



Thesis Project

Applied Artificial Intelligence 180 credits

An Experimental Study on Object Tracking

Data Science 15 credits

Halmstad June 9, 2025

Mahmoud Alshaikh

Abstract

This thesis investigated the robustness of 3D object-tracking algorithms under snowy weather conditions, focusing particularly on snowy scenarios affecting autonomous vehicle perception systems. The principal objective was to evaluate and compare the performance of four tracking methods: Kalman Filter, Extended Kalman Filter, Particle Filter, and ByteTrack. Each method was assessed using LiDAR data obtained from the Canadian Adverse Driving Conditions (CADC) dataset, representing harsh winter conditions, and the nuScenes dataset, representing clear, optimal weather conditions.

The methodology involved processing sequential frames of LiDAR data, detecting 3D bounding boxes, and tracking objects through association and state estimation. Standard metrics such as HOTA, IDF1, AMOTA, and AMOTP were used to measure tracking accuracy and consistency across both datasets. Results indicated significant performance degradation for all algorithms under snowy weather conditions compared to clear weather. The Kalman methods suffered from the linear behavior and the noise, while the Particle Filter provided a more robust estimation due to its ability to cope with the uncertainty via its multiple hypotheses.

Deep learning-based solution ByteTrack demonstrated better performance, with better accuracy and fewer identity switches in challenging scenarios. It was found that deep learning based tracking can provide more solid guarantee on point trajectory

The study concluded that deep learning-based tracking methods offer enhanced reliability for autonomous vehicles in challenging environments.

Keywords Autonomous Vehicles, LiDAR, 3D Object Tracking, snowy Weather Conditions, Kalman Filter (KF), Extended Kalman Filter (EKF), Particle Filter (PF), ByteTrack, LSTM (Long Short-Term Memory), Canadian Adverse Driving Conditions (CADC) Dataset, Multi-Object Tracking (MOT), Tracking Algorithms, Machine Learning, Bird's-Eye-View (BEV)

Sammanfattning

Denna avhandling undersökte robustheten hos 3D-objektspårningsalgoritmer under snöiga väderförhållanden, med särskilt fokus på snöiga scenarier som påverkar perceptionssystem i autonoma fordon. Det huvudsakliga målet var att utvärdera och jämföra prestandan hos fyra spårningsmetoder: Kalmanfilter, Utökat Kalmanfilter, Partikelfilter och ByteTrack. Varje metod bedömdes med hjälp av LiDAR-data från CADC-datamängden, som representerar hårda vinterförhållanden, samt nuScenes-datamängden, som representerar klart och optimalt väder.

Metodiken innebar bearbetning av sekventiella LiDAR-ramar, detekterade av 3D-bounding boxes och spårning av objekt genom association och tillståndsestimering. Standardmått som HOTA, IDF1, AMOTA och AMOTP användes för att mäta spårningsnoggrannhet och konsistens i båda datamängderna.

Resultaten visade en tydlig försämring i prestanda för alla algoritmer under snöiga väderförhållanden jämfört med klart väder. Kalmanbaserade metoder hade svårt att hantera detta på grund av sina linjära antaganden och känslighet för brus, medan Partikelfiltret visade ökad robusthet genom att hantera osäkerhet effektivt via flera tillståndshypoteser. ByteTrack, som bygger på djupinlärning, presterade konsekvent bäst, med högre noggrannhet och färre identitetsfel under svåra förhållanden.

Studien drog slutsatsen att djupinlärningsbaserade spårningsmetoder erbjuder högre tillförlitlighet för autonoma fordon i utmanande miljöer.

Nyckelord Autonomous Vehicles, LiDAR, 3D Object Tracking, Adverse Weather Conditions, Kalman Filter (KF), Extended Kalman Filter (EKF), Particle Filter (PF), ByteTrack, LSTM (Long Short-Term Memory), Canadian Adverse Driving Conditions (CADC) Dataset, Multi-Object Tracking (MOT), Tracking Algorithms, Machine Learning, Bird's-Eye-View (BEV)

Preface

I would like to thank my supervisor for the helpful comments and guidance during this thesis. I also thank the examiner for the valuable feedback that helped improve the work

This thesis work as co-funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Climate, Infrastructure and Environment Executive Agency (CINEA). Neither the European Union nor the granting authority can be held responsible for them. Project grant no. 101069576.

Contents

Abstract	ii
Sammanfattning	iii
Preface	iv
Contents	v
List of Figures	vii
List of Tables	viii
1 Introduction	1
1.1 Background	1
1.1.1 Multiple Object Tracking (MOT) in Autonomous Vehicles . .	1
1.1.2 Kalman Filter (KF) and Extended Kalman Filter (EKF)	1
1.1.3 ByteTrack	2
1.1.4 Particle Filters	3
1.2 Problem Statement	3
1.3 Purpose and goals	4
1.4 Requirements	5
1.5 Related works	5
1.5.1 3D Multi-Object Tracking with Adaptive Cubature Kalman Filter	5
1.5.2 Solving Occlusion in Multi-Object Tracking with DeepSORT and Quantum Computing	5
1.5.3 MUTR3D: A Multi-camera Tracking Framework via 3D-to- 2D Queries	6
2 Technical Background	7
3 Method	9
3.1 3D Detected bounding boxes	9
3.2 Kalman Filter-Based 3D Bounding Box Tracking	9
3.3 Extended Kalman Filter for 3D Bounding Box Tracking	10
3.4 ByteTrack for 3D Bounding	11
3.5 Particle Filter for 3D Box Tracking	12
3.6 LSTM	13
4 Implementation	15
5 Results	16
5.1 Tracking Results on CADK Dataset (Snowy Conditions)	16
5.1.1 Performance Summary	16

5.1.2 Overall Analysis	17
5.2 Tracking Results on nuScenes Dataset (Clean Weather)	17
5.2.1 Performance Summary	17
5.3 Failure Case Analysis	18
5.4 Run-time	18
6 Discussion	19
6.1 Societal Aspects	20
7 Conclusion	22
7.1 Future Work	22
Bibliography	24

List of Figures

3.1	Training loss curves (MAE) for right and left regions.	14
3.2	Predictions vs. ground truth for both regions.	14

List of Tables

- 5.1 Tracking results (higher \uparrow is better; lower \downarrow is better) 16
- 5.2 Tracking results (higher \uparrow better; lower \downarrow better). 17

Chapter 1

Introduction

1.1 Background

1.1.1 Multiple Object Tracking (MOT) in Autonomous Vehicles

3D multi-object tracking (MOT) helps autonomous vehicles detect and follow objects like cars and pedestrians in 3D space [1], [2]. It's important for safety and decision-making. Challenges include sensor noise, occlusion, and complex motion [3]. LiDAR, cameras, and radar are used together. LiDAR gives depth, cameras add visual details, and radar works in snowy weather [2]. Sensor fusion improves tracking but makes the system more complex [1]. VoxelNet and PointPillars are methods to process 3D point clouds for object detection [4], [5]. Combining 2D and 3D views improves accuracy [2]. MOT systems first detect objects, then link them across frames using algorithms like the Hungarian method [1]. Kalman Filters help track motion despite noise. Datasets like CAD90 and nuScenes support tracking research but don't fully reflect real-world edge cases [1], [3].

1.1.2 Kalman Filter (KF) and Extended Kalman Filter (EKF)

Kalman Filter (KF) and Extended Kalman Filter (EKF) are widely used in autonomous vehicles for estimating position and motion by combining models with sensor data. They work in real time and handle noisy or incomplete measurements well.

Researchers use Kalman filters for many driving tasks. Wang et al. showed that adding map data to sensor input improves vehicle tracking at intersections [6]. Guo et al. proposed splitting motion models to get better localization [7]. Bertipaglia et al. improved sideslip angle estimates by combining a KF with a neural

network [8].

Liu et al. proposed an adaptive KF to fuse GPS and inertial sensors which can reduce drift and deal with GPS dropouts [9]. Jing et al. employed EKF in agriculture for land orientation to demonstrate that it could perform well in off-road terrains [10].

KF and EKF are also robust, computationally efficient, and flexible. They assist in handling uncertainty, in manipulation of multiple sensors and in real-time decision-making. EKF is more robust with respect to nonlinear systems.

1.1.3 ByteTrack

The ByteTrack is a Multi-Object Tracking (MOT) method which is known for its simplicity yet effective data-association. It is different from other approaches in the way that it also takes into account low-certainty detections when reconstructing under-painted or missing objects. [11], [12].

The pipeline uses object detection, a Kalman filter for motion prediction, and association through the Hungarian algorithm. Tracks are created from unmatched detections and removed after being unmatched for several frames [12].

ByteTrack's strength is tracking every detection, including low-confidence ones, to increase recall without much loss in precision. This leads to better performance on MOT benchmarks like MOT17, MOT20, and BDD100K [11], [13].

In autonomous driving, ByteTrack is used with detectors like YOLOv8 for real-time vehicle tracking. It handles occlusions and crowded scenes well, which is critical in traffic scenarios [12], [14].

ByteTrack is also used in surveillance to track people in crowds, benefiting from its ability to keep tracks through occlusions. It has been extended for multi-camera and city-scale systems [15], [12].

Other fields have adopted ByteTrack too. In medical imaging, ByteTrack-Line tracks thyroid nodules in ultrasound videos [16]. In robotics, it helps improve SLAM by tracking and removing moving objects from scenes [17].

1.1.4 Particle Filters

Particle Filters (PFs) are Bayesian estimators that track a system's state using many random samples (particles), each representing a possible state with a weight [18]. Unlike Kalman Filters, PFs work well with non-linear motion and non-Gaussian noise [19].

Each particle follows the motion model to predict a new state. Sensor data is then used to weight the particles based on how likely each state is. Resampling focuses on high-probability particles, improving estimation over time [18].

PFs handle multiple possible states at once, making them useful in uncertain environments (e.g., when a vehicle's location is ambiguous) [20]. They also handle outliers and support complex sensor fusion better than linear filters [18].

A key use in autonomous driving is Monte Carlo Localization (MCL), where PFs estimate a car's position using LIDAR or camera data matched to a map [19]. PFs are also used to track other road users, especially when motion is unpredictable or occluded.

Although PFs are more computationally demanding, modern processors make them practical. Their ability to model uncertainty is essential for robust localization, tracking and sensor fusion in autonomous vehicles.

1.2 Problem Statement

Self-driving cars have improved in leaps and bounds, but they're still lousy at bad weather. Snow reduces visibility and causes sensor interference, which make it difficult to detect and track objects. Such a reliance jeopardizes safety and public confidence in autonomous vehicles. Snow causes unique problems. It interferes with LiDAR, obscures lane markings and generates shiny surfaces and slush that confound sensors. These issues make it harder to track vehicles, people, and obstacles accurately.

Many tracking algorithms like Kalman Filters, Particle Filters, and deep learning methods work well in good weather but fail in snow. Classical models can't handle the noise and occlusion from falling snow. Deep learning systems may lack enough training data from harsh conditions.

Since tracking is central to how autonomous vehicles perceive their environment, failures in snowy weather lead to safety risks and reduced performance. Vehicles

may drive too cautiously, stop working, or make unsafe decisions, especially in snowy regions.

This thesis tests how well different tracking algorithms handle snow, using real-world data from the CADDC and ROADVIEW datasets. It compares classical and deep learning methods to find out which are most reliable in winter driving.

The goal is to study tracking under snow, helping autonomous vehicles stay safe and practical in all weather.

1.3 Purpose and goals

This study evaluates four object-tracking algorithms Kalman Filter (KF), Extended Kalman Filter (EKF), Particle Filter (PF), and ByteTrack to understand how well they perform in snowy conditions. It compares classical model-based methods with a learning-based tracker to show their strengths and weaknesses in real-world winter driving.

Kalman Filters are fast and efficient for simple motion, making them useful for real-time tracking, but they struggle with occlusions and nonlinear motion. Particle Filters handle more uncertainty by keeping multiple possible states, which helps in complex scenes. ByteTrack, a deep learning-based method, uses visual features learned from data. It adapts better to dynamic environments but needs more computation and diverse training data.

The goal is to test all four algorithms using real snowy driving data from CADDC and ROADVIEW. The study will measure how well each method deals with occlusion, sensor noise, and reduced visibility. Performance will be evaluated using HOTA, IDF1, AMOTA, AMOTP, and IDS.

By directly comparing non-learning and learning-based approaches, this research aims to give practical insights for building better perception systems in autonomous vehicles. The results support the ROADVIEW project's goal of study tracking reliability in winter conditions.

1.4 Requirements

Background Knowledge

Illuminating fundamental object-tracking concepts is essential to compare different algorithms. It is useful to understand classical filters (Kalman), deep learning trackers (ByteTrack), and motion models (linear vs non-linear). It also helps to have a feel for sensors like LiDAR, and their behavior when in contact with snow. Knowledge of datasets such as CAD3D and ROADVIEW, including how they are labeled and evaluated, supports accurate performance analysis. Basic Python skills and experience with tools like NumPy, OpenCV, and Pandas help in running and testing these algorithms.

Technical Functional

All four algorithms (KF, EKF, PF, ByteTrack) will be tested under the same snowy datasets and tracking metrics. The goal is to measure how well each one handles snow, low visibility, and occlusions. Key performance indicators include accuracy, false detections, and identity switches. This setup allows a fair comparison of classical and deep learning-based methods in real-world winter conditions.

1.5 Related works

1.5.1 3D Multi-Object Tracking with Adaptive Cubature Kalman Filter

Guo and Zhao (2023) propose a 3D tracking method using an Adaptive Cubature Kalman Filter (ACKF) with a constant turn rate and velocity (CTRV) motion model [21]. The ACKF adapts to sensor bias and uncertainty, improving tracking robustness. They also introduce a graph-based data association model to better match predictions with detections. Tested on the KITTI dataset, their method outperforms baseline trackers in accuracy and speed. This shows how KF variants and improved association strategies enhance tracking in real-world driving.

1.5.2 Solving Occlusion in Multi-Object Tracking with DeepSORT and Quantum Computing

Ngeni et al. (2024) enhance DeepSORT by adding a quantum computing module for better tracking under occlusion [22]. The quantum approach solves data association by evaluating many match options in parallel, reducing identity switches

when objects overlap. This method improves accuracy in dense scenes and shows how deep learning-based trackers can be improved with emerging computing methods. It highlights a path to handle persistent occlusions in autonomous driving.

1.5.3 MUTR3D: A Multi-camera Tracking Framework via 3D-to-2D Queries

MUTR3D is an end-to-end framework for tracking 3D objects using multiple cameras. Unlike traditional methods that depend on matching object appearances or positions, MUTR3D introduces "3D track queries" to model objects across different cameras and time frames[23]. It links 3D object tracks to 2D images through camera transformations and refines them using image features. The system employs a set-to-set loss function, eliminating the need for post-processing steps like non-maximum suppression. Tested on the nuScenes dataset, MUTR3D outperforms previous methods by 5.3 percent in AMOTA, demonstrating its effectiveness in multi-camera 3D tracking scenarios[23].

Chapter 2

Technical Background

Self-driving cars rely on tracking objects in order to understand the movements of other cars, pedestrians and cyclists around them. This involves predicting the position and velocity of objects over a period of time, using sensor data (typically LiDAR). Tracking is vital in more cluttered scenes or where targets move [13], [1].

Tracking methods can be classified as either model-based or learning-based. Model-based approaches such as KF, EKF and PF are flexible. They are based on fixed motion models and are computationally cheap and intuitive. KF is effective for pure motion, EKF is useful for weakly nonlinear process, and PF works in case of strong uncertainty with multiple state samples. [24], [7].

learning-based approaches (ByteTrack) leverage neural networks to learn features from data. They can also cope with complex scenes with varying illumination or occlusion, but require strong hardware and large annotated data. [25], [12].

Researchers evaluate these methods using benchmarks and metrics like MOTA and IDF1, which measure accuracy and identity tracking [13], [1]. Studies show both types can be improved or combined to increase reliability in difficult conditions [25].

In this study, the following algorithms are compared:

- **KF, EKF, PF:** Classical, fast, and effective in predictable settings with clear motion assumptions.
- **ByteTrack:** A learning-based method suited for complex urban scenes with high visual variability.

The goal of this research is to investigate which is the best method of the approach for vehicle tracking in snow.

Long Short-Term Memory (LSTM)

The LongShort-Term Memory (LSTM) networks, introduced towards the end of the 1990s, avoids the vanishing gradient problem in vanilla RNNs, using internal memory and gates to control the flow of information [26]. This architecture enables LSTMs to learn long-termdependencies in sequential data and thus makes them applicable for motion pattern prediction.

In autonomous driving, LSTM models help predict future paths of dynamic agents like pedestrians and vehicles [27]. For example, Social LSTM [28] uses shared information among LSTM networks to capture pedestrian interactions, outperforming earlier methods.

Vehicle trajectory prediction also benefits from LSTMs. Altché and de La Fortelle [29] showed LSTMs accurately forecast future positions, while Hou et al. [30] improved accuracy further by sharing information between nearby vehicles. Ego-vehicle path prediction uses similar ideas; Fan et al. [27] integrated driving style data in an LSTM to forecast lane changes.

Chapter 3

Method

3.1 3D Detected bounding boxes

The dataset used for tracking consists of a list of 100 frames, where each frame contains a set of detected bounding boxes. Each bounding box is represented as a 7-dimensional array in the format $[x, y, z, dx, dy, dz, yaw]$, where (x, y, z) is the 3D position of the box center, (dx, dy, dz) defines the box dimensions along each axis, and yaw specifies the orientation around the vertical (z) axis. This consistent encoding provides a unified input format for all four tracking algorithms: Kalman Filter (KF), Extended Kalman Filter (EKF), Particle Filter (PF), and ByteTrack. The goal of each algorithm is to associate the same object across frames with a consistent identity, to assign and maintain the same ID for each detected object throughout the sequence. Using the same detection input across all methods ensures that performance differences can be attributed solely to the tracking logic, not the object detection step. This standardized input is crucial for fair and interpretable evaluation of multi-object tracking accuracy and identity preservation.

3.2 Kalman Filter-Based 3D Bounding Box Tracking

This method uses a Kalman Filter (KF) to track 3D bounding boxes over time. It estimates object position and velocity in 3D space from frame-by-frame sensor data, such as bounding boxes generated from LiDAR.

The tracking pipeline includes: (1) loading bounding boxes and configuration, (2) initializing KF state, (3) associating detections with tracks using the Hungarian algorithm, (4) creating updated 3D boxes, and (5) outputting annotated results.

The KF uses a six-dimensional state vector:

$$\mathbf{x} = [x \quad y \quad z \quad v_x \quad v_y \quad v_z]^T,$$

with a constant velocity model to predict new states. The transition matrix F includes the time step Δt , and the measurement matrix H maps position from the state. Noise matrices Q and R reflect uncertainty in motion and sensor data.

Data association uses Euclidean distance between predicted and detected box centers. Matches below a distance threshold are accepted; others are ignored. The KF then updates matched tracks and predicts the next state for all trackers.

Additional features include 2D velocity estimation from center displacement and optional point cloud integration for checking if objects contain sufficient LiDAR points. Bounding boxes are annotated with metadata and saved in a JSON-friendly format.

This KF-based approach offers a fast, interpretable solution for 3D object tracking and can be extended by adjusting the motion model or integrating richer sensor data.

3.3 Extended Kalman Filter for 3D Bounding Box Tracking

The Extended Kalman Filter (EKF) builds on the KF method but supports non-linear system dynamics. While the general pipeline data input, matching, and tracking remains the same, EKF replaces matrix operations with functions and Jacobians to support more complex models.

The EKF defines:

- **State Transition Function:** $\mathbf{f}(\mathbf{x}, \Delta t)$, which updates position using velocity.
- **Measurement Function:** $\mathbf{h}(\mathbf{x})$, which extracts position from state.

Since \mathbf{f} and \mathbf{h} can be non-linear, the EKF computes their Jacobians:

$$F = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}, \quad H = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}.$$

Update (Correction) Step Given a measurement \mathbf{z} , the EKF computes the predicted measurement $\hat{\mathbf{z}} = \mathbf{h}(\mathbf{x})$ and the Jacobian H . The measurement residual \mathbf{y} is

$$\mathbf{y} = \mathbf{z} - \hat{\mathbf{z}}.$$

then compute the measurement residual covariance,

$$S = HPH^T + \mathbf{R},$$

and the Kalman gain,

$$K = \mathbf{P}H^T S^{-1}.$$

Finally, update the state and covariance:

$$\mathbf{x} \leftarrow \mathbf{x} + K\mathbf{y}, \quad \mathbf{P} \leftarrow (\mathbf{I} - KH)\mathbf{P}.$$

Because the Jacobian H is reevaluated at each update to account for potential non-linearity, the EKF can correct the state even if \mathbf{h} is significantly more complicated than a simple linear projection.

In the predict step:

$$\mathbf{x} \leftarrow \mathbf{f}(\mathbf{x}, \Delta t), \quad \mathbf{P} \leftarrow F\mathbf{P}F^T + Q.$$

Though the provided functions are linear in this case, the EKF framework allows for future extensions such as adding yaw, acceleration, or non-Cartesian sensor models. This flexibility makes EKF more suitable for real-world scenarios with non-linear motion or sensor readings.

3.4 ByteTrack for 3D Bounding

This approach uses ByteTrack, originally designed for 2D tracking, to track 3D bounding boxes by projecting them onto a 2D plane. The process follows a similar structure to Kalman Filter methods: receiving frames, associating detections, and updating trackers. The main difference is that ByteTrack uses a refined data association strategy developed for YOLOX.

3D to 2D Projection

Each 3D box $[x, y, z, \text{length}, \text{width}, \text{height}, \text{yaw}]$ is projected onto 2D as

$$(x_1, y_1, x_2, y_2)$$

by ignoring height and yaw. Dummy confidence scores are added to meet ByteTrack's input needs.

ByteTrack Association

ByteTrack performs multi-step association: first matching high-confidence detections, then refining with lower-confidence ones. This produces stable track IDs even if some detections are missed. It replaces the Hungarian algorithm used directly in Kalman Filter methods.

Speed-Aware Expansion

To help with fast-moving objects, boxes are expanded or shifted based on a simple velocity estimate:

$$(v_x, v_y) = (c_x - c_{x,\text{prev}}, c_y - c_{y,\text{prev}}),$$

where (c_x, c_y) is the box center. The box is then adjusted by a small fraction of this velocity, improving association in dynamic scenes.

Reconstruction in 3D

After ByteTrack finishes, 2D boxes with track IDs are matched back to their original 3D boxes using index alignment. This ensures each 3D bounding box is assigned a persistent ID across frames.

While ByteTrack excels at 2D tracking, projecting 3D boxes into 2D simplifies some spatial details. Still, this method is practical and efficient for 3D object tracking in autonomous driving.

3.5 Particle Filter for 3D Box Tracking

This method uses a dense Particle Filter (PF) to track 3D bounding boxes in LiDAR data. Each object is represented by $N = 1,000,000$ particles, with 6D states for position (x, y, z) and velocity (v_x, v_y, v_z) . Particles start from a Gaussian distribution around the detected bounding box, with zero initial velocities and equal weights.

Prediction and Update

In each frame, particles are predicted forward using a constant velocity model:

$$x_i^{(t)} = F \cdot x_i^{(t-1)} + \mathcal{N}(0, Q),$$

where Q adds noise to model uncertainty. After a new observation, weights are updated based on the Euclidean distance to the detected center, normalized to sum to 1. If weight diversity drops too low, resampling replicates high-weight particles to avoid degeneracy.

State Estimation and Track Management

The estimated object position is the weighted mean of the particles:

$$\hat{x} = \sum_{i=1}^N w_i \cdot x_i.$$

The `bb_tracking` class manages active, lost, and new tracks. Active tracks match to detections by nearest-neighbor association. Lost tracks are restored if a new observation matches within a threshold.

Bounding Box Output

Bounding boxes are visualized with `Open3D`, assigned unique UUIDs, and optionally include point cloud counts. Results are saved in JSON, with position, dimensions, rotation, UUID, and object state.

3.6 LSTM

This section presents an LSTM-based recurrent model for predicting 3D bounding box trajectories. The model uses past positions and velocities to predict future object centers. Two region-specific models are trained: one for right-lane motion ($y \in [-3, 7.5]$) and one for left-lane motion ($y \in (7.5, 20]$) to account for different traffic behaviors.

Data Preparation and Normalization

Bounding box data is loaded from JSON files. Trajectories are filtered by spatial bounds and split into:

- Past sequence: ($T_p = 5, 4$) with positions and computed velocities.
- Future sequence: ($T_f = 3, 2$) as displacement from the last past position.

Inputs and outputs are normalized using dataset statistics.

Model Architecture

The network uses a sequence-to-sequence LSTM structure:

- Five-layer LSTM encoder with dropout.
- A compressed vector is repeated to initialize the decoder.
- Five-layer LSTM decoder predicts future displacements.
- A dense layer maps to 2D displacements.

The model is trained with the Adam optimizer and MAE loss.

Region-Specific Modeling

Separate models are trained for the right and left lane regions to improve accuracy by handling differences in driving patterns.

Training and Evaluation

Models are trained for 100 epochs with a batch size of 256. Metrics include ADE, FDE, MAE, and RMSE to evaluate prediction accuracy.

Stepwise Prediction

During inference, the model predicts one step at a time, updating the input sequence with predicted positions and recalculated velocities.

Model Saving and Visualization

Models and normalization data are saved for future use. Training loss and prediction performance are visualized (see Figures 3.1 and 3.2).

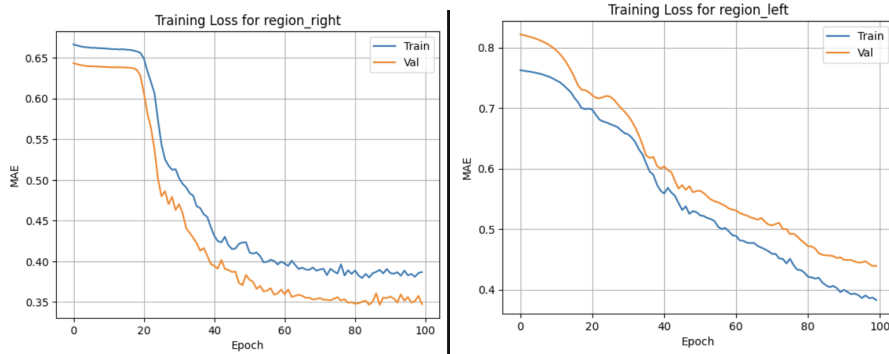


Figure 3.1: Training loss curves (MAE) for right and left regions.

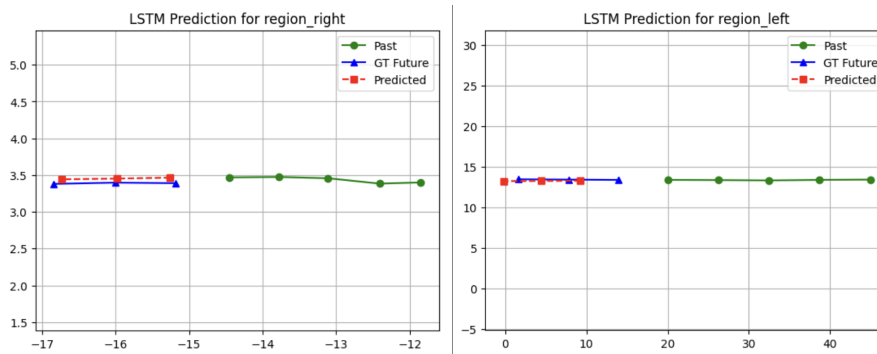


Figure 3.2: Predictions vs. ground truth for both regions.

Chapter 4

Implementation

Implementation Details

The full implementation is available in the following GitHub repository

The Link → <https://github.com/Mahmoudalshaikh00/An-Experimental-Study-on-ObjectTracking>

It includes four tracking methods: **Kalman Filter**, **Extended Kalman Filter**, **Particle Filter**, and **ByteTrack**, each implemented in a separate Jupyter notebook. Additionally, a trained **LSTM model** is included for trajectory prediction, along with code for LSTM training using BEV (Bird's Eye View) inputs.

In the GIF directory, output visualizations such as `traj_KF52.gif`, `traj_EKF52.gif`, and `traj_true52.gif` show the tracking performance of different filters compared to ground truth.

Chapter 5

Results

5.1 Tracking Results on CADC Dataset (Snowy Conditions)

This section compares Kalman Filter (KF), Extended Kalman Filter (EKF), Particle Filter (PF), and ByteTrack tracking performance using HOTA, IDF1, AMOTA, AMOTP, and IDS metrics. The “True” row in Table 5.1 shows ideal scores for reference.

Table 5.1: Tracking results (higher \uparrow is better; lower \downarrow is better).

Method	HOTA \uparrow	IDF1 \uparrow	AMOTA \uparrow	AMOTP \downarrow	IDS \downarrow
True	100	100	100	0.0	0
KF	43.2	63.2	55.8	83.7	490
PF	45.8	66.5	56.1	84.3	467
EKF	44.6	63.6	55.9	83.5	484
ByteTrack	53.0	70.0	60.0	86.2	209

5.1.1 Performance Summary

HOTA: ByteTrack leads (53.0), while PF (45.8) outperforms KF (43.2) and EKF (44.6).

IDF1: ByteTrack again scores highest (70.0), indicating best identity consistency. PF (66.5) improves over KF (63.2) and EKF (63.6).

AMOTA and AMOTP: ByteTrack has the best AMOTA (60.0). KF and EKF have

similar AMOTA scores (around 55.8–55.9) and slightly better AMOTP (83.5–83.7). PF’s AMOTA (56.1) and AMOTP (84.3) slip in between.

IDS: ByteTrack reduces the number of ID switches (209), PF (467) also achieves improvement compared to KF (490) and EKF(484).

5.1.2 Overall Analysis

ByteTrack maintains the strongest tracking despite backdrops in detection and ID stability, even if its higher AMOTP is due to a compromise in spatial precision. PF generally performs better than KF and EKF based methods with respect to different criteria due to the probabilistic tracking strategy. KF and EKF show a similar performance in snow condition and both were degraded compared to learning based ByteTrack and sampling based PF in maintaining identities and occlusions.

The evidence for stating that “Kalman-based approaches are least effective in adverse conditions” comes directly from the metrics in Table 5.1. KF and EKF have the lowest HOTA and IDF1 scores, showing they struggle more with tracking accuracy and identity consistency in snowy scenes.

5.2 Tracking Results on nuScenes Dataset (Clean Weather)

This section compares four tracking methods Kalman Filter (KF), Extended Kalman Filter (EKF), Particle Filter (PF), and ByteTrack on the nuScenes dataset in clear weather. Metrics include HOTA, IDF1, AMOTA, AMOTP, and IDS, with a “True” row for perfect tracking reference.

Table 5.2: Tracking results (higher \uparrow better; lower \downarrow better).

Method	HOTA \uparrow	IDF1 \uparrow	AMOTA \uparrow	AMOTP \downarrow	IDS \downarrow
True	100	100	100	0.0	0
KF	49.90	56.88	54.63	53.33	63
EKF	55.50	56.90	54.63	52.94	67
PF	56.35	57.00	54.56	52.92	65
ByteTrack	63.01	58.40	66.01	52.95	59

5.2.1 Performance Summary

ByteTrack consistently achieved the best performance in HOTA, IDF1, AMOTA, and IDS, confirming its ability to accurately track and maintain object identities.

The Particle Filter showed slight improvements over KF and EKF in HOTA and IDF1, due to its probabilistic motion modeling. AMOTP scores were nearly the same for all methods, indicating similar precision in 3D position estimates.

5.3 Failure Case Analysis

Kalman Filter and EKF: They fail when the boxes are too close to each other and overlap within the rectangular threshold. The tracker can't distinguish the objects because they're treated as one merged detection.

ByteTrack: It fails when boxes are close and overlap within the same `expansion_ratio`. It can't resolve the identities because it links tracks based on overlap, and overlapping boxes confuse the association step.

Particle Filter: It fails in the same way as Kalman Filter. Since it also uses a rectangular threshold for measurement updates, it can't separate close objects when their detections overlap.

5.4 Run-time

Kalman Filter and EKF are fast, usually taking 1 to 2 seconds.

ByteTrack takes 2 to 4 seconds because it has more complex data association and extra tracking logic.

Particle Filter with 1 million particles is very slow. It takes 15 to 20 minutes because it must evaluate each particle at each step.

Chapter 6

Discussion

This study evaluated the performance of four different object-tracking algorithms Kalman Filter (KF), Extended Kalman Filter (EKF), Particle Filter (PF), and ByteTrack in snowy weather conditions using LiDAR data. Each algorithm demonstrated specific strengths and limitations influenced by the complexity of environmental conditions, particularly snowy weather such as snow.

The results from Chapter 5 indicate that snowy weather conditions significantly degrade the performance of tracking algorithms. For instance, metrics such as HOTA, IDF1, and AMOTA revealed reduced accuracy and consistency across all methods when tested with the CADC dataset. This is consistent with the findings outlined in Section 1.1, where it was established that snowflakes and other environmental factors create noise and reduce sensor accuracy, presenting substantial challenges for LiDAR-based perception systems.

The KF and EKF showed relatively stable performance, benefiting from their ability to handle linear and slightly nonlinear motion models efficiently. However, their reliance on Gaussian noise assumptions and linearization limited their effectiveness in the complex and highly variable snow scenarios observed in the CADC dataset. As discussed in Chapter 2, while Kalman-based methods are computationally lightweight, they struggle under significant nonlinearities and erratic environmental noise.

The Particle Filter offered improved performance in snowy conditions by utilizing a set of hypotheses (particles) to represent state distributions. This capability allows PF to handle multi-modal uncertainties effectively, making it suitable for unpredictable movements and sporadic occlusions frequently encountered in snowy weather. This is consistent with previous research aspects mentioned in the line above in Section 1.1.4 that emphasized the merits of PFs in the contexts of high uncertainty and non-linear dynamics.

ByteTrack, a deep learning-based method, achieved the highest overall score on

the CADC dataset and performed best in terms of general applicability to low-confidence detection and occlusion. However, ByteTrack’s tracking is 2D box-level tracking by projecting 3D boxes into 2D space and associating them based on overlap. While this flattening process achieves impressive performance in short-term tracking, it does not fully solve the 3D tracking problem in crowded and complex 3D scenes.

A comparison of the nuScenes dataset (in good weather conditions) and the CADC dataset (insnow weather) showed a difference. All models performed significantly better on the nuScenes dataset with the overall scores for all metrics HOTA, IDF1, AMOTA, AMOTP, IDS consistently higher. This comparison highlights the significant impact of environmental conditions on the reliability of tracking. The reduced performance highlights that tracking algorithms need to be specifically fine-tuned or trained on datasets that fairly represent snowy weather conditions to retain an acceptable accuracy and reliability in real-world scenarios.

It’s important to note that this study did not include other adverse weather conditions like rain, fog, or dust, or alternative datasets such as the ithaca365 dataset (which includes snow, rain, and sun) or the Boreas Dataset (which covers clear, rain, and snow). The reason is that these datasets and conditions lack pre-annotated detection boxes needed for tracking. Tracking algorithms rely on pre-computed detections to generate tracks; without them, direct evaluation of tracking performance is not possible.

In summary, the study clearly indicates that deep learning-based methods like ByteTrack currently offer the best performance under challenging weather conditions, followed closely by particle filtering methods. Kalman-based filters, while computationally efficient, require enhancements to handle the complexity inherent in snowy weather scenarios effectively.

6.1 Societal Aspects

This thesis investigates object tracking for autonomous vehicles which challenges recent societal issues. Tracking systems also contribute to SDG 3 (“Good Health and Well-being”) by reducing accidents and enhancing road safety. They are also part the SDG 11 (“Sustainable Cities and Communities”) movement by providing safer and more efficient urban transportation. SDG 13 (“Climate Action”) is indirectly benefited with accurate prediction, leading to lesser fuel use and emissions.

ethically. . . first, accurate tracking is being “enforced” in service of sustaining a basic objective of preserving human life, which in turn respects something like SDG

9 (“Industry, Innovation and Infrastructure”) by responsibly advancing technology, on the net. In general, this thesis contributes in fulfillment of multiple SDGs through safety and sustainability, innovation, and healthier cities.

Chapter 7

Conclusion

In this thesis, studied how well object tracking algorithms cope with different challenging weather conditions and validated Kalman Filter, Extended Kalman Filter, Particle Filter and ByteTrack algorithms on the CADC dataset. The main issue that has been dealt with was the accurate and reliable management of autonomous vehicle localization systems in wheather snowy scenarios, which is crucial for the broader adoption of autonomous driving technologies.

The study confirmed that traditional tracking methods like KF and EKF experience significant performance degradation under harsh conditions due to their reliance on linear assumptions and Gaussian noise models. The Particle Filter provided notable improvements, handling uncertainty more effectively through a multi-hypothesis representation. However, ByteTrack, a deep learning-based approach, delivered the most robust results, maintaining higher accuracy and consistent object identification despite severe environmental challenges.

Comparison against performance on the nuScenes dataset (good weather) and the CADC dataset (snowy weather) showed that the performance decreased when there are bad visual conditions, using more specialized training and algorithm adjustments for real-world application requirements. The work presented in this paper is crucial for AUTONOMOUS vehicle perception systems to-ward safe and secure driving autonomously in severe weather conditions.

7.1 Future Work

Further testing on a wider range of real-world scenarios and more diverse datasets can help verify and increase the robustness of the methods. and more diverse datasets can help reveal unobserved constraints and enhance generalizability. In future work, it would also be interesting to test on other object classes such as

bicycles, pedestrians, and different datasets with other types of detections. By continuing to refine these areas, future research can improve autonomous vehicle safety and efficiency, making them more reliable in complex, real-world driving conditions.

Bibliography

- [1] X. Weng, J. Wang, D. Held, and K. Kitani, “3d multi-object tracking: A baseline and new evaluation metrics,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 10 359–10 366. DOI: 10.1109/IROS45743.2020.9341164.
- [2] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, “Multi-view 3d object detection network for autonomous driving,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6526–6534. DOI: 10.1109/CVPR.2017.691.
- [3] T. Yin, X. Zhou, and P. Krährenbühl, “Center-based 3d object detection and tracking,” in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 11 779–11 788. DOI: 10.1109/CVPR46437.2021.01161.
- [4] Y. Zhou and O. Tuzel, “Voxelnet: End-to-end learning for point cloud based 3d object detection,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4490–4499. DOI: 10.1109/CVPR.2018.00472.
- [5] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, “Pointpillars: Fast encoders for object detection from point clouds,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12 689–12 697. DOI: 10.1109/CVPR.2019.01298.
- [6] C. Wang, H. Huang, Y. Ji, B. Wang, and M. Yang, “Vehicle localization at an intersection using a traffic light map,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1432–1441, 2019. DOI: 10.1109/TITS.2018.2851788.
- [7] G. Guo, J. Liu, and X. Sun, “A model decomposition kalman filter for enhanced localization of land vehicles,” *IEEE Transactions on Vehicular Technology*, vol. 72, no. 8, pp. 10 013–10 023, 2023. DOI: 10.1109/TVT.2023.3254560.
- [8] A. Bertipaglia, M. Alirezaei, R. Happee, and B. Shyrokau, “An unscented kalman filter-informed neural network for vehicle sideslip angle estimation,” *IEEE Transactions on Vehicular Technology*, vol. 73, no. 9, pp. 12 731–12 746, 2024. DOI: 10.1109/TVT.2024.3389493.

- [9] Y. Liu, X. Fan, C. Lv, J. Wu, L. Li, and D. Ding, "An innovative information fusion method with adaptive kalman filter for integrated ins/gps navigation of autonomous vehicles," *Mechanical Systems and Signal Processing*, vol. 100, pp. 605–616, 2018, ISSN: 0888-3270. DOI: <https://doi.org/10.1016/j.ymsp.2017.07.051>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0888327017304211>.
- [10] Y. Jing, Q. Li, W. Ye, and G. Liu, "Development of a gnss/ins-based automatic navigation land levelling system," *Computers and Electronics in Agriculture*, vol. 213, p. 108 187, 2023, ISSN: 0168-1699. DOI: <https://doi.org/10.1016/j.compag.2023.108187>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0168169923005756>.
- [11] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, *Bytetrack: Multi-object tracking by associating every detection box*, 2022. arXiv: 2110.06864 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2110.06864>.
- [12] J. Liu, Y. Xie, Y. Zhang, and H. Li, "Vehicle flow detection and tracking based on an improved yolov8n and bytetrack framework," *World Electric Vehicle Journal*, vol. 16, no. 1, 2025, ISSN: 2032-6653. [Online]. Available: <https://www.mdpi.com/2032-6653/16/1/13>.
- [13] D. Gragnaniello, A. Greco, A. Saggese, M. Vento, and A. Vicinanza, "Benchmarking 2d multi-object detection and tracking algorithms in autonomous vehicle driving scenarios," *Sensors*, vol. 23, no. 8, 2023, ISSN: 1424-8220. [Online]. Available: <https://www.mdpi.com/1424-8220/23/8/4024>.
- [14] Y. Gladiensyah Bihanda, C. Fatichah, and A. Yuniarti, "Multi-vehicle tracking and counting framework in average daily traffic survey using rt-detr and bytetrack," *IEEE Access*, vol. 12, pp. 121 723–121 737, 2024. DOI: 10.1109/ACCESS.2024.3453249.
- [15] H. Zhang, R. Fang, S. Li, Q. Miao, X. Fan, J. Hu, and S. Chan, "Multi-camera multi-vehicle tracking guided by highway overlapping fovs," *Mathematics*, vol. 12, no. 10, 2024, ISSN: 2227-7390. [Online]. Available: <https://www.mdpi.com/2227-7390/12/10/1467>.
- [16] S. Gao, Y. Li, and H. Luo, "Detecting thyroid nodules along with surrounding tissues and tracking nodules using motion prior in ultrasound videos," *Computerized Medical Imaging and Graphics*, vol. 117, p. 102 439, 2024, ISSN: 0895-6111. DOI: <https://doi.org/10.1016/j.compmedimag.2024.102439>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0895611124001162>.
- [17] Y. Chang, J. Hu, and S. Xu, "Ote-slam: An object tracking enhanced visual slam system for dynamic environments," *Sensors*, vol. 23, no. 18, 2023, ISSN: 1424-8220. [Online]. Available: <https://www.mdpi.com/1424-8220/23/18/7921>.

- [18] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002. DOI: 10.1109/78.978374.
- [19] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial Intelligence*, vol. 128, no. 1, pp. 99–141, 2001, ISSN: 0004-3702. DOI: [https://doi.org/10.1016/S0004-3702\(01\)00069-8](https://doi.org/10.1016/S0004-3702(01)00069-8). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0004370201000698>.
- [20] G. G. Rigatos, "Extended kalman and particle filtering for sensor fusion in motion control of mobile robots," *Mathematics and Computers in Simulation*, vol. 81, no. 3, pp. 590–607, 2010, The Sixth IMACS Seminar on Monte Carlo Methods Applied Scientific Computing VII. Forward Numerical Grid Generation, Approximation and Simulation, ISSN: 0378-4754. DOI: <https://doi.org/10.1016/j.matcom.2010.05.003>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0378475410001515>.
- [21] G. Guo and S. Zhao, "3d multi-object tracking with adaptive cubature kalman filter for autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 512–519, 2023. DOI: 10.1109/TIV.2022.3158419.
- [22] F. Ngeni, J. Mwakalonge, and S. Siuhi, "Solving traffic data occlusion problems in computer vision algorithms using deepsort and quantum computing," *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 11, no. 1, pp. 1–15, 2024, ISSN: 2095-7564. DOI: <https://doi.org/10.1016/j.jtte.2023.05.006>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2095756424000072>.
- [23] T. Zhang, X. Chen, Y. Wang, Y. Wang, and H. Zhao, *Mutr3d: A multi-camera tracking framework via 3d-to-2d queries*, 2022. arXiv: 2205.00613 [cs.CV]. [Online]. Available: <https://arxiv.org/abs/2205.00613>.
- [24] H. Putra, H. H. Nuha, M. Irsan, A. G. Putrada, and S. B. I. Hisham, "Object tracking in surveillance system using particle filter and acf detection," in *2024 International Conference on Decision Aid Sciences and Applications (DASA)*, 2024, pp. 1–7. DOI: 10.1109/DASA63652.2024.10836358.
- [25] L. You, Y. Chen, C. Xiao, C. Sun, and R. Li, "Multi-object vehicle detection and tracking algorithm based on improved yolov8 and bytetrack," *Electronics*, vol. 13, no. 15, 2024, ISSN: 2079-9292. [Online]. Available: <https://www.mdpi.com/2079-9292/13/15/3033>.
- [26] M. Waqas and U. W. Humphries, "A critical review of rnn and lstm variants in hydrological time series predictions," *MethodsX*, vol. 13, p. 102946, 2024, ISSN: 2215-0161. DOI: <https://doi.org/10.1016/j.mex.2024.102946>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2215016124003972>.

- [27] Y. Fan, W. Zhang, W. Zhang, D. Zhang, and L. He, "A double-layer lstm model based on driving style and adaptive grid for intention-trajectory prediction," *Sensors*, vol. 25, no. 7, 2025, Cited by: 0; All Open Access, Gold Open Access. DOI: 10.3390/s25072059. [Online]. Available: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-105002277798&doi=10.3390%2fs25072059&partnerID=40&md5=852375432189f3353e67aafacc48a8ca>.
- [28] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 961–971. DOI: 10.1109/CVPR.2016.110.
- [29] F. Altché and A. de La Fortelle, "An lstm network for highway trajectory prediction," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 353–359. DOI: 10.1109/ITSC.2017.8317913.
- [30] L. Hou, L. Xin, S. E. Li, B. Cheng, and W. Wang, "Interactive trajectory prediction of surrounding road users for autonomous driving using structural-lstm network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 11, pp. 4615–4625, 2020. DOI: 10.1109/TITS.2019.2942089.