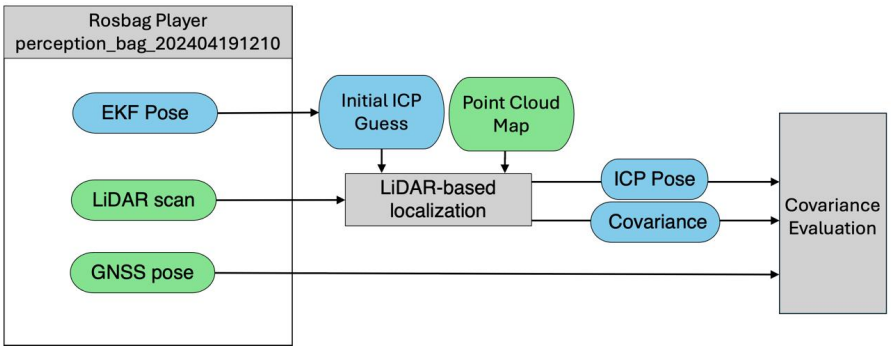


Figure 3.6.: Open-loop test diagram

In the investigation of uncertainty sources, as detailed in Section 3.4, there are no instances of wrong convergence, as the initial guess poses in the rosbag dataset were within the attraction basin of the true solution. However, in future races or with different datasets, there is no guarantee that the initial guess poses will always lie within this basin. To evaluate the performance of different ICP covariance methods under conditions of incorrect convergence, a new dataset was generated with intentionally inaccurate initial guess poses.

This was achieved by re-executing the closed-loop system using the same rosbag dataset, as explained in Subsection 3.2.1, on a computer with limited computational resources. The necessary topics for the open-loop test were recorded in a new rosbag file (called perception_bag_202404191210_with_inaccurate). Due to the system's reduced computational performance, the state estimation poses were less accurate, thereby generating inaccurate initial guesses for the ICP in the open-loop tests. This setup allows for the evaluation of ICP covariance methods in scenarios where initial guesses fall outside the attraction basin of the true solution. However, this dataset with inaccurate initial guesses is limited to the time range between 640 and 690 seconds, as

the trajectory diverged at 690 seconds.

3.2.3. Ground Truth Trajectory

The GNSS poses in the rosbag dataset, obtained from the highly accurate GNSS sensor, are used to define the ground truth trajectory. However, GNSS accuracy decreases when the vehicle is near tall buildings, leading to high uncertainty in those regions. To ensure reliable evaluation, sections with uncertainty greater than 10 cm are excluded from the analysis. These poses are already in ENU coordinates, so no further conversion is necessary for the trajectory.

3.2.4. Pose Conversion from ENU to Vehicle Frame

True ICP pose errors and estimated ICP pose covariances are compared in the lateral and longitudinal directions for a more insightful evaluation. Since the timestamps of the GNSS and ICP poses are different, the GNSS poses are first interpolated to match the ICP pose timestamps. Then, the ICP poses and covariance matrices, relative to the ENU frame, are transformed into the vehicle's current frame, as defined by the interpolated GNSS poses, using Equations 3.1, 3.2 and 3.3.

Longitudinal and lateral positions in vehicle frame:

$$\begin{bmatrix} x_{\text{vehicle}} \\ y_{\text{vehicle}} \end{bmatrix} = \begin{bmatrix} \cos(\theta_{\text{gnss}}) & \sin(\theta_{\text{gnss}}) \\ -\sin(\theta_{\text{gnss}}) & \cos(\theta_{\text{gnss}}) \end{bmatrix} \begin{bmatrix} x_{\text{enu}} - x_{\text{gnss}} \\ y_{\text{enu}} - y_{\text{gnss}} \end{bmatrix} \tag{3.1}$$

Orientation in vehicle frame:

$$\theta_{\text{vehicle}} = \theta_{\text{enu}} - \theta_{\text{gnss}} \tag{3.2}$$

Covariance matrix in vehicle frame:

$$Q_{\text{vehicle}} = \begin{bmatrix} \cos(\theta_{\text{gnss}}) & \sin(\theta_{\text{gnss}}) \\ -\sin(\theta_{\text{gnss}}) & \cos(\theta_{\text{gnss}}) \end{bmatrix} Q_{\text{enu}} \begin{bmatrix} \cos(\theta_{\text{gnss}}) & \sin(\theta_{\text{gnss}}) \\ -\sin(\theta_{\text{gnss}}) & \cos(\theta_{\text{gnss}}) \end{bmatrix}^T \tag{3.3}$$

where:

- $\begin{bmatrix} x_{\text{vehicle}} & y_{\text{vehicle}} & \theta_{\text{vehicle}} \end{bmatrix}^T$ is ICP pose in the 2D vehicle frame.
- $\begin{bmatrix} x_{\text{enu}} & y_{\text{enu}} & \theta_{\text{enu}} \end{bmatrix}^T$ is ICP pose in the 2D ENU frame.
- $\begin{bmatrix} x_{\text{gnss}} & y_{\text{gnss}} & \theta_{\text{gnss}} \end{bmatrix}^T$ is GNSS pose in the 2D ENU frame.

- Q_{vehicle} and Q_{enu} are ICP covariance matrices in 2D vehicle frame and 2D ENU frame, respectively.

3.2.5. Normal Estimation for 3D Map Points

For the point-to-plane residuals in ICP, the normal vectors are required for each 3D point in the map. For the 3D normal vector estimation, NormalEstimation class from Point Cloud Library [44] is used and defined in Algorithm 4.

Algorithm 4 Normal Estimation from Point Cloud Library

Require: Point cloud P, threshold distance ϵ
for each point p in P **do**
Step 1: Find Nearest Neighbors
Find the set $N(p)$ of nearest neighbor points within threshold ϵ of p
Step 2: Estimate Planar Parameters using Least Squares
Compute the centroid $\bar{p}$ of the neighbors in $N(p)$
Compute the covariance matrix C_p for the points in $N(p)$ using Equation 3.4
Set the normal vector n_p to the eigenvector corresponding to the smallest eigenvalue of C_p
end for

The covariance matrix C for each point p is computed as:

$$C = \frac{1}{k} \sum_{i=1}^{k} (p_i - \bar{p}) \cdot (p_i - \bar{p})^T \tag{3.4}$$

where k is the number of nearest neighbors of point p, p_i represents the i-th neighbor point in the set $N(p)$, $\bar{p}$ is the centroid of the neighboring points.

The 3D normal vectors for each point in the point cloud map were estimated with a neighborhood radius of 30 cm. The results of this normal estimation process are shown in Figure 3.7, where the direction of the normal vectors is visualized. These normals are crucial for computing point-to-plane residuals in ICP and for the clustering process explained in Section 3.2.6.

Figure 3.7.: 3D normal vectors (green)

3.2.6. Surface Extraction & Clustering

In Section 4.2, covariance is estimated based on the segmentation of the map, which requires extracting distinct segments. Since the map predominantly consists of planar surfaces, these planes are extracted using the RegionGrowing class from the Point Cloud Library (PCL) [44], defined in Algorithm 5. The algorithm identifies surfaces by applying a smoothness criterion and grouping regions with minimal variation in surface normals and curvatures.

The RegionGrowing algorithm from the Point Cloud Library (PCL) [44] was applied to the point cloud map to extract planar regions. Key parameters were adjusted to ensure effective segmentation of the map (minimum cluster size of 10,000 points, 30 neighbors considered for each point, a smoothness threshold of $\pi/9$ radians, and a curvature threshold of 1.5). The result of the clustering process is visualized in Figure 3.8, where different colors represent different clusters, though clusters of the same color may represent separate regions

Algorithm 5 Region Growing Algorithm

Require: Point cloud P, curvature threshold ϵ_c, angle threshold ϵ_a
Ensure: Labeled regions of the point cloud

```
Sort the points in P by their curvature values
Initialize an empty set of labeled points
while there are unlabeled points in P do
    Pick the point p with the minimum curvature value from the unlabeled points
    Add p to the set of seeds
    while seeds are not empty do
        for each seed point s in seeds do
            Find the neighboring points of s
            for each neighbor point n do
                Compute the angle between the normal of n and the normal of s
                if angle is less than ε_a then
                    Add n to the current region
                    Compute the curvature of n
                    if curvature of n is less than ε_c then
                        Add n to the seeds
                    end if
                end if
            end for
            Remove s from seeds
        end for
    end while
end while
```

Figure 3.8.: Map segments

3.3. Visualization of Dataset

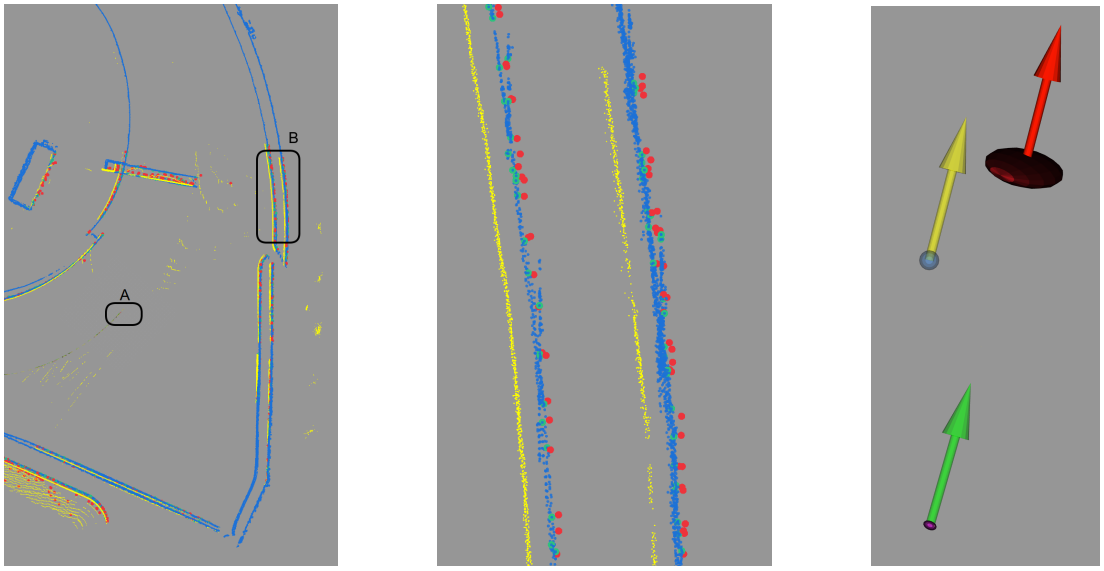
For the investigation of the ICP process and EKF, the 3D data is visualized in RViz using a top-down orthographic projection onto a 2D view, as shown in Subfigure 3.9a.

In Figure 3.9b, the data related to the ICP process in region B is zoomed in. The blue points represent the 3D point cloud map, while the yellow points correspond to the 3D scan points transformed by the initial guess. Green points indicate paired map points in the ICP process, and red points represent paired scan points but transformed by the final ICP pose result. Ideally, the red points (paired and transformed scan points) should align with the green points (paired map points) and the blue points (map points).

Figure 3.9c shows a zoomed-in view of pose results in region A. The red arrow and ellipse represent the ICP pose and estimated covariance. The yellow arrow and blue ellipse depict the GNSS pose and its covariance. The green arrow and purple ellipse represent the state estimation (EKF) pose and its associated covariance. Since their

timestamps differ, the poses do not necessarily coincide. However, due to the non-holonomic constraint of the vehicle — where the wheels and steering system restricts lateral movement — the lateral positions should ideally remain consistent. Additionally, previous poses are retained in the visualization to visually inspect all the trajectories visually.

The same color scheme is used across all figures generated by the visualization tool.



(a) Data visualization in top-down 2D view

(b) Zoomed-in view of region B

(c) Zoomed-in view of region A

Figure 3.9.: Visualization of 3D ICP and EKF results in RViz using top-down orthographic projection.

3.4. Investigation of the ICP Uncertainty Sources

Before developing and evaluating new ICP covariance estimation approaches, a visual investigation was conducted into the accuracy of ICP poses and the underlying sources of ICP uncertainty and error. In the experiments, the primary source of uncertainty stems from sensor noise and bias, as explained in Subsection 2.2.3. Additionally, the map contains long featureless straight sections, which increase uncertainty due to the under-constrained nature of the environment, as detailed in Subsection 2.2.2.

The initial pose guesses for ICP are provided by the highly accurate state estimation, explained in Section 3.1. Since this initial pose guess lies within the attraction basin of the true ICP solution, no instances of incorrect convergence were observed during testing, as discussed in Subsection 2.2.1. However, to evaluate ICP performance in cases of incorrect convergence, the dataset with inaccurate initial pose guesses was generated, as explained in Subsection 3.2.2. In the ICP results with this dataset, wrong convergences were observed, and their reasons were investigated.

3.4.1. Sensor Noise

As discussed in Subsection 2.2.3, sensor noise is an inherent characteristic of real-world data acquisition systems. Even with a well-constrained environment and proper alignment, small residual errors between the paired source points and the map can be observed. In Figure 3.10, despite the correct overall alignment of the map and the sensor data, some paired source points are not perfectly aligned with the map, and these errors can be attributed to sensor noise.

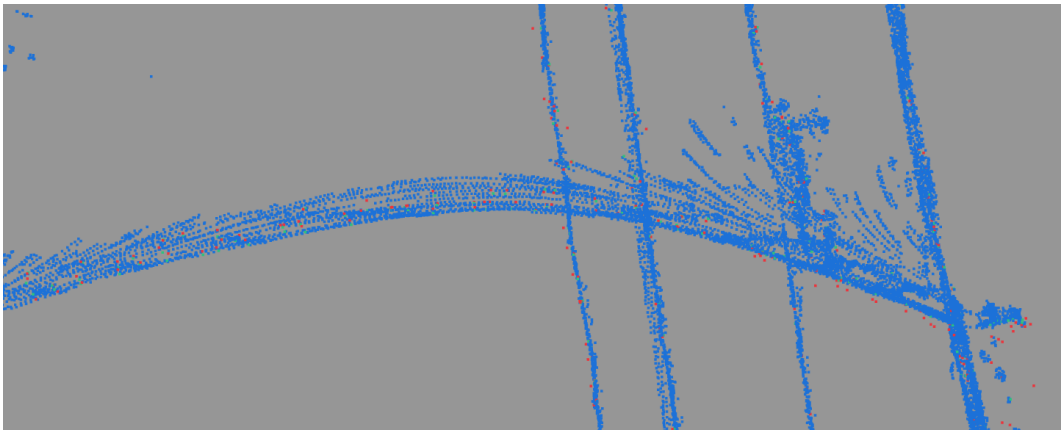
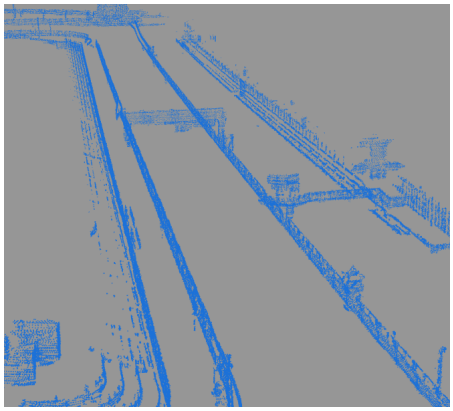


Figure 3.10.: Point pairs affected by noise

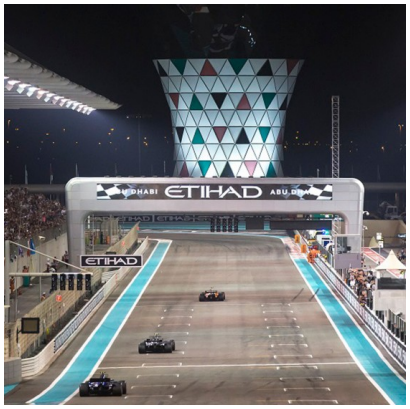
3.4.2. Underconstrained Environments

The map contains long straight segments where intersections with the scan may lack features to create constraints in the longitudinal direction (Figure 3.11). For instance, in Figure 3.12, the billboard is captured by the LiDAR and provides a constraint, whereas in Figure 3.13, there is no constraint in the longitudinal direction. In such

cases, even with perfect alignment, the ICP position in the longitudinal direction can still be incorrect.



(a) Start straight in 3D point cloud map



(b) Billboard on the start straight

Figure 3.11.: Start straight with billboard

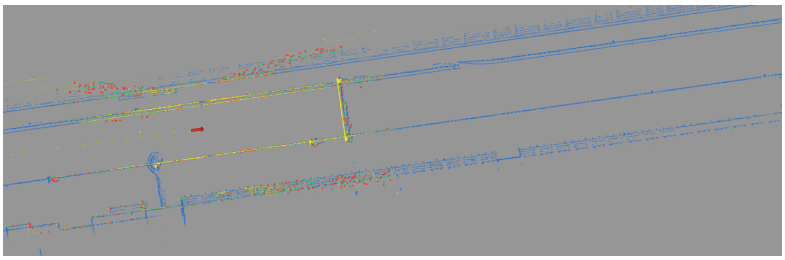


Figure 3.12.: Constrained part of the straight

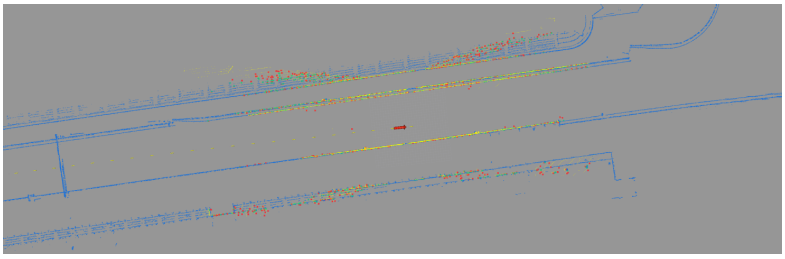
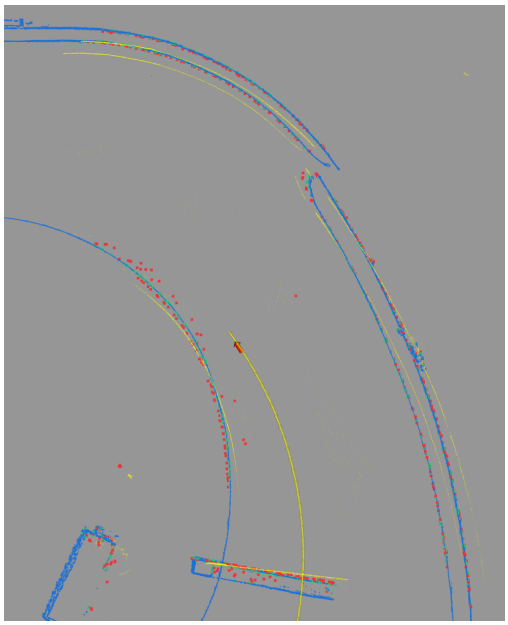


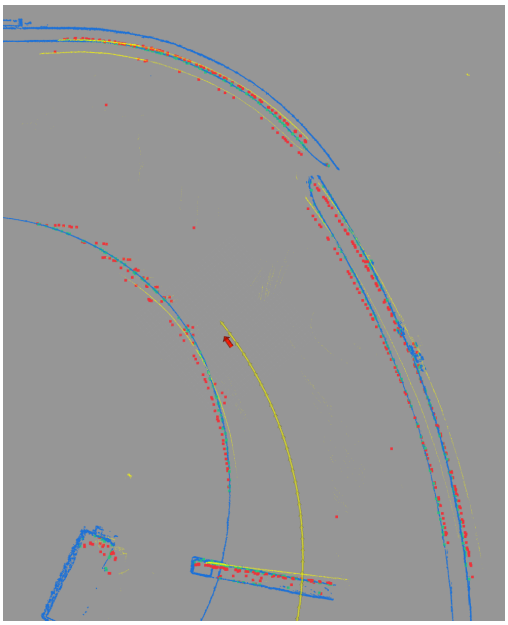
Figure 3.13.: Under-constrained part of the straight

3.4.3. Wrong Convergence

As discussed in Section 2.2.1, when the initial guess provided to the ICP is not within the attraction basin of the true solution, the pose estimate can converge to a wrong local minimum. This issue is observed in a real-world scenario, visualized in Figure 3.14b. In the upper part of the map, two similar surfaces (shown as billboards in Figure 3.15) are positioned close to each other. Due to their similarity and proximity, ICP wrongly matches points from one surface to another. Although the point pairs in the lower and lower-right sections of Figure 3.14b are correctly matched and ICP also tries to minimize the residuals between them, the wrong point correspondences in the upper section cause the overall pose estimate to be trapped in a local minimum.



(a) Correct convergence and accurate alignment



(b) Wrong convergence and poor alignment

Figure 3.14.: Correct and wrong convergence examples

Figure 3.15.: Surfaces causing wrong convergence

3.4.4. ICP Parameters

ICP parameters can affect the accuracy and pose errors. For instance, the threshold for the maximum allowable distance between paired points is critical. If the threshold is too high, wrong points may be paired. Conversely, if it is too low, even correct points may not be paired, preventing the ICP algorithm from converging to the correct solution. In the dataset, there are noisy points on the ground, and some of them are paired with the map, as seen in Figure 3.16, which may have affected the results.

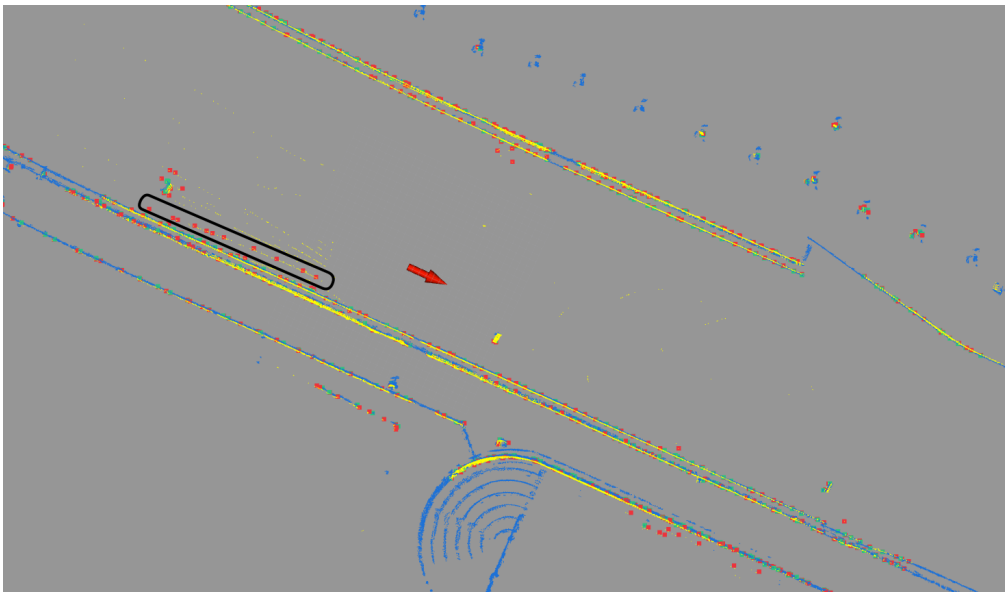


Figure 3.16.: Wrong pairs between map and noisy ground points

4. Methods

In this chapter, two new approaches are introduced for the covariance calculation: Error Distribution method and Segmentation method.

4.1. Error Distribution Method

The transformation parameters are optimized through the ICP process, ideally resulting in a mean error of zero. However, in an optimization problem, the magnitude of the residuals can vary depending on the specific circumstances, shown in Figure 4.1.

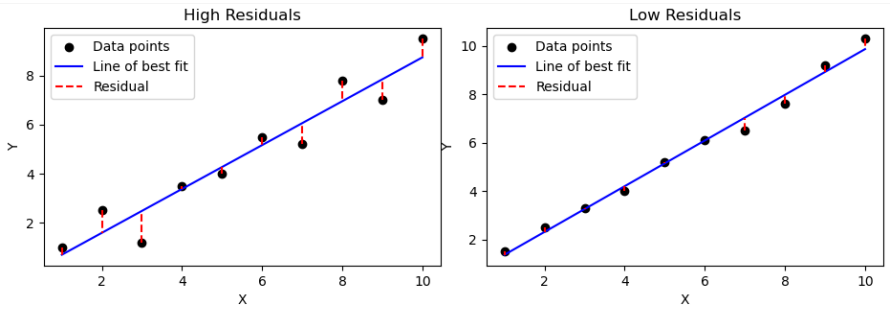
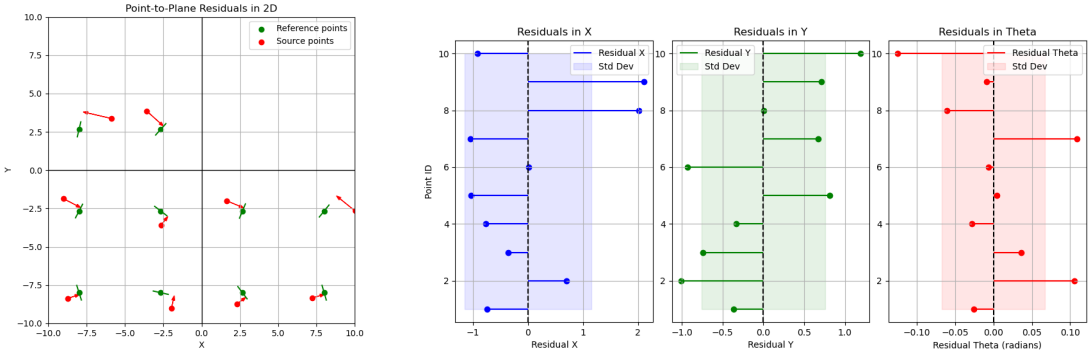


Figure 4.1.: Regression with high and low residuals

This approach assumes that the pairs of point clouds are correctly matched and evaluates the alignment. The idea is to create a distribution based on the errors between these pairs, as shown in Figure 4.2 for a simple 2D case. For instance, in the ICP context, all residuals would be zero in a perfect match without sensor noise, leading to a covariance matrix that is essentially a zero matrix. In a more realistic scenario, residuals may be large along one axis and small along others. In such cases, the position estimation may be inaccurate in directions exhibiting large residuals.



(a) 2D ICP point-to-plane residuals (b) 2D ICP point-to-plane residuals' distribution

Figure 4.2.: 2D ICP Point-to-Plane ICP

Variances for each component of the residual vectors can be computed using their respective components, as illustrated in Figure 4.2b.

Furthermore, the complete covariance matrix, cov $\in \mathbb{R}^{6\times6}$, can be calculated for 3D cases in matrix form.

Let $r_t \in \mathbb{R}^3$ be the positional residual vector, and $r_R \in \mathbb{R}^3$ the rotational residual vector. The vector r_t is derived from either the point-to-plane or point-to-point ICP residuals, while r_R is defined as the cross product of the source point position s and the translation residual r_t, such that:

$$r_R = s \times r_t \quad \text{, where } s \text{ is the position of the source point} \tag{4.1}$$

Let $r_{full,i} \in \mathbb{R}^6$ be a full residual vector consisting of both translational and rotational residuals for the analysis of the residual for pair i.

$$r_{full,i} = \begin{pmatrix} r_{t,i} \\ r_{R,i} \end{pmatrix} = \begin{pmatrix} r_{t,i} \\ s \times r_{t,i} \end{pmatrix} \in \mathbb{R}^6 \tag{4.2}$$

The covariance matrix of the full residual vectors can be calculated as:

$$cov = \frac{1}{n}\sum_{i=1}^{n} r_{full,i} r_{full,i}^T = \frac{1}{n}\sum_{i=1}^{n} \begin{pmatrix} r_{t,i} \\ r_{R,i} \end{pmatrix} \begin{pmatrix} r_{t,i}^\top & r_{R,i}^\top \end{pmatrix} \tag{4.3}$$

Point-to-Point Error

For the point-to-point error metric, the translational residual, $r_{t,i} \in \mathbb{R}^3$, between the map point and the source point is defined in Equation 2.10.

$$r_{t,i} = r_i \in \mathbb{R}^3 \tag{4.4}$$

The full residual vector is given by:

$$r_{full,i} = \begin{pmatrix} r_{t,i} \\ r_{t,i} \times s \end{pmatrix} = \begin{pmatrix} r_i \\ r_i \times s \end{pmatrix} \in \mathbb{R}^6 \tag{4.5}$$

In Equation 2.12, the Jacobian matrix, $\mathrm{J} \in \mathbb{R}^{3\times6}$, consists of an identity matrix in the first part and a linear transformation (hat operator) representing the cross product in the second part. The full residual vector can be expressed as:

$$r_{full,i} = J_i^T \, r_i \in \mathbb{R}^6 \tag{4.6}$$

Recalling the covariance equation in 4.3 and the covariance matrix:

$$\begin{aligned} cov &= \frac{1}{n} \sum_{i=1}^{n} r_{full,i} \, r_{full,i}^T \\ &= \frac{1}{n} \sum_{i=1}^{n} \mathrm{J}_i^T \, r_i \left(\mathrm{J}_i^T \, r_i \right)^T \\ &= \frac{1}{n} \sum_{i=1}^{n} \mathrm{J}_i^T \, r_i \, r_i^T \, \mathrm{J}_i \end{aligned} \tag{4.7}$$

Point-to-Plane Error

Recalling Equation 2.13 for the point-to-plane error metric, the translational residual $r_{t,i} \in \mathbb{R}^3$ is defined as:

$$r_{t,i} = n_i \, r_i \in \mathbb{R}^3 \quad \text{where} \quad r_i \in \mathbb{R} \tag{4.8}$$

$$r_{full,i} = \begin{pmatrix} r_{t,i} \\ r_{t,i} \times s \end{pmatrix} = \begin{pmatrix} n_i \, r_i \\ s \times n_i \, r_i \end{pmatrix} \in \mathbb{R}^6 \tag{4.9}$$

In Equation 2.15, the Jacobian matrix $\mathrm{J} \in \mathbb{R}^{1\times6}$ consists of the normal vector n in the first part and the cross product of the source point s and the the normal vector n in the second part. The full residual vector can be expressed as:

$$r_{full,i} = \mathrm{J}_i^T \, r_i \in \mathbb{R}^6 \tag{4.10}$$

Recalling the covariance equation in 4.3 and the covariance matrix:

$$
\begin{aligned}
cov &= \frac{1}{n}\sum_{i=1}^{n} r_{full,i}\, r_{full,i}^{T} \\
&= \frac{1}{n}\sum_{i=1}^{n} \mathrm{J}_i^T\, r_i \left(J_i^T\, r_i\right)^T \\
&= \frac{1}{n}\sum_{i=1}^{n} \mathrm{J}_i^T r_i r_i \mathrm{J}_i = \frac{1}{n}\sum_{i=1}^{n} r_i^2 \mathrm{J}_i^T \mathrm{J}_i
\end{aligned}
\tag{4.11}
$$

Generalization

In this method, the goal is to calculate the covariance for the distribution of the full residual vectors $r_{full,i}$. This vector corresponds to the right-hand side term in the normal equation, Equation 2.7, specifically $\mathrm{J}^T r \in \mathbb{R}^6$, for any 3D ICP case. The residual vector, $r \in \mathbb{R}^n$, which varies depending on the specific case, is mapped into the full residual vector, $r_{full,i} = \mathrm{J}^T r \in \mathbb{R}^6$ using the Jacobian matrix, $\mathrm{J} \in \mathbb{R}^{n\times 6}$, of the residual metric. Notably, the full residual vector is consistently $\mathbb{R}^6$ and is invariant to changes in the error metric, weighting kernel, and other parameters. For any ICP problem, the term $\mathrm{J}^T r \in \mathbb{R}^6$ in the normal equation 2.7 represents the full residual vector in Equation 4.2 . This term, which is already computed during the ICP process, can be directly used in Equation 4.3 to calculate the covariance without requiring any additional computation.

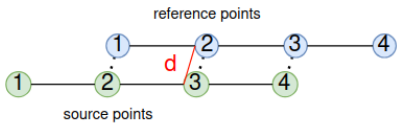
$$
cov = \frac{1}{n}\sum_{i=1}^{n} \mathrm{J}_i^T r_i r_i^T \mathrm{J}_i \tag{4.12}
$$

It is crucial to note that the impact of the residuals, both in the ICP process to estimate the final pose and in the covariance estimation, must be consistent. For an accurate and realistic covariance estimation, the same conditions must be applied to the covariance calculation to ensure consistency in the impact of the residuals. For instance, if the residuals are computed using the point-to-point error and are weighted by a kernel during the ICP process, the identical weighting kernel and error metric must be employed in the covariance calculation to replicate the same effects.

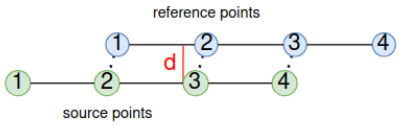
Generalized covariance matrix with weight adjustments:

$$
\begin{aligned}
cov &= \frac{1}{n}\sum_{i=1}^{n} \mathrm{J}_i^T w_i r_i r_i^T w_i \mathrm{J}_i \\
&= \frac{1}{n}\sum_{i=1}^{n} w_i^2 \mathrm{J}_i^T r_i r_i^T \mathrm{J}_i
\end{aligned}
\tag{4.13}
$$

This method primarily accounts for sensor noise and evaluates the fitness between the provided point pairs. However, it can be overly optimistic in under-constrained scenarios, particularly in unobservable directions. For point-to-point error, since source points are always matched with the closest map points, the positional residual often remains small, even if the pairing is incorrect, as shown in Figure 4.3a. On the other hand, in point-to-plane error, the residual component along the unobservable axis is ignored due to the projection onto the plane's normal vector, shown in Figure 4.3b. As a result, this approach fails to capture uncertainty arising from underconstrained scenarios. To handle these situations effectively, an additional covariance matrix representing unobservability should be incorporated into the covariance structure of this method.



(a) Point-to-Point residual between wrong pairs



(b) Point-to-Plane residual between wrong pairs

Figure 4.3.: Residuals between wrong pairs

4.2. Clustering Methods

My other approach involves evaluating the residuals and transformations between point clusters in the scan and reference (map) point clouds rather than between individual point pairs, as described in Section 4.1. The map consists of various elements, such as planes, buildings, billboards, and other structures. This creates a constrained environment for matching, leading to more stable localization results. However, if there is a misalignment in any of these elements, it can cause inaccurate convergence, resulting in errors. If the alignment is correct (Figure 3.14a) and the pose estimation is close to true transformation, the additional transformation results for each segment

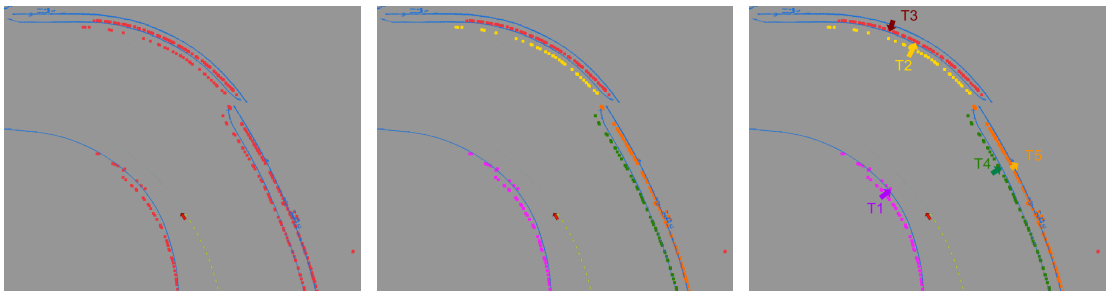
should be consistent and close to the identity matrix (transformation vector $\delta X_i = 0$ and $\exp(\delta X_i) = I_4$). However, if the alignment is wrong (Figure 3.14b), for example, due to a mismatch in a part of the scan points, the transformation results for each segment differs significantly, leading to a higher covariance in the transformation distribution. For the ICP covariance estimation, the distribution of these transformation results for each cluster can be used.

As discussed in Subsection 3.4.3, the main reason for the wrong convergence in Figure 3.14b is the mismatch of a single segment of the scan point cloud. This approach allows for the detection of incorrect convergences, which can be reflected in the covariance matrix.

4.2.1. ICP for each Scan Cluster

Ideally, the segmentation can be applied to the current scan point cloud, and new transformations can be computed with ICP for each scan cluster, as described in Algorithm 6 and shown in Figure 4.4. In this process, point pairs are rematched, and transformations are refined iteratively (ICP). Subsequently, the covariance matrix can be derived using the transformations $T_i \in \mathrm{SO}(3)$:

$$\mathrm{cov} = \frac{1}{n}\sum_{i=1}^{n} \delta X_i \delta X_i^T \quad \text{where} \quad \delta X_i = \log(T_i) \in R^6 \tag{4.14}$$



(a) Map (blue) and source points (red)

(b) Map (blue) and clustered source points (other colors)

(c) Registrations of source point clusters via ICP

Figure 4.4.: ICP for each scan cluster

The number of points in the clusters can be taken into account, and a weighted covariance matrix can be derived as follows:

$$\text{cov} = \frac{\sum_{i=1}^{n} n_i \delta X_i \delta X_i^T}{\sum_{i=1}^{n} n_i} \quad \text{, where } n_i \text{ is the number of points in cluster i} \tag{4.15}$$

Algorithm 6 Uncertainty Calculation with Scan Segmentation

Input: Source point cloud, S, reference point cloud M, ICP and segmentation algorithms
Output: Covariance matrix

Segment source point cloud and create new scan segments S_i
for each scan segments S_i **do**
  $T_i = ICP(S_i, M)$
  $\delta X_i = \log(T_i)$
end for
Compute covariance matrix using Equation 4.14 or 4.15.

For robust results, a threshold can be set for the minimum number of points required to consider a segment as valid.

There are several concerns associated with this method. The first problem is that the scan point cloud differs and requires segmentation of each timestamp. However, segmentation is a computationally expensive process, increasing the runtime and the number of clusters in the new scan point cloud may be unknown, causing not controllable clustering process. Another issue is that running ICP for each cluster increases the computational load and extends the runtime, making it potentially infeasible for real-time systems, especially in complex environments.

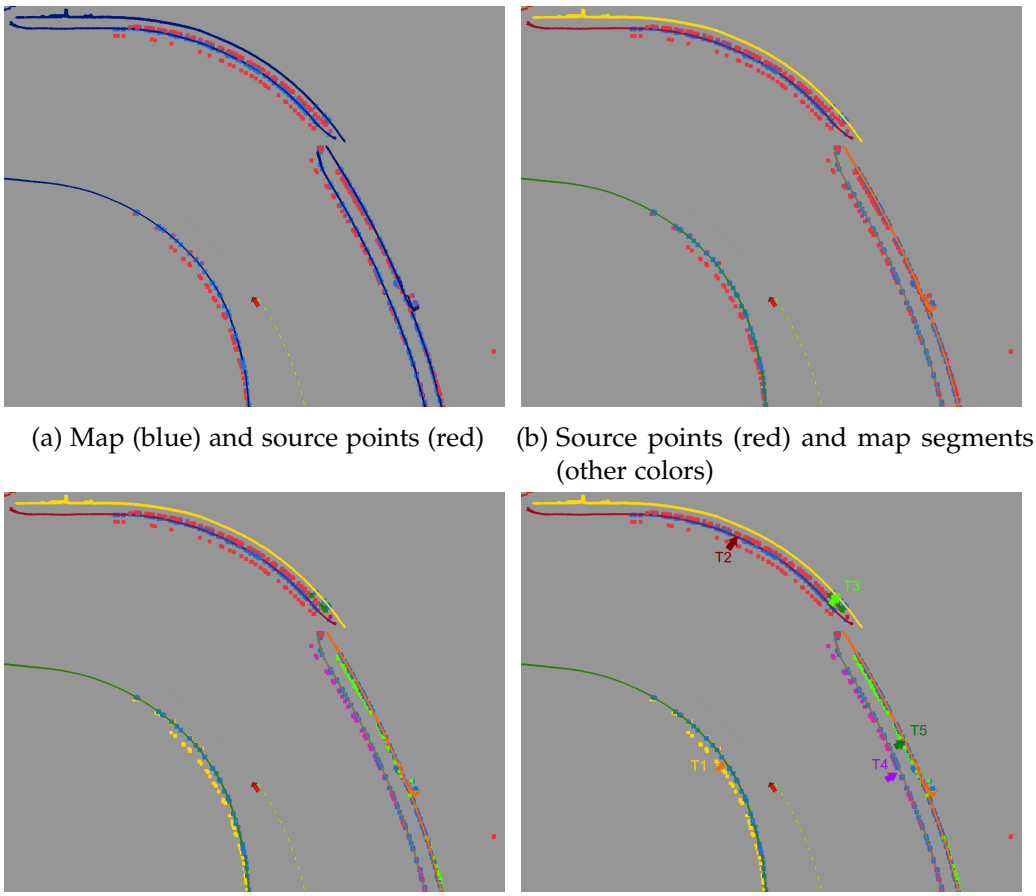
4.2.2. Transformation for each Map Cluster

A simple and effective method is proposed to address the concerns mentioned above. Instead of performing clustering on the scan point cloud for each timestamp and running ICP for each cluster in real-time, clustering can be applied to the reference point cloud (map), which remains the same. Since the map does not change, the clusters can be determined offline as a one-time operation. Additionally, the clusters can be manually adjusted, and the parameters of the segmentation algorithm can be tuned to achieve better segmentation results. This approach also allows for an intuitive

understanding of the segments and subsequent transformations.

Since running multiple ICP processes increases runtime, and considering that KISS-ICP is designed to register the scan to the map (not vice versa), the transformations between the map segments and corresponding source points can be computed for ICP covariance estimation, as described in Algorithm 7 and shown in Figure 4.5.

Registrations of source point segments via ICP



(a) Map (blue) and source points (red)

(b) Source points (red) and map segments (other colors)

(c) Map segments and paired source points (squares with same color) to map segments

(d) Transformations between map segments and corresponding source points

Figure 4.5.: Transformation for each corresponding scan cluster

The concerns discussed in the latter part of Subsection 4.1 are also applicable to these

Algorithm 7 Uncertainty Calculation with Map Segmentation

Input: Source point cloud, S, reference point cloud M and its segments M_i
Output: Covariance matrix

for each map segment M_i **do**
  Get corresponding scan points S_i for map segment M_i
  Calculate transformation T_i between S_i and M_i
  $\delta X_i = \log(T_i)$
end for
Compute covariance matrix using Equation 4.14 or 4.15.

two methods and must be taken into consideration. To obtain realistic uncertainty results, the Jacobian matrix J and the residual r must be computed in the same manner in the ICP process.

4.3. Cramér–Rao Bound

If the point-to-plane error is used in ICP and Jacobian matrices consist of normal vectors of the reference (map) points, e.g., Equation 2.15, Fisher's information matrix provides insight into the shape and constraints of the environment. Censi [45] utilized the information matrix to detect observable and unobservable manifolds, explained in Section 2.3.3. This information matrix can be used to represent epistemic uncertainty.

Recall Fisher's information matrix Equation 2.31:

$$I(S_c, M_c) = \sum_{i=1} \mathrm{J}_i^{\mathrm{T}} \mathrm{J}_i \tag{4.16}$$

Censi [45] mentions that the inverse of Fisher's information matrix $I(S_c, M_c)$ [63] is Cramér–Rao bound (CRB) [67] and it is the lower boundary of the uncertainty:

$$\mathrm{CRB} = (I(S_c, M_c))^{-1} \quad \text{and} \quad \mathrm{cov}(\hat{X}) \geq \mathrm{CRB} \tag{4.17}$$

The CRB represents the lowest achievable uncertainty for ICP pose estimation. Even if the ICP accurately estimates the position in an under-constrained direction due to a correct initial guess, this accuracy cannot be guaranteed due to the lack of information (epistemic uncertainty).

The CRB can be applied alongside methods that evaluate alignment performance, such as the Error Distribution method 4.1 and Segmentation method 4.2. The lowest uncertainty value due to epistemic uncertainty and the alignment error $\mathrm{COV}_{\mathrm{error}}$ can be combined as visualized in Figure 4.18.

$$\mathrm{COV} = \mathrm{CRB} + \mathrm{COV}_{\mathrm{error}} \tag{4.18}$$

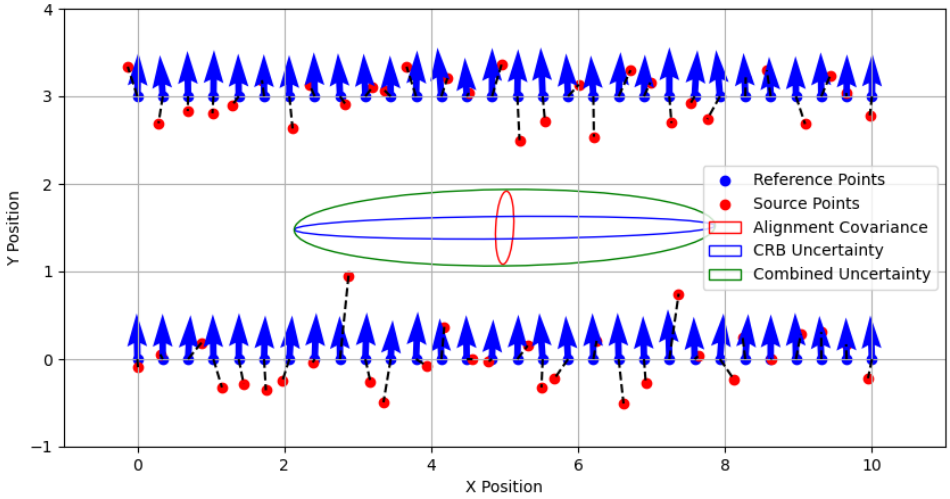


Figure 4.6.: Alignment error and epistemic uncertainty

5. Results

This chapter presents the metrics used for quantitative analysis, followed by a comparison of true ICP errors and estimated values in the open-loop tests. Finally, the impact of the methods on state estimation is assessed in the closed-loop tests.

For further details on the implementation and test setup, please refer to Appendix A.1.

5.1. Metrics

In this section, two metrics used in the literature are described.

5.1.1. Normalized Norm Error

The first metric to evaluate covariance estimation is the Normalized Norm Error (NNE). It is defined as:

$$\text{NNE} = \left(\frac{1}{N} \sum_{n=1}^{N} \| \exp^{-1}(T_{\text{true,n}}^{-1} T_{\text{icp},n}) \|_2^2 \, / \text{trace}(Q_{icp,n}) \right)^{1/2} \tag{5.1}$$

where $T_{\text{true,n}}$ is the true transformation, $T_{\text{icp,n}}$ is the ICP pose, and $Q_{\text{icp},n}$ is the covariance estimation for the n-th pose.

Results in longitudinal and lateral directions and rotations are evaluated separately. For 1D evaluations, the equation becomes:

$$\text{NNE} = \left(\frac{1}{N} \sum_{n=1}^{N} \frac{\xi_n^2}{\sigma_{\text{icp},n}^2} \right)^{1/2} \tag{5.2}$$

where ξ_n is the pos error and $\sigma_{\text{icp},n}^2$ is the variance for the uncertainty estimation.

The ideal result is 1.0. A value less than 1 indicates pessimism, while a value greater than 1 indicates optimism. However, it should be noted that this is a cumulative metric,

and optimistic and pessimistic values may counterbalance each other, resulting in a value close to 1.0.

5.1.2. Difference between True Error and Uncertainty Estimation

Another metric is the distribution of the differences between true error and error estimation:

$$\delta_n = \sigma_{\text{icp},n} - \xi_n \tag{5.3}$$

where $\sigma^2_{\text{icp},n}$ is the square root of variance for uncertainty estimation and ξ_n is the absolute pos error.

Ideal δ_n values are zero, with the mean and standard deviation of the set of δ_n also being zero. A positive mean indicates that the uncertainty estimation is higher than the true error, reflecting a pessimistic estimate.

In the evaluations, true poses are provided by GNSS. For the quantitative analyses, GNSS poses with errors higher than 10 cm are excluded, and the remaining poses are assumed to be correct.

5.2. Open Loop Tests

In this section, the true pose errors and the covariance method estimations are compared visually and quantitatively under various conditions, including:

- Sensor noise, leading to low pose error when ICP converges correctly with no significant mismatches
- Wrong convergence, resulting in highly inaccurate poses due to incorrect matches
- Under-constrained environments

These comparisons are performed using the open-loop tests described in Section 3.2.2. For the sensor noise and under-constrained environment conditions, the first lap of the dataset perception_bag_202404191210, between 640 and 840 seconds, described in Section 3.2 is used. For the wrong convergence conditions, the dataset perception_bag_202404191210_with_inaccurate, between 640 and 690 seconds, is employed.

For the comparison with true errors, the longitudinal and lateral positions are calculated relative to the current vehicle frame, provided by the GNSS pose, as described in Section 3.2.4. These positions are also referred to as position errors, with the GNSS poses assumed to represent the ground truth.

5.2.1. Sensor Noise

Brossard [46] stated that the most significant source of uncertainty arises from wrong convergence when it occurs. Additionally, an under-constrained environment introduces further uncertainty. To evaluate the methods under sensor noise (with low ICP error), position errors and uncertainty estimations are compared in the lateral directions using an open-loop test with accurate initial guesses. In this test, wrong convergence is avoided due to the accurate initial guess, and the lateral direction remains constrained within the map.

Since the position error and uncertainty estimation are small and not visible in 2D visualization, results are evaluated in a time-lateral position plot in Figure 5.1 using the distribution of the difference between true error and estimation and NNE results in Table 5.1.

Table 5.1.: Numerical results in lateral direction under sensor noise conditions

Method	**Average ICP Error (m)**	**Difference Mean (m)**	**Difference Std (m)**	**NNE**
Base Method	0.1123	-0.0123	0.0997	1.5018
Censi Method	0.1125	-0.0446	0.0986	2.2904
Brossard (12 ICP) Method	0.1126	-0.0951	0.1131	71.7044
Monte Carlo	0.1126	0.0045	0.2671	59.1939
Error Dist. Point-to-Point	0.1122	0.0221	0.0977	1.0945
Error Dist. Point-to-Plane	0.1122	0.0136	0.0983	1.1980
Segmentation Point-to-Point	0.1112	-0.0055	0.1202	2.8441
Segmentation Point-to-Plane	0.1112	0.0439	0.1587	2.3171

In the numerical results, shown in Table 5.1 and Figure 5.2, the best-performing methods are the Base method, both Error Distribution methods, and the Segmentation method with point-to-point error. Regarding the difference between true error and error estimation values, the mean values are close to zero (indicating neither optimism

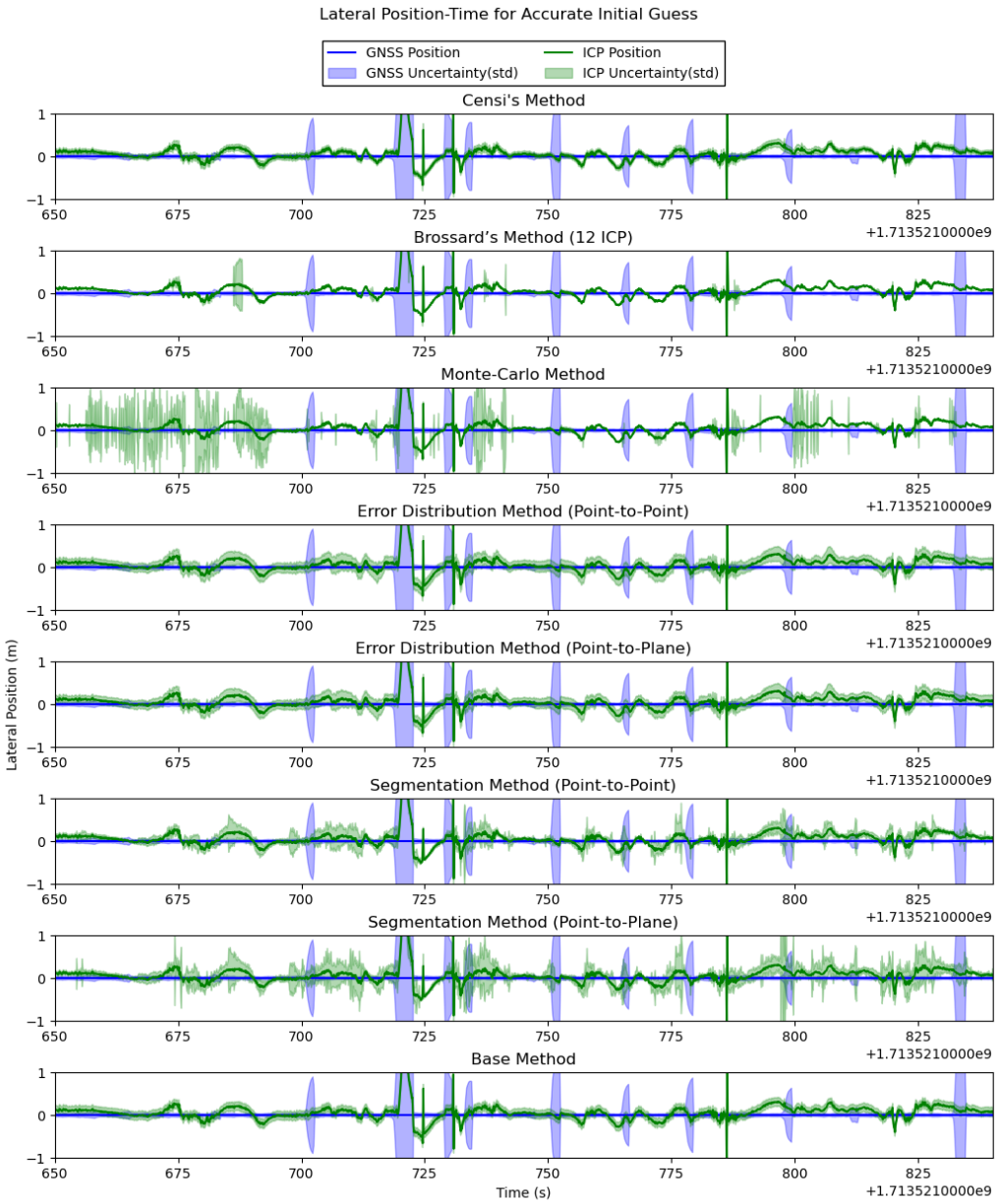


Figure 5.1.: Time vs. lateral position plot under sensor noise conditions

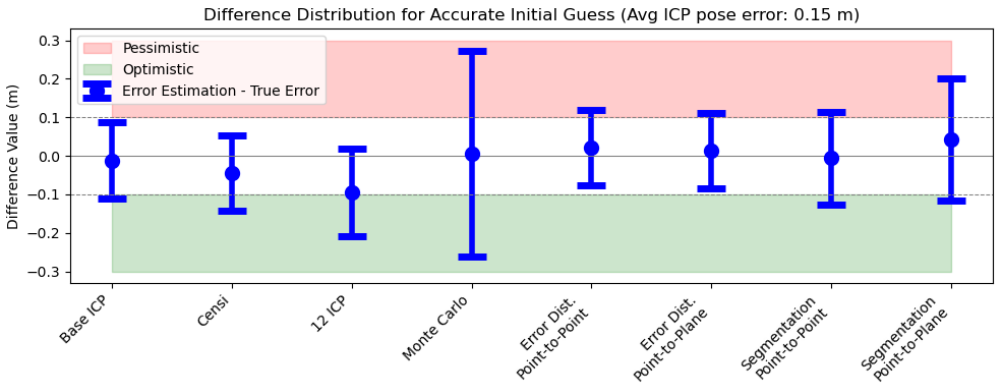


Figure 5.2.: Distributions of the differences between true error and error estimation under sensor noise conditions (visualization of Table 5.1)

nor pessimism overall), with standard deviations around 10 cm.

However, in the visual inspection of the methods in Figure 5.1, the results from the Base and Censi, and both Error Distribution methods appear nearly constant and small.

In Table 5.2 and in Figure 5.3, this situation was further investigated by analyzing the distributions of the uncertainty estimation values corresponding to true position errors, which have an average of around 11 cm and a standard deviation of 10 cm (the true position errors are low but not constant). The results of the Base, Censi, and both Error Distribution methods have standard deviations of less than 3 cm, indicating low changes in the uncertainty estimates. Their performance may be constrained by the small ICP pose errors, where nearly constant and small uncertainty is a reasonable estimation. Their performance in less accurate ICP positions is evaluated at the end of this Subsection and Subsection 5.2.2.

Monte Carlo and Brossard (12 ICP) Methods show considerable instability in the results, as shown in Figure 5.1. The Brossard (12 ICP) Method tends to provide overly optimistic estimations, rarely producing pessimistic results. On average, the method underestimates the true error by approximately 9 cm, as indicated by the mean difference between the true error and the estimation, which is around -9 cm.

In contrast, the Monte Carlo Method has a smaller mean difference, less than 1 cm, indicating it estimates error values closer to the true error on average. However, its performance is not reliable, showing both overly pessimistic and optimistic values. The

Table 5.2.: Distribution values of true errors and error estimations in lateral direction under sensor noise conditions

Method	**Estimation mean (m)**	**Estimation std (m)**	**True Error mean (m)**	**True Error std (m)**
Base Method	0.1000	0.0004	0.1123	0.0997
Censi Method	0.0678	0.0133	0.1125	0.1003
Brossard (12 ICP) Method	0.0176	0.0595	0.1126	0.1011
Monte Carlo	0.1171	0.2444	0.1126	0.1011
Error Dist. Point-to-Point	0.1343	0.0283	0.1122	0.1001
Error Dist. Point-to-Plane	0.1258	0.0274	0.1122	0.1001
Segmentation Point-to-Point	0.1057	0.0813	0.1112	0.0978
Segmentation Point-to-Plane	0.1551	0.1405	0.1112	0.0978

standard deviation of the differences between the true errors and the estimations for Monte Carlo is around 26 cm, indicating the method's significant instability in error estimation.

In Figure 5.1, segmentation methods appear to provide more sensitivity to the true error values. Comparing the segmentation methods' results with the base method in Figure 5.4, it is observed that the base method results (highlighted in red) remain nearly constant, whereas the segmentation methods show more variation, with estimations tending to increase or decrease based on the true error, particularly in the regions marked by red rectangles.

The uncertainty estimation values are examined for different true error regions in Figure 5.5 for a more detailed quantitative analysis. As previously discussed, the Base, Censi, and both Error Distribution methods provide nearly constant error estimation results across the different error regions. In contrast, the segmentation methods exhibit a weak correlation between the true error and the corresponding estimation values.

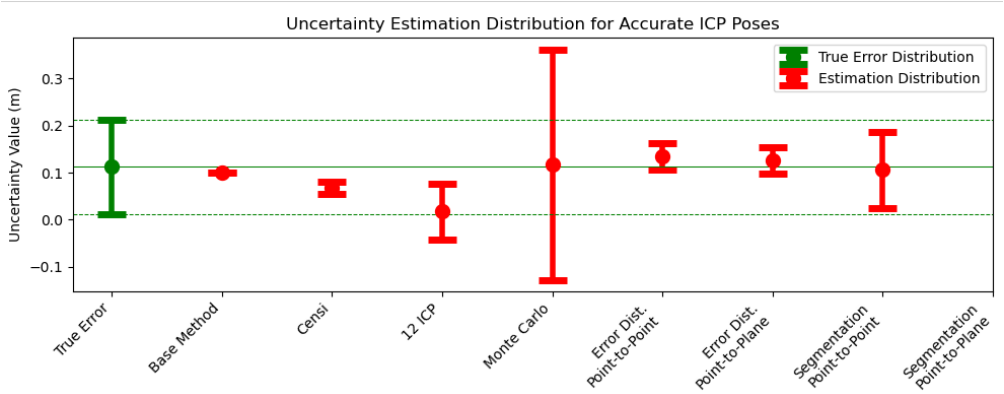


Figure 5.3.: Distributions of true errors and error estimations in lateral direction under sensor noise conditions (visualization of Table 5.2)

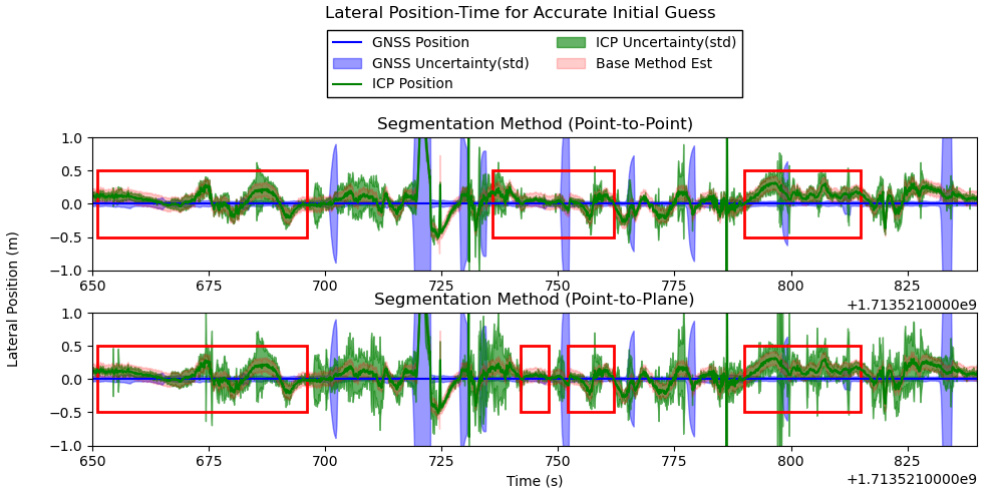


Figure 5.4.: Time-lateral position plot for segmentation methods (closer view to Figure 5.1)

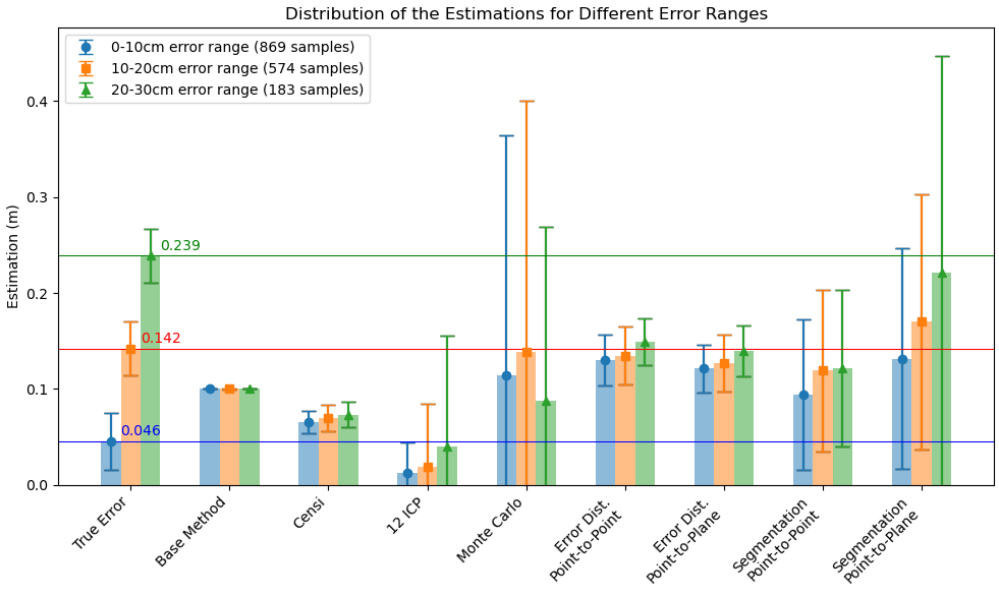


Figure 5.5.: Distributions of the true error and error estimations for different ICP error ranges under sensor noise conditions

5.2.2. Wrong Convergence

To evaluate the covariance methods in cases of wrong convergence (high ICP pose error), an open-loop test was performed using the set of inaccurate initial guesses described in Subsection 3.2.2, and it causes wrong convergences as described in Subsection 3.4.3 and high ICP pose errors. The true error and the estimated uncertainty values are compared in the lateral direction in Figure 5.6.

As discussed in Subsection 5.2.1, the Base method, Censi method, and both Error Distribution methods consistently provide nearly constant and small error estimation results, even for high ICP errors around 2 meters. These methods cannot capture the wrong convergences in the region highlighted by the red rectangle in Figure 5.6. On the other hand, the Monte Carlo and Brossard (12 ICP) Methods successfully capture these wrong results, estimating high uncertainty around the magnitudes of true error values while maintaining small uncertainty for correct convergences. Both segmentation methods perform similarly to the Monte Carlo and Brossard (12 ICP) methods but miss the first incorrect convergence in the region highlighted by the red rectangle.

In the quantitative results in Table 5.3 and Figure 5.7, the mean values of the differ-

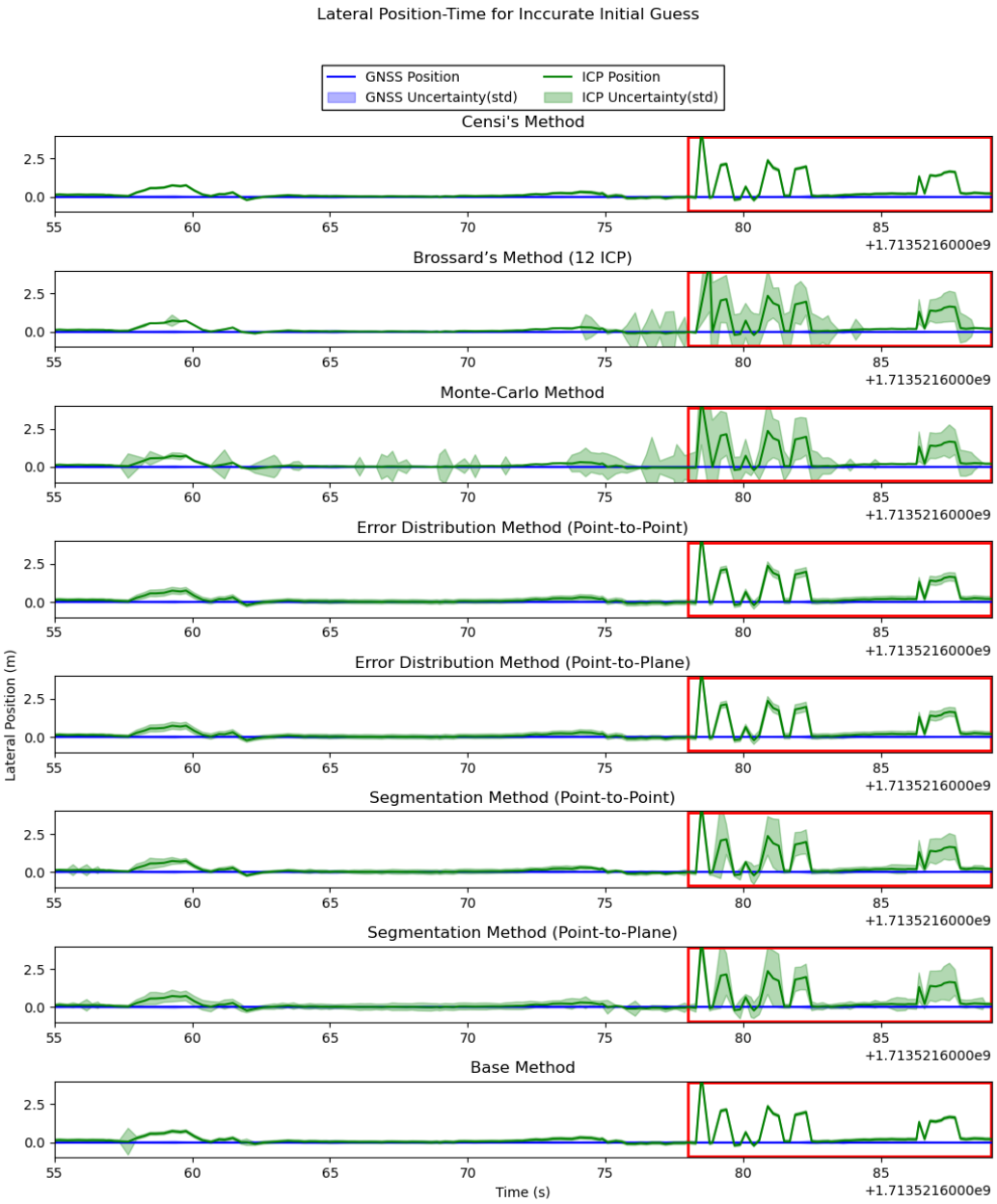


Figure 5.6.: Time-lateral position plot under wrong convergence

Table 5.3.: Numerical results in lateral direction under wrong convergence

Method	**Average ICP Error (m)**	**Difference Mean (m)**	**Difference Std (m)**	**NNE**
Base Method	0.6725	-0.2033	0.6032	6.7254
Censi Method	0.6755	-0.2474	0.5917	7.9508
Brossard (12 ICP) Method	0.6889	0.0199	0.5451	50.2262
Monte Carlo	0.6911	0.1064	0.5315	34.7497
Error Dist. Point-to-Point	0.6758	-0.1503	0.5703	3.1817
Error Dist. Point-to-Plane	0.6758	-0.1479	0.5668	3.2398
Segmentation Point-to-Point	0.6751	-0.0810	0.4230	1.7879
Segmentation Point-to-Plane	0.6751	-0.0008	0.3987	1.1715

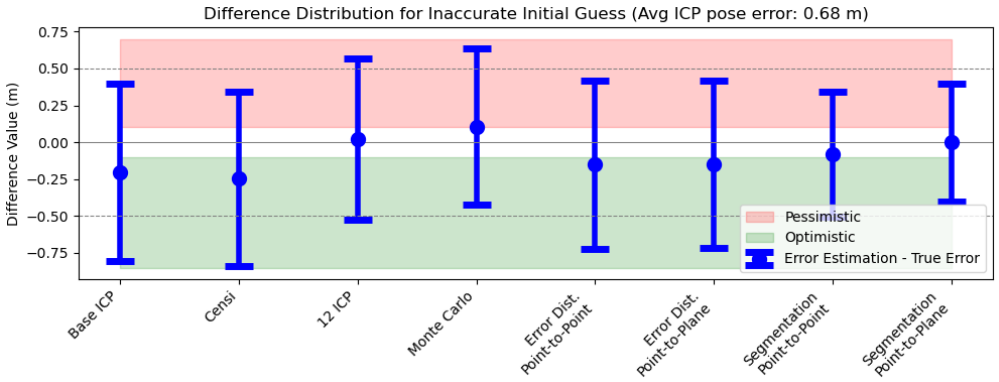
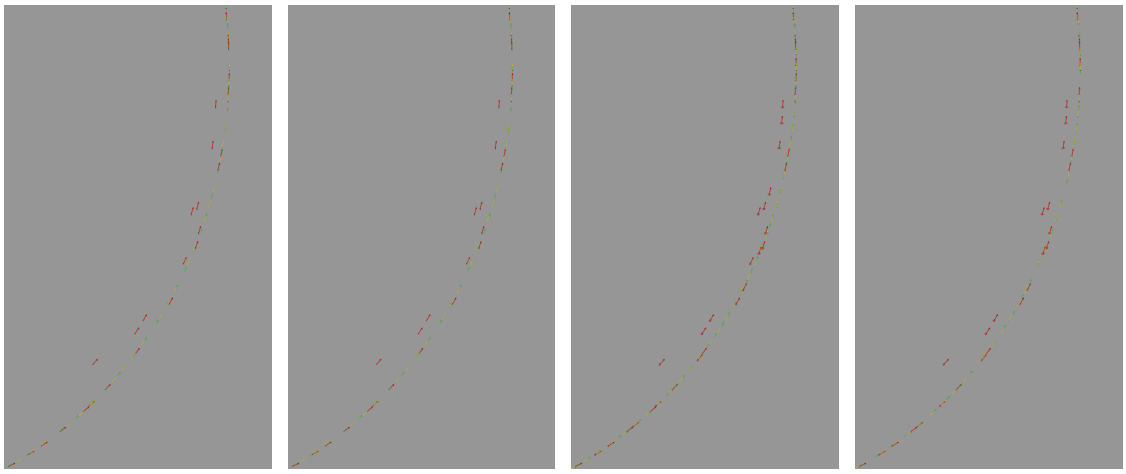


Figure 5.7.: Distributions of the differences between true error and error estimation under wrong convergence (visualization of Table 5.3)

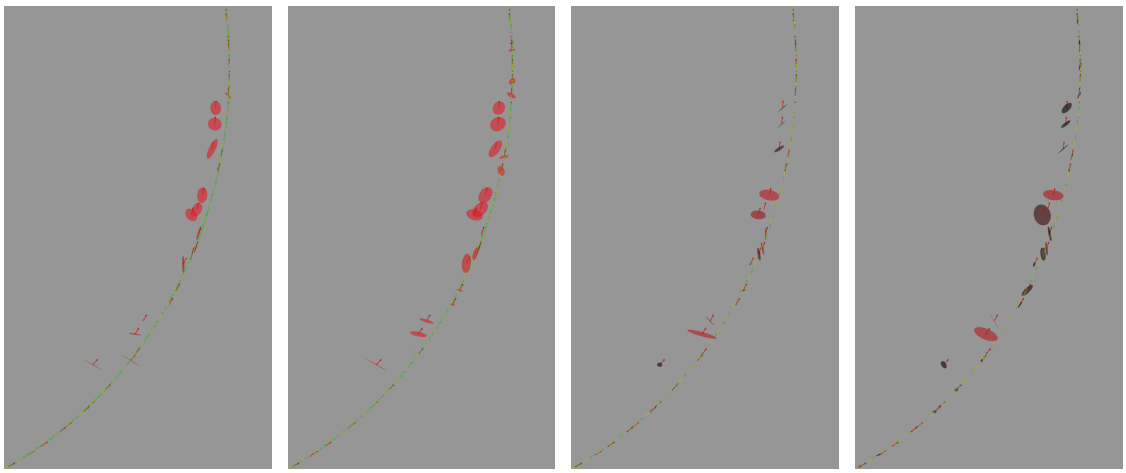
ence between true error and estimation for nearly constant methods (the Base, Censi, and Error Distribution methods) are negative (between -14 and -25 cm), indicating that their estimations are significantly lower than the true error, making them optimistic. In contrast, the mean differences for the other methods (Monte Carlo, Brossard (12 ICP), and Segmentation methods) are closer to zero, reflecting more accurate estimations.

Furthermore, covariance values for the inaccurate ICP poses, highlighted in the red regions in Figure 5.6, are visualized in Figures 5.8 and 5.9. In the 2D views, the red arrows and ellipses indicate the ICP poses and their associated covariances, while the yellow arrows represent the GNSS poses.



(a) Base Method (b) Censi Method (c) Error Dist. Method point-to-point (d) Error Dist. Method point-to-plane

Figure 5.8.: Covariance results in wrong convergence for nearly constant methods



(a) Brossard (12 ICP) Method

(b) Monte Carlo Method

(c) Segmentation Method point-to-point

(d) Segmentation Method point-to-plane

Figure 5.9.: Covariance results in wrong convergence for successful methods

5.2.3. Under-Constrained Environments

In the experiments, a distance offset between the GNSS (true) poses, and ICP poses was observed in the longitudinal direction due to a time synchronization issue between the GNSS and LiDAR sensors. Since ground truth positions in the longitudinal direction are unavailable due to this issue, the magnitude of uncertainty values is inspected in Figure 5.10 instead of comparing them to position error for the uncertainty due to under-constrained parts of the map, as discussed in Section 3.4.2.

In the in Figure 5.10, long straights are highlighted with red rectangles, and ideally, the under-constrained sections of these straights should exhibit high uncertainty. However, it is important to note that these straights may not necessarily be under-constrained due to features like billboards, bridges, and other structures. For instance, between timestamps 740 and 757, there is a long straight where the only under-constrained section is between timestamps 752 and 756. Features such as the billboard, along with other structures, reduce uncertainty, keeping it low as long as they remain visible; otherwise, the uncertainty increases.

The nearly constant methods (The Base, Censi, and Error Distribution methods) provide small and consistent uncertainty estimations in the longitudinal direction throughout the entire lap, where some parts are fully constrained, and others are under-

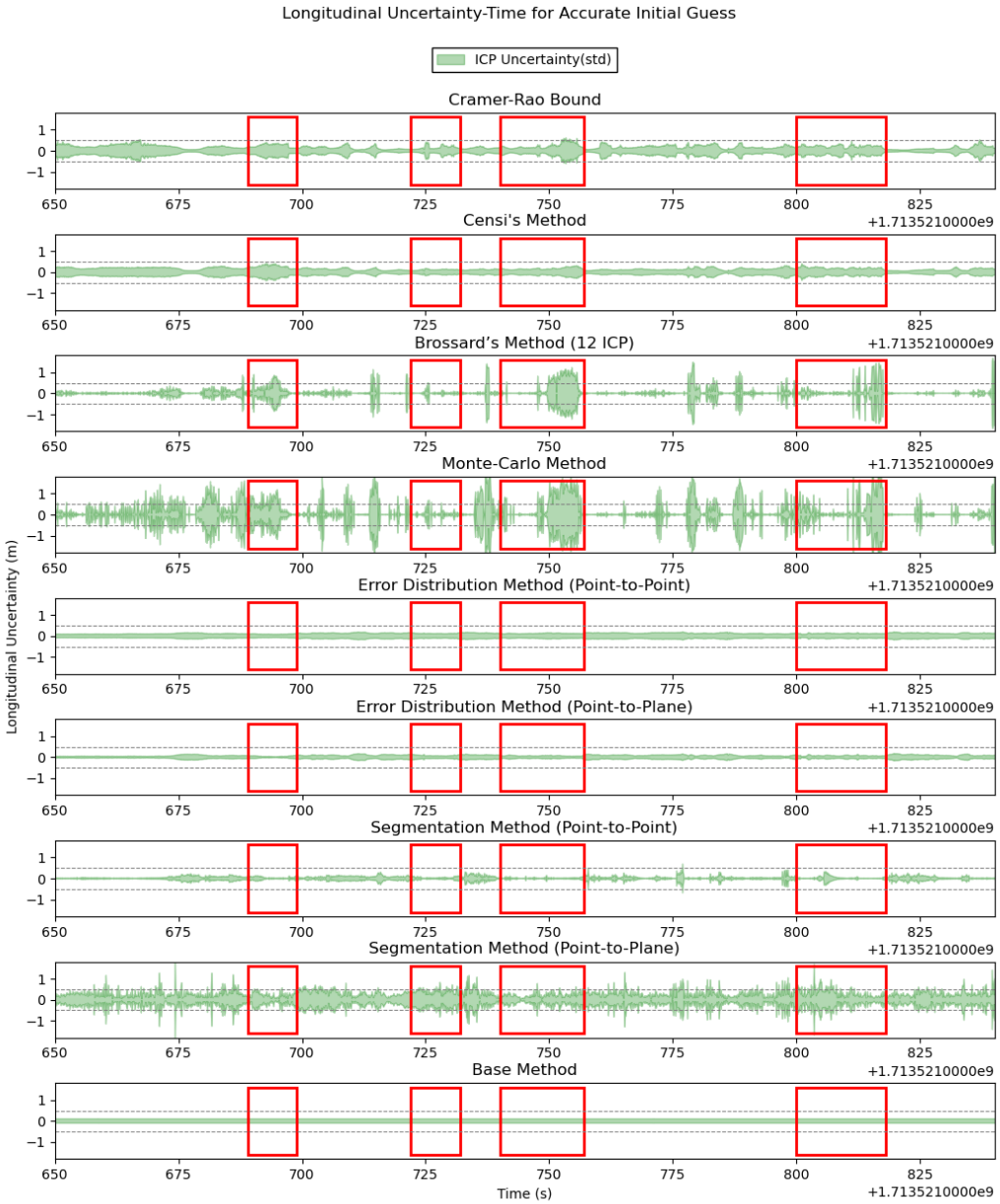
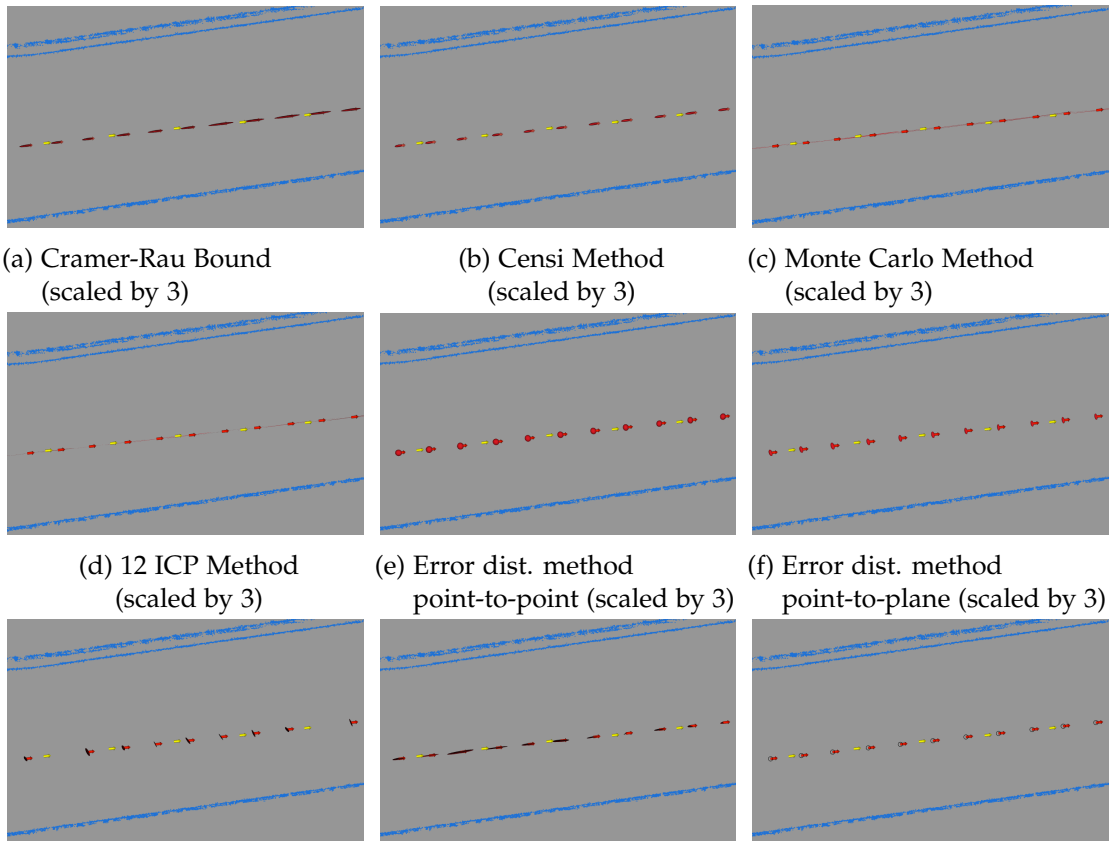


Figure 5.10.: Uncertainty estimations in longitudinal direction and long straights on the map (red rectangles)

constrained, similar to the results in the lateral direction discussed in the previous section. The Monte Carlo and Brossard (12 ICP) methods estimate high uncertainty in the red regions. The Segmentation method with the point-to-point also provides small and constant uncertainty values. However, the Segmentation method with the point-to-plane estimates high uncertaint in the red regions and other areas. For this type of uncertainty, the Cramer-Rao lower bound is also evaluated. While the lower bound for uncertainty values is generally not high, it tends to increase in the red regions.

Another important aspect of evaluation is the shape of the covariance matrix. Regardless of the magnitude, the covariance matrix should ideally reflect the shape of the environment, with higher uncertainty in the under-constrained directions if they exist.

In Figure 5.11, the shapes of the covariance matrices in the under-constrained environment for each method are visualized. The red arrows and ellipses denote the ICP poses and their covariances, scaled by a factor of 3 or 10, while the yellow arrows represent the GNSS poses. The Cramer-Rao lower bound, Censi, Monte Carlo, Brossard (12 ICP), and Segmentation with point-to-plane methods can capture the under-constrained direction. Longitudinal uncertainty along the road is significantly higher than the lateral uncertainty, with the covariance alignment exactly along the longitudinal direction. However, the covariance matrices of the Segmentation method with point-to-point and Error Distribution method with point-to-plane are aligned along the lateral direction. On the other hand, the Error Distribution method with point-to-point and the Base method's covariance matrices are invariant to the environment.



(a) Cramer-Rau Bound (scaled by 3)
(b) Censi Method (scaled by 3)
(c) Monte Carlo Method (scaled by 3)
(d) 12 ICP Method (scaled by 3)
(e) Error dist. method point-to-point (scaled by 3)
(f) Error dist. method point-to-plane (scaled by 3)
(g) Segmentation method point-to-point (scaled by 10)
(h) Segmentation method point-to-plane (scaled by 3)
(i) Base Method (scaled by 3)

Figure 5.11.: Covariance shapes in under-constrained straight (timestamp 751)

5.3. Closed Loop Tests

After comparing the uncertainty estimation values to the true error, this section evaluates their impact on the state estimation process using the first lap of the dataset perception_bag_202404191210, between 640 and 840 seconds, described in Section 3.2. Trajectory results are analyzed for different initial pose guesses and modalities, simulating scenarios with GNSS dropouts.

Due to the time synchronization problem with the GNSS sensor, trajectory error values become high when the GNSS poses are used as the ground truth poses, as described in Section 3.2.1. To overcome this issue, the EKF results utilizing only GNSS as an extero-

ceptive sensor, (without LiDAR and RADAR-based localization) are used as the ground truth poses. Furthermore, the sections containing inaccurate GNSS signals are excluded.

Furthermore, all diagnostics messages and ICP validity checks, rejecting results with problems - such as not converged ICP, poses that are behind the last state estimation pose, or far away - are excluded to observe the performance of the methods in cases involving wrong ICP poses. Ideally, these errors should be detected by the covariance methods.

5.3.1. Accurate ICP Initial Guess and 1/10 GNSS

In the first modality, 9 out of 10 GNSS messages are excluded, and initial ICP guesses are not perturbed to evaluate the effects of accurate ICP results in the state estimation.

In the 2D view figures, the yellow arrows represent GNSS poses used in the state estimation, while the blue arrows indicate GNSS poses not used but included for the accuracy evaluation. The green arrows correspond to the state estimation results, and the ICP poses are illustrated by red arrows, with their corresponding uncertainty matrices shown as red ellipses. This color convention is consistently used throughout the closed-loop results.

In Figures 5.12, 5.13 and 5.14, the ICP poses are accurate and ICP covariance matrices are small. All methods complete the route with small root mean square errors (RMSE). Among all methods, the Base and Censi methods and the Segmentation method with point-to-point show the lowest RMSE values, as shown in Table 5.4.

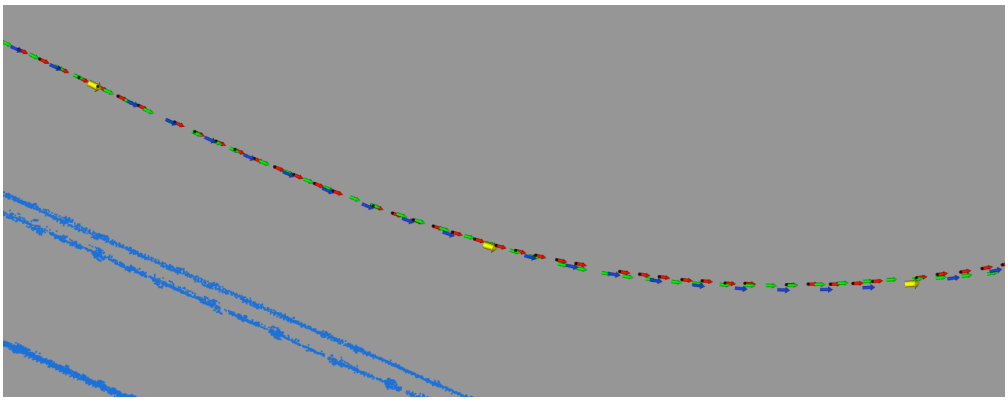


Figure 5.12.: GNSS, ICP and EKF poses with Base method

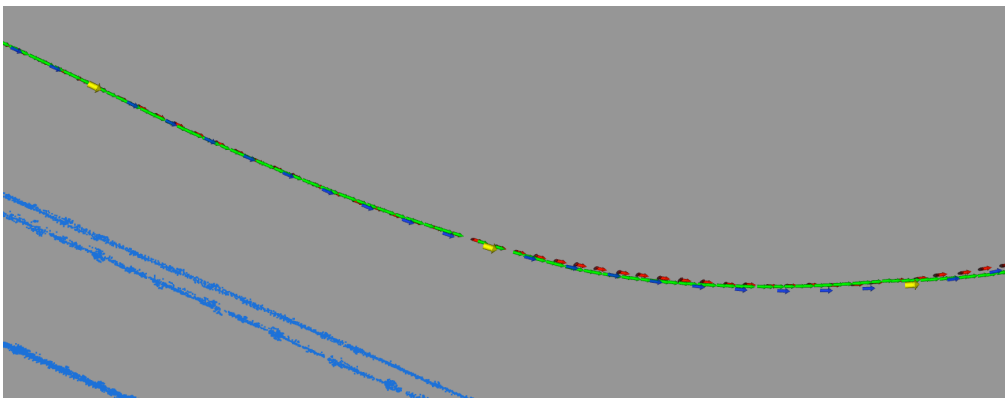


Figure 5.13.: GNSS, ICP and EKF poses with Segmentation method (point-to-point)

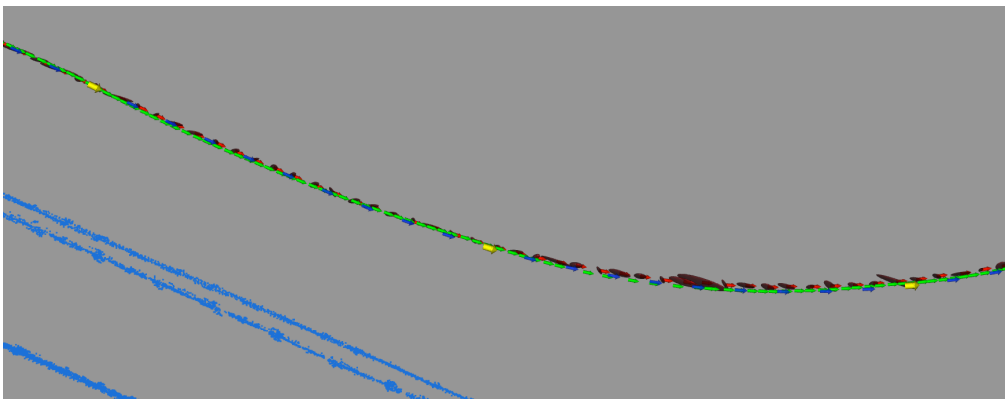


Figure 5.14.: GNSS, ICP and EKF poses with Segmentation method (point-to-plane)

Table 5.4.: Trajectory error results without perturbing ICP initial guesses and 1/10 GNSS messages

	RMSE (m)	Mean (m)	Median (m)	Std (m)	Min (m)	Max (m)	SSE (m)
Base method	0.477	0.369	0.286	0.302	0.001	2.136	3766.210
Censi method	0.479	0.369	0.285	0.305	0.000	1.892	3795.554
Error Dist. point-to-plane	0.684	0.456	0.311	0.509	0.002	4.067	7739.504
Error Dist. point-to-point	0.528	0.409	0.318	0.333	0.001	2.098	4611.308
Segmentation point-to-plane	0.727	0.527	0.374	0.500	0.002	3.417	8746.330
Segmentation point-to-point	0.486	0.377	0.289	0.307	0.003	1.790	3913.556

5.3.2. Accurate ICP Initial Guess and without GNSS

In the second modality, initial ICP guesses are not perturbed, similar to the previous modality, but the state estimation is performed without any GNSS messages. Despite the absence of GNSS poses, three methods, the Base and both Segmentation methods, complete the route with RMSE results similar to those in the first modality (Figure 5.15, 5.16, 5.17, and Table 5.5).

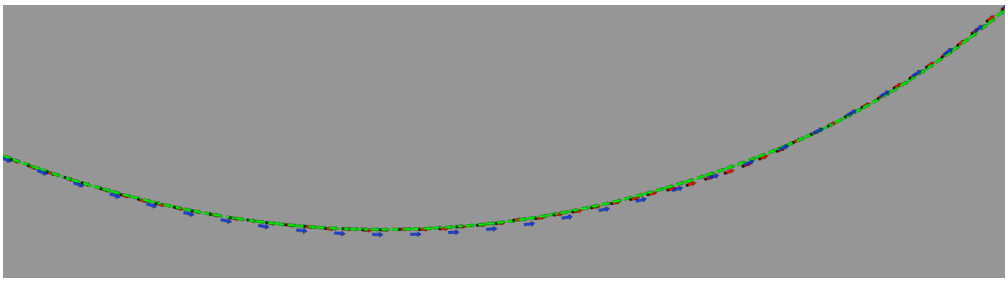


Figure 5.15.: GNSS, ICP and EKF poses with Base method

5.3.3. Inccurate ICP Initial Guess and Full GNSS

In this modality, the initial pose guesses are disturbed by a fixed pose offset ($\delta x = 0.5$ m, $\delta y = 0.5$ m, $\delta \psi = 0.05$ rad) to achieve inaccurate ICP results. These inaccurate ICP poses are used to observe the effects of the wrong ICP poses and the corresponding uncertainty values estimated by three methods, the Base and both Segmentation methods, on the state estimation. To ensure the stability of the state estimation, full GNSS

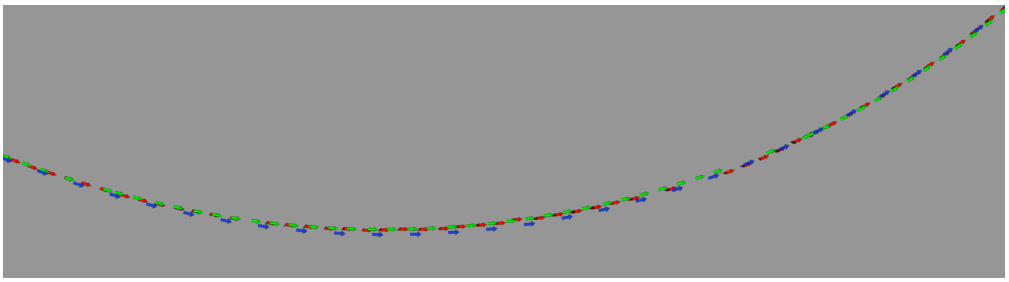


Figure 5.16.: GNSS, ICP and EKF poses with Segmentation method (point-to-point)

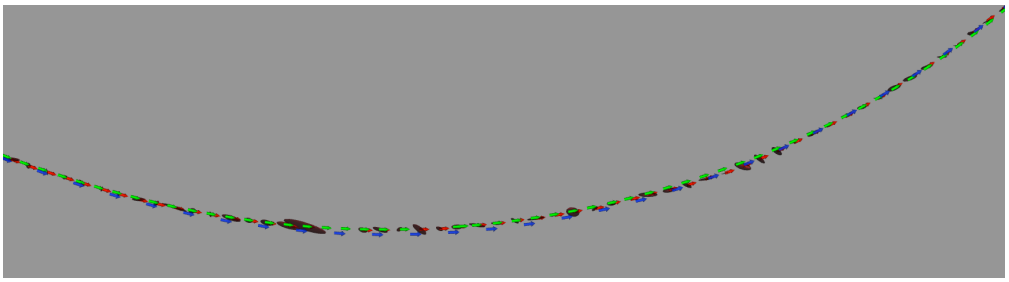


Figure 5.17.: GNSS, ICP and EKF poses with Segmentation method (point-to-plane)

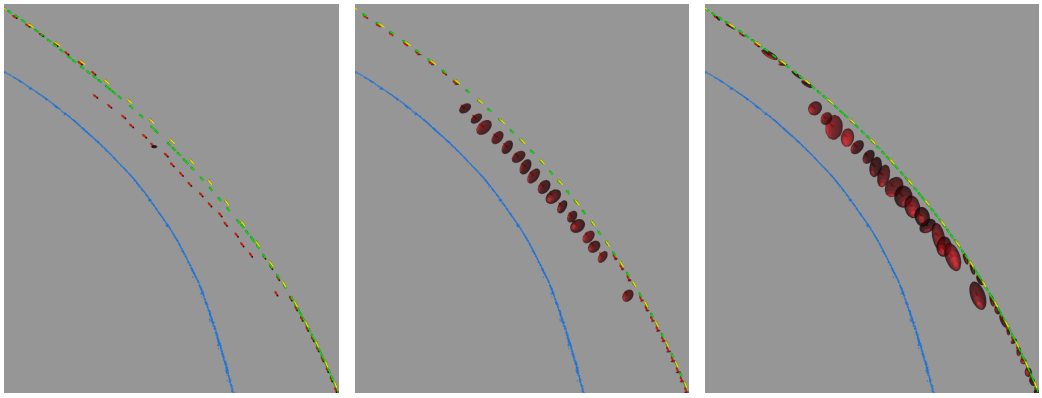
Table 5.5.: Trajectory error results without perturbing ICP initial guesses and without GNSS messages

	RMSE (m)	Mean (m)	Median (m)	Std (m)	Min (m)	Max (m)	SSE (m)
Base method	0.493	0.403	0.338	0.283	0.002	1.946	4018.277
Segmentation point-to-plane	0.646	0.511	0.410	0.396	0.002	2.833	6915.636
Segmentation point-to-point	0.694	0.474	0.343	0.507	0.001	4.012	7982.720

messages are provided in this modality.

While the Segmentation methods can capture wrong convergences and provide high uncertainties for them (Figures 5.18b and 5.18c), the Base method produces nearly constant and small uncertainty values, and fails to detect wrong ICP poses 5.18a.

Despite the inaccurate ICP poses and over-optimistic uncertainty values in the Base method, all three methods completed the route with lower RMSE results compared to the previous modalities (Table 5.6). The trajectory generated by the Base method exhibits more errors, as the state estimation is affected more by incorrect ICP poses with low uncertainty. In Figure 5.19, the trajectories produced by the Segmentation methods closely align with the reference trajectory (GNSS), even though the ICP poses are highly inaccurate. On the other hand, the trajectory for the Base method diverges slightly from the reference and converges toward the ICP poses due to the over-optimistic ICP uncertainty values.



(a) With Base method (b) With Segmentation method (point-to-point) (c) With Segmentation method (point-to-plane)

Figure 5.18.: GNSS, ICP, EKF poses and ICP covariances

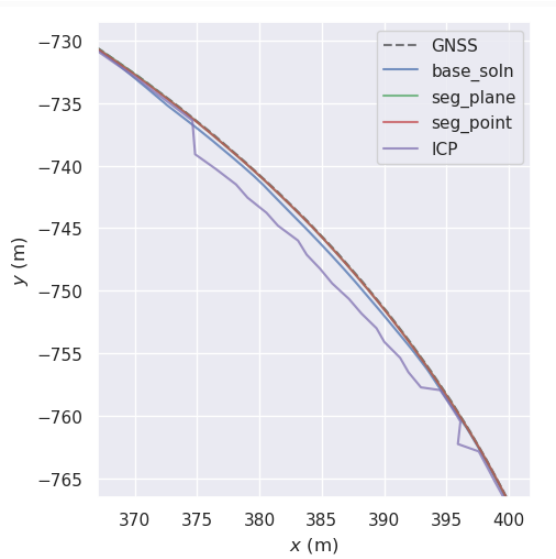


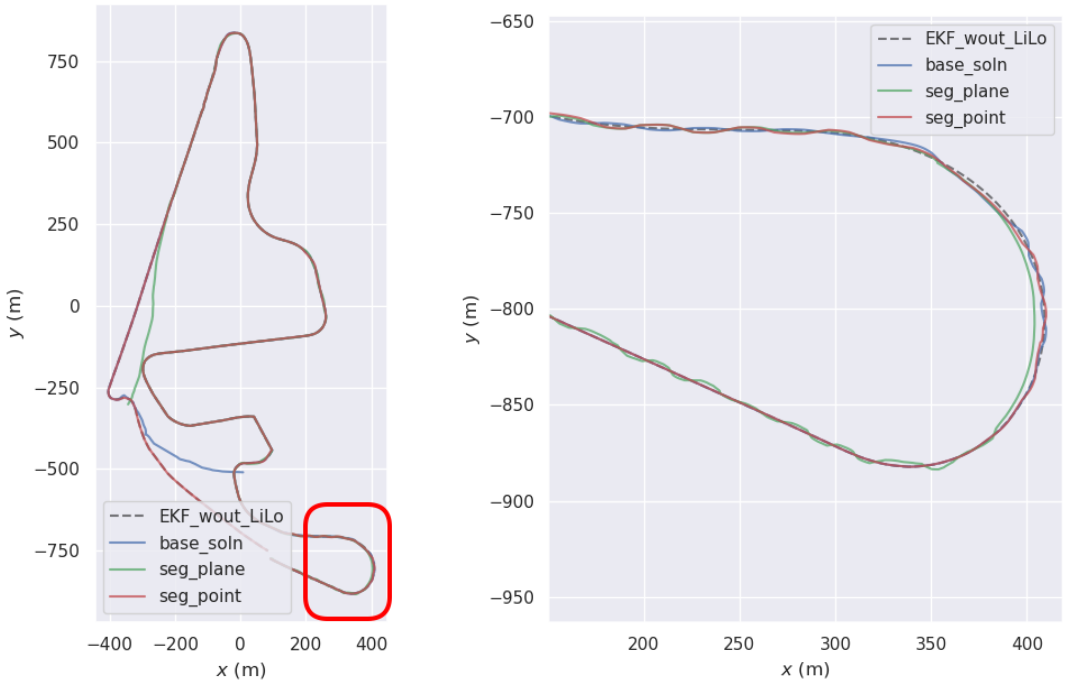
Figure 5.19.: Closer view of GNSS, ICP and EKF trajectories for different methods

Table 5.6.: Trajectory error results with perturbing ICP initial guesses and with full GNSS messages

	RMSE (m)	Mean (m)	Median (m)	Std (m)	Min (m)	Max (m)	SSE (m)
Base method	0.359	0.257	0.178	0.251	0.000	2.025	2133.349
Seg. method plane	0.329	0.235	0.159	0.230	0.002	1.960	1790.647
Seg. method point	0.342	0.241	0.162	0.242	0.000	1.762	1934.416

5.3.4. Inaccurate ICP Initial Guess and 1/5 GNSS

In the last modality, to increase the impact of the ICP on the state estimation, 4 out of 5 GNSS messages were excluded. As in the previous modality, the initial guesses were disturbed using the same pose offset ($\delta x = 0.5$ m, $\delta y = 0.5$ m, $\delta \psi = 0.05$ rad), resulting in wrong ICP poses. In these experiments, both the Base method and the Segmentation method with point-to-plane failed to complete the route, while the Segmentation method with point-to-point completed it (Figure 5.20a).

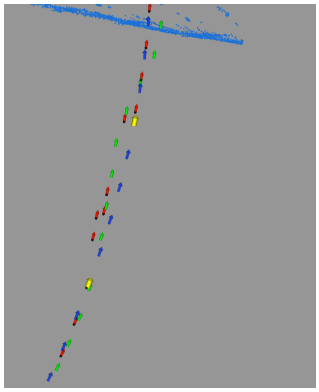


(a) Trajectories

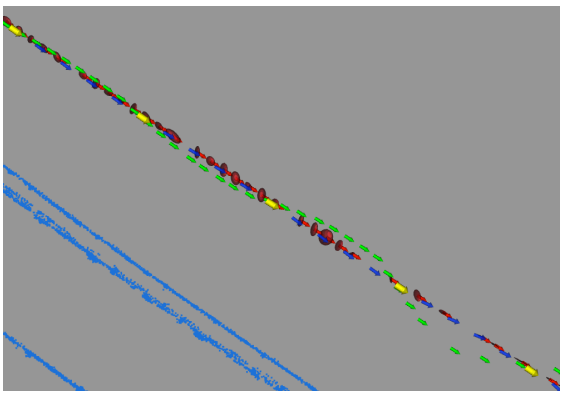
(b) Closer view to red region

Figure 5.20.: Reference and EKF trajectories for different methods

However, when the trajectories are investigated visually in Figure 5.20b, oscillations are observed in all trajectories. The main reasons for these unstable results may be inaccurate ICP poses with low uncertainty (Figure 5.21a) and accurate ICP poses with high uncertainty (Figure 5.21b).



(a) Oscillations with the Base method (inaccurate pose with low uncertainty)



(b) Oscillations with the Segmentation method (point-to-plane) (accurate pose with high uncertainty)

Figure 5.21.: Oscillations in the state estimations

For the Segmentation method with point-to-point, ICP poses errors, associated uncertainty values, and their effects on the trajectory were analyzed using the time-lateral position plot (Figure 5.22) and the 2D visualization (Figure 5.23). In the red regions, inaccurate ICP poses with low uncertainty values are observed, and they start the first oscillation in the trajectory. Between these red regions, there are accurate ICP poses with low uncertainty values. However, these poses cannot suppress the oscillations.

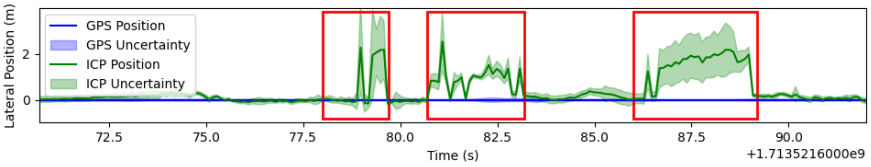


Figure 5.22.: Time-lateral position plot for the Segmentation method with point-to-point

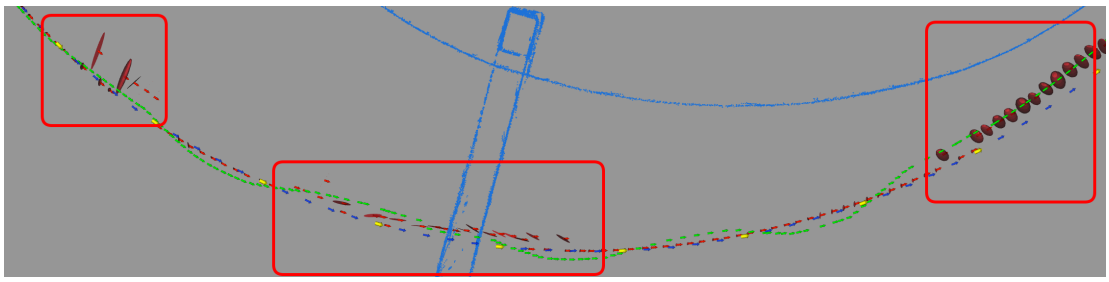


Figure 5.23.: GNSS, ICP, EKF poses and ICP covariances

5.4. ICP Error and Estimation in Rotation

True ICP errors and uncertainty estimations in rotation were examined in the experiments, but the results (Figures 5.24 and 5.25) were not realistic, and the underlying issue could not be identified.

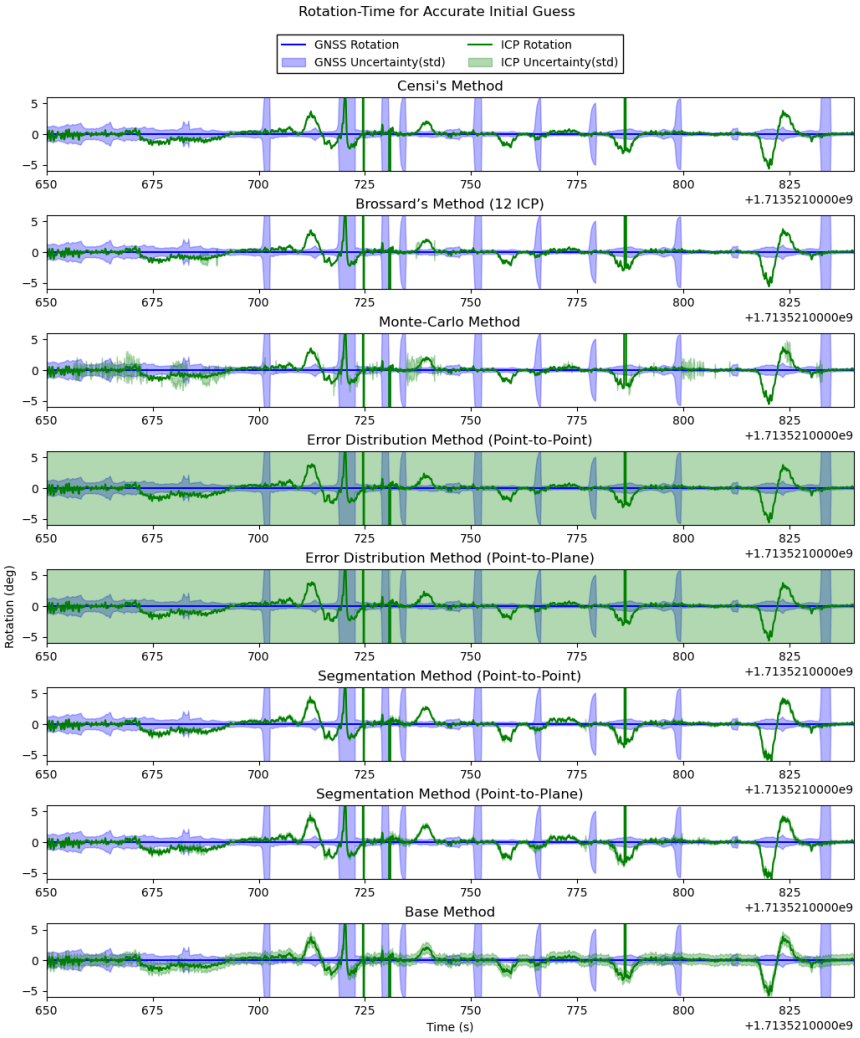


Figure 5.24.: Time-rotation plot under sensor noise conditions

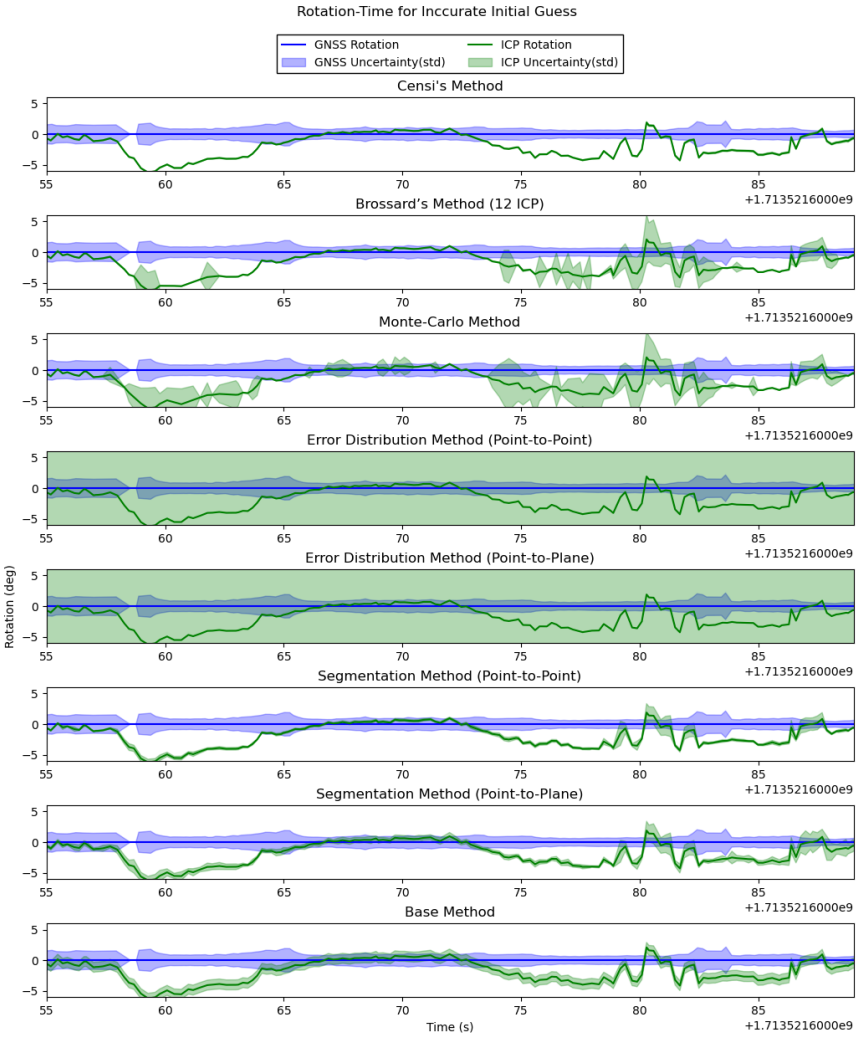


Figure 5.25.: Time-rotation plot under under-constrained environment

5.5. Runtime

Since ICP is used in real-time systems, particularly in autonomous race cars, the runtime of the covariance calculation process is critical. The Brossard (12 ICP) method requires approximately 250 milliseconds, as it runs 6 ICP processes to estimate the covariance. Other methods rely on the source and reference point pairs (data association) to estimate uncertainty, which is the most time-consuming part of the process. There are two different implementations for data association. The first is to get the closest 3D map points for methods using point-to-point error, while the second one is to get the closest 3D map points along with the corresponding normal vector for methods using point-to-plane error. To calculate the total runtime for covariance estimation, the runtime of the respective data association process must be included. The Table 5.7 lists each method's runtime values in microseconds, based on 748 point pairs for 773 scan points.

Method	**Runtime (microseconds)**
Brossard (12 ICP) Method	255586
Data Association (only map point)	544
Data Association (map point and normal)	1111
Base Method*	29
Cramer-Rau Lower Bound†	28
Censi Method†	133
Error Distribution (Point-to-Point) Method*	18
Error Distribution (Point-to-Plane) Method†	22
Segmentation (Point-to-Point) Method*	107
Segmentation (Point-to-Plane) Method†	67

Table 5.7.: Runtime of methods in microseconds
*: using point-to-point error
†: using point-to-plane error

5.6. Methods Summary

In the scenarios with slightly inaccurate ICP poses (average error of 11 cm) caused by sensor noise in constrained environments without wrong convergence, the Monte Carlo and Brossard (12 ICP) methods produce unstable results, with either overly optimistic or overly pessimistic uncertainty estimates (Figure 5.1). The Base, Censi, and Error Distribution methods provide nearly constant and small error estimation values (Figure

5.3). Segmentation methods estimate small uncertainty values, but they show slight variation corresponding to the magnitude of the true error (Figure 5.5).

In cases where the ICP error is high (around 2 meters) due to wrong convergence (Figure 5.6), the methods capable of detecting these incorrect convergences are Monte Carlo, Brossard (12 ICP), and Segmentation methods (Figure 5.9). In contrast, other methods fail to detect them (Figure 5.8). However, there is no guarantee that the Segmentation methods will detect incorrect convergences in all instances, and they may miss or underestimate these high errors (Figures 5.23 and 5.22).

The uncertainty caused by under-constrained environments is not accounted for by the nearly constant methods (the base, Censi, and Error Distribution methods) or the Segmentation method with the point-to-point error (Figure 5.10). In contrast, the Monte Carlo and Brossard (12 ICP) methods produce high uncertainty estimates in these conditions. The Segmentation method with point-to-plane error also estimates high uncertainty values, but not only in under-constrained regions—this method exhibits high uncertainty even in constrained environments. The Cramer-Rao lower bound tends to increase in under-constrained environments, though not as significantly as the Monte Carlo and Brossard (12 ICP) methods.

Except for the Monte Carlo and Brossard (12 ICP) methods, all methods are feasible for real-time systems due to their low runtime (approximately 1 ms).

Table 5.8.: Comparison of different methods based on the results

Method	**Covariance Estimation in Accurate ICP Poses**	**Considers Sensor Noise**	**Captures Under Constrained Directions**	**Captures Wrong Convergence**	**Covariance Matrix Shape (Captures the Env.)**	**Runtime**
Base Method	moderate	+[1]	-	-	-	**low**
Cramer-Rao	N/A	N/A	+[3]	N/A	**+**	**low**
Censi	moderate	+[1]	+[3]	-	**+**	**low**
Monte Carlo	unstable	+[2]	**+**	**+**	**+**	high
12 ICP	optimistic	+[2]	**+**	**+**	**+**	high
Error Dist. point-to-point	moderate	+[1]	-	-	-	**low**
Error Dist. point-to-plane	moderate	+[1]	-	-	+[4]	**low**
Segmentation point-to-point	moderate	**+**	-	**+**	-	**low**
Segmentation point-to-plane	moderate	**+**	-	**+**	-	**low**

N/A: not applicable
1: nearly constant uncertainty estimation
2: considers sensor noise but either over optimistic or over pessimistic
3: captures but optimistic
4: captures but the alignment is in the opposite direction

6. Discussion

This chapter discusses the methods' performances in terms of accuracy in different uncertainty sources, run-time, and feasibility. Later, the effects of the Base and Segmentation methods on the state estimation and the stability are discussed. Finally, further investigations and improvements are discussed.

6.1. Base Method

The results show that the base method provides a constant covariance estimation. This is because the transformation result of the additional iteration in the covariance calculation is the identity matrix if the ICP pose estimate converges to a local or global minimum. If convergence has yet to occur, e.g. ICP process is terminated by the time out, the covariance estimation may become larger.

It performs well in the cases where ICP poses are accurate. However, the covariance estimations are unrealistic as they tend to provide constant and small uncertainty values. It underestimates the errors for the inaccurate ICP results and under-constrained environments.

6.2. Censi Method

The Censi method theoretically considers the sensor noise and the uncertainty due to under-constrained environments. However, it underestimates the error in the results using real-world data. It gives over-optimistic uncertainty values.

6.3. Monte Carlo and Brossard (12 ICP) Methods

The two methods are fundamentally similar, as both assess the stability of the ICP process by perturbing the initial guess and analyzing the distribution of the resulting ICP outputs. However, they differ in the number of ICP processes and the nature of the initial guess perturbations. In the Brossard Method, the number of ICP processes is

fixed at 12 for 3D cases (6 for 2D cases), whereas the Monte Carlo method does not prescribe a specific number, though it generally exceeds 12. In Brossard's work [46], the number was set to 65, while in my experiments, I selected 20. Additionally, the Monte Carlo method perturbs the initial guess randomly based on the initial guess uncertainty, while the Brossard method employs fixed perturbations, also determined by the initial guess uncertainty.

In these experiments, both Monte Carlo and Brossard (12 ICP) Methods perform well in detecting wrong convergence and handling under-constrained environments. However, they tend to estimate over-optimistic or over-pessimistic uncertainty values for slightly inaccurate ICP results. The performance of these methods depends heavily on the initial guess uncertainty and a realistic perturbation of the initial guess. The initial guess is derived from the state estimation stack, but associated uncertainty values are too small. In these methods, small perturbations to the initial guess can lead to over-optimistic uncertainty estimations. To address this, the initial guess was perturbed by 94 cm in the Brossard method. In contrast, the Monte Carlo method was perturbed by a value drawn from a uniform distribution between -1 and 1 meter in the lateral direction. These values may be too large for experiments with highly accurate initial guesses. Therefore, selecting appropriate parameters is critical for obtaining realistic results.

Another important point is that these methods compute the stability of ICP around the initial guess rather than estimating the error of the ICP pose result. Low uncertainty indicates that the ICP processes for different initial guesses converge to the same pose. In contrast, high uncertainty indicates that they converge to different poses.

The Brossard method requires approximately 250 ms for 6 ICP runs, while the Monte Carlo method takes even more. As a result, they are not feasible for the real-time systems.

6.4. Error Distribution Methods

Like the Base and Censi methods, Error Distribution methods estimate nearly constant and small uncertainty values. These methods evaluate the alignment of the scan to the map by considering the errors between map and source point pairs. In the experiments with inaccurate initial guesses, there are wrong convergence cases where the alignment is poor, as shown in Figure 3.14b; however, the estimated uncertainty values remain

small. This may be attributed to the weighting kernel, which reduces the influence of large error values between points.

Regarding covariance shapes, the covariance matrices for the point-to-point method tend to be circular, even in under-constrained environments (Figure 5.11f). On the other hand, the covariance matrices for the point-to-plane method are typically elliptical, with alignment in the lateral direction. This occurs because the residuals between point pairs are projected onto the normal vectors of the map points, leading to small uncertainty values in under-constrained directions, as shown in Figure 5.10.

6.5. Segmentation Methods

Segmentation methods perform well with slightly inaccurate ICP results (Figure 5.5), similar to nearly constant methods (the Base, Censi, and Error Distribution method). However, segmentation methods exhibit more variation in response to changes in error.

In cases of inaccurate ICP results due to incorrect convergence, segmentation methods are the only ones capable of detecting these errors and remain feasible for real-time systems (Figures 5.6 and 5.9). However, it may miss some wrong convergences or provide less uncertainty than the magnitude of the error (Figures 5.22) and 5.23).

Although both methods provide similar uncertainty estimations in the lateral direction (Figures 5.5 and 5.9), the method using point-to-point error produces small and nearly constant uncertainty values in the longitudinal direction, whereas the method using point-to-plane error results in higher uncertainty values (Figures 5.10 and 5.11). The reason for the higher uncertainty in the longitudinal direction for the point-to-plane method was investigated, but a solid explanation could not be identified.

6.6. Effects on the State Estimation

With accurate initial guesses, the trajectories with the lowest RMSE values are achieved by the Base method in both the cases with dropped GNSS messages and without GNSS messages. When accurate initial guesses are provided to ICP, the pose results are precise (with an average error of around 11 cm, as shown in Table 5.1). In such cases, using constant and small uncertainty values makes sense. Additionally, in the absence of GNSS, where the only measurements come from ICP poses, small uncertainty values

help keep the state estimation close to the ICP poses, preventing divergence. The second-best method is another nearly constant method, the Censi method. For accurate ICP poses, using small uncertainty values leads to better trajectory results.

When the GNSS messages are completely excluded, the error (RMSE) values increase for the Base and the Segmentation method with point-to-point. However, when the segmentation method with point-to-plane is utilized, the trajectory error decreases. This is because the low-frequency GNSS updates cause oscillations in the state estimation result. In Figure 6.1, the ICP poses (red arrows) are accurate and have small uncertainty. However, the state estimation results (green arrows) fail to follow the ICP poses, leading to divergence in the absence of GNSS messages. When a GNSS message (yellow arrows) is received, the state estimation result converges toward the true poses (GNSS poses - blue arrows represent dropped GNSS poses and not used by state estimation). This issue may be related to the EKF parameters and how the pose sources and corresponding uncertainty values are incorporated into the state estimation.

It should be kept in the mind that EKF adapts the covariance matrices before the state estimation as described in Section 3.1

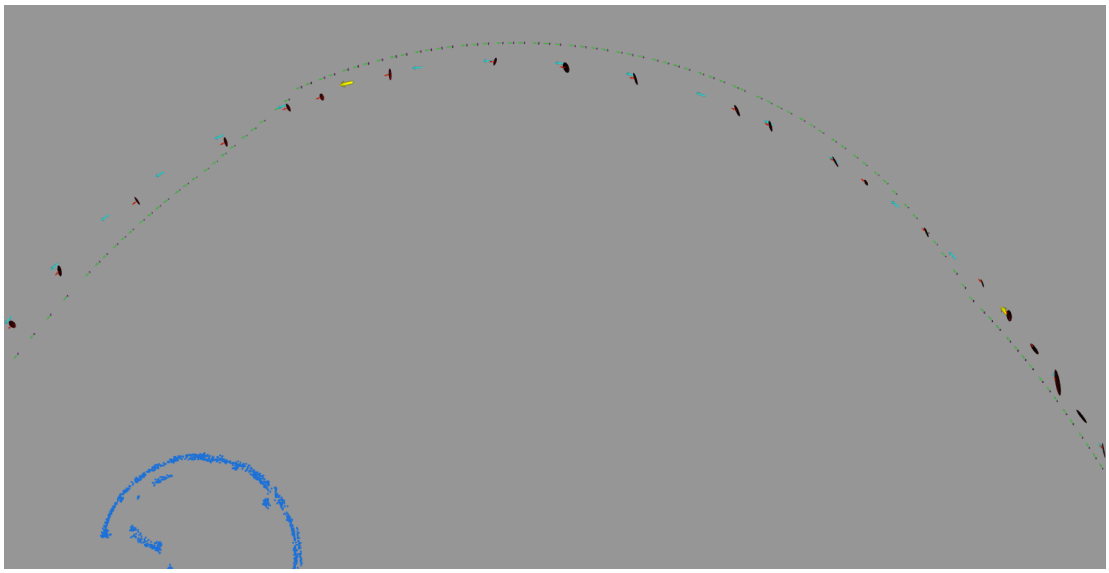


Figure 6.1.: Oscillation problem in low-frequency GNSS messages

In the third modality, where all GNSS messages and inaccurate ICP poses are utilized in the state estimation, all trajectories for different ICP covariance methods have fewer

errors than the error values in the first two modalities. It can be explained as that uncertainty values for GNSS are less than the ICP uncertainty values, and GNSS poses are used more in the state estimation. Furthermore, in the third modality, the trajectory with the Base method has more error than the trajectories with Segmentation methods, and the reason is that Segmentation methods provide high uncertainty values for the inaccurate ICP poses and the impact of these wrong poses in the final estimation is much less. The Segmentation methods outperform the Base method in this modality.

In the last modality, where only one out of every five GNSS messages was used alongside inaccurate ICP poses, only the Segmentation method with point-to-point was able to complete the route, while the trajectories from the Base method and the Segmentation method with point-to-plane diverged from the race track in the later parts. However, it does not necessarily indicate that the Segmentation method with point-to-point outperformed others in this modality, as oscillations were present in all trajectories, and these oscillations caused the divergence (Figure 5.20). These oscillations appear to have resulted from inaccurate ICP poses with low uncertainty (Figure 5.21a) and accurate ICP poses with high uncertainty (Figure 5.21b).

Similarly, in the first modality, where ICP poses were only slightly inaccurate, and uncertainty estimates were low, oscillations were observed (Figure 6.1). Other possible causes of these oscillations could be the handling of the intermittent reception of GNSS messages in the state estimation process or issues related to time synchronization between GNSS and other sensors.

6.7. Further Improvements and Works

For slightly inaccurate ICP poses, which is common in the dataset, the Error Distribution methods can be effectively utilized. In the experiments, the results were nearly constant, but more realistic outcomes for the alignment of source points to the map can potentially be achieved with smaller weighting kernels. Additionally, the magnitude of the sensor noise estimates could be subtracted from the residuals between the source and map points.

Regarding the covariance calculation with segments, further investigation and refinement of the map segments could be beneficial, and new segmentation strategies could be developed. In the segmentation process (Section 3.2.6, the threshold parameters for smoothness and curvature were not strict, leading to cases where a single segment consists of two perpendicular planes or fails to separate two parallel planes, as shown

in the red rectangles in Figure 6.2.

Figure 6.2.: Segments on the map

Error Distribution and Segmentation methods cannot estimate the uncertainty arising from under-constrained environments, as they primarily focus on evaluating the alignment process. For more realistic uncertainty estimates, these error estimates could be combined with the Cramer-Rao bound value, as discussed in Section 4.3. However, this strategy could not be properly tested in the experiments due to the lack of ground truth poses in the longitudinal direction. This approach could be further investigated and tested.

Another critical issue is the error estimation in the rotation. Realistic values could not be obtained in the experiments, and the methods could not be properly evaluated for rotation. This issue could be investigated in future work to enable the use of the entire covariance matrix in state estimation.

For the ICP covariance effects on EKF and the stability of the state estimation, the reasons could be determined by investigating the covariance estimation values by the methods and actual covariance values used in the state estimation after the covariance adaptation process. After that, EKF parameters could be tuned for better and more stable state estimations.

In the open-loop tests, GNSS poses were selected as the ground truth. However, there is a time synchronization issue with GNSS. In the closed-loop tests, state estimation results using GNSS, but without Lidar localization, were selected as the ground truth to overcome the time synchronization issue between GNSS and EKF poses. Finally, these

methods could be tested with different datasets, and ground truth poses without time synchronization issues.

A. Appendix

A.1. Implementation Details

The covariance methods methods listed below were implemented in icp-uncertainty-new branch of the scanmatcher repository (CovarianceCalcuation.hpp and Covariance-Calculation.cpp files) and these implementations are used for the experiments. In this chapter, important implementation details are explained.

Repository link: https://gitlab.lrz.de/sensor-fusion-slam/scanmatcher/-/tree/icp-uncertainty-new

Commit ID: `0d0ad902086bb42c04b34ba0870c69ac7d28355a`

- Cramer-Rao bound
- Censi method
- Brossard Method (12 ICP method)
- Monte Carlo Method
- Error distribution method (point-to-point)
- Error distribution method (point-to-plane)
- Segmentation method (point-to-point)
- Segmentation method (point-to-plane)

A.1.1. Covariance Conversion between Map and Vehicle Frames

In the dataset, both the point cloud map and the point cloud source (LiDAR scan) are defined in the map frame. If these covariance methods are applied directly without any transformation, the covariance of the map origin relative to the map frame will be calculated. However, the objective is to compute the covariance of the vehicle relative

to the map frame. Except for Monte-Carlo and Brossard methods, this problem can be easily solved by assuming the ICP pose result is correct and subtracting the 3D vehicle position (ICP pose) from each map and source point prior to covariance calculations. In this case, the vehicle is centralized, and the covariance matrix is computed for the vehicle and defined in the map frame.

However, since Monte-Carlo and Brossard methods require multiple ICP runs and points cannot be changed before the ICP process, pose results are transformed into the vehicle frame after the ICP process. In Algorithm 3, $\xi^j_{\text{icp}} = \exp^{-1}(\hat{\text{T}}^{-1}_{\text{icp}}\hat{\text{T}}^j_{\text{icp}})$ is defined in the map frame. Firstly, this transformation, representing the difference between ICP poses, $\hat{\text{T}}^{-1}_{\text{icp}}\hat{\text{T}}^j_{\text{icp}}$ is transformed into vehicle frame, then logarithmic map $\exp^{-1}(.)$ is applied.

$$\xi^j_{\text{icp}} = \exp^{-1}(\text{T}^{\text{map}}_{\text{vehicle}}{}^{-1}\ \hat{\text{T}}^{-1}_{\text{icp}}\hat{\text{T}}^j_{\text{icp}}\ \text{T}^{\text{map}}_{\text{vehicle}}) \tag{A.1}$$

where $\text{T}^{\text{map}}_{\text{vehicle}}$ is the transformation from vehicle to map.

With this change, the final result in Algorithm 3, $\frac{1}{12}\sum_{j=1}^{12}\xi^j_{\text{icp}}(\xi^j_{\text{icp}})^T$, becomes the covariance of the vehicle defined in the vehicle frame (Figure A.1). The magnitude of the covariance is correct, however the orientation is wrong. For instance, the first element in the covariance matrix is the variance in the longitudinal direction, rather than in the X-axis of the map.

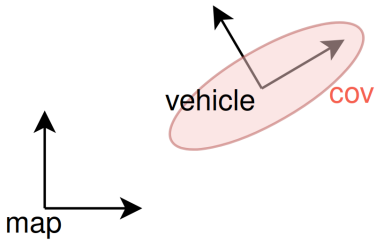


Figure A.1.: Covariance matrix

The covariance matrix defined in the vehicle frame is rotated using

$$T_{\text{cov}} = \begin{bmatrix} R^{\text{map}}_{\text{vehicle}} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & R^{\text{map}}_{\text{vehicle}} \end{bmatrix} \tag{A.2}$$

where $R^{\text{map}}_{\text{vehicle}} \in \text{SO}(3)$ is the rotation from vehicle to map. The covariance matrix of

the vehicle defined in the map frame is:

$$\mathrm{COV}_{\mathrm{icp,map}} = T_{\mathrm{cov}}\ \mathrm{COV}_{\mathrm{icp,vehicle}}\ T_{\mathrm{cov}}^{T} \tag{A.3}$$

Another conversion is necessary to introduce noise into the initial ICP guess to generate new estimates for the Monte Carlo and Brossard methods. $\mathbf{Q}_{\mathrm{ini}}$ in Algorithm 3 represents the uncertainty of the initial ICP guess. Ideally, this initial uncertainty should be derived from the odometry stack, specifically the uncertainty of the EKF result. However, since these uncertainties were small, the initial guess uncertainty is instead parameterized in the longitudinal, lateral, and rotational directions, $\mathbf{Q}_{\mathrm{ini,vehicle}}$. Firstly, a new initial guess, $\mathrm{T}_{\mathrm{ini,vehicle}}$, is determined in the vehicle frame using $\mathbf{Q}_{\mathrm{ini,vehicle}}$. Then this initial guess, defined in the vehicle frame, is transformed into the map frame.

$$\mathrm{T}_{\mathrm{ini,map}} = \mathrm{T}_{\mathrm{vehicle}}^{\mathrm{map}}\ \mathrm{T}_{\mathrm{ini,vehicle}}\ \mathrm{T}_{\mathrm{vehicle}}^{\mathrm{map}\ -1} \tag{A.4}$$

A.1.2. Method Parameters

In Brossard Method, initial covariance matrix in vehicle frame is defined as

$$\mathbf{Q}_{\mathrm{ini,vehicle}} = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.15 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.01 \end{bmatrix}$$

and sigma points become:

$$\xi_1 = \pm \begin{bmatrix} 1.732 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \xi_2 = \pm \begin{bmatrix} 0 \\ 0.948 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \xi_3 = \pm \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.245 \end{bmatrix}$$

In the paper [46], for a full 6 degrees of freedom, 12 sigma points are generated for the three translational and three rotational parameters. However, in the 2D case, sigma points are only generated for translation in the X and Y directions, and for rotation around the Z-axis. As a result, 6 ICP runs are performed instead of 12.

In the Monte Carlo method (Algorithm 8), 20 ICP runs are performed, and the initial guess is generated randomly from a uniform distribution between $\pm \begin{bmatrix}2.5 & 1.0 & 0 & 0 & 0 & 0.1\end{bmatrix}^T$.

Algorithm 8 ICP Uncertainty Calculation with Monte Carlo for Localization

Input: Point cloud scan S, point cloud map M, initial guess $\mathrm{T}_{\mathrm{ini}}$, ICP pose result $\hat{\mathrm{T}}_{\mathrm{icp}} = \mathrm{ICP}(\mathcal{M}, \mathrm{T}_{\mathrm{ini}}, \mathcal{S})$, $\mathbf{Q}_{\mathrm{ini,vehicle}}$
Output: Covariance matrix

for each Monte Carlo iteration k (N=20 times) **do**
  Get initial guess $\xi_{\mathrm{ini,vehicle,k}}$ from $\mathbf{Q}_{\mathrm{ini,vehicle}}$
  $\mathrm{T}_{\mathrm{ini,vehicle,k}} = \exp(\xi_{\mathrm{ini,vehicle,k}})$
  Transform initial guess to map frame $\mathrm{T}_{\mathrm{ini,map,k}} = \hat{\mathrm{T}}_{\mathrm{icp}} \mathrm{T}_{\mathrm{ini,vehicle,k}} \hat{\mathrm{T}}_{\mathrm{icp}}^{-1}$
  Run ICP: $\hat{\mathrm{T}}_{\mathrm{icp,k}} = \mathrm{ICP}(\mathcal{M}, \mathrm{T}_{\mathrm{ini,map,k}} \mathrm{T}_{\mathrm{ini}}, \mathcal{S})$
  ICP pose difference: $\xi_{\mathrm{icp,vehicle,k}} = \exp^{-1}(\hat{\mathrm{T}}_{\mathrm{icp}}^{-1} \hat{\mathrm{T}}_{\mathrm{icp}}^{-1} \hat{\mathrm{T}}_{\mathrm{icp,k}} \hat{\mathrm{T}}_{\mathrm{icp}})$
end for
Covariance matrix in vehicle frame: $\mathrm{COV}_{\mathrm{vehicle}} = \frac{1}{N} \sum_{k=1}^{N} \xi_{\mathrm{icp,vehicle,k}} (\xi_{\mathrm{icp,vehicle,k}})^T$
Transform covariance matrix into map frame using Equation A.3.

In the Error Distribution methods, Equation 4.13 is applied, with the same kernel from ICP for the weighting.

In the segmentation methods, Algorithm 7 and Equation 4.15 are applied, using the same ICP kernel for weighting. A threshold of 50 points is set as the minimum required to consider a segment valid.

A.2. Experiment Setup

In the open-loop tests, the related ICP covariance method implemented in CovarianceCalculation.cpp file is enabled in the Registration.cpp file and the Lidar-based localization is run. Test commands are listed below:

- Start LiDAR-based localization

```
$ ros2 launch iac_kiss_icp LiLo.launch
map_path:=/../berkay_media/DATA2/thesis/maps/AbuDhabi.pcd
```

- Play rosbag data

```
$ ros2 bag play perception_bag_202404191210/ --clock --topics /tf
/core/perception/lidar_negative_track_crop_filtered /tf_static
/core/state/odometry /vectornav/raw/gps /vectornav/raw/common
/vectornav/raw/attitude /vehicle/sensor/virtual_heading
```

In the closed-loop tests, to run the whole localization system, iac_launch repository with commit ID `263549353a95ae54f5988853b388bd31ffd51f38` is used. A new docker compose file (replay_loc_wout_lilo_ralo.yml) is created to exclude LiDAR-based and RADAR-based localization systems and include a GNSS-dropout node.

- Start LiDAR-based localization

```
$ ros2 launch iac_kiss_icp LiLo.launch
map_path:=/../berkay_media/DATA2/thesis/maps/AbuDhabi.pcd
```

- Start other localization systems

```
$ docker compose -f replay_loc_wout_lilo_ralo.yml --env-file .env up
```

- Play rosbag data

```
$ ros2 bag play  perception_bag_202404191210/ --clock --remap
/tf_static:=/mull3 /tf:=/mull /core/state/odometry:=/mull2
/core/perception/lidar_filtered:=/mull4
/core/perception/lidar_ground_filtered:=/mull5
/core/perception/lidar_negative_track_crop_filtered:=/mull6
/debug/core/perception/LidarPreprocessing/signal_names:=/mull7
/debug/core/perception/LidarPreprocessing/values:=/mull8
/debug/core/perception/RadarPreprocessing/signal_names:=/mull9
/debug/core/perception/RadarPreprocessing/values:=/mull10
/debug/core/perception/LocalizationPreprocessing/signal_names:=/mull11
/debug/core/perception/LocalizationPreprocessing/values:=/mull12
/core/state/lidar_odometry:=/mull13
/core/state/radar_odometry:=/mull14
/debug/perception/loc_trackfilter_viz:=/mull15
/debug/perception/trackfilter_viz:=/mull16
/debug/perception/trackfilter_radar_viz:=/mull17
```

```
/core/perception/radar_filtered:=/mull18
/core/perception/radar_negative/track_filtered:=/mull19
```

- After the pos is initialized, kill the GNSS if necessary

```
$ docker compose -f replay_loc_wout_lilo_ralo.yml
--env-file .env down vectornav_gnss
```

List of Figures

List of Tables

Bibliography

[1] J. Deichmann, E. Ebel, K. Heineke, R. Heuss, M. Kellner, and F. Steiner. "Autonomous driving's future: Convenient and connected." Accessed: 19 September 2024. (2023), [Online]. Available: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/autonomous-drivings-future-convenient-and-connected#/.

[2] K. Houser. "The rise of the semi-autonomous car." Accessed: 19 September 2024. (2024), [Online]. Available: https://www.freethink.com/robots-ai/driving-automation.

[3] R. Limon. "Study finds self-driving cars are safer than human-driven vehicles." Accessed: 19 September 2024. (2024), [Online]. Available: https://english.elpais.com/technology/2024-06-18/study-finds-self-driving-cars-are-safer-than-human-driven-vehicles.html.

[4] "Partially autonomous cars forecast to comprise 10% of new vehicle sales by 2030." Accessed: 19 September 2024. (2024), [Online]. Available: https://www.goldmansachs.com/insights/articles/partially-autonomous-cars-forecast-to-comprise-10-percent-of-new-vehicle-sales-by-2030.

[5] D. Chiao, J. Deichmann, K. Heineke, A. Kelkar, M. Kellner, E. Scarinci, and D. Tolstinev. "Autonomous vehicles moving forward: Perspectives from industry leaders." Accessed: 19 September 2024. (2024), [Online]. Available: https://www.mckinsey.com/features/mckinsey-center-for-future-mobility/our-insights/autonomous-vehicles-moving-forward-perspectives-from-industry-leaders.

[6] "Indy autonomous challenge." Accessed: 19 September 2024. (2024), [Online]. Available: https://www.indyautonomouschallenge.com/.

[7] "The abu dhabi autonomous racing league." Accessed: 19 September 2024. (2024), [Online]. Available: https://a2rl.io/.

[8] M. Behl. "How autonomous racing is pushing self-driving cars forward." Accessed: 19 September 2024. (2024), [Online]. Available: https://www.inverse.com/science/self-driving-cars-ai-autonomous-racing.

[9] J. Betz, T. Betz, F. Fent, M. Geisslinger, A. Heilmeier, L. Hermansdorfer, T. Herrmann, S. Huch, P. Karle, M. Lienkamp, B. Lohmann, F. Nobis, L. Ögretmen, M. Rowold, F. Sauerbeck, T. Stahl, R. Trauth, F. Werner, and A. Wischnewski, "Tum autonomous motorsport: An autonomous racing software for the indy autonomous challenge," *Journal of Field Robotics*, vol. 40, no. 4, pp. 783–809, Jan. 2023, ISSN: 1556-4967. DOI: 10.1002/rob.22153.

[10] "Formula 1 could see driverless race cars as early as 2025." Accessed: 19 September 2024. (2022), [Online]. Available: https://www.mos.ed.tum.de/en/ftm/news/article/2025-koennten-fahrerlose-rennwagen-bei-der-formel-1-mitfahren/.

[11] "Tum autonomous motorsport." Accessed: 19 September 2024. (2024), [Online]. Available: https://www.mos.ed.tum.de/en/ftm/main-research/intelligent-vehicle-systems/tum-autonomous-motorsport/.

[12] M. Pielmeier, *Multi-LiDAR Pipeline for Robust Localization and Mapping of Autonomous Vehicles*. Master Thesis, Technical University of Munich, München, 2023.

[13] F. Sauerbeck, D. Kulmer, M. Pielmeier, M. Leitenstern, C. Weiß, and J. Betz, "Multi-lidar localization and mapping pipeline for urban autonomous driving," 2023. arXiv: 2311.01823 [cs.RO].

[14] A. Wischnewski, T. Stahl, J. Betz, and B. Lohmann, "Vehicle dynamics state estimation and localization for high performance race cars**research was supported by the basic research fund of the institute of automotive technology of the technical university of munich.," *IFAC-PapersOnLine*, vol. 52, no. 8, pp. 154–161, 2019, 10th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2019, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2019.08.064.

[15] S. Goblirsch, M. Weinmann, and J. Betz, "Three-dimensional vehicle dynamics state estimation for high-speed race cars under varying signal quality," 2024. arXiv: 2408.14885 [cs.RO].

[16] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss, "Kiss-icp: In defense of point-to-point icp – simple, accurate, and robust registration if done the right way," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 1029–1036, Feb. 2023, ISSN: 2377-3774. DOI: 10.1109/lra.2023.3236571.

[17] P. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992. DOI: 10.1109/34.121791.

[18] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, 1991, 2724–2729 vol.3. DOI: 10.1109/ROBOT.1991.132043.

[19] Z. Zhang, "Iterative point matching for registration of free-form curves," no. RR-1658, p. 42, 1992.

[20] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, pp. 119–152, 1994.

[21] E. Smistad, T. Falch, M. Bozorgi, A. Elster, and F. Lindseth, "Medical image segmentation on gpus - a comprehensive review," *Medical Image Analysis*, vol. 20, pp. 1–18, Feb. 2015. DOI: 10.1016/j.media.2014.10.012.

[22] F. Pomerleau, F. Colas, and R. Siegwart, "A review of point cloud registration algorithms for mobile robotics," *Foundations and Trends® in Robotics*, vol. 4, pp. 1–104, May 2015. DOI: 10.1561/2300000035.

[23] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke, "Registration with the point cloud library: A modular framework for aligning in 3-d," *IEEE Robotics Automation Magazine*, vol. 22, no. 4, pp. 110–124, 2015. DOI: 10.1109/MRA.2015.2432331.

[24] A. V. Segal, D. Hähnel, and S. Thrun, "Generalized-icp," in *Robotics: Science and Systems*, 2009.

[25] J. Konecny, M. Prauzek, and J. Hlavica, "Icp algorithm in mobile robot navigation: Analysis of computational demands in embedded solutions**this work was supported by the project sp2016/162, development of algorithms and systems for control, measurement and safety applications ii of student grant system, vsb-tu ostrava.," *IFAC-PapersOnLine*, vol. 49, no. 25, pp. 396–400, 2016, 14th IFAC Conference on Programmable Devices and Embedded Systems PDES 2016, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2016.12.079.

[26] D. Holz and S. Behnke, "Sancta simplicitas - on the efficiency and achievable results of slam using icp-based incremental registration," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 1380–1387. DOI: 10.1109/ROBOT.2010.5509918.

[27] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, 2001, pp. 145–152. DOI: 10.1109/IM.2001.924423.

[28] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing icp variants on real-world data sets," *Autonomous Robots*, pp. 133–148, Apr. 2013. DOI: 10.1007/s10514-013-9327-2.

[29] S. Fontana and D. Sorrenti, "A termination criterion for probabilistic pointclouds registration," Oct. 2020. DOI: 10.48550/arXiv.2010.04979.

[30] G. Sun, Y. Wang, L. Gu, and Z. Liu, "An improved icp algorithm for point cloud registration," in *2021 6th IEEE International Conference on Advanced Robotics and Mechatronics (ICARM)*, 2021, pp. 582–585. DOI: 10.1109/ICARM52023.2021.9536137.

[31] D. A. Simon, *Fast and Accurate Shape-Based Registration*. PhD Thesis, Carnegie Mellon University, USA, 1996.

[32] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sep. 1975, ISSN: 0001-0782. DOI: 10.1145/361002.361007.

[33] E. Vespa, N. Nikolov, M. Grimm, L. Nardi, P. H. J. Kelly, and S. Leutenegger, "Efficient octree-based volumetric slam supporting signed-distance and occupancy mapping," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1144–1151, 2018. DOI: 10.1109/LRA.2018.2792537.

[34] M. Zeng, F. Zhao, J. Zheng, and X. Liu, "Octree-based fusion for realtime 3d reconstruction," *Graphical Models*, vol. 75, no. 3, pp. 126–136, 2013, Computational Visual Media Conference 2012, ISSN: 1524-0703. DOI: https://doi.org/10.1016/j.gmod.2012.09.002.

[35] P. Dellenbach, J.-E. Deschaud, B. Jacquet, and F. Goulette, "Ct-icp: Real-time elastic lidar odometry with loop closure," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 5580–5586. DOI: 10.1109/ICRA46639.2022.9811849.

[36] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3d reconstruction at scale using voxel hashing," *ACM Trans. Graph.*, vol. 32, no. 6, Nov. 2013, ISSN: 0730-0301. DOI: 10.1145/2508363.2508374.

[37] D. W. Eggert, A. Lorusso, and R. B. Fisher, "Estimating 3-d rigid body transformations: A comparison of four major algorithms," *Machine Vision and Applications*, vol. 9, pp. 272–290, 1997.

[38] S. Bonnabel, M. Barczyk, and F. Goulette, "On the covariance of icp-based scan-matching techniques," in *2016 American Control Conference (ACC)*, 2016, pp. 5498–5503. DOI: 10.1109/ACC.2016.7526532.

[39] P. Glira, N. Pfeifer, C. Briese, and C. Ressl, "A correspondence framework for als strip adjustments based on variants of the icp algorithm," *Photogrammetrie - Fernerkundung - Geoinformation*, vol. 2015, Aug. 2015. DOI: 10.1127/pfg/2015/0270.

[40] K.-L. Low, "Linear least-squares optimization for point-to-plane icp surface registration," 2004.

[41] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "MeshLab: an Open-Source Mesh Processing Tool," in *Eurographics Italian Chapter Conference*, V. Scarano, R. D. Chiara, and U. Erra, Eds., The Eurographics Association, 2008, ISBN: 978-3-905673-68-5. DOI: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.

[42] F. Pomerleau, S. Magnenat, F. Colas, M. Liu, and R. Siegwart, "Tracking a depth camera: Parameter exploration for fast icp," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE Press, 2011, pp. 3824–3829.

[43] P. Glira, N. Pfeifer, C. Briese, and C. Ressl, "A correspondence framework for als strip adjustments based on variants of the icp algorithm," *Photogrammetrie-Fernerkundung-Geoinformation*, vol. 2015, no. 4, pp. 275–289, 2015.

[44] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China: IEEE, May 2011.

[45] A. Censi, "An accurate closed-form estimate of icp's covariance," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 3167–3172. DOI: 10.1109/ROBOT.2007.363961.

[46] M. Brossard, S. Bonnabel, and A. Barrau, "A new approach to 3d icp covariance estimation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 744–751, 2020. DOI: 10.1109/LRA.2020.2965391.

[47] A. de Maio and S. Lacroix, "Deep Bayesian ICP Covariance Estimation," in *IEEE International Conference on Robotics and Automation (ICRA 2022)*, Philadelphia, United States: IEEE, May 2022, pp. 6519–6525. DOI: 10.1109/ICRA46639.2022.9811899.

[48] Z. Feng, "An efficient initial guess for the icp method," *Pattern Recognition Letters*, vol. 125, pp. 721–726, 2019, ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2019.07.019.

[49] F. A. Maken, F. Ramos, and L. Ott, "Stein icp for uncertainty estimation in point cloud matching," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1063–1070, 2022. DOI: 10.1109/LRA.2021.3137503.

[50] D. Landry, F. Pomerleau, and P. Giguère, "Cello-3d: Estimating the covariance of icp in the real world," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8190–8196. DOI: 10.1109/ICRA.2019.8793516.

[51] F. Pomerleau, A. Breitenmoser, M. Liu, F. Colas, and R. Siegwart, "Noise characterization of depth sensors for surface inspections," Sep. 2012. DOI: 10.1109/CARPI.2012.6473358.

[52] Z. Wang, Y. Liu, Q. Liao, H. Ye, M. Liu, and L. Wang, "Characterization of a rs-lidar for 3d perception," in *2018 IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2018, pp. 564–569. DOI: 10.1109/CYBER.2018.8688235.

[53] J. Laconte, S.-P. Deschênes, M. Labussière, and F. Pomerleau, "Lidar measurement bias estimation via return waveform modelling in a context of 3d mapping," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8100–8106. DOI: 10.1109/ICRA.2019.8793671.

[54] J.-E. Deschaud, "Imls-slam: Scan-to-model matching based on 3d data," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2480–2485. DOI: 10.1109/ICRA.2018.8460653.

[55] T. M. Iversen, A. G. Buch, and D. Kraft, "Prediction of icp pose uncertainties using monte carlo simulation with synthetic depth images," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 4640–4647. DOI: 10.1109/IROS.2017.8206335.

[56] O. Bengtsson and A.-J. Baerveldt, "Robot localization based on scan-matching—estimating the covariance matrix for the idc algorithm," *Robotics and Autonomous Systems*, vol. 44, no. 1, pp. 29–40, 2003, Best Papers of the Eurobot '01 Workshop, ISSN: 0921-8890. DOI: https://doi.org/10.1016/S0921-8890(03)00008-3.

[57] S. M. Prakhya, L. Bingbing, Y. Rui, and W. Lin, "A closed-form estimate of 3d icp covariance," in *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*, 2015, pp. 526–529. DOI: 10.1109/MVA.2015.7153246.

[58] I. Torroba, C. I. Sprague, N. Bore, and J. Folkesson, "Pointnetkl: Deep inference for gicp covariance estimation in bathymetric slam," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4078–4085, 2020. DOI: 10.1109/LRA.2020.2988180.

[59] F. Lu, *Shape Registration Using Optimization for Mobile Robot Navigation*. PhD Thesis, University of Toronto, USA, 1995.

[60] S. M. Kay, *Fundamentals of statistical signal processing: estimation theory*. USA: Prentice-Hall, Inc., 1993, ISBN: 0133457117.

[61] O. Bengtsson and A. Baerveldt, "Localization in changing environments - estimation of a covariance matrix for the idc algorithm," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, vol. 4, 2001, 1931–1937 vol.4. DOI: 10.1109/IROS.2001.976356.

[62] S. Krantz and H. Parks, "The implicit function theorem : History, theory, and applications / s.g. krantz, h.r. parks.," pp. 36–41, Jan. 2003. DOI: 10.1007/978-1-4612-0059-8.

[63] Wikipedia contributors, *Fisher information — Wikipedia, the free encyclopedia*, [Online; accessed 29-July-2024], 2024.

[64] D. Brujic and M. Ristic, "Monte carlo simulation and analysis of free-form surface registration," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 211, Aug. 1997. DOI: 10.1243/0954405981516544.

[65] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart, "Challenging data sets for point cloud registration algorithms," *The International Journal of Robotics Research*, vol. 31, no. 14, pp. 1705–1711, Dec. 2012.

[66] K. Koide, J. Miura, M. Yokozuka, S. Oishi, and A. Banno, "Interactive 3d graph slam for map correction," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 40–47, 2021. DOI: 10.1109/LRA.2020.3028828.

[67] Wikipedia contributors, *Cramér–rao bound — Wikipedia, the free encyclopedia*, [Online; accessed 21-September-2024], 2024.