Université de Technologie
de Compiègne

Master Automation and Robotics in Intelligent Systems (ARS)

ARS4 Project Report

Localization using poles and signs detected by a Lidar

Prepared by :

Yassine Mathlouthi
Adnane Sallili

Supervisor: Prof. Dr. Philippe Bonnifait

2024-2025

Contents

1 Introduction

Accurate localization in urban environments remains a challenging problem despite advancements in Global Navigation Satellite Systems (GNSS). The limitations of GNSS in urban canyons and the need for sub-meter accuracy necessitate the integration of additional sensors. This project leverages a 360° vehicle-mounted LiDAR to detect geo-referenced landmarks, such as poles and signs, to enhance positioning accuracy.

The objectives of this work include:

- Developing a stochastic state space representation for sensor fusion.
- Implementing and validating an Extended Kalman Filter (EKF) on simulated data.
- Integrating a data association algorithm to link LiDAR measurements with map landmarks.
- Applying the EKF to real data for trajectory estimation.
- Implementing and evaluating an Unscented Kalman Filter (UKF) on simulated and real data.
- Comparing the performance of the EKF and UKF in terms of accuracy and robustness.

2 Stochastic State Space Representation

To fuse data from lidar, GNSS, and kinematic sensors, a stochastic state space representation is proposed. The components are as follows:

- **State Vector (x):**

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \theta \\ v \\ \omega \end{bmatrix}$$

  where x and y represent the 2D position, θ the heading angle, v the longitudinal speed, and ω the angular velocity.

- **Input (u):** The input vector includes controls:

$$\mathbf{u} = \begin{bmatrix} a \\ b \end{bmatrix}$$

  where a is the acceleration and b is the angular velocity.

- **Evolution Model :**

  The system evolves according to:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) + \alpha,$$

  where

$$f(\mathbf{x}_k, \mathbf{u}_k) = \begin{bmatrix} x_k + v_k \cos(\theta_k)\Delta t \\ y_k + v_k \sin(\theta_k)\Delta t \\ \theta_k + \omega_k \Delta t \\ v_k + a_k \Delta t \\ \omega_k + b_k \Delta t \end{bmatrix},$$

  and, Δt is the time step, $\alpha \sim \mathcal{N}(0, \mathbf{Q})$ is the noise model.

- **Covariance matrix Q :**

  Uncertainty in the evolution model is represented by the covariance matrix:

  $$Q = \mathrm{diag}(\sigma_x^2, \sigma_y^2, \sigma_\theta^2, \sigma_v^2, \sigma_\omega^2)$$

- **Observation Model :**
  - **GNSS Observation Model:**

    The GNSS provides measurements of the vehicle pose:

    $$h_{GNSS}(x) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

    The corresponding observation noise is:

    $$R_{GNSS} = \mathrm{diag}(\sigma_{GNSS_x}^2, \sigma_{GNSS_y}^2, \sigma_{GNSS_\theta}^2)$$

  - **Lidar Observation Model:**

    The lidar detects landmarks in the vehicle's mobile frame. These detections are transformed into the map frame using:

    $$h_{Lidar}(x, z) = \begin{bmatrix} x + z_1\cos(\theta) - z_2\sin(\theta) \\ y + z_1\sin(\theta) + z_2\cos(\theta) \end{bmatrix}$$

    where $z = \begin{bmatrix} z_1 & z_2 \end{bmatrix}^T$ are the lidar detections in the mobile frame. The corresponding observation noise is:

    $$R_{Lidar} = \mathrm{diag}(\sigma_{Lidar_x}^2, \sigma_{Lidar_y}^2)$$

3 Implementation of Extended Kalman Filter (EKF)

3.1 Simulated Data

The EKF was tested on simulated data with the following steps:

1. **Initialization:** The state vector was initialized using the reference dataset, and the covariance matrix was set to an identity matrix.
2. **Prediction Step:** The state and covariance were predicted using:

   $$\hat{\mathbf{x}}_{k+1|k} = f(\hat{\mathbf{x}}_k, \mathbf{u}_k)$$
   $$\mathbf{P}_{k+1|k} = \mathbf{F}_k\mathbf{P}_k\mathbf{F}_k^T + \mathbf{Q},$$

   where $\mathbf{F}_k = \frac{\partial f}{\partial \mathbf{x}}$ is the Jacobian matrix of the evolution Model.

   $$F = \frac{\partial f(x,u)}{\partial x} = \begin{bmatrix} 1 & 0 & -v\sin(\theta)\Delta t & \cos(\theta)\Delta t & 0 \\ 0 & 1 & v\cos(\theta)\Delta t & \sin(\theta)\Delta t & 0 \\ 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

3. **Update Step:** When GNSS observations were available:

$$
\begin{aligned}
\mathbf{y}_k &= \mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1}) \\
\mathbf{S}_k &= \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R} \\
\mathbf{K}_k &= \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1} \\
\hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \mathbf{y}_k \\
\mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \mathbf{R} {\mathbf{K}_k}^T,
\end{aligned}
$$

   where $\mathbf{H}_k = \frac{\partial h}{\partial \mathbf{x}}$.

   – **GNSS Jacobian matrix:**

$$
H_{GNSS} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}
$$

   – **Lidar Jacobian matrix:**

$$
H_{Lidar} = \begin{bmatrix} 1 & 0 & -z_1 \sin(\theta) - z_2 \cos(\theta) & 0 & 0 \\ 0 & 1 & z_1 \cos(\theta) - z_2 \sin(\theta) & 0 & 0 \end{bmatrix}
$$

4. **Tuning and Validation:** Process and observation noise covariances were tuned to achieve 95% consistency, and the results were compared to reference trajectories.

3.2 Data Association Algorithm

To integrate lidar measurements into the EKF, a data association algorithm is used to match lidar detections to known map landmarks.

So, the Algorithm Steps :

1. **Global Transformation:** Transform lidar detections from the vehicle's local frame to the global frame using the estimated state.

$$
\mathbf{z}_{global} = \begin{bmatrix} x_{est} + z_x \cos(\theta_{est}) - z_y \sin(\theta_{est}) \\ y_{est} + z_x \sin(\theta_{est}) + z_y \cos(\theta_{est}) \end{bmatrix}.
$$

2. **Nearest-Neighbor Matching:** Compute the Euclidean distance between each transformed detection and all map landmarks:

$$
d_{ij} = \|\mathbf{z}_{global,i} - \mathbf{m}_j\|,
$$

   where $\mathbf{m}_j$ is the jth map landmark.

3. **Association Criterion:** Associate detections to landmarks if the distance is below a predefined threshold (e.g., 2 meters).

4. **Output:** Return an array of associated landmark indices for each detection.

3.3 Implementation on Real Data

Real-world data introduces additional challenges, including sensor noise and missing detections. The EKF is applied as follows:

1. **Prediction Step** The state and covariance are propagated using the evolution model:

$$\hat{\mathbf{x}}_{k+1|k} = f(\hat{\mathbf{x}}_k, \mathbf{u}_k), \quad \mathbf{P}_{k+1|k} = \mathbf{F}_k \mathbf{P}_k \mathbf{F}_k^T + \mathbf{Q},$$

   where $\mathbf{F}_k$ is the Jacobian of the evolution model.

2. **Update Step**

   GNSS Update: GNSS measurements are incorporated using:

$$\mathbf{y} = \mathbf{z}_{GNSS} - h_{GNSS}(\hat{\mathbf{x}}_{k|k-1}),$$

   with Kalman gain:

$$\mathbf{K} = \mathbf{P}_{k|k-1} \mathbf{H}_{GNSS}^T (\mathbf{H}_{GNSS} \mathbf{P}_{k|k-1} \mathbf{H}_{GNSS}^T + \mathbf{R}_{GNSS})^{-1}.$$

   Lidar Update: Lidar measurements are first associated with map landmarks using the data association algorithm. For each matched detection:

$$\mathbf{y} = \mathbf{z}_{lidar} - h_{lidar}(\hat{\mathbf{x}}_{k|k-1}),$$

   with similar Kalman gain and update equations.

4 Implementation of Unscented Kalman Filter (UKF)

The UKF was implemented following the same state space model. It uses a sigma point representation to handle non-linearities more effectively.

4.1 UKF steps :

1. **Sigma Point Generation:** Generate sigma points using the Cholesky decomposition of the state covariance matrix:

$$\mathbf{X}_i = \mathbf{x} \pm \sqrt{n+\lambda}\mathbf{P}, \quad i = 1, \ldots, 2n,$$

   with
   - $\lambda = \alpha^2(n+\kappa) - n$,
   - n the state vector dimension,
   - α, β, κ parameters defining the spread of sigma points.

2. **Prediction Step:**
   - Propagate sigma points through the motion model:

$$\mathbf{X}_i^{\text{pred}} = \mathbf{f}(\mathbf{X}_i, \mathbf{u}_t).$$

   - Compute predicted mean and covariance:

$$\mathbf{x}_{k+1|k} = \sum_{i=0}^{2n} w_i \mathbf{X}_{i,k+1|k},$$

$$\mathbf{P}_{k+1|k} = \sum_{i=0}^{2n} w_i (\mathbf{X}_{i,k+1|k} - \mathbf{x}_{k+1|k})(\mathbf{X}_{i,k+1|k} - \mathbf{x}_{k+1|k})^T + \mathbf{Q}.$$

3. **Update Step:**
   - Transform sigma points into the measurement space:
$$\mathbf{Z}_i = \mathbf{h}(\mathbf{X}_i).$$
   - Compute predicted measurement mean and innovation covariance:
$$\mathbf{z}_{k+1|k} = \sum_{i=0}^{2n} w_i \mathbf{Z}_i,$$
$$\mathbf{S} = \sum_{i=0}^{2n} w_i (\mathbf{Z}_i - \mathbf{z}_{k+1|k})(\mathbf{Z}_i - \mathbf{z}_{k+1|k})^T + \mathbf{R}.$$
   - Compute cross covariance and Kalman gain:
$$\mathbf{P}_{xz} = \sum_{i=0}^{2n} w_i (\mathbf{X}_i - \mathbf{x}_{k+1|k})(\mathbf{Z}_i - \mathbf{z}_{k+1|k})^T,$$
$$\mathbf{K} = \mathbf{P}_{xz}\mathbf{S}^{-1}.$$
   - Update state and covariance:
$$\mathbf{x}_{k+1} = \mathbf{x}_{k+1|k} + \mathbf{K}(\mathbf{z}_{k+1} - \mathbf{z}_{k+1|k}),$$
$$\mathbf{P}_{k+1} = \mathbf{P}_{k+1|k} - \mathbf{K}\mathbf{S}\mathbf{K}^T.$$

4.2 Real Data Implementation

4.2.1 Challenges with Real Data

- Sensor noise and missing observations.
- Need for data association to link LiDAR detections with map landmarks.

4.2.2 UKF Updates for Poles and Signs

Separate update steps are implemented for:

- **Poles Observations:** Matched with pole landmarks in the map.
- **Signs Observations:** Matched with sign landmarks in the map.

Each update follows the standard UKF procedure with its respective measurement model and noise covariance.

5 Results and Comparison

5.1 Simulated Data Results

5.1.1 EKF

The results from the two plots highlight the effectiveness of the filter in estimating the trajectory and handling sensor noise. In the time-series plot, the estimated X, Y, and heading closely align with the reference values, with minimal deviations, demonstrating high accuracy and robustness. The filter effectively tracks both linear and angular states, even during dynamic transitions such as sharp turns. In the spatial trajectory plot, the estimated trajectory (red dashed line) aligns well with the reference (blue line), significantly reducing the noise present in the GNSS measurements (blue stars). Despite noisy lidar observations (black stars), the filter integrates them effectively, leveraging the map landmarks (red stars) for corrections. Overall, the filter successfully mitigates sensor noise, ensures smooth trajectory estimation, and maintains reliability under challenging conditions.

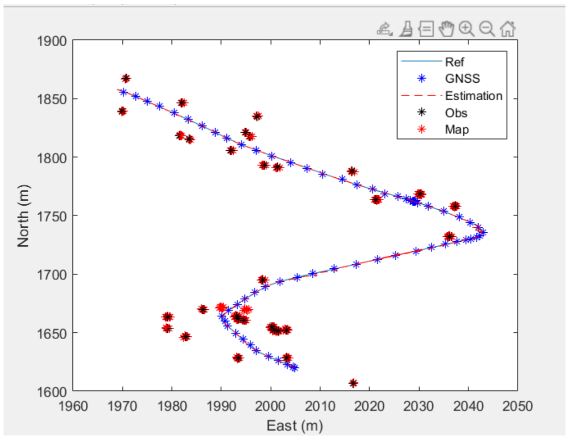

Figure 1: EKF trajectory with simulated data

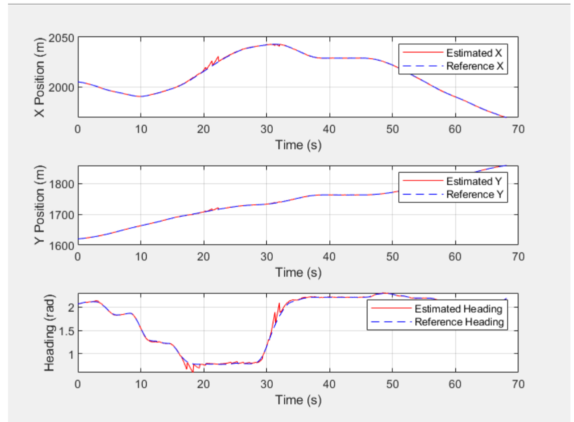

Figure 2: EKF trajectory with simulated data: estimation vs ref

	Mean Error	Max Error	MSE	Consistency
x	0.10774	5.6711	0.34878	0.90762
y	0.16108	5.5497	0.35163	0.93255
h	0.0025468	0.28781	0.0011024	0.85924

Table 1: Metrics EKF

The table showcases the performance of the Extended Kalman Filter (EKF) in estimating the x, y, and heading (h) states. The filter demonstrates excellent accuracy, with low mean errors ($x = 0.10774$, $y = 0.16108$, and $h = 0.0025468$), indicating precise average estimation for both position and heading. The maximum errors ($x = 5.6711$, $y = 5.5497$, and $h = 0.28781$) reveal occasional deviations, possibly due to transient disturbances or outliers. The mean square errors (MSE) further affirm the filter's precision, particularly for heading (MSE $= 0.0011024$). Consistency values are high for x (0.90762) and y (0.93255), confirming that the filter's predicted uncertainties align well with the actual errors. The heading consistency (0.85924) is slightly lower but remains reliable. Overall, the EKF provides robust and consistent performance across all states, with minimal deviations from the true values.

5.1.2 UKF

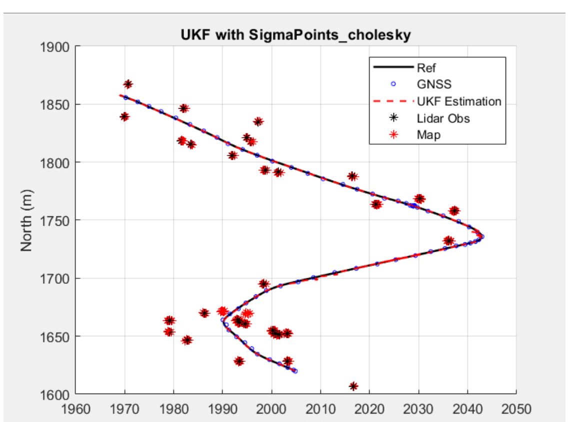

Figure 3: UKF trajectory with simulated data

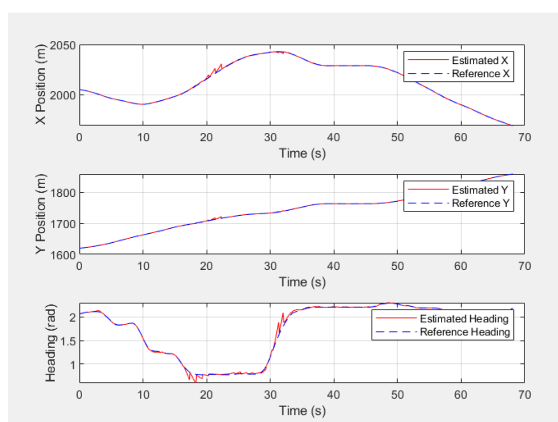

Figure 4: UKF with simulated data: estimation vs ref

The plots illustrate the performance of the Unscented Kalman Filter (UKF) in estimating the trajectory compared to the reference. In the time-series analysis, the estimated X, Y, and heading closely align with the reference values, showing minimal deviations. The UKF effectively handles transitions, such as sharp changes in heading around $t = 20$ to $t = 30$ seconds, demonstrating robustness in angular state estimation. In the spatial trajectory comparison, the UKF estimation significantly reduces noise from GNSS measurements while maintaining strong alignment with the reference trajectory, even in curved sections. The filter successfully integrates noisy lidar observations using map landmarks for correction, resulting in a smooth and accurate trajectory estimate. Overall, the UKF shows high precision and reliability, effectively mitigating noise and handling non-linearities, particularly during dynamic transitions.

	Mean Error	Max Error	MSE	Consistency
x	0.1074	5.6724	0.34884	0.90909
y	0.16124	5.5504	0.35161	0.93255
h	0.0025519	0.28729	0.001097	0.85924

Table 2: Metrics UKF

he table summarizes the performance metrics of the Unscented Kalman Filter (UKF) for estimating x, y, and heading (h) states. The UKF demonstrates excellent accuracy, with very low mean errors for x (0.1074), y (0.16124), and heading (0.0025519), indicating precise average estimates. The maximum errors ($x = 5.6724$, $y = 5.5504$, $h = 0.28729$) highlight occasional deviations, likely due to transient noise or dynamic changes in the system. The mean square errors (MSE) are low ($x = 0.34884$, $y = 0.35161$, $h = 0.0011097$), showcasing consistent performance across all states. The consistency values are high for x (0.90909) and y (0.93255), indicating that the UKF's predicted uncertainties align well with the actual estimation errors. Although the consistency for heading (0.85924) is slightly lower, it remains acceptable, reflecting robust angular state estimation.
Overall, the UKF achieves reliable and accurate performance in both positional and angular states. However, there is no significant difference between the results of the UKF and the Extended Kalman Filter (EKF). Both filters exhibit similar levels of accuracy, MSE, and consistency across all metrics, making either a viable choice.

5.2 Real Data Results

5.2.1 EKF

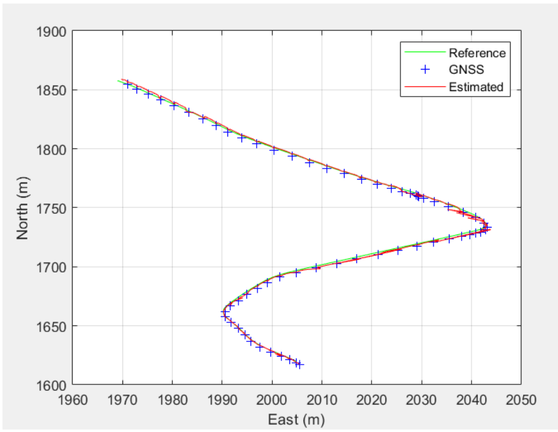

Figure 5: EKF trajectory with real data

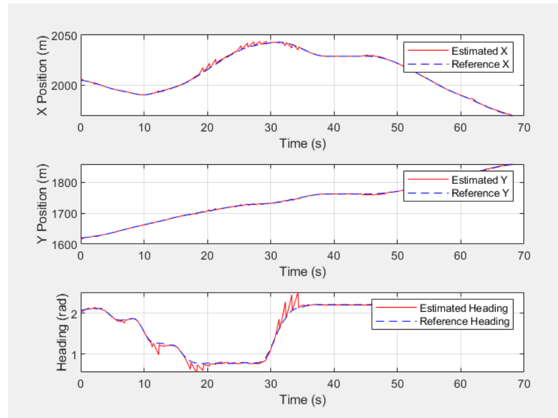

Figure 6: EKF with real data: Estimated vs Ref

The plots highlight the performance of the EKF in estimating trajectories with real data and reveal notable differences compared to simulated data. In the time-series analysis, the estimated X, Y, and heading closely align with the reference values, with minimal deviations. However, real-world noise, such as GNSS inaccuracies and sensor imperfections, introduces small discrepancies that are not present in simulated data. For instance, the real data exhibit significant noise, especially in curved sections, which the EKF successfully mitigates through effective sensor fusion. In contrast, simulated data typically assumes ideal or bounded noise distributions, resulting in smoother and more predictable behavior for both GNSS and lidar observations.

State	Mean Error	Max Error	Mean Square Error	Consistency
x	0.33011	5.1279	0.9003	0.74927
y	-0.015229	3.4896	1.1983	0.48094
h	-0.0075687	0.40309	0.0037677	0.81672

Table 3: Performance metrics for EKF with real data.

The EKF demonstrates reliable performance with real data, though its accuracy and consistency are slightly degraded compared to results from simulated data. In real data, the mean errors for x (0.33011) and y (-0.015229) remain low, but the mean square errors (MSE) are larger ($x = 0.9003$, $y = 1.1983$) due to increased noise and variability in real-world conditions. The consistency values are significantly lower, particularly for y (0.48094), indicating that the filter's predicted uncertainties are less aligned with actual errors compared to the simulated environment. For heading (h), the mean error remains minimal (-0.0075687), and the consistency (0.81672) is still acceptable, though both metrics are slightly worse than in simulated data.

Overall, the real-world data introduces challenges s leading to higher maximum errors and reduced consistency compared to the controlled conditions of simulation. Despite these challenges, the EKF proves robust, maintaining small average errors and providing reliable estimates. However, further tuning of noise parameters and consideration of real-world disturbances could enhance its performance.

5.2.2 UKF

When applying the UKF with real data, the results were almost identical to those obtained with the EKF.

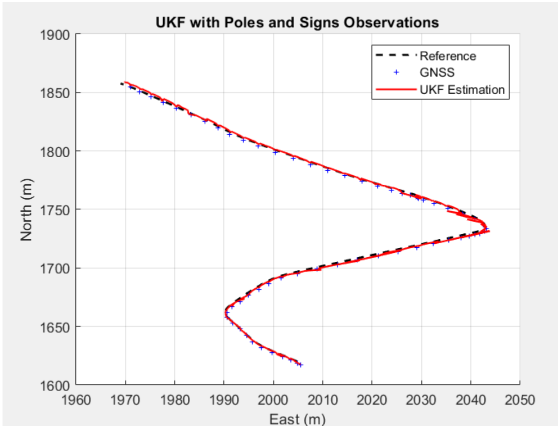

Figure 7: UKF trajectory with real data

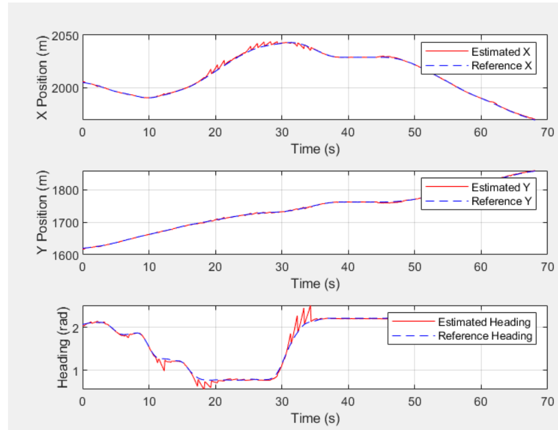

Figure 8: UKF with real data: Estimated vs Ref

Overall, the UKF achieves reliable and accurate performance in both positional and angular states. However, there is no significant difference between the results of the UKF and the Extended Kalman Filter (EKF). Both filters exhibit similar levels of accuracy, MSE, and consistency across all metrics, making either a viable choice.

6 Conclusion

This project successfully implemented the EKF and UKF on both simulated and real data for vehicle localization using GNSS and LiDAR observations. The filters demonstrated their ability to handle non-linearities effectively and provided robust trajectory estimates, even in challenging real-world conditions. The key findings include:

- Unscented Kalman Filter (UKF) and the Extended Kalman Filter (EKF) exhibit similar levels of accuracy, MSE, and consistency across all metrics when using real data. This indicates that both filtering techniques are robust to the nuances of real-world sensor data and dynamics but do not show significant performance advantages over one another in this specific application.
- The performance varies significantly between simulated and real data scenarios. With simulated data, the EKF shows relatively lower mean errors and mean square errors (MSE) across all states (x, y, h). For real data, these errors increase, suggesting that real-world conditions introduce complexities and noise that are not perfectly modeled in simulations.
- Data association algorithms were critical for integrating LiDAR observations with map landmarks, ensuring accurate updates.