GENG4412 Engineering Research Project Part 2

Final Report

Uncertainty-Aware Sequential Modelling for End-to-End Autonomous Vehicles

Kate Kalugina
22931707
School of Engineering, University of Western Australia

Supervisor: Prof. Thomas Bräunl
School of Engineering, University of Western Australia

Co-Supervisor: Kieran Quirke-Brown
School of Engineering, University of Western Australia

Word count: 7,077

School of Engineering
University of Western Australia

Submitted: 15 October 2025

DECLARATION OF CONTRIBUTION

My contribution

- Design and implementation of the SparseResNet9D_Mamba_evidential model
- Adapted training script accounting for the new model
- The development of inference script
- The design and execution of the project's evaluation
- The visualisation scripts for the analysis and interpretation of the results

Contribution of others

- Base model was developed by Patrick Reidel
- Data preprocessing scripts for sorting data were developed by Zhihui Lai
- Base training script was developed by Kieran Quirke-Brown
- Main developers of the software stack on nUWAy2 were Kieran Quirke-Brown, Zhihui Lai

Use of AI tools

I have used AI tools in the preparation of my report: Yes

Details of how AI tools were used: AI was used to check grammar, vary similar words of high use, and keep style even throughout the report.

In accordance with University Policy, I certify that:

The above information is correct, and the attached work submitted for assessment is my own work and that all material drawn from other sources has been fully acknowledged and referenced.

Student signature  Date 14/10/25

Supervisor confirmation

To the best of my knowledge, the student's contribution outlined above is correct.

Supervisor signature  Date 14/10/25

Project Summary

Introduction and Motivation

The transition of autonomous driving (AD) from simulation to real-world application remains a significant challenge, hampered by the public distrust and cautious regulatory frameworks. A primary limitation of current systems is their lack of temporal context, leading to reactive movements and inability to handle complex scenarios smoothly. Their failure to quantify predictive uncertainty undermines their reliability and safety. This project addresses these critical gaps by developing a novel AD system that integrates temporal sequence modelling with real-time uncertainty estimation, aiming to bridge the gap between theoretical performance and safe, real-world deployment.

Project Objectives

This research was guided by three core objectives:

1. Develop a Mamba-based temporal architecture with an inference speed of 20 Hz, capable of replicating smooth, safe, human-like driving.
2. Integrate an evidential regression framework that quantifies both aleatoric (sensor noise) and epistemic (model confidence) uncertainties.
3. Utilise the estimated uncertainty to dynamically adapt driving behaviour for enhanced safety.

Approach and Methodology

The project developed SparseResNet9D_Mamba_evidential, an end-to-end neural network built upon a baseline LiDAR model. Key methodological innovations:

- Shifting from single to 10-frame LiDAR sequence processing.
- Using a selective state-space model to efficiently capture long-range temporal dependencies with linear computational complexity.
- Replacing deterministic outputs with a head predicting the parameters of a Normal Inverse-Gamma distribution, enabling simultaneous estimation of driving commands and their associated aleatoric and epistemic uncertainty.

The model was trained on real-world data from the nUWAy2 vehicle, using a loss function that penalises overconfident predictions. This constitutes, to the best of current knowledge, the first integration of the Mamba architecture with evidential deep learning for end-to-end autonomous vehicles.

Key Findings and Future Work

Offline tests demonstrated superior performance: an 86% lower test loss, smoother control in complex scenarios and interpretable saliency maps comparing to a baseline model. The new model also successfully quantified predictive uncertainty. However, real-world deployment was hindered by latency from data buffering, which destabilised the control loop and prevented testing uncertainty-aware control. This project successfully provided the value of temporal and uncertainty-aware modelling. Future work must prioritise latency mitigation through predicate control and pipeline optimisation to unlock the model's full potential for safe, real-world AD.

Acknowledgements

I would like to express my appreciation to my supervisor, Prof. Thomas Bräunl, for providing me with an opportunity to work on this challenging project. I would also like to express my gratitude to Kieran Quirke-Brown for his extensive assistance and mentorship. His support and insightful feedback were instrumental in navigating the complexities of this research. I am deeply thankful to Zhihui Lai for facilitating the data collection and testing at the Eglinton site, enabling real-world validation that was crucial for this work.

Table of Contents

DECLARATION OF CONTRIBUTION i
Project Summary ii
Acknowledgements iii
List of Figures iv
List of Tables v
Nomenclature vi
1. Introduction 1
1.1 Introduction 1
1.2 Background 1
1.3 Project Objectives 3
2. Model Formulation 5
2.1 Base Model Architecture 5
2.2 Model Modifications 6
2.3 Model Configuration 7
2.4 Software Tools Used 10
2.5 Model Input Configuration 11
2.6 Model Validation and Final Selection 11
2.7 Limitations 12
3. Results & Discussion 14
3.1 Training and Validation Loss 14
3.2 Saliency 15
3.3 Ground Truth Scene Analysis 19
3.4 Ground Truth Time Analysis 20
3.5 Uncertainty vs Driving Prediction in Real Life Testing 26
4. Conclusions & Future Work 29
References 30
Appendices 32
Appendix A: Literature Review 32
Appendix B: Saliency Map with Different Scenarios 43
Appendix C: Evaluation of Transformation Matrices 47

List of Figures

Figure 1.1: AD Architectures (Li et al., 2025) 2
Figure 2.1: Base Model with a Single Input Frame (Riedel, 2025) 5
Figure 2.2: SparseResNet9D_Mamba_evidential Model Architecture 7
Figure 2.3: Point Cloud vs Voxel (Gao et al., 2022) 8
Figure 2.4: Voxelising with different resolutions of 0.1 m (A), 0.5 m (B) and 1 m (C) (Lecigne et al., 2017) 8
Figure 2.5: Mamba Architecture (Gu & Dao, 2023) 10
Figure 3.1: Baseline Model, Roundabout with Obstacles 15
Figure 3.2: SparseResNet9D_Mamba_evidential Model, Roundabout with Obstacles 16
Figure 3.3: Baseline Model, Roundabout with Obstacles 17
Figure 3.4: SparseResNet9D_Mamba_evidential Model, Roundabout without Obstacles 18
Figure 3.5: Good Alignment of Prediction with the Ground Truth. 19
Figure 3.6: Oversteering Comparing to the Ground Truth 20

Figure 3.7: Understeering Comparing to the Ground Truth 20
Figure 3.8: Driving Command of the Baseline Model Comparing to Ground Truth, Lane Following 21
Figure 3.9: Driving Command of the New Model Comparing to Ground Truth, Lane Following ..22
Figure 3.10: Driving Command of the Baseline Model Comparing to Ground Truth, Roundabout 23
Figure 3.11: Driving Commands of the New Model Comparing to Ground Truth, Roundabout.....24
Figure 3.12: Path with Multiple Roundabouts and Turns 25
Figure 3.13: Velocity and Corresponding Uncertainty in Real Life Driving 26
Figure 3.14: Uncertainty Trend vs Velocity in Real Life Driving.....27
Figure A.1: Inference throughput on A100 80GB (prompt length 2048) (Gu & Dao, 2023).....33
Figure A.2: Gantt chart for the Project, Part 1 41
Figure A.3: Gantt chart for the Project, Part 2.....42
Figure A.4: SparseResNet9D_Mamba_evidential Model, Lane Bay Pass.....43
Figure A.5: SparseResNet9D_Mamba_evidential Model, Lane Following.....44
Figure A.6: SparseResNet9D_Mamba_evidential Model, Intersection 45
Figure A.7: SparseResNet9D_Mamba_evidential Model, Intersection with Obstacle 46

List of Tables

Table 3.1: Saliency Point Distribution by Intensity 18
Table A.1: Comparison of Uncertainty Methods in Deep Learning 34

Nomenclature

AD	Autonomous Driving
AV	Autonomous Vehicle
BNN	Bayesian Neural Network
GT	Ground Truth
LiDAR	Light Detection and Ranging
LSTM	Long Short-Term Memory
N	Number of tokens
NN	Neural Network
RNN	Recurrent Neural Network
ROS	Robot Operating System

α	Shape Parameter
β	Scale Parameter
υ	Degrees of Freedom
γ	Prediction Mean
ω	Trainable Parameters of the Neural Network
$\mathcal{L}_i^{\mathrm{NLL}}$	Negative Log-Likelihood
$\mathcal{L}_i^{\mathrm{R}}$	Evidence Regulariser
T_{front}	Transformation Matrix for Front LiDAR
T_{rear}	Transformation Matrix for Rear LiDAR

1. Introduction

1.1 Introduction

Despite its advancements in simulation, real-world autonomous driving (AD) remains a challenge. In Perth, the technology has yet to be integrated into public infrastructure or the lifestyle of its residents, leading to public distrust. This scepticism stems from safety incidents and the perceived unpredictability of autonomous systems around the world. Consequently, this translates to governments adopting cautious regulatory frameworks, industries delaying commercialisation, and the public expressing persistent safety concerns.

Although promising models exist, these are not temporally dependent, meaning they only respond to the current input without considering previous timesteps. A complex task, such as driving, requires past context to ensure smooth movements and informed decisions.

Humans naturally leverage historical experience when making decisions, adjusting their behaviour based on previous interactions and road conditions. Early autonomous vehicles (AV) used a simple neural network to map an input to a command (Pomerleau, 1988), without any capacity for long-term behaviour modelling. Traditional machine learning models, especially the ones based on static perception without temporal context, struggle to replicate this capability. This results in jerky movements, inconsistent decision-making, and reduced safety in real-world scenarios, as demonstrated by the model (Riedel, 2025) upon which this project is built.

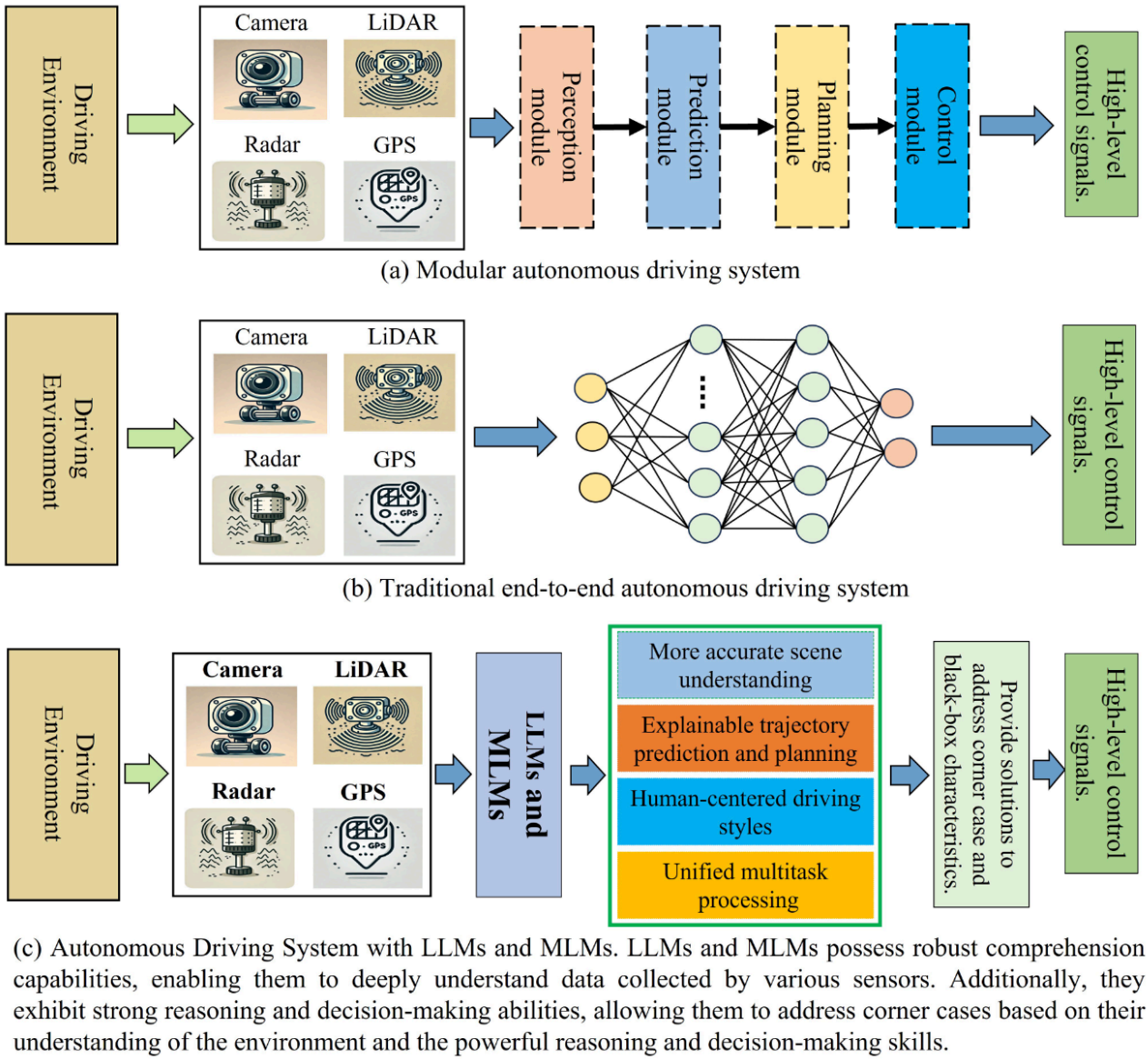
This research addresses the inability to model sequential dependencies effectively and the failure to quantify predictive uncertainty. Quantifying uncertainty is critical for assessing the reliability of predictions. Even a well-known route can create uncertainty if the data the model takes in is not clear. By developing a computationally efficient architecture that incorporates temporal depth and uncertainty awareness, this research aims to create a more robust and reliable AD system, moving beyond simulation and closer to safe real-world application. Such an advancement would not only improve vehicle performance but also provide interpretable signals of model confidence. In the near future, this will enable safer and broader acceptance of human-machine handovers.

1.2 Background

Recent progress in machine learning has accelerated AD development, but most advances remain confined to simulation, creating a significant gap between theoretical performance and practical application. Bridging this gap requires tackling two crucial components: system design and sensor selection.

Architecturally, the three most common paradigms defining AD include: modular, end-to-end and hybrid, as can be seen in Figure 1.1. The modular pipeline decomposes perception, prediction, planning, and control driving into sequential stages. This approach is used for its ease of interpreting and debugging, but is error-prone due to the construction of multiple modules.

The end-to-end system predicts driving commands from raw point cloud data without relying on intermediate modules such as object detection or path planning. It can learn complex behaviours directly from data. Two main drawbacks of such an approach are that it is hard to debug and that it requires a substantial amount of real-world data.



(c) Autonomous Driving System with LLMs and MLMs. LLMs and MLMs possess robust comprehension capabilities, enabling them to deeply understand data collected by various sensors. Additionally, they exhibit strong reasoning and decision-making abilities, allowing them to address corner cases based on their understanding of the environment and the powerful reasoning and decision-making skills.

Figure 1.1: AD Architectures (Li et al., 2025)

Combining the strengths of both the modular and end-to-end pipelines creates a hybrid system. This approach offers a smarter, safer driving system, resulting in more accurate scene understanding, human-like driving styles and explainable trajectory predictions, but is complex to design and maintain. In contrast, end-to-end implementation is much more efficient and still offers a safe driving system, hence its application in this project.

Alongside these architectural considerations, a robust sensor must be used. Among these, LiDAR has emerged as particularly promising due to its reliable performance in real-world situations where other sensors may fail. Compared to cameras, LiDAR provides accurate 3D spatial data and is immune to visual noise from lighting or colour variation. For AV, driving in a 3D space requires them to have

vision in all directions, which is not possible with a single camera (Mugunthan et al., 2020). Although cameras can differentiate colours, which is beneficial for object recognition, they struggle with depth estimation. They are also prone to disconnections, as observed during real-world visits to the shuttle bus in Eglinton, where safety LiDARs were required as a fallback.

Although a strong alternative, LiDAR's point cloud data is dimensionally high and sparse, making it computationally intensive to process. Each point cloud frame is a raw output of LiDAR sensors, containing unstructured, irregularly spread 3D points in space. This challenge is offset using voxelisation, the process of converting unstructured data into a regular 3D grid of voxels. This enables efficient sparse tensor operations, preserves spatial structure, which reveals environmental features, and reduces processing cost. For this reason, LiDAR was the chosen sensor for this study.

Once features are extracted, the model needs to understand these sequences. Currently, transformer-based architectures are the state of the art for sequence modelling, but they are computationally heavy due to their quadratic complexity with sequence length (Vaswani et al., 2017). For resource-constrained platforms, such as AV, processing long sequences of high-dimensional LiDAR data with optimal computational load is vital for both training and real-time performance. This creates a need for a model that retains the functionality of transformers but is more efficient. The Mamba architecture presents a solution through its selective state space model, which maintains linear complexity while achieving five times higher inference throughput than similarly sized transformers (Gu & Dao, 2023). This makes Mamba ideal for real-time LiDAR data processing.

Most autonomous systems provide deterministic outputs, specifically commands containing velocity and steering. Uncertainty plays a critical role in real-world environments, meaning models must quantify their confidence in their predictions. A common method, Monte Carlo Dropout offers a lightweight approximation of Bayesian inference (Gal & Ghahramani, 2016) but introduces latency. Ensembles capture both aleatoric (sensor noise) and epistemic (model confidence) uncertainties, but they are memory-intensive (Lakshminarayanan et al., 2017), making them unsuitable for an on-board computer. This project instead employs evidential regression (Amini et al., 2020), which estimates both aleatoric and epistemic uncertainties in a single forward pass, thereby improving computational efficiency and making it suitable for real-time systems.

For further information regarding the trajectory prediction methods or uncertainty estimation, please refer to Appendix A.

1.3 Project Objectives

This research aims to narrow the margin between theoretical AD models and real-world deployment by integrating temporal coherence and uncertainty quantification into a LiDAR-based AD system. The hypothesis states, incorporating Mamba-based temporal modelling and probabilistic uncertainty estimation into the decision-making and training pipeline, will enhance the performance of the AD, helping with the safety of the vehicle. To reach this, the research pursues the following objectives:

- Develop a Mamba-based temporal architecture with inference speed of 20 Hz, capable of replicating smooth, safe, human-like driving.

- Integrate uncertainty estimation within the AD pipeline to support principled decision-making under uncertainty.

- Develop uncertainty aware control to adapt driving behaviour.

Achieving these goals will enhance public safety by reducing human error, which is currently responsible for approximately 90% of road crashes (National Transport Commission, 2017, as cited in Australian Government Department of Infrastructure, Transport, Regional Development and Communications, 2021). For the industry, this will provide a framework for developing more reliable, self-sufficient systems that can operate safely in real-life conditions. Scientifically, it will advance the state of the art in spatiotemporal modelling by demonstrating practical application of the Mamba architecture and evidential regression within a safety-critical, real-world domain.

2. Model Formulation

2.1 Base Model Architecture

The original LiDAR-based end-to-end model (Riedel, 2025) serves as a baseline architecture for this research. As shown in Figure 2.1, the model consists of several components: a sparse convolution backbone, a Mamba-based block, and a linear projection head that outputs deterministic velocity and steering commands. The input is a single voxelised point cloud created from merging the front and rear Velodyne VLP-16 LiDAR sensors.

Feature extractor starts with a sparse 3D convolution with batch normalisation and Rectified Linear Unit (ReLU) activation. Normalisation is used to standardise the input for a more stable training, while the activation introduces non-linearity by setting negative values to zero. Afterwards, three residual blocks are applied, with each block using two sparse convolutions, another normalisation, an activation, and an identity shortcut that adds unprocessed input to the output to preserve gradients. Each of these blocks are followed by a downsampling (stride of two), while channels are doubled to prevent information loss.

The input to the two Mamba blocks is put through a linear layer with layer normalisation and an activation. The selective scan mechanism combines current input with past states and uses skip connections, allowing spatial information to bypass layers, ensuring no information is permanently lost.

The last block consists of three dense layers: linear layers with batch normalisation and an activation. The final output is a single steering angle and velocity command per input.

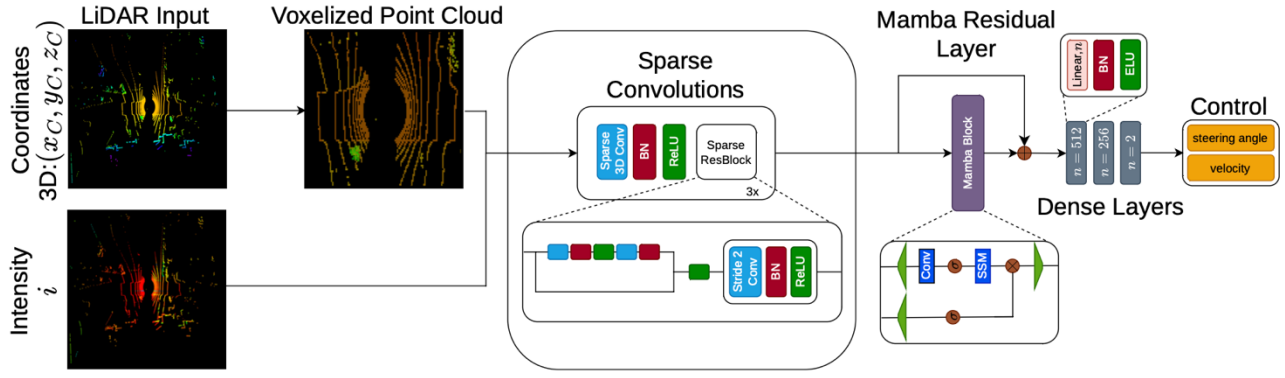


Figure 2.1: Base Model with a Single Input Frame (Riedel, 2025)

The base model shows promising results running at 20 Hz on the NVIDIA Jetson AGX Orin edge hardware installed on the nUWAy2. The model shows weak performance on turns and roundabouts, requiring 14 human interventions on a 4.2 km route (Riedel, 2025). This lightweight spatially focused network processes one LiDAR scan at a time, lacking meaningful temporal understanding. Despite including Mamba in the architecture, the input for this model is only a single point cloud frame, which is a potential contributor for the necessary interventions. The model's robustness is enhanced by

transforming the unused potential into genuine temporal learning with significant modifications in the following key areas.

2.2 Model Modifications

2.2.1 Enhanced Feature Extraction

A stem layer, comprising of a sparse 3D convolution, is introduced as an initial feature projection. Now the feature extractor consists of residual blocks only. This introduces computational efficiency since early convolution reduces sparse tensor density. The stem layer highlights the major scene features first, enabling the residual blocks to focus on refining semantically meaningful representation.

2.2.2 Temporal Sequence Integration

The most significant change to the model is the shift from a single-frame input to a 10-frame one. The model now processes a sequence of multiple consecutive voxelised point clouds stored in a sliding window buffer. To process this sequence, the sparse convolution backbone extracts features from each frame independently. These feature maps are collapsed into a 1D vector via global average pooling and stacked to form a temporal sequence tensor.

2.2.3 Uncertainty Quantification

The deterministic projection head is replaced with an evidential regression head. This allows the model to make predictions with aleatoric and epistemic uncertainty. Aleatoric uncertainty refers to natural ambiguity in the sensory input or environment itself, such as noise in a LiDAR point cloud, for example, due to rain. Epistemic uncertainty reflects the model's lack of knowledge due to encountering a scenario that was absent in its training. These are included for scenarios when human intervention is needed, for example, to retrain a model for a specific course or to fix the input data.

The model predicts four parameters of a Normal Inverse-Gamma distribution for velocity and steering: mean γ, degrees of freedom υ, shape α and scale β. The Tanh activation function is retained for the prediction mean to bound the control commands. Softplus activations are applied to the other parameters, ensuring strict positivity.

The implementation of evidential head necessitates the removal of batch normalisation, dropouts and the activations of some linear layers. Batch normalisation standardises the activations to zero mean and unit variance, which can distort the natural scale and relationships between the evidence parameters υ, α and β. For example, β can be greater than other parameters indicating very noisy data; batch normalisation would supress this. Dropout, a source of epistemic uncertainty, is redundant since the evidential framework explicitly captures this form of uncertainty. Non-linear processing is handled by Mamba layers, while the final linear layer is for mapping the features to evidential parameters. Softplus on the final layer is used instead of Exponential Linear Unit (ELU) since the model needs to guarantee a smooth, strictly positive output, while ELU keeps negative values.

2.3 Model Configuration

The SparseResNet9D_Mamba_evidential (refer to Figure 2.2) is an end-to-end neural network that is based on spatiotemporal sequence processing on sparse point cloud data. It is designed to process sequential 3D data, and it predicts control outputs with uncertainty quantifications.

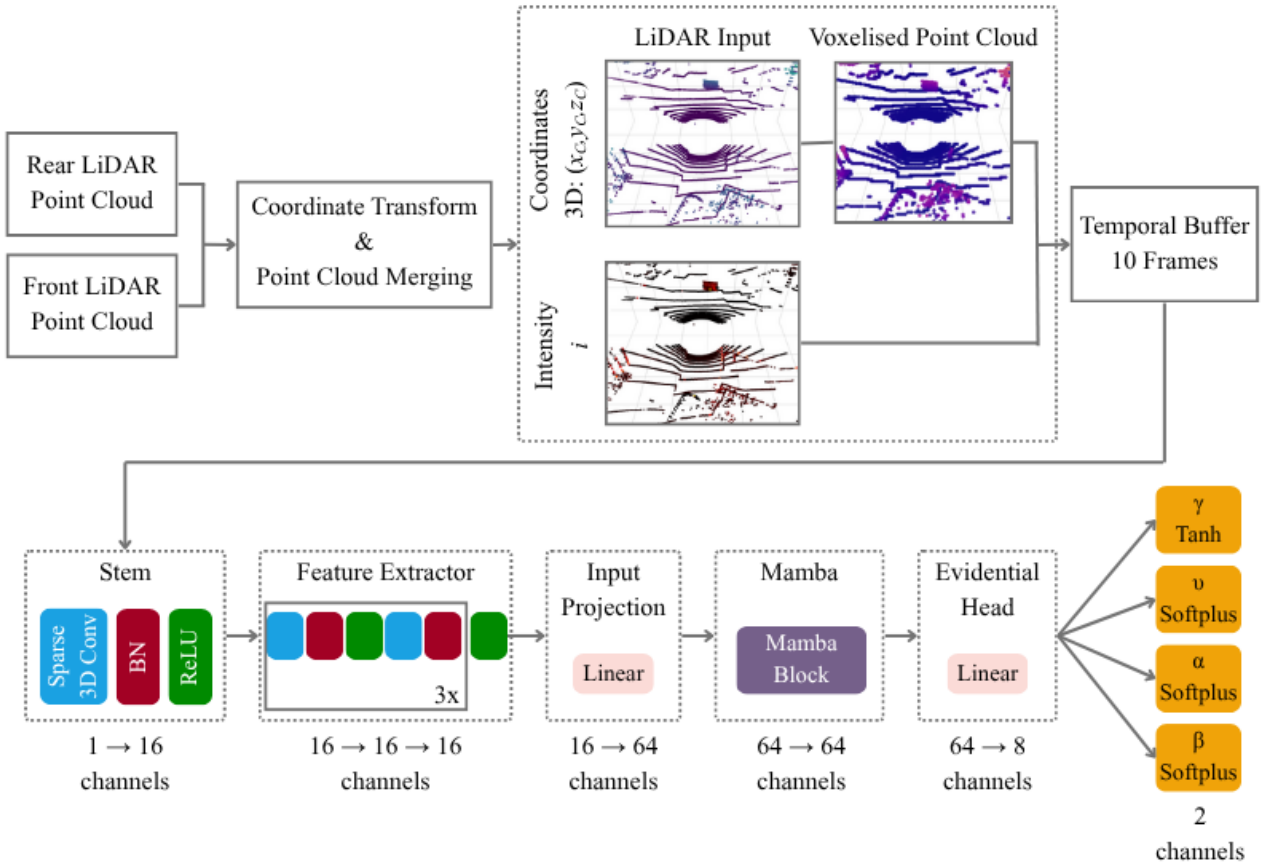


Figure 2.2: SparseResNet9D_Mamba_evidential Model Architecture

2.3.1 Data Preprocessing

The raw data points from LiDARs are transformed from the sensor frame to the vehicle's base frame using the sensors' calibrated extrinsic parameters. Front, T_{front}, and rear, T_{rear}, transformation matrices are as follows:

$$T_{\text{front}} = \begin{bmatrix} -\cos(8°) & 0 & -\sin(8°) & -1.689 \\ 0 & -1 & 0 & 0 \\ -\sin(8°) & 0 & \cos(8°) & 0.9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

$$T_{\text{rear}} = \begin{bmatrix} \cos(8°) & 0 & \sin(8°) & 1.689 \\ 0 & 1 & 0 & 0 \\ -\sin(8°) & 0 & \cos(8°) & 0.9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

Refer to Appendix C for the calculation and evaluation of the above matrices.

The transformed points from both sensors are merged. For temporal sequence modelling, the system maintains a sliding window buffer of the 10 most recent consecutive point cloud frames. Each frame undergoes a quantisation with 0.2 m voxels and a sparse tensor representation to create a structured format suitable for neural network processing.

Voxel grid downsampling (Figure 2.3) is a widely adopted technique that improves computational and storage efficiency by converting raw points into a 3D regular grid cell. Using a sparse_quantize function from TorchSparse, raw coordinates are mapped onto a discrete grid by rounding each position to its nearest grid cell based on a voxel size. Figure 2.4 demonstrates how a voxel size affects information. Smaller voxels preserve fine geometric features by assigning points to more densely spaced grid cells, however, they exponentially increase memory usage. Larger voxels merge points over broader regions but risk the loss of critical features. If multiple points fall into the same voxel, they are collapsed into a single representative location, removing redundancy and ensuring spatial sparsity.

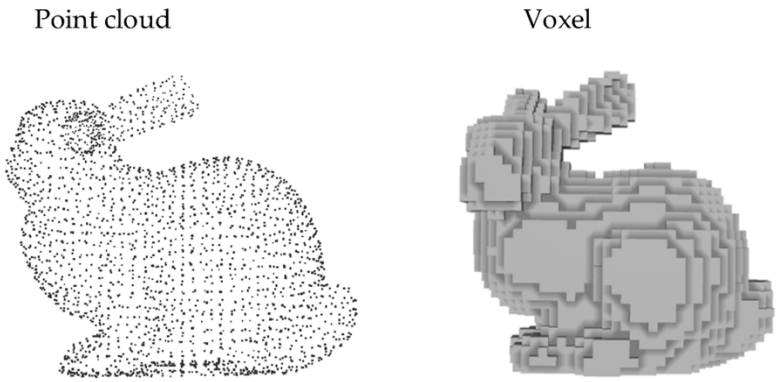


Figure 2.3: Point Cloud vs Voxel (Gao et al., 2022)

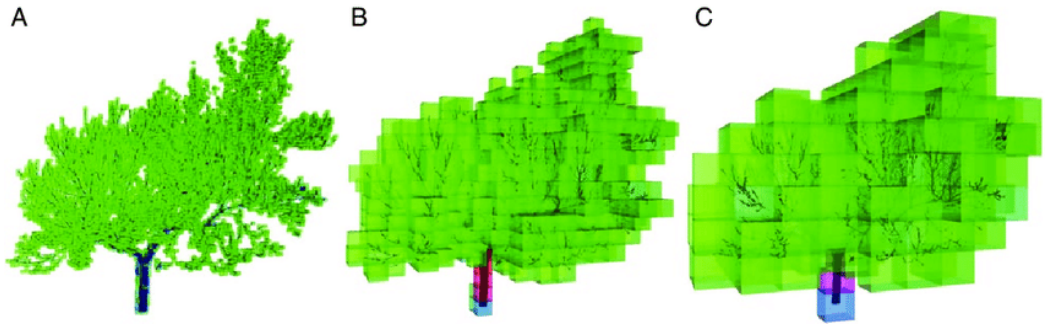


Figure 2.4: Voxelising with different resolutions of 0.1 m (A), 0.5 m (B) and 1 m (C) (Lecigne et al., 2017)

2.3.2 Model Forward Pass

The model takes a sequence of 10 sparse tensors and feeds it through the convolution backbone. It processes each timestep independently and identically. It first passes through the stem layer projecting features onto a 16-channel space. This is followed by groups of residual blocks performing feature extraction.

All the processed information is aggregated into a single feature vector per timestep using global average pooling. The output at this stage is a single tensor.

Single tensors are stacked into a sequence, forming a temporal tensor which is further projected into a single Mamba layer using a linear layer. The Mamba' selective scan captures temporal dependencies across the sequence, modelling evolution of 3D scene.

The output from the Mamba layer is passed to the evidential head. The output consists of four parameters for velocity and steering for the future 10 steps. The predicted γ serves as a driving command, while uncertainty is characterised by concentration υ, tail heaviness α, and variance β. The velocity is an average of these future steps to improve smoothness, while steering is not aggregated, rather only the first prediction is taken in order to account for turns.

2.3.3 Training and Loss Function

Training was modified to suit the multi-frame input. Data augmentations are applied consistently across all frames to maintain consistency.

The model is trained using a multi-task loss that maximises the model evidence to ensure good fit to observed data and penalises overconfident wrong predictions. The trainable parameters of the neural network, ω, are automatically adjusted during training to minimise the loss:

$$\mathcal{L}_i(\omega) = \mathcal{L}_i^{\mathrm{NLL}}(\omega) + \lambda \mathcal{L}_i^{\mathrm{R}}(\omega) \tag{2.3}$$

The first component corresponds to the negative log-likelihood, $\mathcal{L}_i^{\mathrm{NLL}}$, ensuring a good fit to the training data (Amini et al., 2020):

$$\mathcal{L}_i^{\mathrm{NLL}}(\omega) = \frac{1}{2}\log\left(\frac{\pi}{\upsilon}\right) - \alpha\log(\Omega) + \left(\alpha + \frac{1}{2}\right)\log((y_i - \gamma)^2\upsilon + \Omega) + \log\left(\frac{\Gamma(\alpha)}{\Gamma\left(\alpha+\frac{1}{2}\right)}\right) \tag{2.4}$$

where $\Omega = 2\beta(1 + \upsilon)$.

The second component is an evidence regulariser, $\mathcal{L}_i^{\mathrm{R}}$, (Amini et al., 2020), which penalises overconfidence when prediction errors are large:

$$\mathcal{L}_i^{\mathrm{R}}(\omega) = |y_i - \gamma| \cdot (2\upsilon + \alpha) \tag{2.5}$$

The coefficient λ is tuneable and was chosen to be 0.01. The selection of the parameter was guided by the established literature on Evidential Deep Learning (Amini et al., 2020) for real-world data. The parameter controls the trade-off between the data fit and the penalty on incorrect evidence.

2.4 Software Tools Used

Due to hardware constraints and an emphasis on open-source libraries, there was a scarce selection of software tools considered, of these, the ones used needed to be computationally efficient, compatible and easy to integrate.

PyTorch serves as the core deep learning framework. It provides support for GPU-accelerated tensor operations, which is essential for training deep learning models with substantial amounts of 3D data.

TorchSparse is a high-performance computing library used for efficient 3D sparse convolution on point clouds directly on GPU. Processing LiDAR point clouds as sparse tensors is dramatically more computationally and memory efficient than using dense voxel grids.

Mamba is a sequence modelling architecture based on a selective state space model, chosen for its linear complexity in sequence handling. Due to its selective scanning mechanism, the model maintains temporal context within each processed sequence by evolving a hidden state that accumulates relevant information from previous frames. As shown in Figure 2.5, the architecture of a Mamba block begins by projecting each frame's features linearly into a hidden state space as two separate branches. In one branch, a convolutional layer mixes local temporal information across nearby frames to capture short-term patterns, followed by an activation. At this stage, the core state space model block transforms the sequence using a continuous-time state equation that efficiently models long-range dependencies across all frames by updating a hidden state with learned dynamics. The other branch is also activated, and the outputs of both branches are combined via multiplication before the gated result is passed through a final linear projection to produce an output.

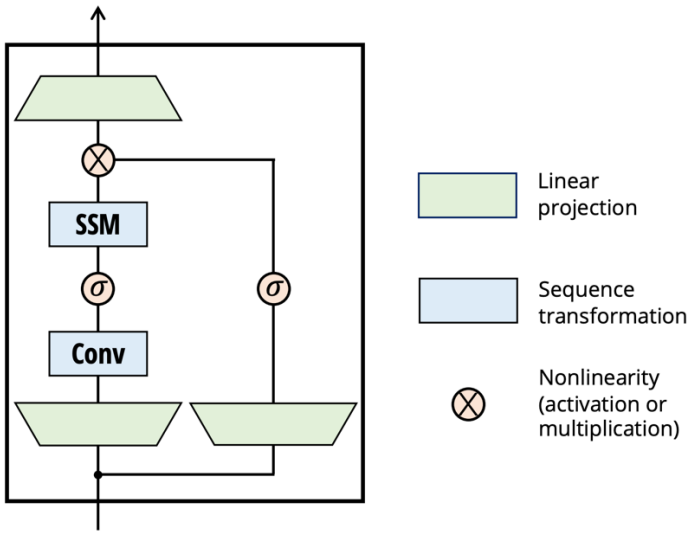


Figure 2.5: Mamba Architecture (Gu & Dao, 2023)

Lastly, the system is integrated within Robot Operating System (ROS) 2 Humble, an open-source software framework used for communication between different components of the autonomous system including sensor data and vehicle control. The deployment target is an AV, which is itself built on ROS2. It is based on service/client interactions to exchange data, serving as inputs and outputs for the system. The framework allows the model to ingest LiDAR data from a ROS topic,

publish its predicted driving commands back to the control system and integrate with other critical modules.

2.5 Model Input Configuration

The project utilised the nUWAy2 vehicle, located in Eglinton, Western Australia, which, due to restrictions, operated with a speed limit of 20 km/h. This suburban test environment featured narrow, single-laned roads with closely positioned housing, numerous turns, intersections and roundabouts. The route also largely featured roadside curbing, which if recognised by safety LiDARs as being too close, would stop the vehicle accordingly.

The input to the model is a temporal sequence of 10 LiDAR point clouds, which covers 0.5 seconds of the drive, represented as a sparse tensor object list.

Cropping imposes a spatial boundary, where the model is blind to anything outside of the predefined grid, which is 50 m in forward direction, 30 m back, right and left, and 10 meters up. Cropping manages potentially unbounded irrelevant input, creating immediate region of interest. The sensor setup provides sufficient coverage of the ego-vehicle's surroundings to make driving decisions. Front and back LiDARs leave gaps in coverage on both sides, these blind spots are a physical constraint that the model must learn to handle.

2.6 Model Validation and Final Selection

The final model was determined through an iterative training process, and quantitative and qualitative analysis, ensuring the model converged towards a robust solution.

2.6.1 Convergence Monitoring

Model convergence is primarily monitored using training and validation loss, where the evidential regression loss is the key metric. A model is considered to have converged when the validation loss plateaus, however, if the validation loss does not improve for over 10 epochs, the training must be stopped prematurely and the parameters adjusted.

2.6.2 Saliency Computation

The model's spatial perception was analysed using comprehensive saliency mapping to visualise the regions of the input point cloud that most strongly influence the driving command. Perturbation analysis was used across all points in the processed point cloud. For the best results, the saliency map should contain fewer strongly activated points accompanied by clustered areas of mid importance, often representing dynamic and static obstacles.

Point-based saliency is computed by perturbing individual points in the point cloud and measuring the change in the predictions. For each point, the intensity is set to zero, and the model is rerun. The observed output change is compared to the baseline prediction, from when all points exhibited full intensity. The absolute difference in prediction values quantifies the importance of each point, with

larger changes indicating higher saliency. These saliency scores are normalised and visualised using colour mapping to distinguish spatial regions most influential in the model's steering decisions.

The density-based heatmap collaborates the spatial patterns by aggregating point-level saliency into regional importance maps. The heatmap can reveal trending and clustered data, whereas saliency maps highlight specific points deemed important. Visualising thousands of individual points means saliency maps can sometimes be hard to interpret, making heatmaps a valuable addition.

2.6.3 Ground Truth Analysis

The model's performance was quantitatively evaluated by comparing its driving command predictions against the ground truth (GT) derived from human driving. This GT was sourced from prerecorded ROS bags, which included a front camera view as well as driving commands. To ensure a thorough assessment, the model was tested using unseen controlled data. Analysis of the GT driving commands establishes the benchmark for smooth predictable vehicle control. It can be used to evaluate the model's abilities when compared to human standards.

This test utilised LiDAR and camera data to display the model's predictions. LiDAR point cloud served as the model input, while the camera enabled visual assessment of the output against the GT values: the velocity and steering collected during the human drive.

The above validation strategies ensured that the model's performance was measured thoroughly, providing a robust comparison to the GT and supporting real-world implementation.

2.7 Limitations

The model expects an input of 10 timesteps in a sequence and cannot dynamically process variable-length sequences. The fixed cropping assumes areas outside this boundary do not affect the vehicle.

The model's performance is highly dependable on precise sensors calibration. If the accuracy of transformation matrices is affected, meaning miscalibration in the sensor's extrinsic parameters, the world model would be misaligned, leading to the model acting on incorrect spatial information.

The model's capabilities are confined to the specific scenarios present in its training data. These include lane following, bay passing, intersection, and roundabout manoeuvres (driving straight through and performing right turns). It possesses no learned ability for other critical driving tasks, such as lane changing or U-turns. This limited skill range means the model would be unable to recover from its own errors, or outside intervention, if they were to lead it outside of its trained scenarios.

Heavy rain can introduce significant noise into the point cloud data, disturbing the model's perception and leading to unpredictable behaviour. This limitation brings the model's reliability and safety during harsh weather into question, creating a challenge for real-world implementation in adverse weather.

The model relies on multiple specific libraries, meaning it is imperative that the compatible versions are chosen. For more information on what version to use, please, refer to the literature review in Appendix A.

3. Results & Discussion

3.1 Training and Validation Loss

The model was trained on 287,702 samples, with a validation size of 50,772, and a testing size of 13,323. The manually filtered data included scenarios like lane following, intersections, bay passing and roundabouts. Size of 25 epochs was considered optimal as it had already extracted most of the useful information from the data due to the plateau of validation loss. Training a model beyond this is only useful if new patterns are discovered, otherwise overfitting occurs.

The consistent decrease in training and validation loss showed how the model learned meaningful features from sequential point cloud data. Validation loss dropped from 0.0952 during the first epoch to 0.0064 by the last epoch, a 93% reduction, indicating that substantial learning occurred. Close alignment between training and validation loss, where the values were 0.0075 and 0.0064 respectively, suggest good generalisation capabilities.

A key finding was the model's performance on the completely unseen test set. The final test loss of 0.0061 shows highly reliable and stable predictions, representing an 86% improvement over the baseline model's 0.0426. The baseline model was trained over 129 epochs, with a best validation loss of 0.0324. This reduction in test loss confirms that the integration of the sequence fed Mamba-based temporal architecture combined with evidential output is vastly more effective at capturing the complex spatiotemporal dependencies necessary for driving than the baseline's single input. The model's ability to utilise 10 timesteps allowed it to learn coherent vehicle dynamics rather than reacting instantaneously. This improvement is a critical prerequisite for achieving the smooth, predictable behaviour outlined in objective one.

The choice of the evidential loss function (Equation 2.3) was critical to the results. The close alignment between training and validation loss and the superior test performance can be attributed to $\mathcal{L}_i^{\mathrm{R}}$. It actively penalised overconfidence that guided the model to memorise sequences rather than operating on the dataset. This minimal gap between training and validation loss is a direct indicator that the regularisation was effective. This successfully meets the second objective, to integrate uncertainty estimation within the AD pipeline, as the model inherently quantified its confidence with every prediction, forming the basis for principled decision-making.

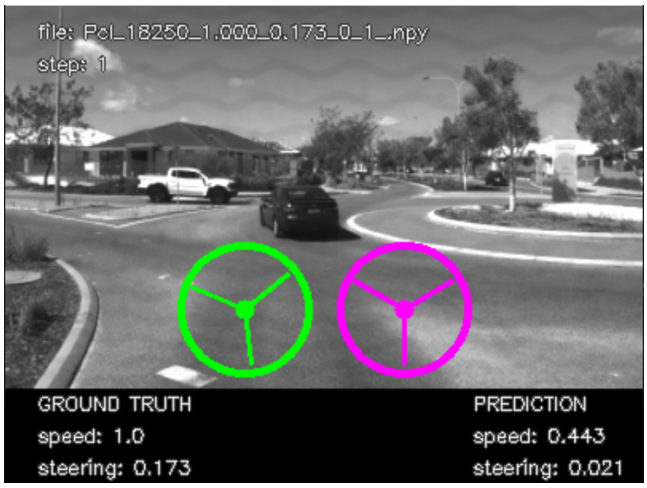
The outperformance in test loss over the base model provides initial support for the hypothesis. The successful integration of temporal input and evidential deep learning directly addresses a key challenge: moving beyond single-frame predictions to a robust context-aware driving model.

The limitations of these findings must be considered. While the model successfully generated uncertainty estimates, the third objective to develop uncertainty-aware control was not completely met. The loss, while demonstrating excellent convergence, does not guarantee flawless real-world performance. A low-test loss indicates accurate prediction of the training distribution's driving commands, but it does not capture the model's ability to handle novel, out of distribution scenarios. Furthermore, this metric is an aggregate, it does not reveal if the performance is consistent across all manoeuvre types.

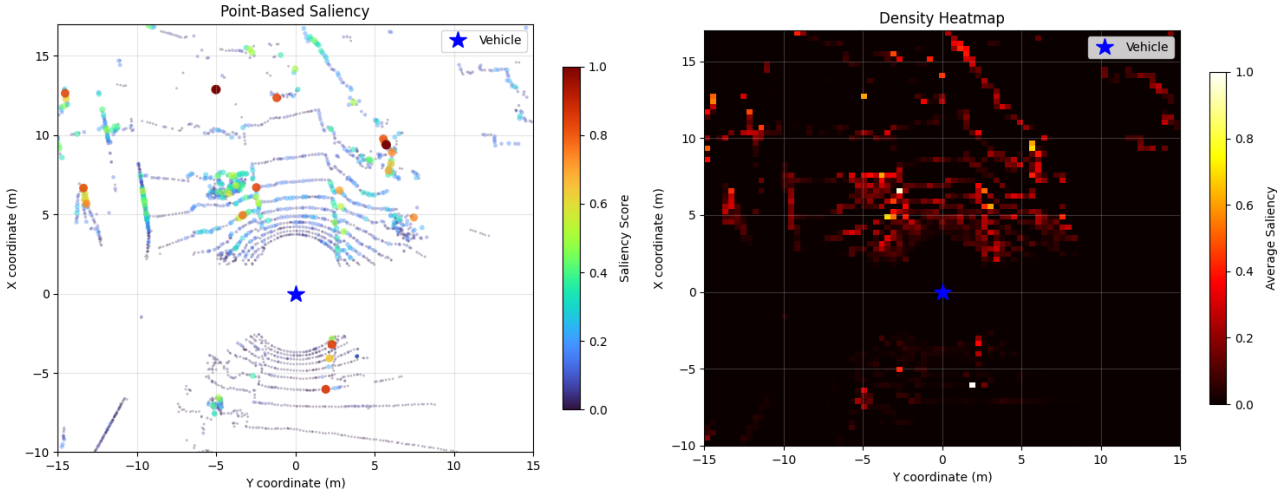
3.2 Saliency

The saliency analysis provides qualitative validation that the model's learned behaviour aligns with the project's aim. The analysis was performed on point clouds derived from Velodyne sensor data. Every frame of approximately 30,000 points was reduced to 3,000-5,000 via cropping and voxelisation. Visualising the spatial attention on this processed data indicates the temporal modelling functioned as intended, leading to a more interpretable perception, a foundation for human-like behaviour, and robust decision making under uncertainty, as mentioned in objectives one and two.

Representative frames were chosen at random from diverse driving scenarios encountered along the route. For more scenarios, please refer to Appendix B. A direct comparison between the models reveals a fundamental difference in perceptual strategy. The baseline model (Figure 3.1(b)) exhibits a diffuse attention pattern, with points of interest spread indiscriminately throughout the frame rather than focusing on the dynamic obstacles in front. Figure 3.1(c) highlights the real sparsity of the model's attention, expanding on the saliency map's information. This indicates that the model struggled to interpret the scene from a single frame, processing many points with similar priority due to its lack of temporal context.



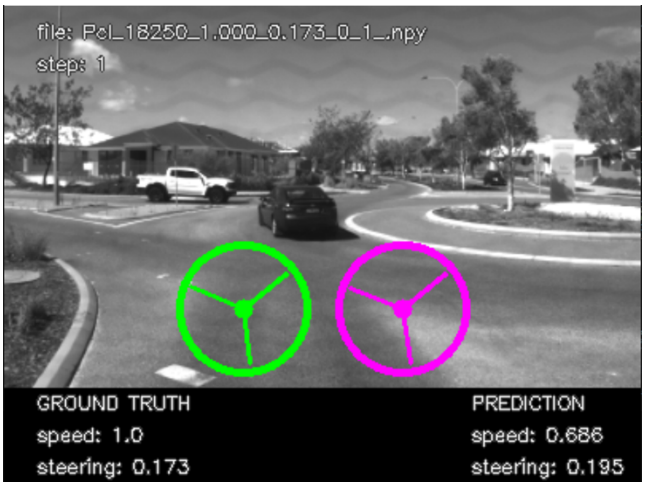
(a) Camera View. A comparison of the model's predicted driving command (pink wheel, bottom right) against the recorded dataset's ground truth driving command (green wheel, bottom left).



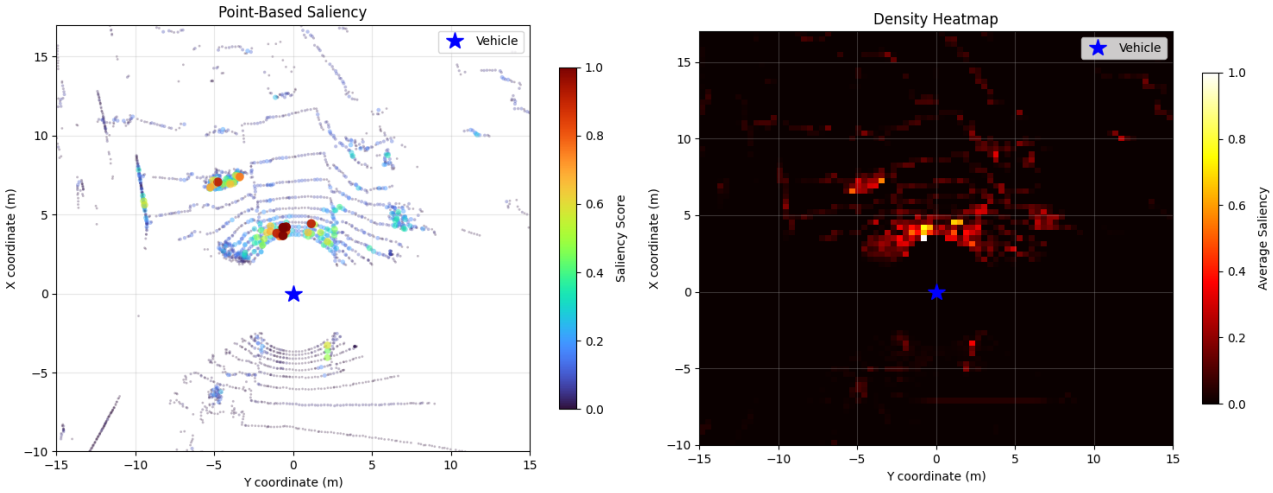
(b) Point-Based Saliency (c) Density Heatmap

Figure 3.1: Baseline Model, Roundabout with Obstacles

In stark contrast, Figure 3.2 demonstrates the new model's meaningful focus and environmental sense. The number of saliency scores tending to 1 indicates that the model put high relevance in the road and vehicles before it. The scores of around 0.5 indicate the model's moderate attention in the surrounding environment. The scale used shows significance levels for each point, red being the largest, visually expressing its importance. When compared to the baseline model, the focus is much more localised, rather than spread throughout the frame.



(a) Camera View. A comparison of the model's predicted driving command (pink wheel, bottom right) against the recorded dataset's ground truth driving command (green wheel, bottom left).



(b) Point-Based Saliency (c) Density Heatmap

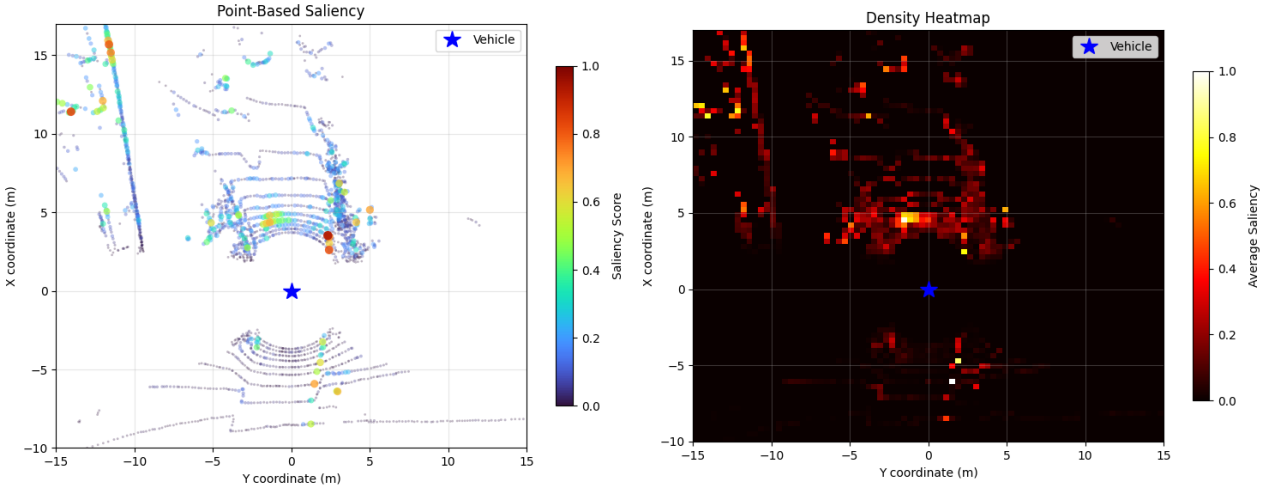
Figure 3.2: SparseResNet9D_Mamba_evidential Model, Roundabout with Obstacles

The highly localised focus of the proposed model (Figure 3.2), contrasted with the spread attention of the baseline (Figure 3.1), provides direct visual evidence that the Mamba block temporal modelling allows it to disambiguate the scene. Instead of processing all points equally, it learns to identify persistent task-relevant entities.

As Figures 3.3(b) and (c) illustrate, the baseline model generates valid maps. The model uses spatial features within the entire frame when no immediate obstacles are present. Aside from a high density of points in the top left corner, seemingly redundant to the human eye, helping the model to distinguish scenarios, it highlights the road directly ahead as the primary feature. Since the saliency

map in Figure 3.3(b) shows only a few important points, it is expected that there would be high densities of lower score features, as proven by the heatmap in Figure 3.3(b).

(a) Camera View. A comparison of the model's predicted driving command (pink wheel, bottom right) against the recorded dataset's ground truth driving command (green wheel, bottom left).



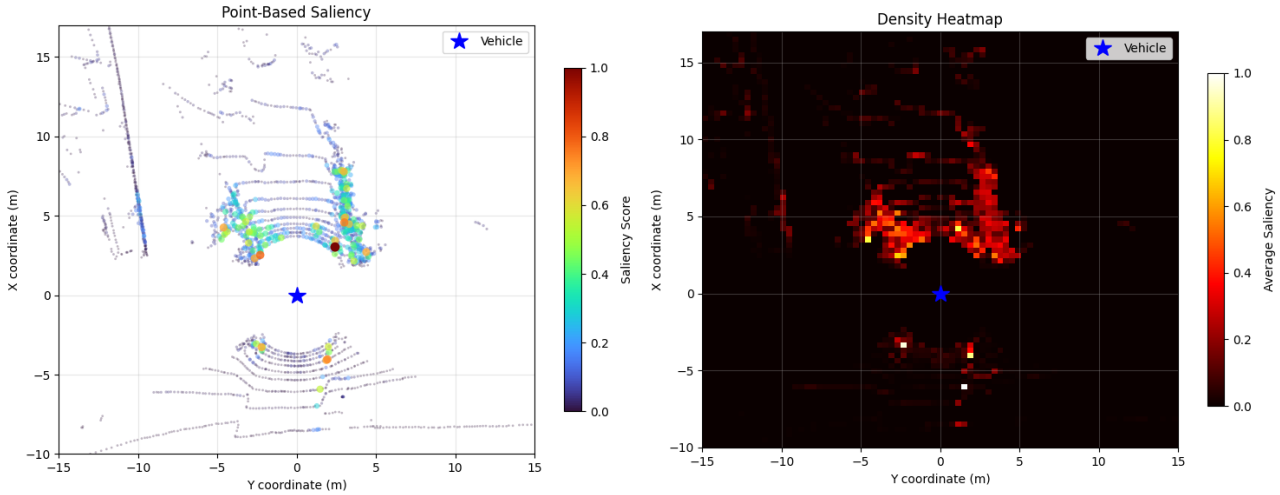
(b) Point-Based Saliency (c) Density Heatmap

Figure 3.3: Baseline Model, Roundabout with Obstacles

A similar circumstance can be seen in Figure 3.4, where, in a roundabout, even in the absence of dynamic obstacles, the new model's attention remained focused to the right, a region where vehicles would most likely pose a threat; consistent with where objects had appeared in past scenarios. The cluster of points of various interest indicates high influence on the steering decision. Occasional points can be seen to the left side and straight ahead in multiple roundabout scenarios, a pattern emphasising the fact that the model anticipated hazards, focusing selective attention, even in static environments; this is not a reaction to a present obstacle, but an expectation of a potential threat based on learned contextual knowledge.

Consistent across multiple frames, the saliency maps for the new model show that high-attention regions typically form clusters rather than isolated points. These clusters often consist of high-saliency points (red, > 0.7) surrounded by points of moderate relevancy; these lesser points typically capture environmental elements.

(a) Camera View. A comparison of the model's predicted driving command (pink wheel, bottom right) against the recorded dataset's ground truth driving command (green wheel, bottom left).



(b) Point-Based Saliency (c) Density Heatmap

Figure 3.4: SparseResNet9D_Mamba_evidential Model, Roundabout without Obstacles

As shown in Table 3.1, the new model's focus is localised, with less than 5% of input points (the sum of high and medium saliency scores) significantly influencing steering decisions. This shows that it does not search extensively in the image of the frame, instead efficiently allocating computational resources, a necessity for real-world performance These patterns mirror human behaviour by employing selective processing rather than exhaustive scene analysis, supporting objective two.

Table 3.1: Saliency Point Distribution by Intensity

	Total points assessed	**High saliency points (> 0.7)**	**Medium saliency points (0.3 - 0.7)**	**Low saliency points (< 0.3)**
Figure 3.2	3815	11 – front 0 – rear	73 – front 3 – rear	3728
Figure 3.4	3696	3 – front 1 – rear	109 – front 9 – rear	3574

Consistent across all frames, the model showed its strongest activation in the forward-facing direction, with most of mid- to high-saliency points concentrated within a 7-metre radius of the

vehicle, indicating that the model primarily attends to its immediate path. Occasionally, points of interest from the rear LiDAR were also emphasised, suggesting the architecture successfully learned to factor in rearward information, in line with its design to prioritise near-navigation cues over environmental details.

Collectively, the saliency maps prove that the architecture is capable of finding and differentiating the most important parts of the route to form its next driving commands. They provide strong evidence that the model's attention mechanisms are both spatially precise and semantically meaningful, focusing computational resources on relevant features. When combined with the quantitative improvement in test loss, it can be seen the model performs better because it processes information more intelligently. It focuses computational resources on relevant features, leverages temporal context to anticipate potential hazards and ignores redundant information. The ability to focus selectively represents a significant qualitative improvement over the baseline model, which aligns with the goal of developing a more sufficient and interpretable AD.

3.3 Ground Truth Scene Analysis

The model's performance was evaluated against the GT, a human-driven data that was manually sorted to represent smooth lawful driving serving as the expert benchmark. The model was evaluated on its ability to replicate the dataset's driving behaviour, where both ground truth and prediction were normalised. The below images show a comparison of the model's predicted driving command (pink wheel, bottom right text) against the recorded dataset's ground truth driving command (green wheel, bottom left text). In general, the new model's velocity and steering closely aligned with the GT. Figure 3.5 shows an almost identical steering angle, with a difference of 1.375º (0.024 radians) and the GT's speed at 20 km/h, while the predicted velocity shows 18 km/h, an accuracy of 90%. This indicates it prioritises caution, exhibiting a consistent underprediction of speed during roundabouts.

Figure 3.5: Good Alignment of Prediction with the Ground Truth

Figure 3.6 illustrates a rare occasion in which the model exhibited a slightly higher speed than the GT. As for the steering, the predicted angle aiming 26.814º further right was taken at a time when the GT experienced irregular movement (as was seen during the sequential frame analysis). This demonstrates the model not only mirrored the GT, but at times reached a smoother performance.

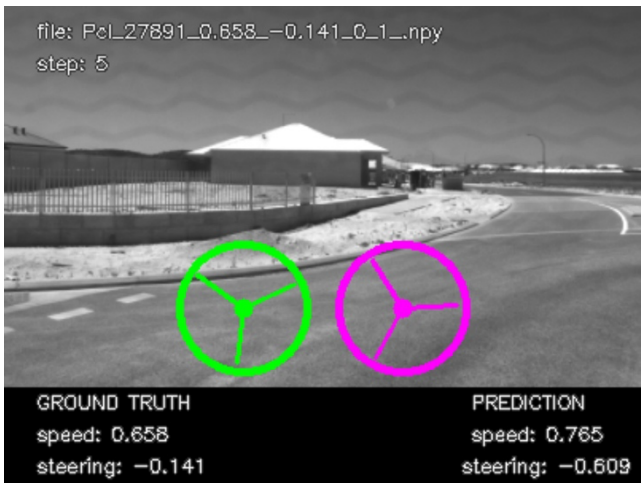


Figure 3.6: Oversteering Comparing to the Ground Truth

In Figure 3.7, the GT's speed of 18.1 km/h is higher than the model's predicted speed of 15.6 km/h, a pattern consistent with the differentials seen in Figure 3.5. However, the steering behaviour of the GT shows a 20º turn to the left, whilst the predicted steering again prioritises a smoother trajectory, a trend aligned with the results noted in Figure 3.6.

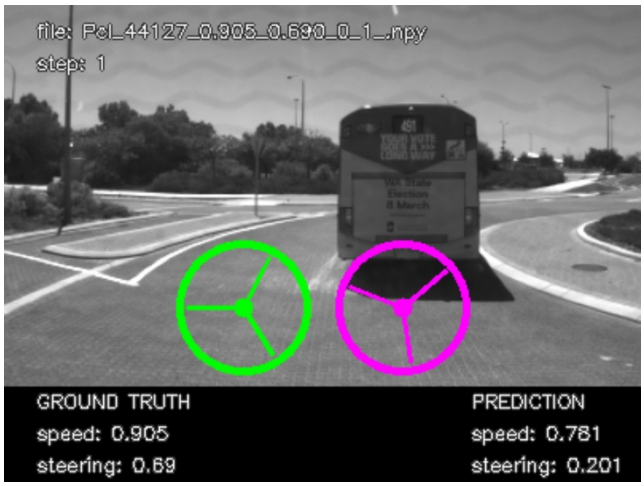


Figure 3.7: Understeering Comparing to the Ground Truth

These results show that although the model is able to shadow the inclinations of the GT's data, it still prioritises smoothness and safety over outright mimicry. The sequential processing of the Mamba blocks allows the model to perceive not just an instant but a trajectory of time. The model's under-prediction during roundabouts, as can be seen in Figure 3.5 and 3.7 is a direct indicator of the evidential loss function, which actively penalises overconfident errors, incentivising a risk-averse strategy.

3.4 Ground Truth Time Analysis

A sequential analysis of the models' performance over time provides clear evidence of the advantages presented through temporal modelling. Both models, shown in Figures 3.8 and 3.9 exhibited excellent steering during lane following, and thus are both suitable options for real-world deployment. The baseline model verged on maximum speed for majority of the snippet. In contrast, the new model remained consistently cautious, typically around 94% of the maximum velocity. This is not a failure,

but a result of the regulariser used during training (objective two), which actively discouraged overconfidence. By penalising high-speed errors more severely, the training process formed a risk-averse model, that ranked safety over pace.

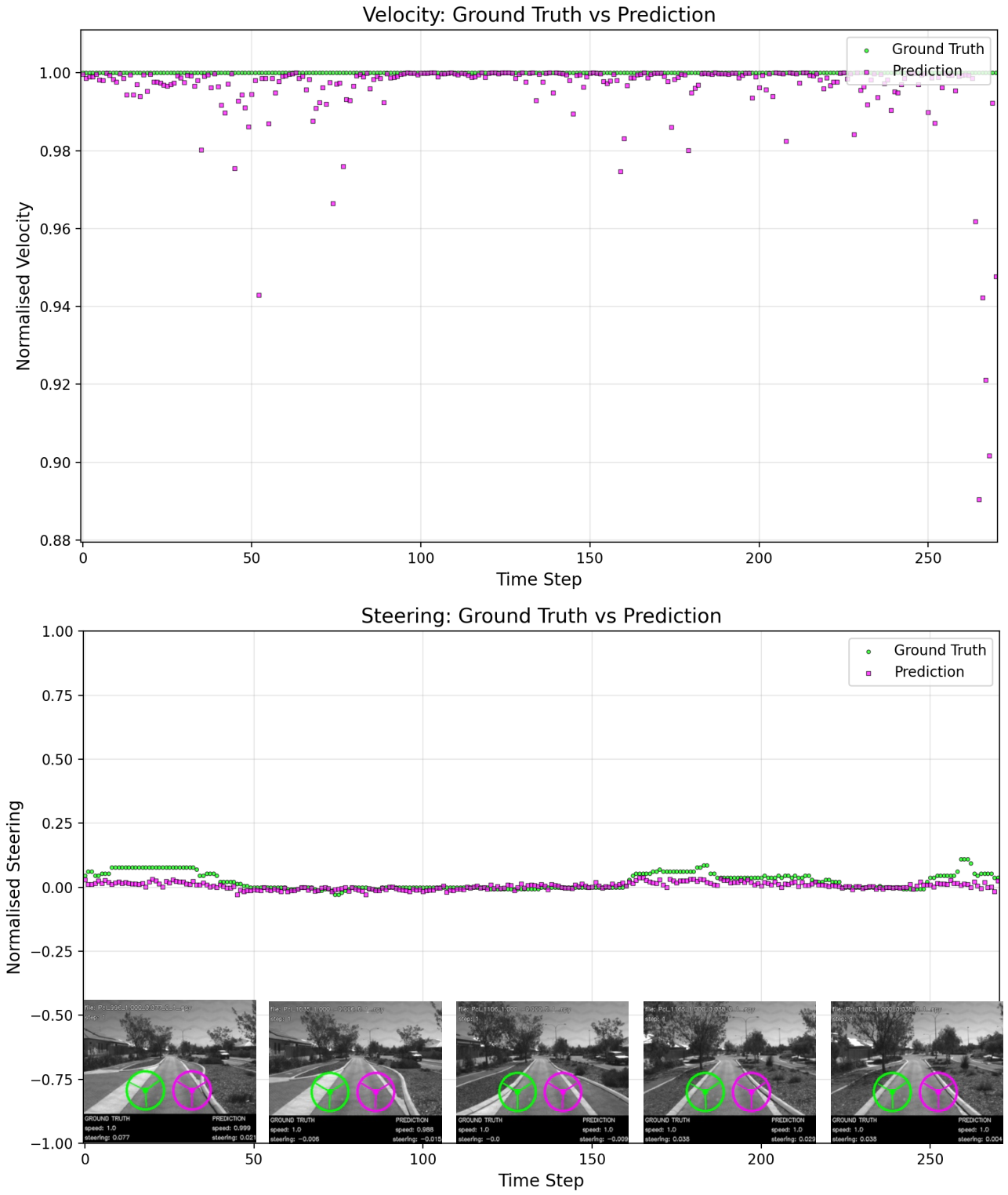


Figure 3.8: Driving Command of the Baseline Model Comparing to Ground Truth, Lane Following

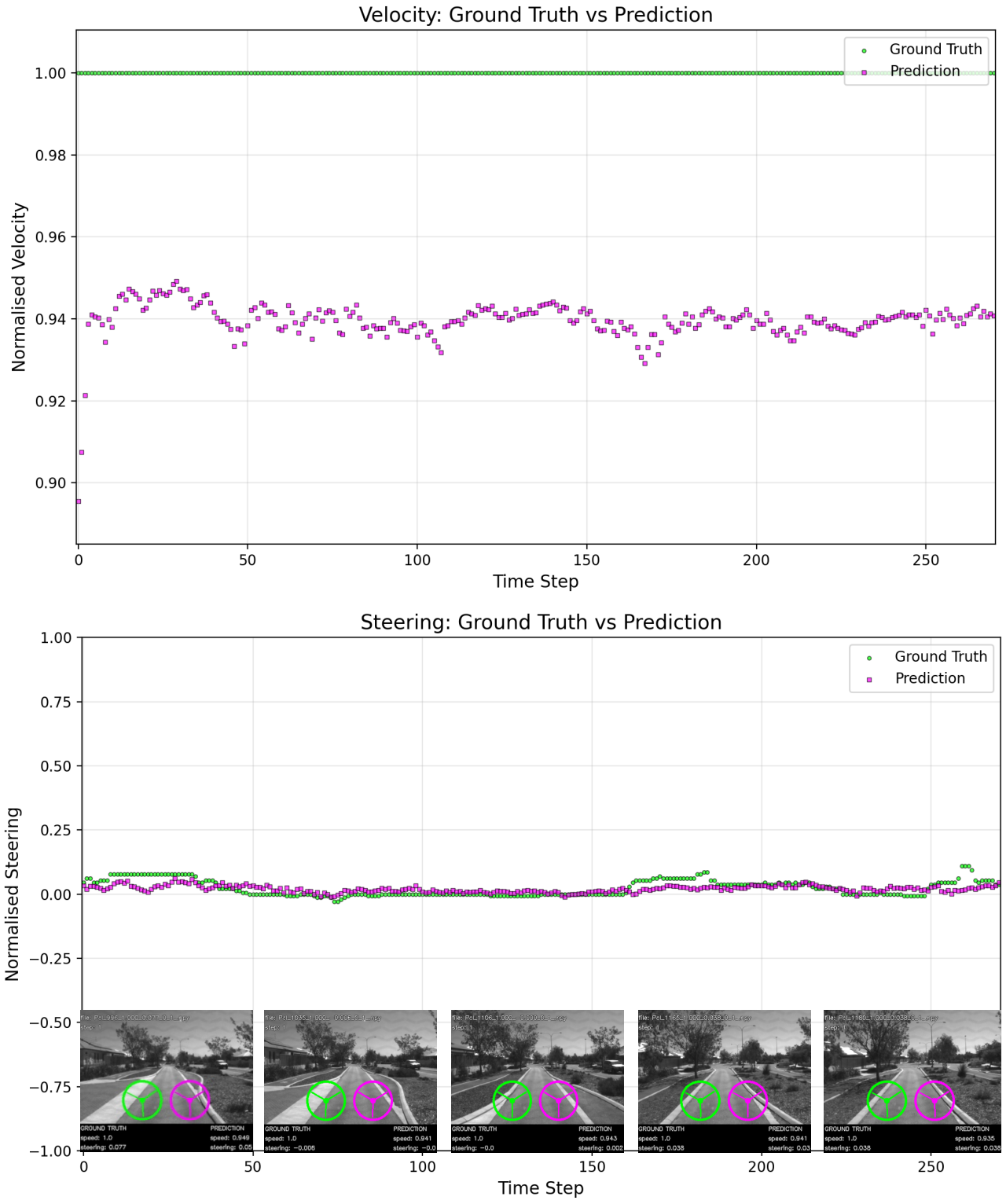


Figure 3.9: Driving Command of the New Model Comparing to Ground Truth, Lane Following

The superior performance of the new model becomes undeniable in complex situations. Figures 3.10 and 3.11, which analyse roundabout manoeuvres, reveal this stark contrast. The baseline model exhibits erratic overreactive commands to each LiDAR frame, a direct consequence of its lack of historical context, whereas, with the inclusion of past timesteps, the new model showed significantly smoother operation outputs. The jittery control of the baseline model is likely a primary cause for its manual interventions in all roundabouts.

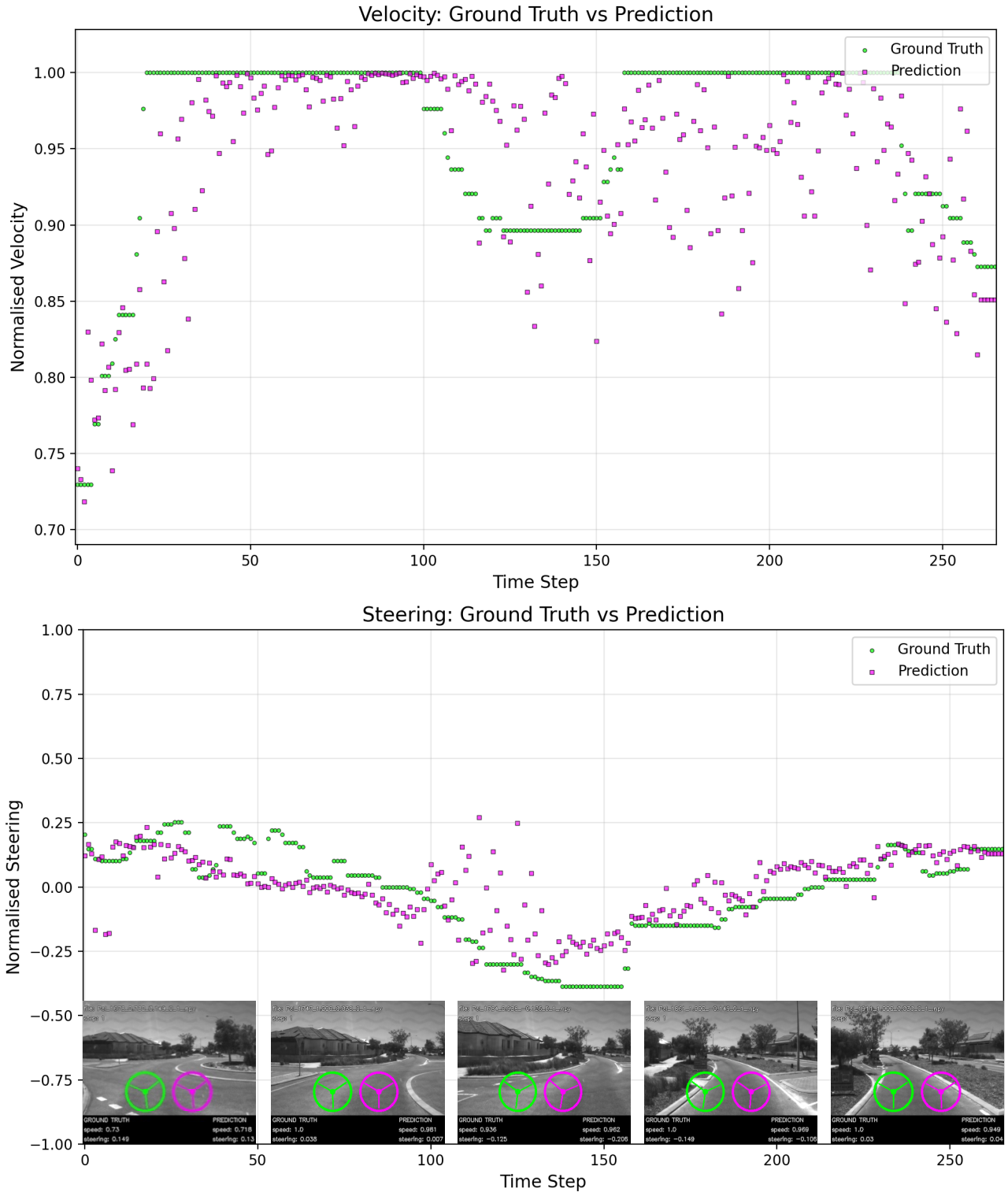


Figure 3.10: Driving Command of the Baseline Model Comparing to Ground Truth, Roundabout

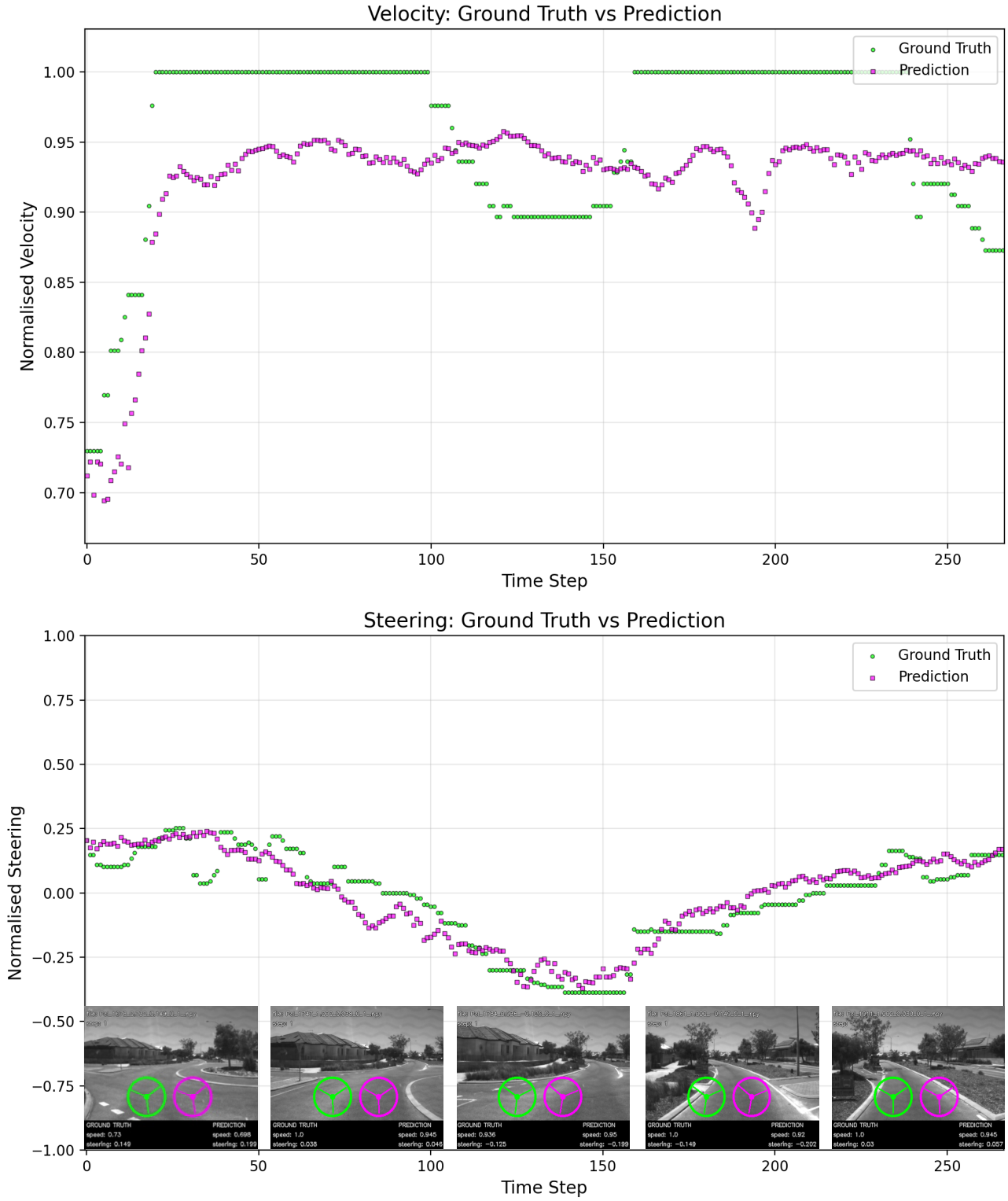


Figure 3.11: Driving Commands of the New Model Comparing to Ground Truth, Roundabout

This contrast in complex scenarios underscores the core advantage of the proposed temporal model. The baseline, lacking historical context, reacts erratically to each instantaneous input. In contrast, the new model leverages the history to curve its steering commands predictably, and transition velocity gradually. Although, sometimes steeper in appearance, the new model's velocity is majority of the time lower, this results in less aggressive inclines and declines during speed changes, equating to an overall smoother and more stable drive. This allowed the new model to plan its manoeuvres rather

than simply react, fulfilling the objective of achieving smooth, human like driving, as well as enhancing performance through temporal modelling.

This performance advantage is consistent across an extended route of multiple roundabouts and turns, as shown in Figure 3.12.

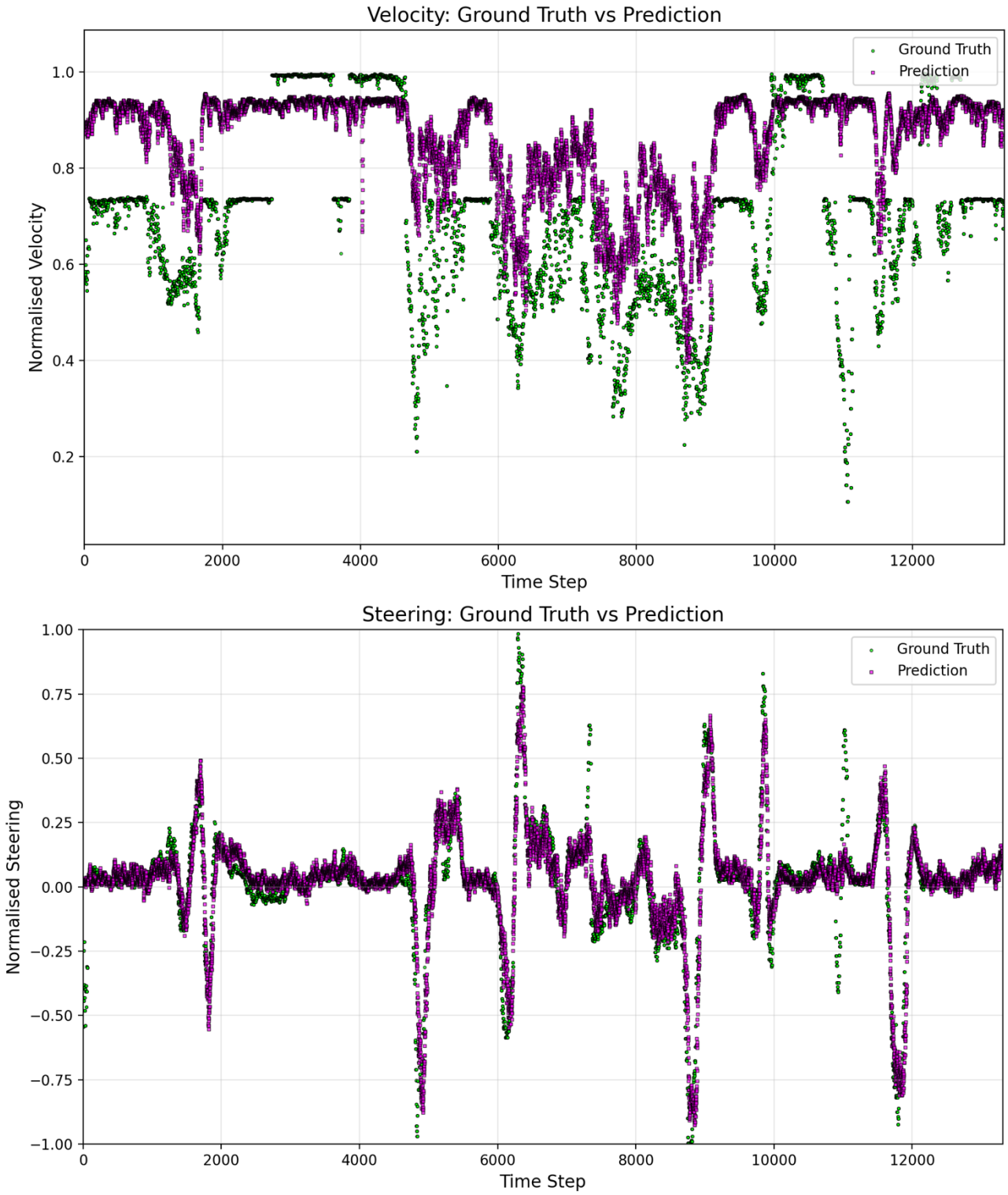


Figure 3.12: Path with Multiple Roundabouts and Turns

The new model's velocity reliably tracks the cautions trend of the GT, which slows appropriately for turns, prioritising safety. The troughs and peaks seen in the Steering graph indicate roundabouts and

turns, where positive are left and negative are right turns. The relatively calm moments indicate lane following, during which, velocity prediction stayed around 90% full speed. The training sets mostly included data with full speed, rarely slower. With the addition of the regulariser, punishing overconfidence, the model learned to behave at around 94% speed during lane following and other straight parts of the path. As for steering predictions, they were generally well-aligned, though the model failed to recognise two specific turns. This is an example of imbalanced data in the training set; the model was exposed to fewer examples of this turn geometry, causing the more frequent patterns to dominate the final command. Despite these minor omissions, the overall temporal smoothness and safety conscious behaviour confirm the new model was successful in learning a robust and predictable driving strategy.

3.5 Uncertainty vs Driving Prediction in Real Life Testing

Despite strong offline validation of the model through the various tests analysed above, road testing revealed a latency-induced instability. A short snippet from the uncertainty and relevant velocity collected from the drive can be found in Figure 3.13, where aleatoric is higher than epistemic; this is consistent with operation on a known route. Both uncertainties show limited fluctuation as the training data was almost perfect with minimal sensor noise (resulting in low, constant aleatoric) and ideal GT values (low, constant epistemic). The low and constant aleatoric uncertainty further suggests the training data was too clean with minimal sensor noise, preventing the model from learning to robustly handle real-world sensory variance.

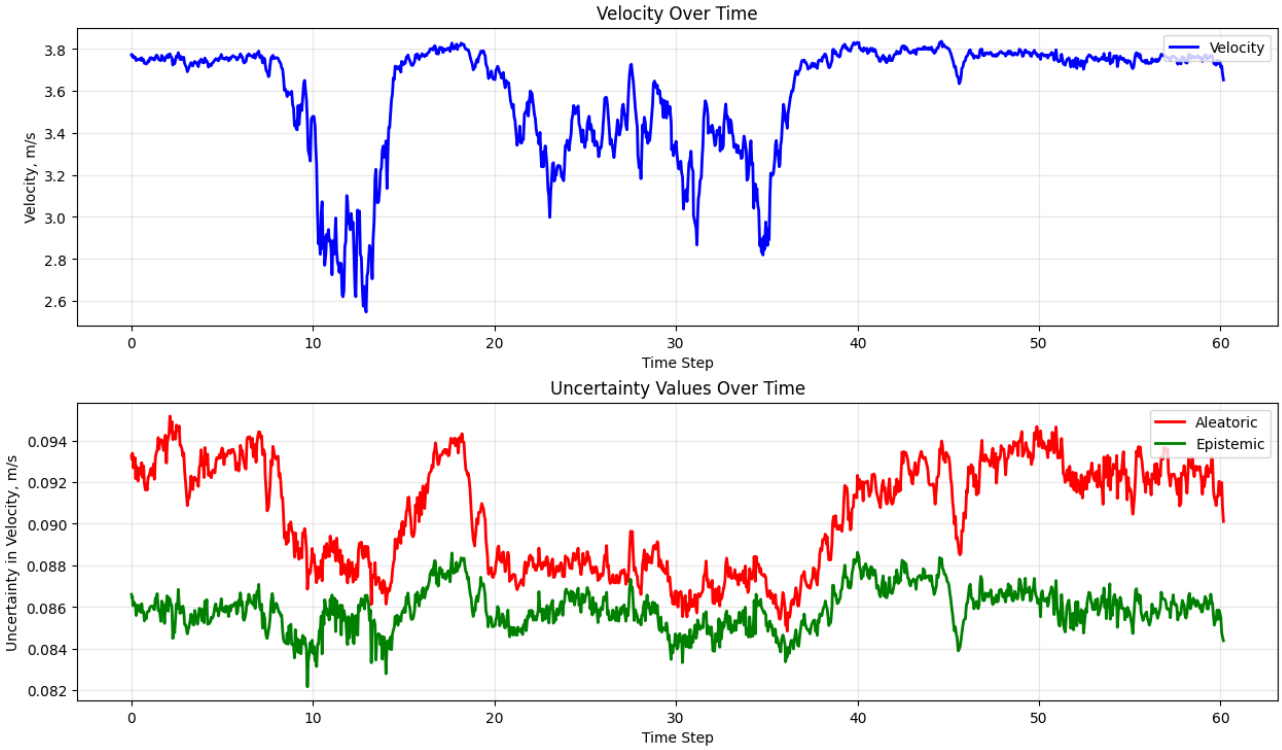


Figure 3.13: Velocity and Corresponding Uncertainty in Real Life Driving

Figure 3.14 confirms an expected relationship, where uncertainty become higher with an increase in velocity. Higher speed reduces the margin for error, naturally increasing the inherent noise in the driving command.

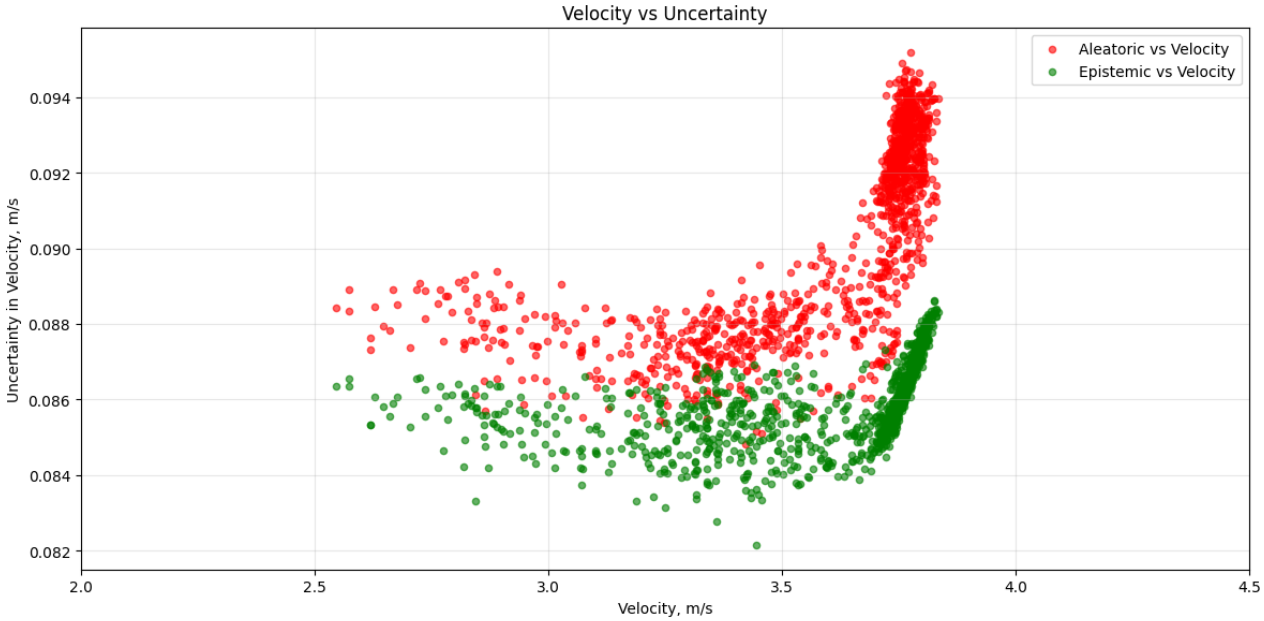


Figure 3.14: Uncertainty Trend vs Velocity in Real Life Driving

The real-world performance was subjected to the latency introduced in the control loop. While quantitative analysis against recorded GT demonstrated smooth temporally coherent driving, real-life implementation suffered from unstable behaviour. This stemmed from latency inherent in the sequential data aggregation process. Buffering 10 consecutive frames introduces a 0.5 second delay before computation can even start. This means that by the time he model made its real-time decisions the vehicle had already moved on.

This latency put the model in a state of persistent error correction, constantly issuing commands to correct a position that had already changed. The vehicle's actual position would always be ahead of the model's latest processed frame, causing it to consistently turn too late or oscillate as it chased an outdated trajectory.

A significant limitation of the deployment was the failure to implement uncertainty-aware control. While the model successfully generated uncertainty estimates, these values were not used to modulate the driving command in real-time. This was primarily due to the unanticipated severity of the latency issue, which became the dominant failure mode and made the closed-loop system unstable before the benefits of the uncertainty-scaling could be evaluated.

The strong performance in GT comparison in contrast with the challenges encountered during real-world testing, further highlight the critical difference between a controlled offline validation and the complexities of real-time deployment. This result underscores a pivotal limitation and critical finding in transitioning temporal models from offline to real-time deployment, the computational pipeline for building temporal context must be optimised to minimise latency.

The hypothesis was strongly supported in offline validation, where the Mamba based temporal model and evidential uncertainty enhanced driving smoothness and performance. However, real-

world deployment revealed performance gains were dependent on a low-latency pipeline, as the benefits of a temporally aware model were negated by the delay.

4. Conclusions & Future Work

This research project succeeded in creating the SparseResNet9D_Mamba_evidential, a LiDAR based AD model that effectively addresses critical shortcomings found in other systems. Through the implantation of a Mamba-based temporal block that processes 10 LiDAR frames, the model was able to learn from past context, which resulted in vastly smoother, more predictable driving commands, than the baseline model it was drafted from, particularly in roundabouts. By incorporating evidential regression head the model efficiently quantified both aleatoric and epistemic uncertainty in a single forward pass.

An 86% lower test loss compared to the baseline model confirms the new model's superiority when it comes to generalisation and learning prowess. Qualitative analysis through saliency maps visualised the model's attention to relevant features such as dynamic obstacles and the immediate path. Introduced evidential regression loss into the training caused it to underperform when it came to velocity predictions compared to GT.

Objective one was fully achieved by including Mamba-based temporal architecture which contributed to smooth driving from sequential LiDAR data. Objective two was also successfully met providing an insight to the model's epistemic and aleatoric uncertainties. However, objective three was not achieved due to the unanticipated dominance of latency from sequential data buffer destabilising the control loop. This indicates that the performance benefits of temporal models are critically dependant on low-latency computations.

The most critical direction for future work is in latency mitigation and real-time system integration. A predictive control framework should be implemented, where the model is trained to output commands for a future timestep to compensate for the known processing delay.

Once the delay is accounted for, next step is to implement and evaluate the uncertainty-aware control. This involves dynamically adjusting the vehicle's speed based on the real-time epistemic and aleatoric uncertainty.

Finally, to enhance real-world robustness, the model must be trained on a broader and more challenging dataset. This includes a wider variety of manoeuvres to reduce impact of class imbalance and prepare the system for full operational deployment.

References

Pomerleau, D. A. (1988). *ALVINN: An Autonomous Land Vehicle in a Neural Network.* Advances in Neural Information Processing Systems. Retrieved May 1, 2025, from https://proceedings.neurips.cc/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf

Riedel, P. (2025). *LiDAR-based End-to-End Autonomous Driving for a Shuttle Bus Using Deep Learning*. Master's Thesis, University of Stuttgard.

Li, J., Li, J., Yang, G., Yang, L., Chi, H., & Yang, L. (2025). *Applications of Large Language Models and Multimodal Large Models in Autonomous Driving: A Comprehensive Review.* Drones, 9(4), 238. Retrieved Oct 6, 2025, from https://doi.org/10.3390/drones9040238

Mugunthan, N., Balaji, S., Harini, C., Naresh, V. & Venkatesh, P. (2020). *Comparison Review on LiDAR vs Camera in Autonomous Vehicle*. International Research Journal of Engineering and Technology. Retrieved Oct 6, 2025, from https://www.irjet.net/archives/V7/i8/IRJET-V7I8731.pdf

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need*. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS) 2017, Long Beach, CA, USA. Retrieved May 1, 2025, from https://arxiv.org/pdf/1706.03762

Gu, A., & Dao, T. (2023). *Mamba: Linear-Time Sequence Modeling with Selective State Spaces*. Retrieved May 1, 2025, from https://arxiv.org/pdf/2312.00752v1

Gal, Y., & Ghahramani, Z. (2016). *Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning.* In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48. Retrieved May 1, 2025, from https://proceedings.mlr.press/v48/gal16.pdf

Lakshminarayanan. B., Pritzel, A., & Blundell, C. (2017). *Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles*. In Proceedings of 31st Conference on Neural Information Processing Systems (NIPS) 2017, Long Beach, CA, USA. Retrieved May 1, 2025, from https://arxiv.org/pdf/1612.01474

Amini, A., Schwarting, W., Soleimany, A., & Rus, D. (2020). *Deep Evidential Regression*. In Proceedings of 34th Conference on Neural Information Processing Systems (NeurIPS) 2020, Vancouver, Canada. Retrieved May 1, 2025, from https://proceedings.neurips.cc/paper_files/paper/2020/file/aab085461de182608ee9f607f3f7d18f-Paper.pdf

Australian Government Department of Infrastructure, Transport, Regional Development and Communications. (2021). *Final Regulation Impact Statement Reducing Trauma from Light Vehicles: Autonomous Emergency Braking.* Retrieved Oct 6, 2025, from https://oia.pmc.gov.au/sites/default/files/posts/2021/11/Attachment%20B%20-%20DRAFT%20Final%20RIS%209_July%202021.pdf

Gao, M., Ruan, N., Shi, J., & Zhou, W. (2022). *Deep Neural Network for 3D Shape Classification Based on Mesh Feature*. Sensors, 22(18), 7040. Retrieved Oct 6, 2025, from https://doi.org/10.3390/s22187040

Lecigne, B., Delagrange, S., & Messier, C. (2017). Exploring trees in three dimensions: VoxR, a novel voxel-based R package dedicated to analysing the complex arrangement of tree crowns. Annals of Botany, 121(4), 589–601. Retrieved Oct 6, 2025, from https://doi.org/10.1093/aob/mcx095

Waqas, M., & Humphries, U. W. (2024). *A Critical Review of RNN and LSTM Variants in Hydrological Time Series Predictions.* MethodsX, 13, 102946–102946. Retrieved May 1, 2025, from https://doi.org/10.1016/j.mex.2024.102946

Kandadi, T. (2025). *Drawbacks of LSTM Algorithm: A Case Study*. SSRN. Retrieved May 1, 2025, from https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5080605

Velodyne LiDAR, Inc. (2018). *Real-Time 3D Lidar Sensor*. https://www.mapix.com/wp-content/uploads/2018/07/63-9229_Rev-H_Puck-_Datasheet_Web-1.pdf

Learning, New York, NY, USA. (2016). JMLR: W&CP volume 48. Retrieved May 1, 2025, from https://proceedings.mlr.press/v48/gal16.pdf

Appendices

Appendix A: Literature Review

A.1 Introduction, Literature Review and Project Objectives

A.1.1 Introduction

Due to a rising interest in technology and an increase in innovation, demand for autonomous vehicles has had a high influx, however, there are still areas that are yet to meet a satisfactory level of reliability suitable for public use, leaving gaps in the application to be filled. The UWA Renewable Energy Vehicle Project plans to improve this; the aim is to bring transportation into the future by implementing driverless transport in the form of shuttle busses.

One of the challenges currently being faced in the field is the ability to achieve efficient trajectory prediction in end-to-end driving in real-time. Trajectory planning is a critical process in autonomous driving in dynamic environments, and the lack thereof or inefficient implementation of it within the model architecture has real-world implications. These consequences include a lack of manoeuvrability in planning and an inability to adapt to unexpected events. Because of this, there is a building distrust around autonomous vehicles within the general public. Existing methods face challenges in capturing temporal dependencies and quantifying prediction uncertainty. Within the broader scientific community, these inefficient temporal planning and predictions are hindering advancements in driverless systems, stopping them from being implemented as frequently as they could.

The research aims to improve trajectory prediction and adjust the driving behaviour for more reliable and efficient use. The purpose is to build a model that will sway the public's trust back in the favour of advancing technology and autonomous vehicles, henceforth accelerating industry adoption.

A.1.2 Literature Review

Trajectory Prediction

Trajectory prediction within autonomous vehicles has evolved since the beginning of robotics and control theory, driven by long term behaviour modelling and computational efficiency. Earlier models used simple NN (Neural Network) to map inputs to steering commands (Pomerleau, 1988) but lacked the capacity for long-term behaviour modelling, limiting reliability in dynamic environments. Modern approaches employ deep learning architecture to address this limitation.

Traditional approaches to sequence modelling use Recurrent Neural Networks (RNN) or its improved version, LSTM (Long Short-Term Memory). LSTMs are an improvement over RNNs as they do not suffer from vanishing gradients due to the gate structure that controls what information to keep, update or forget and a memory cell to store information for long periods (Waqas & Humphries, 2024). While it is still effective for short term dependencies, for larger datasets, the processing power is greater due to the gated structure (Kandadi, 2025). Because of the aforementioned as well as the

amount of parameters causing a risk of overfitting, LSTMs are not ideal for real-time trajectory prediction in autonomous diving, where efficiency and broad deployment capabilities are crucial.

Transformer models, due to global context awareness through an attention mechanism, can predict multiple future steps simultaneously and do not suffer from overfitting. The major limitation is quadratic computational complexity ($O(N^2)$ for N input tokens) (Vaswani et al., 2017), making them an unrealistic option to use in real-world situations with sequences of LiDAR data with up to 300,000 points per second (for Velodyne VLP-16) (Velodyne LiDAR, Inc., 2018).

To balance the accuracy and efficiency, selective state-space models are a better choice. Noticeably, the Mamba (Gu & Dao, 2023) architecture has shown promise due to its ability to process high amounts of sensor data and time-based information efficiently. With 5 times higher throughput at inference comparing to transformers (Gu & Dao, 2023), Mamba still maintains linear complexity (O(N) for N input tokens) and captures long-term patterns from high-dimensional sensor data. This is achievable through its ability to selectively focus on relevant information, prioritising it. Mamba's linearity is critical for dual (front and back) LiDAR input like the one in the nUWAy bus. Mamba addresses the critical need for computational efficiency and accuracy in autonomous driving.

Figure 1 highlights the performance of Mamba-based and transformer-based architectures at inference. Mamba achieves higher throughput in tokens per second than a transformer of similar size which points to efficiency in processing sequential data. For example, Mamba model with 1.4B parameters and a batch size of 32 can process 1445 tokens per second comparing to a transformer with 1.3B parameters, which has a throughput of 364 tokens per second, almost 4 times slower.

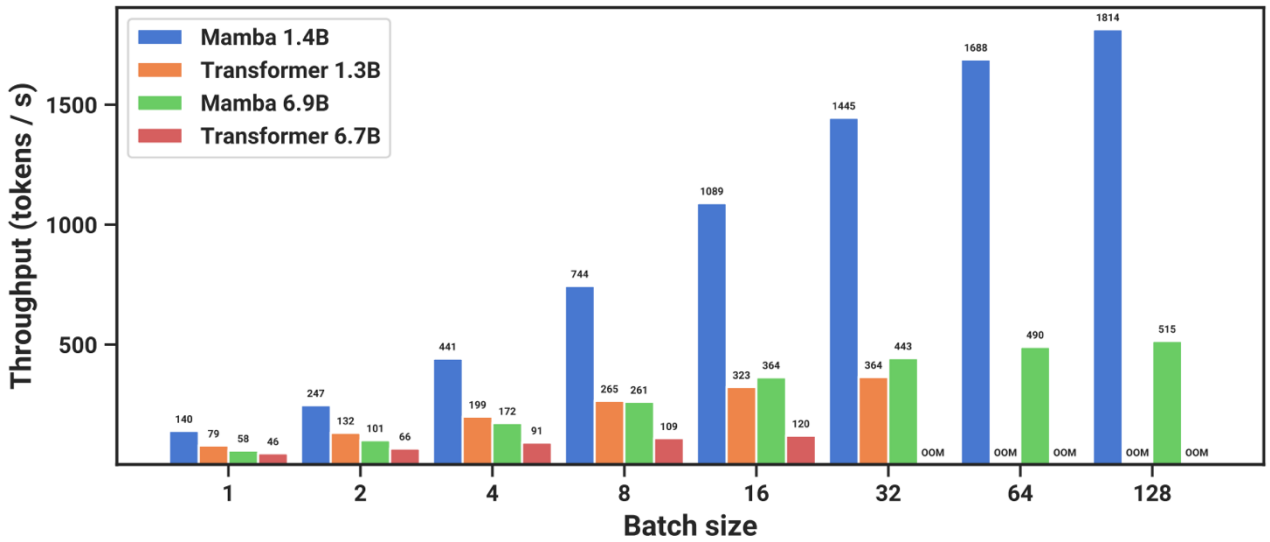


Figure A.1: Inference throughput on A100 80GB (prompt length 2048) (Gu & Dao, 2023)

Uncertainty Estimation

Regarding uncertainty, which is crucial for decision-making in imperfect dynamic conditions, autonomous driving is concerned about two types: aleatoric and epistemic. Aleatoric uncertainty refers to a randomness that cannot be reduced, but can be quantified, such as sensor noise, since

LiDAR data clouds suffer from sparsity and environmental inference. Epistemic refers to the amount of confidence or lack thereof a model has depending on the amount of collected data and its generalisation abilities. This type of uncertainty arises in new environments, where the training data cannot adequately represent real-world dynamics. With more training data and better modelling techniques, epistemic uncertainty is reducible. In autonomous driving, particularly when using LiDAR data, both forms of uncertainty must be modelled carefully to ensure safe planning. Table 1 summarises possible uncertainty methods.

Table A.2: Comparison of Uncertainty Methods in Deep Learning

Method	Strength	Limitation
BNNs	Clear, mathematically sound limits on how uncertain the prediction is	Complex for LiDAR
Monte Carlo Dropout	Lightweight, easy to implement	Multiple forward passes: might introduce lag
Ensembles	Captures both uncertainties	Memory overhead
Evidential regression	Single-pass, distinguishes between both uncertainty types	Overconfident if no proper regulation

Bayesian Neural Networks (BNNs) (Gal and Ghahramani, 2016) use probabilities rather than fixed numbers when making decisions; this helps them to express their confidence, propagating throughout the network. This is particularly useful in epistemic uncertainty as it enables the model to quantify its lack of knowledge if unusual (from training) data is inputted. However, apart from being not feasible for high-dimensional LiDAR data, BNNs are computationally expensive, which makes them unsuitable for real-time autonomous driving.

Monte Carlo Dropout, as a lightweight approximation of BNNs (Gal and Ghahramani, 2016), randomly zeroes parts of the NN during its training and inference stages. By running the same prediction multiple times with random parts of the network “turned off”, leading to different predictions, it helps the model estimate how unsure it is. However, due to multiple forward passes, it introduces latency, hence it is not great for time sensitive systems.

Ensemble methods (Lakshminarayanan et al., 2017) deal with both types of uncertainty. They train several different models independently and combine their predictions, improving reliability, especially when unexpected inputs are encountered. Even though ensembles outperform Monte Carlo Dropout in reliability, they require more memory and computing power, as each model is trained and stored independently, thus deploying this in an autonomous vehicle would be challenging.

Evidential regression shows more promising results, where the model predicts parameters of a Gaussian distribution (Amini et al., 2020) for regression targets. This distinguishes between and quantifies both types of uncertainty in a single forward pass, providing prediction and associated confidence. It offers scalable solutions for real-time systems. Balancing accuracy and computational efficiency is suitable for predicting uncertainty in autonomous driving. The main drawback is it could be overconfident. The efficacy of the method depends on loss calibration.

To implement evidential regression, there are 4 parameters that need to be predicted (γ, υ, α, β), which define Normal Inverse-Gamma distribution, where γ is sample mean from υ observations, α is shape parameter (uncertainty shape) and β is scale parameter (variance scale). The model "learns" these parameters via NN and can be used for aleatoric and epistemic uncertainties (Amini et al., 2020):

$$\underbrace{\mathbb{E}[\mu] = \gamma}_{\text{prediction}}, \qquad \underbrace{\mathbb{E}[\sigma^2] = \frac{\beta}{\alpha - 1}}_{\text{aleatoric}}, \qquad \underbrace{\mathrm{Var}[\mu] = \frac{\beta}{\upsilon(\alpha - 1)}}_{\text{epistemic}}. \tag{A.1}$$

The model is trained using total loss combining maximising and regularising evidence:

$$\mathcal{L}_i(\boldsymbol{w}) = \mathcal{L}_i^{\mathrm{NLL}}(\boldsymbol{w}) + \lambda \, \mathcal{L}_i^{\mathrm{R}}(\boldsymbol{w}). \tag{A.2}$$

The first term is negative log-likelihood which ensures predictions align with observed data, it is calculated using

$$\mathcal{L}_i^{\mathrm{NLL}}(\boldsymbol{w}) = \tfrac{1}{2}\log\left(\tfrac{\pi}{\upsilon}\right) - \alpha\log(\Omega) + \left(\alpha + \tfrac{1}{2}\right)\log((y_i - \gamma)^2\upsilon + \Omega) + \log\left(\tfrac{\Gamma(\alpha)}{\Gamma(\alpha + \frac{1}{2})}\right) \tag{A.3}$$

where $\Omega = 2\beta(1 + \upsilon)$.

Second term (increases uncertainty when errors are high, preventing overfitting) consists of λ, regularisation coefficient, where 0 is overconfident estimate, multiplied by evidence regulariser

$$\mathcal{L}_i^{\mathrm{R}}(\boldsymbol{w}) = |y_i - \mathbb{E}[\mu_i]| \cdot \Phi = |y_i - \gamma| \cdot (2\upsilon + \alpha). \tag{A.4}$$

Evidence regulariser penalises overconfident predictions in low evidence regions.

State of the Vehicle

The Robot Operating System 2 (ROS2) is a framework for writing robot software, which nUWAy utilises (previously, it used ROS). Due to the Data Distribution Service that is utilised for communication, ROS2 ensures reliable data exchange in dynamic environments. It utilises a modular node architecture and enables low latency of high throughput LiDAR data. The bus has multiple LiDAR sensors: SICK 2D LiDAR is for low altitude obstacle detection, IBEO Lux 4-layer Automotive LiDAR is for short-to-medium range perception and Velodyne 360-degree LiDAR, which is used for high-resolution imaging. Further work will focus on the Velodyne VLP-16 LiDAR (front and back) since it has up to a 100-meter range and wide horizontal (360 degrees) and vertical (30 degrees) field of view (Velodyne LiDAR, Inc., 2018).

The nUWAy shuttle bus can predict driving commands (velocity and steering angle) based on camera input, while efficient LiDAR driving is yet to be implemented. There are 4 available architectures: SparseResNet9D uses standard residual blocks, SparseResNet9D_Mamba_base integrates Mamba

layers to model temporal dependencies in sequences, SparseResNet9D_Mamba_Temporal applies Mamba layers temporally after feature extraction, and SparseResNet9D_uncertainty_aware which outputs uncertainty. For a whole architecture, please, refer to Appendix A. Training script preprocesses the data, so it creates a dataset including point cloud and the corresponding driving command. The code uses techniques like data augmentation which improves training efficiency. During inference, the code merges front and rear LiDAR data and transforms point cloud data into information suitable for the NN format to predict driving speed and the steering angle, which are published as control commands to the bus.

A.1.3 Project Objectives

To enhance the state of the art, the research hypothesis proposes that by incorporating Mamba-based temporal modelling and probabilistic uncertainty estimation into the decision-making pipeline based on the deep learning model, it will enhance the performance of the autonomous driving.

Objectives:

- Extend the model to predict 10 timesteps of future trajectory (balancing prediction with computational constrains)
- Integrate uncertainty estimation methods with evidential regression
- Develop uncertainty aware control to adapt driving behaviour

This research will integrate Mamba layer architecture at the inference level and enable adaptive responses to uncertainty, in turn, addressing a major limitation in the current models. By combining trajectory prediction with real-time uncertainty quantification, the state of the current art will be advanced, offering practical insights for deploying safer autonomous systems that lie within industry constraints. These outcomes will help to improve the public's trust in automated systems, helping to speed up the bus's filtration into the field.

A.2 Methodology and Methods

Data Acquisition and Preprocessing

Existing LiDAR ROSbags from the bus (Velodyne sensors) will be used as a data input. They can be found on the lab PC or collected on the bus. The existing code implements preprocessing of the data to a proper format, which can be used by subscribing to the ROSbag playback. If it's real-world data, **data_sorting.py** needs to be used twice: automatic sorting based on GPS and manual sorting, where each frame is inspected. When the data is sorted, **data_grouping_horizon_10.py** groups 10 point clouds into a NumPy array along with the driving commands.

Extension of the Model to Predict Future Trajectory

The existing model in **SparseResNet9D_Mamba_temporal.py** processes the LiDAR data through convolution and Mamba layers, optimised for long sequence processing, but outputs only one driving command. To extend this, the output layer needs to be modified to predict multiple future timestamps. This sequence-to-sequence change would involve adjusting the models' final layers to output 10 steps, each containing velocity and steering angle and the loss function to account for multiple outputs. In theory, this will anticipate for a smoother navigation. Computational efficiency should be ensured due to tensor optimisations for deployment on Orin.

Integration of Uncertainty Estimation Methods

SparseResNet9D_uncertainty_aware.py has needed elements, however, it does not combine the uncertainty with temporal prediction. Using these elements for the model with temporal prediction will expand the output by including velocity, steering angle and the evidential parameters per timestep. During training, evidential loss function, combining already implemented L1 loss, will be used to penalise overconfidence that evidential regression might introduce.

Development of Uncertainty-Aware Control System

To achieve this objective, a control system, that integrates uncertainty methods from earlier, needs to be designed to dynamically adjust the driving commands. Firstly, the parameters from evidential uncertainty will be used to compute aleatoric and epistemic uncertainties. Then, the control system will adjust speed and steering angle based on the uncertainty thresholds (calibrated through the simulation) and uncertainty type.

The success metrics will evaluate the possible improvement. The root mean squared value between predicted and the ground truth trajectories will be calculated to evaluate the trajectory accuracy.

A.3 Timeline and Risk Management

A.3.1 Timeline

The Initiation phase began on the first day of the semester. It involved various meetings specially with Patrick Riedel, a previous developer on the code that will be modified throughout this project. A large reason for the initiation phase was to complete all inductions and installations of important libraries. There was also an attempt at installing a simulation, however, due to the capabilities of the devise, this was not completed, instead a lab computer will be used.

Majority of the Planning phase has been taken up by obtaining a compatible setup and research, due to be completed with the submission of this report. From March 31st through to the report's submission, May 9th, all research found within the literature review was undertaken. A lot of the time within the Planning stage also contributed to understanding the code (as there are multiple variations of the model) and coming to grips with Mamba (it has a steep learning curve but is crucial to the project).

The Execution stage begins on May 12th and includes key milestones such as:

- **May 30th** - Model refinement
  To achieve this the output layer of the architecture needs to be adjusted, as well as the training script. The inference file needs to be modified to be able to handle sequence of data.

- **May 31st** - Evidential loss integration

- **August 18th** - Uncertainty aware control

- **September 12th** - Completed training and inference code

For full Gantt Chart please refer to Appendix B. Note: Critical Paths are in red. In short it follows as model refinement -> Uncertainty integration -> Control system deployment. Also, 2 weeks of buffer time were left towards the end of the project, as a fail-safe. If there are zero delays, this time will be used beneficially for extra training.

A.3.2 Risk Management

Safety Risks:

- Description: Driving errors caused by external individuals not associated with the project, which may lead to collision during testing.
  Likelihood: Medium
  Consequences: Potential for injury or damage to property.
  Mitigations: Use signs along the route, indicating the slow moving vehicle in testing.

Environmental Risks:

- Description: Batteries in the vehicle pose risk of leakage or fire if damaged.
  Likelihood: Low
  Consequences: Fire or chemical leakage can cause environmental contamination, property damage, injury or disruption of operations.
  Mitigations: Always use original charger and avoid overcharging.

Financial Risks:

- Description: Downfall of bus due to physical negligence and improper handling of code.
  Likelihood: Unlikely
  Consequences: The risk poses a possibility of failure to the bus's components or the shell itself, this will incur a need for repurchasing of items, thus creating a financial loss.
  Mitigations: Sound testing in the simulation will decrease the potential of risk. If any physical adjustments must be made to the bus, it will be discussed first with the supervisor. By taking time to properly test the code, as well as treating every physical aspect of the bus, including sensors, with care, this risk can be taken care of.

Project Outcomes:

- Description: Bus breakdown
  Likelihood: Moderate
  Consequences: If the bus was to breakdown this would create an inability to collect LiDAR data, and to deploy and test the code.
  Mitigations: This can be mitigated through the implementation of the alternate buses already on campus, while the affected bus is getting fixed.

- Description: Model is too computationally expensive due to multiple waypoint predictions. with uncertainties
  Likelihood: High
  Consequences: There is a potential for high computational power, and there is a high risk of insufficient training due to time constraints.
  Mitigations: The way point prediction can be decreased to fewer points if needed.

- Description: Lab PC failure
  Likelihood: Unlikely
  Consequences: Inability to test the model in the simulation, and thus, deploy it on the bus.
  Mitigations: The titan PC will be in use in the next couple of weeks and will be used as a fail-safe.

- Description: Loss of Data
  Likelihood: Unlikely
  Consequences: Loss of progress, leading to a need to restart or end the project prematurely.
  Mitigations: Saving a copy of the code on GitHub will help to lower the risk of losing the code.

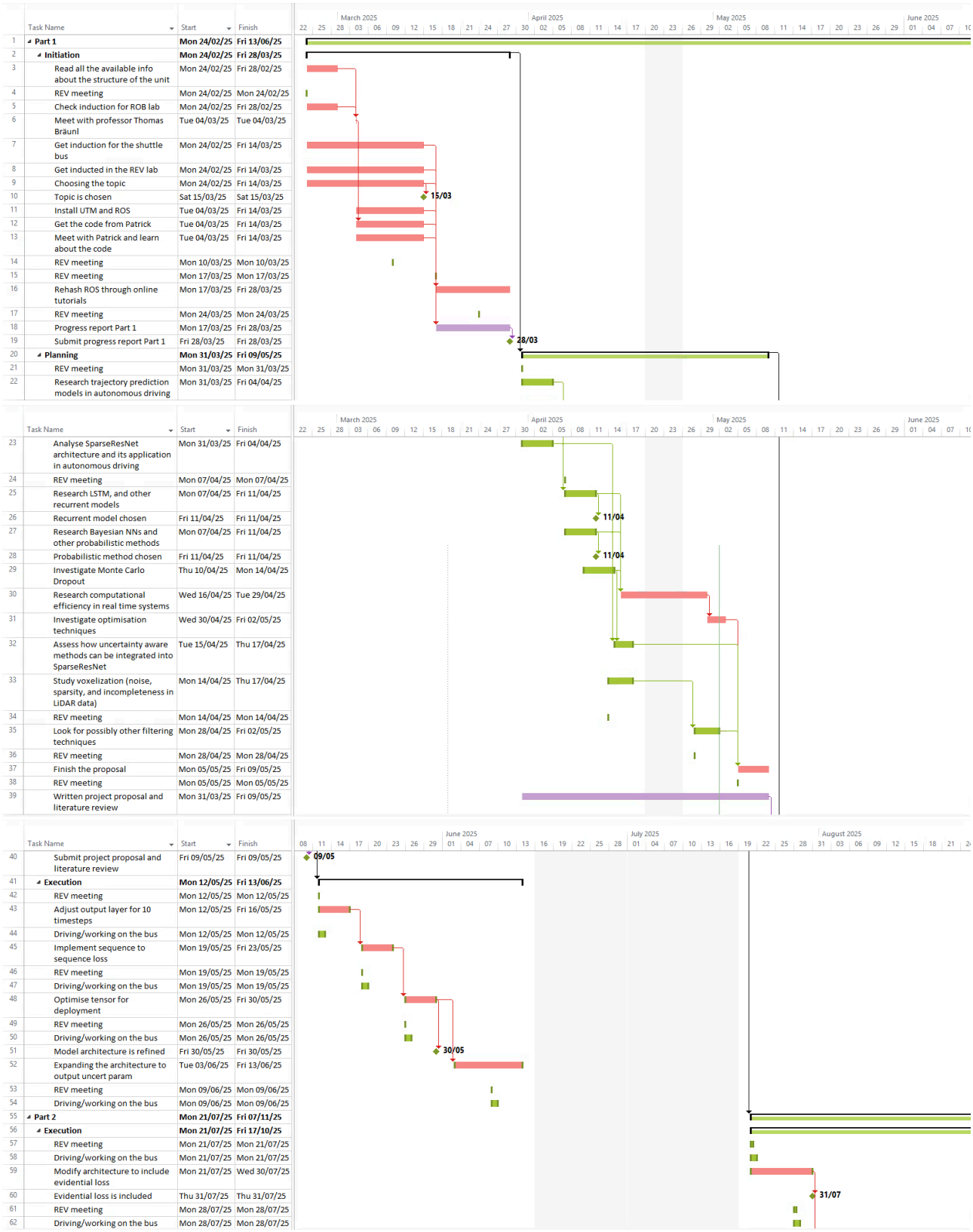
A.4 Progress to Date

While experimental implementation is pending, significant progress has been made in theoretical groundwork. Some time was spent cleaning up the code as it was handed over while the previous researcher was still working on it. There were also delays in the Initiation phase due to the inability to install necessary libraries on the device. This was mitigated by obtaining another device with suitable GPU. The most challenging part of this stage was to attain a setup with all the required libraries that are compatible with each other. The setup for this research is on Ubuntu-22.04:

- cudatoolkit==11.8
- torch==2.1.1
- torchvision==0.16.1
- torchaudio==2.1.1
- causal-conv1d==1.1.1
- mamba-ssm

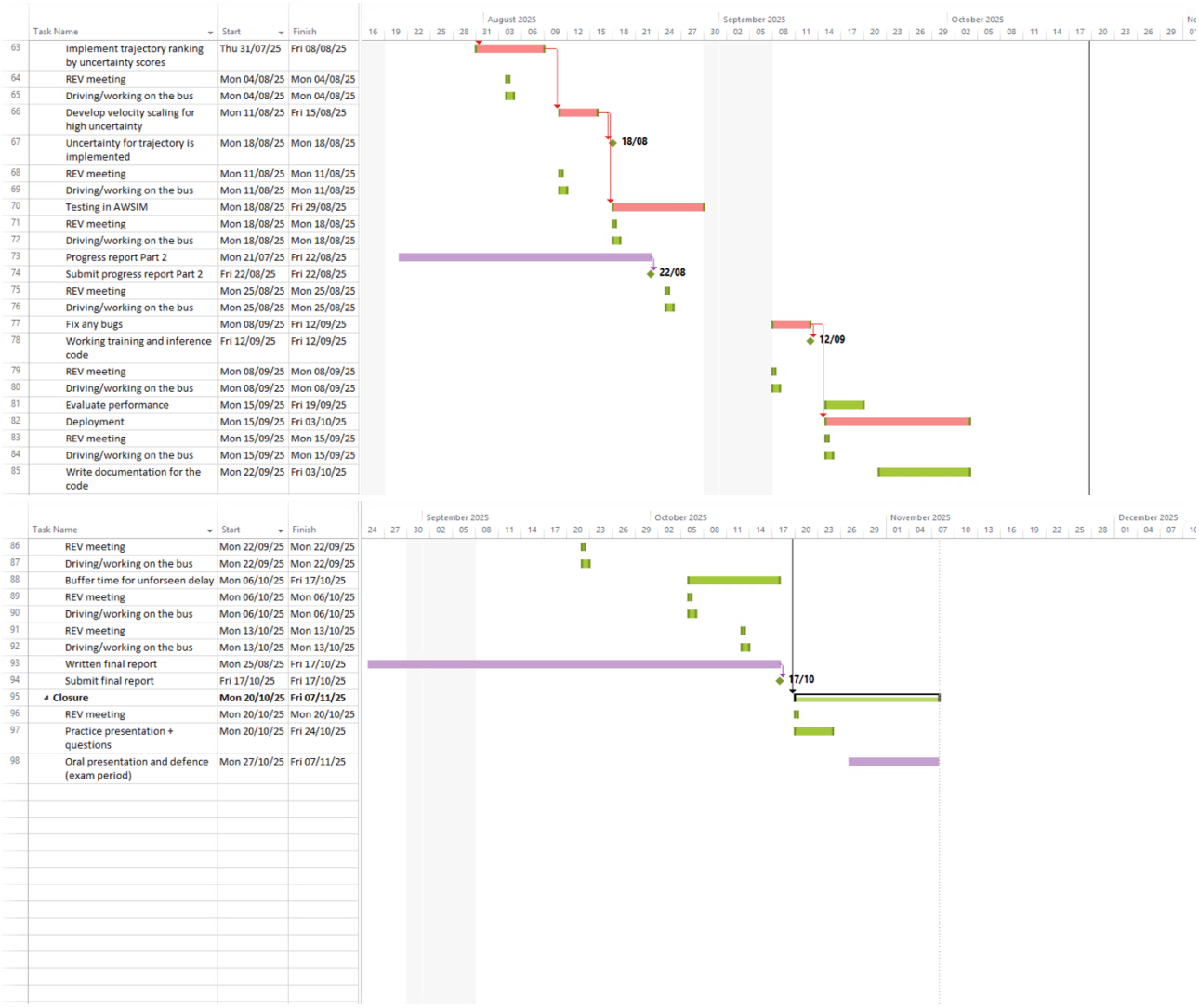
Apart from the delays in the Initiation stage, the Gantt Chart has been followed precisely, keeping everything on track. The next steps include modifying the code to output 10 waypoints, this also includes modifying the training script and rewriting the inference code, since inference does not support the processing of sequence of data.

Also, since nUWAy3 does not do simple LiDAR autonomous driving, a simple code was started. The ROS2 node limits the bus's speed based on the LiDAR data and feedback. It subscribes to /cmd_vel for velocity commands, uses LiDAR scans for obstacle detection and adjust speed accordingly (maximum 5 m/s). This is not the final version and has not been tested yet, as it was started last week.



	Task Name	Start	Finish
1	**Part 1**	**Mon 24/02/25**	**Fri 13/06/25**
2	**Initiation**	**Mon 24/02/25**	**Fri 28/03/25**
3	Read all the available info about the structure of the unit	Mon 24/02/25	Fri 28/02/25
4	REV meeting	Mon 24/02/25	Mon 24/02/25
5	Check induction for ROB lab	Mon 24/02/25	Fri 28/02/25
6	Meet with professor Thomas Bräunl	Tue 04/03/25	Tue 04/03/25
7	Get induction for the shuttle bus	Mon 24/02/25	Fri 14/03/25
8	Get inducted in the REV lab	Mon 24/02/25	Fri 14/03/25
9	Choosing the topic	Mon 24/02/25	Fri 14/03/25
10	Topic is chosen	Sat 15/03/25	Sat 15/03/25
11	Install UTM and ROS	Tue 04/03/25	Fri 14/03/25
12	Get the code from Patrick	Tue 04/03/25	Fri 14/03/25
13	Meet with Patrick and learn about the code	Tue 04/03/25	Fri 14/03/25
14	REV meeting	Mon 10/03/25	Mon 10/03/25
15	REV meeting	Mon 17/03/25	Mon 17/03/25
16	Rehash ROS through online tutorials	Mon 17/03/25	Fri 28/03/25
17	REV meeting	Mon 24/03/25	Mon 24/03/25
18	Progress report Part 1	Mon 17/03/25	Fri 28/03/25
19	Submit progress report Part 1	Fri 28/03/25	Fri 28/03/25
20	**Planning**	**Mon 31/03/25**	**Fri 09/05/25**
21	REV meeting	Mon 31/03/25	Mon 31/03/25
22	Research trajectory prediction models in autonomous driving	Mon 31/03/25	Fri 04/04/25
23	Analyse SparseResNet architecture and its application in autonomous driving	Mon 31/03/25	Fri 04/04/25
24	REV meeting	Mon 07/04/25	Mon 07/04/25
25	Research LSTM, and other recurrent models	Mon 07/04/25	Fri 11/04/25
26	Recurrent model chosen	Fri 11/04/25	Fri 11/04/25
27	Research Bayesian NNs and other probabilistic methods	Mon 07/04/25	Fri 11/04/25
28	Probabilistic method chosen	Fri 11/04/25	Fri 11/04/25
29	Investigate Monte Carlo Dropout	Thu 10/04/25	Mon 14/04/25
30	Research computational efficiency in real time systems	Wed 16/04/25	Tue 29/04/25
31	Investigate optimisation techniques	Wed 30/04/25	Fri 02/05/25
32	Assess how uncertainty aware methods can be integrated into SparseResNet	Tue 15/04/25	Thu 17/04/25
33	Study voxelization (noise, sparsity, and incompleteness in LiDAR data)	Mon 14/04/25	Thu 17/04/25
34	REV meeting	Mon 14/04/25	Mon 14/04/25
35	Look for possibly other filtering techniques	Mon 28/04/25	Fri 02/05/25
36	REV meeting	Mon 28/04/25	Mon 28/04/25
37	Finish the proposal	Mon 05/05/25	Fri 09/05/25
38	REV meeting	Mon 05/05/25	Mon 05/05/25
39	Written project proposal and literature review	Mon 31/03/25	Fri 09/05/25
40	Submit project proposal and literature review	Fri 09/05/25	Fri 09/05/25
41	**Execution**	**Mon 12/05/25**	**Fri 13/06/25**
42	REV meeting	Mon 12/05/25	Mon 12/05/25
43	Adjust output layer for 10 timesteps	Mon 12/05/25	Fri 16/05/25
44	Driving/working on the bus	Mon 12/05/25	Mon 12/05/25
45	Implement sequence to sequence loss	Mon 19/05/25	Fri 23/05/25
46	REV meeting	Mon 19/05/25	Mon 19/05/25
47	Driving/working on the bus	Mon 19/05/25	Mon 19/05/25
48	Optimise tensor for deployment	Mon 26/05/25	Fri 30/05/25
49	REV meeting	Mon 26/05/25	Mon 26/05/25
50	Driving/working on the bus	Mon 26/05/25	Mon 26/05/25
51	Model architecture is refined	Fri 30/05/25	Fri 30/05/25
52	Expanding the architecture to output uncert param	Tue 03/06/25	Fri 13/06/25
53	REV meeting	Mon 09/06/25	Mon 09/06/25
54	Driving/working on the bus	Mon 09/06/25	Mon 09/06/25
55	**Part 2**	**Mon 21/07/25**	**Fri 07/11/25**
56	**Execution**	**Mon 21/07/25**	**Fri 17/10/25**
57	REV meeting	Mon 21/07/25	Mon 21/07/25
58	Driving/working on the bus	Mon 21/07/25	Mon 21/07/25
59	Modify architecture to include evidential loss	Mon 21/07/25	Wed 30/07/25
60	Evidential loss is included	Thu 31/07/25	Thu 31/07/25
61	REV meeting	Mon 28/07/25	Mon 28/07/25
62	Driving/working on the bus	Mon 28/07/25	Mon 28/07/25

Figure A.2: Gantt chart for the Project, Part 1



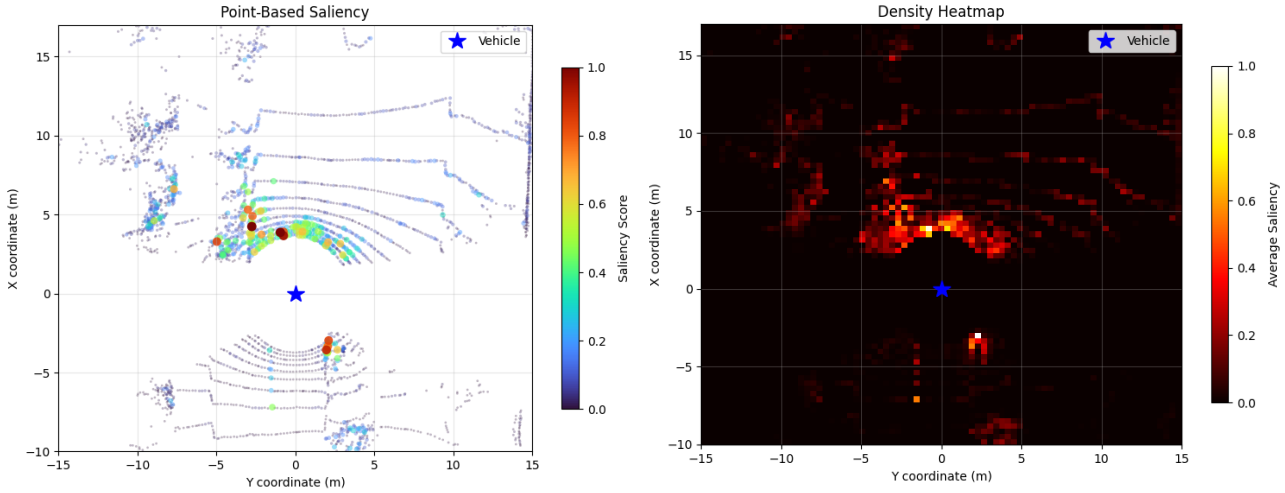
	Task Name	Start	Finish
63	Implement trajectory ranking by uncertainty scores	Thu 31/07/25	Fri 08/08/25
64	REV meeting	Mon 04/08/25	Mon 04/08/25
65	Driving/working on the bus	Mon 04/08/25	Mon 04/08/25
66	Develop velocity scaling for high uncertainty	Mon 11/08/25	Fri 15/08/25
67	Uncertainty for trajectory is implemented	Mon 18/08/25	Mon 18/08/25
68	REV meeting	Mon 11/08/25	Mon 11/08/25
69	Driving/working on the bus	Mon 11/08/25	Mon 11/08/25
70	Testing in AWSIM	Mon 18/08/25	Fri 29/08/25
71	REV meeting	Mon 18/08/25	Mon 18/08/25
72	Driving/working on the bus	Mon 18/08/25	Mon 18/08/25
73	Progress report Part 2	Mon 21/07/25	Fri 22/08/25
74	Submit progress report Part 2	Fri 22/08/25	Fri 22/08/25
75	REV meeting	Mon 25/08/25	Mon 25/08/25
76	Driving/working on the bus	Mon 25/08/25	Mon 25/08/25
77	Fix any bugs	Mon 08/09/25	Fri 12/09/25
78	Working training and inference code	Fri 12/09/25	Fri 12/09/25
79	REV meeting	Mon 08/09/25	Mon 08/09/25
80	Driving/working on the bus	Mon 08/09/25	Mon 08/09/25
81	Evaluate performance	Mon 15/09/25	Fri 19/09/25
82	Deployment	Mon 15/09/25	Fri 03/10/25
83	REV meeting	Mon 15/09/25	Mon 15/09/25
84	Driving/working on the bus	Mon 15/09/25	Mon 15/09/25
85	Write documentation for the code	Mon 22/09/25	Fri 03/10/25
86	REV meeting	Mon 22/09/25	Mon 22/09/25
87	Driving/working on the bus	Mon 22/09/25	Mon 22/09/25
88	Buffer time for unforseen delay	Mon 06/10/25	Fri 17/10/25
89	REV meeting	Mon 06/10/25	Mon 06/10/25
90	Driving/working on the bus	Mon 06/10/25	Mon 06/10/25
91	REV meeting	Mon 13/10/25	Mon 13/10/25
92	Driving/working on the bus	Mon 13/10/25	Mon 13/10/25
93	Written final report	Mon 25/08/25	Fri 17/10/25
94	Submit final report	Fri 17/10/25	Fri 17/10/25
95	**Closure**	**Mon 20/10/25**	**Fri 07/11/25**
96	REV meeting	Mon 20/10/25	Mon 20/10/25
97	Practice presentation + questions	Mon 20/10/25	Fri 24/10/25
98	Oral presentation and defence (exam period)	Mon 27/10/25	Fri 07/11/25

Figure A.3: Gantt chart for the Project, Part 2

Appendix B: Saliency Map with Different Scenarios

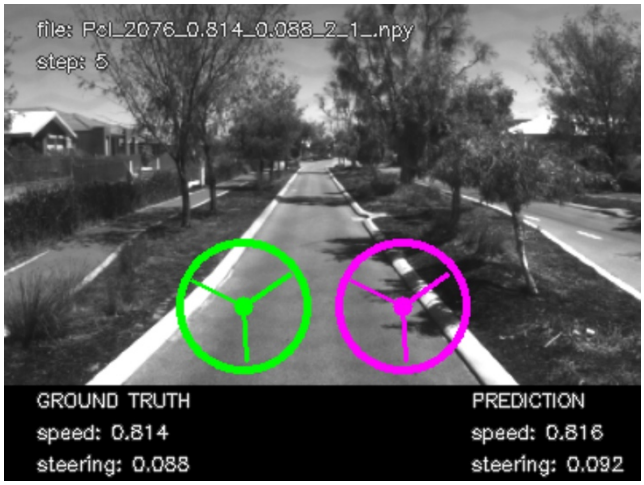
Below you can find more scenarios like lane following and intersections. The saliency maps have shown the model pays attention to the path and obstacles before it. Occasional high saliency score points appear from the rear LiDAR highlighting the model's ability to recognise patterns using both sensors.

(a) Camera View. A comparison of the model's predicted driving command (pink wheel, bottom right) against the recorded dataset's ground truth driving command (green wheel, bottom left).

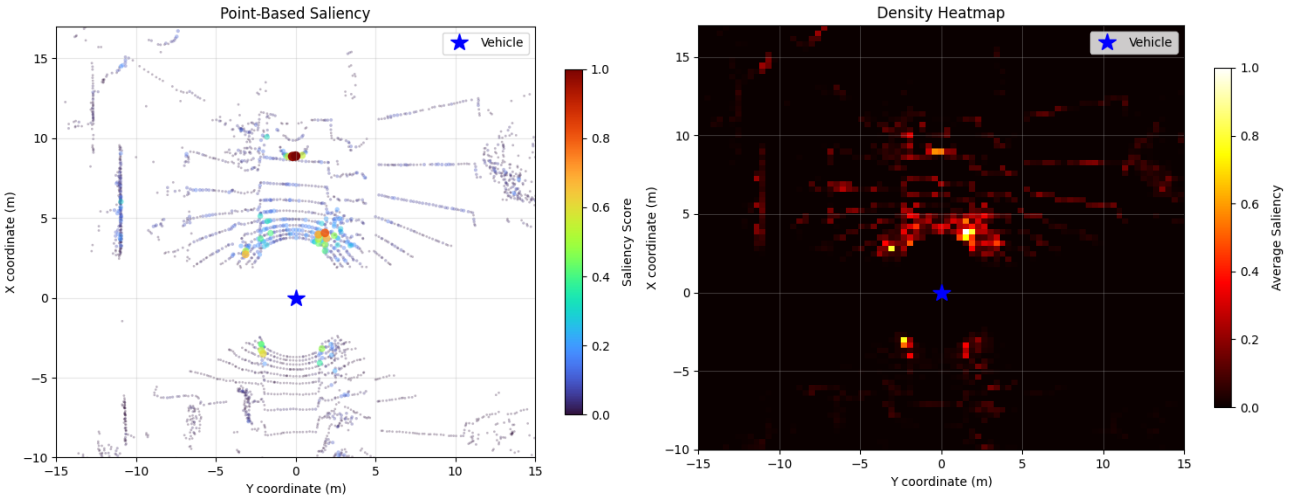


(b) Point-Based Saliency (c) Density Heatmap

Figure A.4: SparseResNet9D_Mamba_evidential Model, Lane Bay Pass

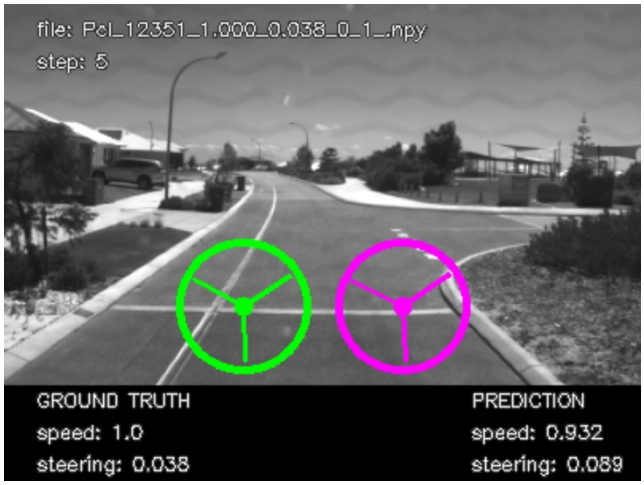


(a) Camera View. A comparison of the model's predicted driving command (pink wheel, bottom right) against the recorded dataset's ground truth driving command (green wheel, bottom left).

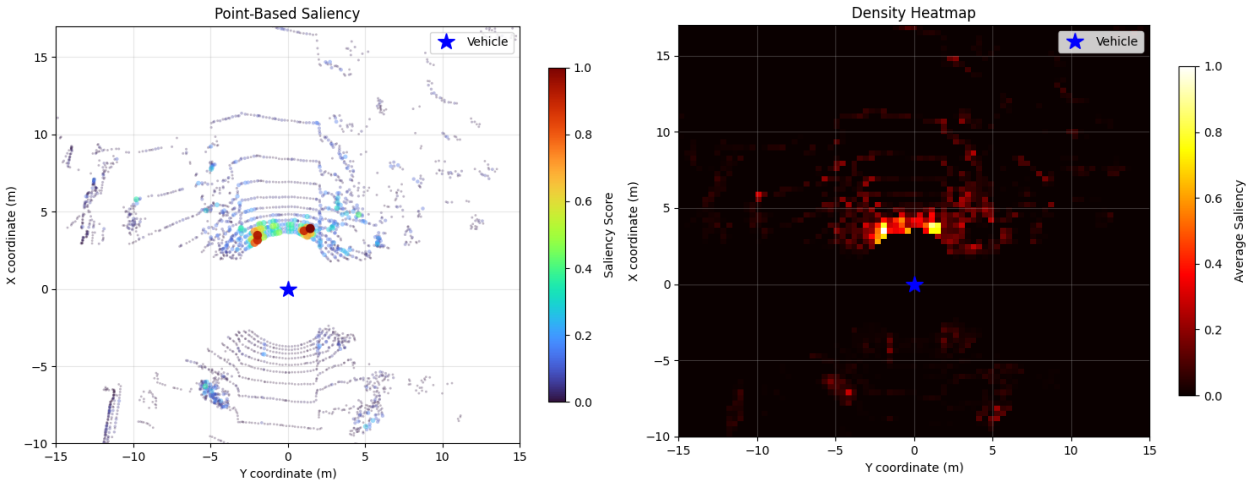


(b) Point-Based Saliency (c) Density Heatmap

Figure A.5: SparseResNet9D_Mamba_evidential Model, Lane Following

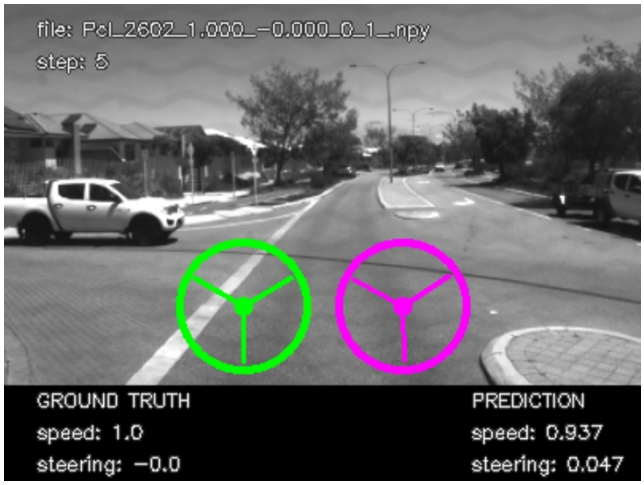


(a) Camera View. A comparison of the model's predicted driving command (pink wheel, bottom right) against the recorded dataset's ground truth driving command (green wheel, bottom left).

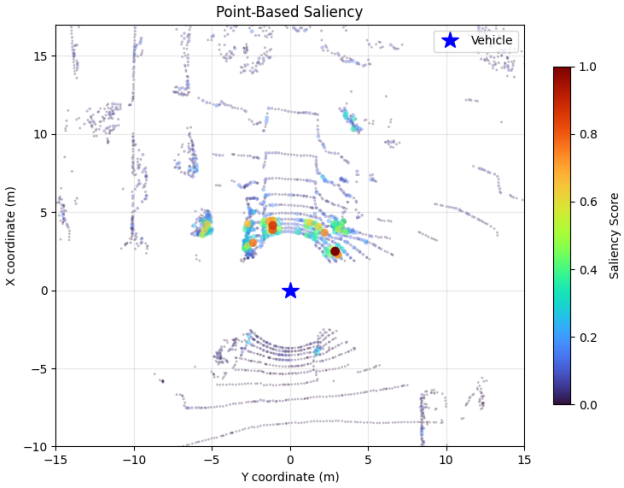


(b) Point-Based Saliency (c) Density Heatmap

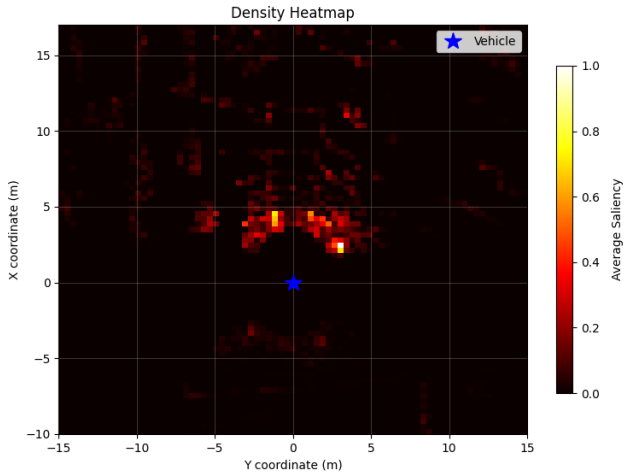
Figure A.6: SparseResNet9D_Mamba_evidential Model, Intersection



(a) Camera View. A comparison of the model's predicted driving command (pink wheel, bottom right) against the recorded dataset's ground truth driving command (green wheel, bottom left).



(b) Point-Based Saliency



(c) Density Heatmap

Figure A.7: SparseResNet9D_Mamba_evidential Model, Intersection with Obstacle

Appendix C: Evaluation of Transformation Matrices

A transformation matrix, T, for rotation, R, and translation, t, follows the standard format:

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{A.5}$$

R is 3x3 rotation matrix and t is a 3x1 translation vector. A yaw of 180º (refer to Equation A.6) and a pitch of -8º (refer to Equation A.7) are also applied and combined (refer to Equation A.4):

$$R_z\,(180º) = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.6}$$

$$R_y\,(\text{-}8º) = \begin{bmatrix} \cos(-8°) & 0 & \sin(-8°) \\ 0 & 1 & 0 \\ -\sin(-8°) & 0 & \cos(-8°) \end{bmatrix} = \begin{bmatrix} \cos(8°) & 0 & -\sin(8°) \\ 0 & 1 & 0 \\ \sin(8°) & 0 & \cos(8°) \end{bmatrix} \tag{A.7}$$

$$R = \begin{bmatrix} \cos(8°) & 0 & -\sin(8°) \\ 0 & 1 & 0 \\ \sin(8°) & 0 & \cos(8°) \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -\cos(8°) & 0 & -\sin(8°) \\ 0 & -1 & 0 \\ -\sin(8°) & 0 & \cos(8°) \end{bmatrix} \tag{A.8}$$

The front sensor is located 0.9 m above the ground and -1.689 m from the centre of the vehicle, which give a translation vector as follows:

$$t = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -1.689 \\ 0 \\ 0.9 \end{bmatrix} \tag{A.9}$$

Combining Equation A.8 and Equation A.9, Equation A.5 gives a transformation matrix for front sensor:

$$T_{\text{front}} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -\cos(8°) & 0 & -\sin(8°) & -1.689 \\ 0 & -1 & 0 & 0 \\ -\sin(8°) & 0 & \cos(8°) & 0.9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.10}$$

Rear sensor has a pitch of 8º, and it is located 0.9 m above the ground and 1.689 m in x direction. Substituting these values into Equation A.7 and Equation A.9, T_{rear} is as follows:

$$T_{\text{rear}} = \begin{bmatrix} \cos(8°) & 0 & \sin(8°) & 1.689 \\ 0 & 1 & 0 & 0 \\ -\sin(8°) & 0 & \cos(8°) & 0.9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.11}$$