Università
di Genova

DIBRIS - Department of Informatics, Bioengineering, Robotics
and Systems Engineering

Analysis of Neural Trajectory Prediction for Collision Avoidance in Smart Wheelchairs

by

Pouya Bathaei Pourmand

Supervisors:

Prof. Luca Oneto

Prof. Maurizio Mongelli

Co-Supervisor:

Dr. Sara Narteni

July 2026

Declaration of Originality

I, Pouya Bathaei Pourmand, hereby declare that this thesis is my own work and all sources of information and ideas have been acknowledged appropriately. This work has not been submitted for any other degree or academic qualification. I understand that any act of plagiarism, reproduction, or use of the whole or any part of this thesis without proper acknowledgment may result in severe academic penalties.

To my family, whose unwavering support, trust, and encouragement
have guided me throughout this journey.

Acknowledgements

I would like to express my sincere gratitude to Prof. Luca Oneto, who supervised this thesis and supported me from the very beginning. His guidance and trust were essential in shaping the direction of this work, and I am grateful for his recommendation that enabled my collaboration with the CNR team.

I also wish to thank Dr. Maurizio Mongelli and Dr. Sara Narteni for their continuous support during the development of this thesis. Their expertise, patience, and constructive feedback greatly improved the quality of my work and made this research a valuable experience.

Finally, I would like to thank my friends for their encouragement and support throughout this journey.

Abstract

This thesis investigates the reliability of neural-network-based motion prediction for autonomous wheelchair navigation under a wide range of operational command conditions. The objective is to analyse when predicted trajectories become unstable or fail to reach the intended goal, without modifying the underlying model architectures or training procedures.

A trajectory-based evaluation framework is applied to several pretrained neural models. Two complementary experiments are conducted. The first examines input sensitivity by analysing how initial orientation, linear velocity, and angular velocity influence prediction stability across the command space. The second evaluates goal difficulty by analysing model behaviour across different target positions.

To better characterise performance, both strict and soft success criteria are introduced. The strict definition uses a fixed distance threshold ($d < 0.30$ m), while the soft criterion distinguishes between successful trajectories, near-miss cases, and clear failures.

To improve interpretability, decision tree classifiers are trained for each model to identify the input conditions associated with prediction failures. The results indicate that goal position and initial orientation are the dominant factors affecting trajectory stability. Models trained with different loss functions exhibit significantly different reliability profiles, with success rates ranging from 25.3% to 99.3%.

Overall, this work provides a transparent trajectory-level evaluation framework for analysing the stability and reliability of neural motion prediction systems, supporting informed model comparison and safer deployment in assistive wheelchair navigation.

Keywords: trajectory analysis, reliability evaluation, trajectory stability, neural networks, autonomous wheelchair, decision tree, interpretability.

Notation

This section summarises the main symbols and abbreviations used throughout this thesis. All quantities are defined upon first appearance in the text.

Symbol / Abbreviation	**Meaning**
ML	Machine Learning
ϕ	Initial orientation (rad)
v	Linear velocity command (m/s)
ω	Angular velocity command (rad/s)
$x_t = (x_t, y_t, \theta_t)$	Wheelchair pose at time step t
$\hat{X}$	Predicted trajectory sequence
T	Prediction horizon length
Δt	Temporal sampling step
d	Final distance from the goal
R_{success}	Strict success threshold radius
R_{near}	Near-success threshold radius
G	Number of evaluated goal configurations
N	Number of sampled input commands

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter introduces the context and motivations of this thesis, focusing on the analysis of neural-network-based trajectory prediction for autonomous wheelchair navigation. It presents the problem addressed in this work, outlines the objectives and contributions, and describes the overall structure of the thesis.

1.1 Background and Motivations

Autonomous wheelchairs increasingly rely on learning-based models to transform user commands into motion trajectories that are safe and comfortable for the user [1; 2; 3]. These models are typically trained on demonstrations or recorded operational data and are expected to generate predictable trajectories during execution.
Figure 1.1 shows representative examples of smart wheelchair platforms considered in assistive robotics applications.

Figure 1.1: Examples of smart wheelchair platforms equipped with sensing and computing units for autonomous navigation and collision avoidance.

Although the navigation task may appear simple at a physical level, the behaviour of data-driven motion predictors can vary significantly depending on the input conditions. In real-world scenarios, variations in initial orientation, velocity commands, or target positions may lead to trajectories that deviate from the expected behaviour, potentially resulting in unsafe or inefficient motion.

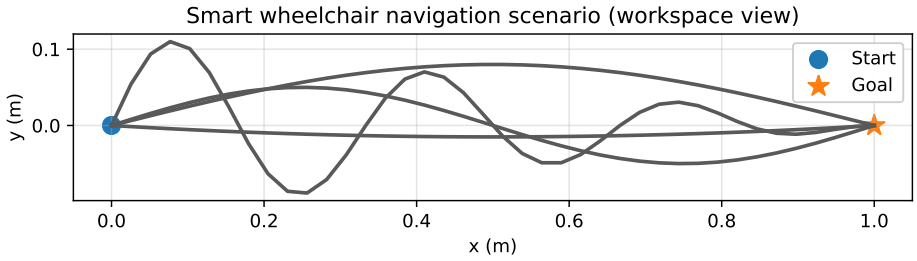

Figure 1.2: Illustrative workspace scenario for smart wheelchair navigation. The system starts from an initial pose and aims to reach a target position. Example predicted trajectories are shown to provide an intuitive view of motion execution in the workspace.

A typical smart wheelchair navigation system consists of perception, global planning, and local motion control modules. Classical approaches often rely on deterministic planners such as graph-based or sampling-based methods [4; 5; 6], to compute collision-free paths in structured environments. While these methods provide strong geometric guarantees, they may struggle to capture user-specific preferences or subtle motion patterns observed in real assistive settings.
From a canonical autonomous control perspective, navigation can be formulated as a model-based control problem in which the wheelchair kinematics are explic-

itly described and a planner generates feasible trajectories that satisfy geometric and collision constraints. In contrast, learning-based trajectory predictors implicitly approximate this mapping from data, without relying on an explicit analytical model. This difference motivates a systematic behavioural evaluation of neural predictors in the context of collision avoidance.
This limitation has motivated the integration of learning-based components, particularly for local trajectory generation [7; 8; 9].
Understanding how neural trajectory predictors behave under different input conditions is therefore essential for assessing their reliability in collision-avoidance-focused assistive applications [10]. Rather than modifying the training procedure or the model architecture, this thesis focuses on analysing the outputs produced by pretrained neural models in order to characterise their performance limits and potential failure modes.

1.2 Thesis Objectives and Contributions

This thesis addresses two high-level problems. First, identifying which combinations of input commands cause a pretrained neural trajectory predictor to fail to reach the intended goal — and characterising the structure of these failure regions in the command space. Second, understanding how the choice of training loss function affects the reliability of different neural models across a range of operational conditions and goal configurations.
To address these problems, the thesis develops a trajectory-level evaluation pipeline applied to five pretrained DNN-LNA models. The evaluation is based on uniform sampling of the input command space, binary and graded success labelling, empirical risk mapping, and decision tree analysis to extract interpretable failure rules — all without modifying the underlying model architectures.
The main contributions of this thesis are:

- **Trajectory-based evaluation framework.** A structured, model-agnostic evaluation pipeline that analyses predicted trajectories directly, enabling systematic assessment of model behaviour across a wide range of input conditions without requiring access to model weights or retraining.

- **Input sensitivity and goal-difficulty analysis.** A dual experimental study characterising how input commands (θ, v, ω) and target positions jointly affect goal-reaching performance, with empirical risk maps identifying failure-prone regions in the command space.

- **Decision tree interpretability layer.** Shallow decision tree classifiers fitted per model, producing explicit if-then rules that identify the precise

input conditions under which each model is likely to fail. These rules are consistent with the quantitative performance metrics and directly applicable to run-time collision-avoidance policies such as command filtering and hybrid fallback planning.

- **Comparative evaluation of multiple neural models.** A systematic comparison of five pretrained DNN-LNA models showing that the choice of loss function during training is the dominant factor determining model robustness, with strict success rates ranging from 25.3% to 99.3%.

1.3 Thesis Structure

The remainder of this thesis is organised as follows:

- **Chapter 2** reviews related work on neural trajectory prediction, learning-based navigation, and evaluation methodologies for autonomous systems.
- **Chapter 3** describes the problem formulation, the neural models under study, the input generation process, and the experimental evaluation framework.
- **Chapter 4** presents the experimental results and discusses the observed model behaviours across the considered scenarios.
- **Chapter 5** concludes the thesis and outlines possible directions for future work.

Chapter 2

Related Work

This chapter reviews literature relevant to learning-based trajectory prediction and trajectory-level evaluation in autonomous robotic navigation, with an emphasis on assistive systems such as smart wheelchairs. The goal is to position this thesis within existing work on neural motion prediction and performance analysis, rather than proposing new training procedures or explicit detection mechanisms. Learning-based approaches have been increasingly adopted to generate motion trajectories for mobile robots and assistive platforms, including neural trajectory predictors capable of modelling complex human and robot motion patterns [10; 11; 12]. Models trained on demonstrations or recorded operational data can learn complex mappings from command inputs to future motion trajectories, capturing non-linear relationships between user commands and resulting motion. Such approaches have demonstrated the ability to produce smooth and realistic trajectories in autonomous navigation tasks [1; 7; 8; 12]. In the context of assistive mobility, learning-based trajectory prediction can be used as a local motion generation module within a broader navigation pipeline [9], supporting responsive control under user commands. However, learned predictors may exhibit heterogeneous behaviour across the operational input space, motivating the need for systematic evaluation beyond average performance indicators.

Smart wheelchair navigation systems are typically organised [2; 13] into interacting components, including perception, global planning, and local motion control. Classical navigation pipelines commonly rely on deterministic planning algorithms, such as graph-based or sampling-based methods, to compute feasible collision-free paths, while a local controller tracks the planned motion under kinematic and collision-avoidance constraints [4; 5]. In assistive settings, navigation must also account for user intent, comfort, and shared autonomy requirements. Prior work has investigated shared autonomy and control-sharing paradigms for assistive mobile robots and smart wheelchairs, highlighting the importance of safety and user-centric performance evaluation [3; 13].

Trajectory-level evaluation aims to characterise the quality of predicted motion by analysing geometric and kinematic properties of trajectories [4; 5]. Common criteria include distance to a target, path length, smoothness, curvature, rotation profiles, and collision-free execution. Such metrics provide interpretable signals of performance and user comfort, and are widely used to benchmark navigation behaviour [6; 14]. For safety-critical assistive applications, it is often insufficient to report a single binary success metric. Recent work emphasises the analysis of failure modes and borderline behaviours in order to better understand model limitations and operational constraints [4; 5].

The sensitivity of learning-based navigation models to variations in command inputs and initial conditions has been explored in different robotic contexts. Changes in initial pose, velocity commands, or goal specifications can significantly affect the generated trajectories, even when the model remains fixed [7; 8]. Input-space analysis provides a practical evaluation strategy to map performance across operational command regions. This evaluation perspective is well-suited to the analysis of pretrained systems, where the focus is on characterising behaviour and identifying input regions associated with reduced goal-reaching performance, rather than retraining or redesigning models.

Comparing multiple neural models under consistent experimental conditions is a common approach to assess performance differences in navigation tasks. Comparative studies typically focus on identifying systematic behavioural differences and performance trade-offs across architectures, rather than training or optimising a single model [1; 8]. This perspective is particularly relevant in applied scenarios where pretrained models are provided and the primary objective is to validate model behaviour and support model selection based on empirical trajectory-level evidence.

The evaluation of safety-critical autonomous systems poses specific challenges that go beyond standard performance benchmarking. In safety-critical applications, it is essential not only to report average performance but also to characterise the conditions under which a system may fail, and to bound the probability of failure under operational constraints. Several frameworks have been proposed for systematic safety validation of autonomous navigation systems, including formal verification methods, runtime monitoring approaches, and data-driven risk characterisation [4; 5]. This thesis adopts a data-driven perspective: rather than providing formal guarantees, it uses empirical risk maps and interpretable classifiers to characterise failure regions in the input space and provide actionable safety guidance for the wheelchair system.

Explainable Artificial Intelligence (XAI) has emerged as a key research area to address the opacity of deep learning models in high-stakes applications. Post-hoc explanation methods aim to provide human-understandable insight into the behaviour of black-box models without modifying their architecture or training.

Among the most widely adopted approaches are feature attribution methods such as SHAP [15] and LIME [16], which assign importance scores to input features for individual predictions, and decision tree surrogates, which approximate the global input-output mapping of a complex model with an interpretable rule set. Decision tree surrogates are particularly attractive for safety-critical applications because they produce explicit if-then rules that can be directly used by a human operator or integrated into a control policy without requiring any probabilistic inference at runtime.

Surrogate modelling is a well-established technique in engineering and scientific computing, where a computationally expensive or opaque model is approximated by a simpler, more interpretable proxy. In the machine learning context, surrogate models are trained to mimic the input-output behaviour of a black-box system on a representative dataset, and are then used to extract global behavioural insights or to support decision-making under uncertainty. The use of shallow decision trees as surrogates for neural classifiers is well supported in the XAI literature [16], and has been applied in diverse domains including medical decision support, autonomous driving, and industrial fault detection. In this thesis, decision tree surrogates are fitted independently for each DNN-LNA model on the labelled dataset of trajectory outcomes, providing a compact and interpretable characterisation of each model's success and failure regions.

In summary, the literature on learning-based navigation highlights the need for trajectory-level analysis to understand model behaviour in autonomous and assistive systems. While many contributions focus on model design and training, this thesis adopts an evaluation-driven perspective by analysing the goal-reaching performance and failure patterns of pretrained neural trajectory predictors under varying command inputs and goal conditions. The use of decision tree surrogates as an interpretability layer connects this thesis to the broader XAI literature, while the focus on smart wheelchair navigation and collision avoidance positions it within the assistive robotics domain.

Chapter 3

Methodology

This chapter describes the methodology adopted in this thesis to analyse and evaluate the behaviour of neural trajectory predictors for smart wheelchair navigation under a wide range of operational conditions. The chapter first introduces the problem formulation, then details the neural models, the generation of input commands, the dataset construction, the success criteria, and the decision tree interpretability framework applied to characterise model behaviour.

3.1 Problem Formulation

Neural models trained on historical data may generate unstable or undesirable trajectories when exposed to certain combinations of input commands. These effects can manifest as abrupt rotations or trajectories that fail to reach the intended goal within acceptable tolerances.

The DNN-LNA model takes as input a full command vector that includes the initial orientation θ, the linear velocity v, the angular velocity ω, an occupancy map $\mathbf{M}$ (empty in the scenarios considered here, since no obstacles are present in the evaluation environment), and the target goal coordinates (x_g, y_g). The occupancy map is included in the interface for completeness, as it is part of the original model specification; in the experiments of this thesis it is set to zero everywhere, reflecting an obstacle-free workspace. Given this input, the model produces a predicted trajectory consisting of a sequence of planar poses and the corresponding predicted linear and angular speeds at each step:

$$\hat{\mathbf{X}} = f_\theta(\theta,\, v,\, \omega,\, \mathbf{M},\, x_g,\, y_g) = \left[(\hat{x}_t,\, \hat{y}_t,\, \hat{\theta}_t,\, \hat{v}_t,\, \hat{\omega}_t)\right]_{t=1}^{T}, \tag{3.1}$$

where $T = 30$ is the prediction horizon with a time step of $\Delta t = 0.1$ s, corresponding to a total horizon of 3 s.

In this thesis, the sensitivity analysis focuses on the three scalar command inputs (θ, v, ω), since the occupancy map is empty and the goal coordinates are held fixed per experiment. The full model interface is used as-is, without any modification to the model architecture or weights.
The primary objective of the navigation system is to reach the specified target while maintaining smooth motion. Depending on the input configuration, the predicted trajectory may deviate from this objective, motivating a trajectory-level evaluation of model behaviour. The goal of this thesis is to identify which combinations of (θ, v, ω) lead to reliable goal-reaching and which are associated with failure, using only the predicted trajectory outputs.

3.2 Overview of the Approach

The proposed methodology characterises when a neural network controller for a smart wheelchair behaves reliably and when its behaviour becomes unreliable under certain combinations of input commands. The core idea is to treat the predicted trajectory as the main object of analysis, rather than focusing only on point-wise prediction errors.
The methodology consists of the following main steps:

1. **Uniform input sampling:** command inputs (θ, v, ω) are sampled uniformly across their operational ranges to ensure broad coverage of the feasible command space.

2. **Trajectory generation:** each pretrained model is queried for each sampled input to obtain a predicted trajectory $\hat{\mathbf{X}}$.

3. **Outcome evaluation:** each trajectory is evaluated against strict and soft success criteria based on the final distance to the goal.

4. **Dataset construction:** a labelled dataset of input–outcome pairs $(\theta, v, \omega, x_g, y_g) \rightarrow y$ is assembled for each model across all goal configurations.

5. **Decision tree fitting:** a shallow decision tree classifier is fitted per model to extract human-readable rules identifying failure-prone input regions.

This pipeline is illustrated in Figure 3.1.

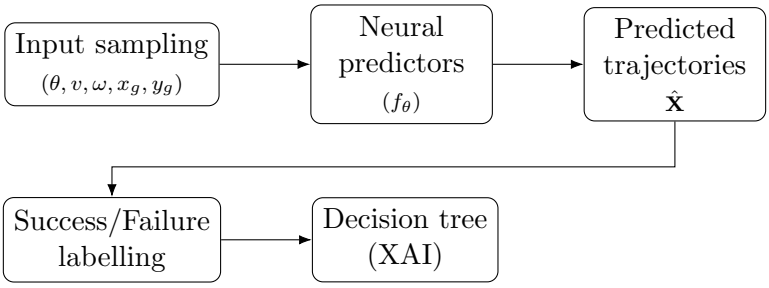

Figure 3.1: Overall pipeline of the proposed methodology. Input commands are sampled uniformly, fed to each pretrained neural model, and the resulting trajectories are evaluated against success criteria to build a labelled dataset. A decision tree is then fitted per model to extract interpretable failure rules.

3.3 Neural Trajectory Prediction Models

3.3.1 From Model-Based Control to Learning-Based Prediction

In canonical autonomous navigation, the motion of a wheelchair is typically governed by a kinematic model (e.g., a unicycle model):

$$\dot{x} = v\cos(\theta), \quad \dot{y} = v\sin(\theta), \quad \dot{\theta} = \omega, \tag{3.2}$$

where the control inputs (v, ω) are determined by a deterministic planner to ensure collision-free paths. In this thesis, we move from this explicit model-based approach to a learning-based paradigm. Here, the neural network f_θ implicitly learns the mapping from commands to feasible trajectories, approximating the underlying physics and navigation constraints directly from data. This motivates the use of the DNN-LNA models as black-box predictors and the need for a systematic evaluation of their behaviour across the input space.

3.3.2 Origin and Training of the Neural Models

The neural trajectory prediction models analysed in this thesis were developed and trained within the EU Horizon Europe project REXASI-PRO. The models implement a Deep Neural Network-based Local Navigation Approach (DNN-LNA) designed for smart wheelchair motion prediction and control. The complete training procedure, including dataset preparation, network architecture design, and optimisation, is described in the project deliverable D3.2 [17]. In this thesis, the models are used as pretrained predictors and are not retrained or modified. The focus of this work is therefore not on model development, but on analysing

the behaviour and reliability of their predicted trajectories under different input conditions.

3.3.3 State Representation and Coordinate System

The wheelchair motion is represented using a planar pose $\mathbf{x}_t = (x_t, y_t, \theta_t)$, where x_t and y_t denote the position in a fixed reference frame and θ_t denotes the absolute orientation at time step t. User commands are modelled as a linear velocity v (forward speed, assumed constant over the prediction horizon) and an angular velocity ω (commanded turning rate).
Throughout this thesis, trajectories are expressed in a robot-centric reference frame attached to the initial pose. The starting position is translated to the origin and the initial orientation is aligned with the x-axis. This normalisation simplifies the comparison between trajectories generated under different input commands and facilitates the definition of model-independent evaluation criteria.

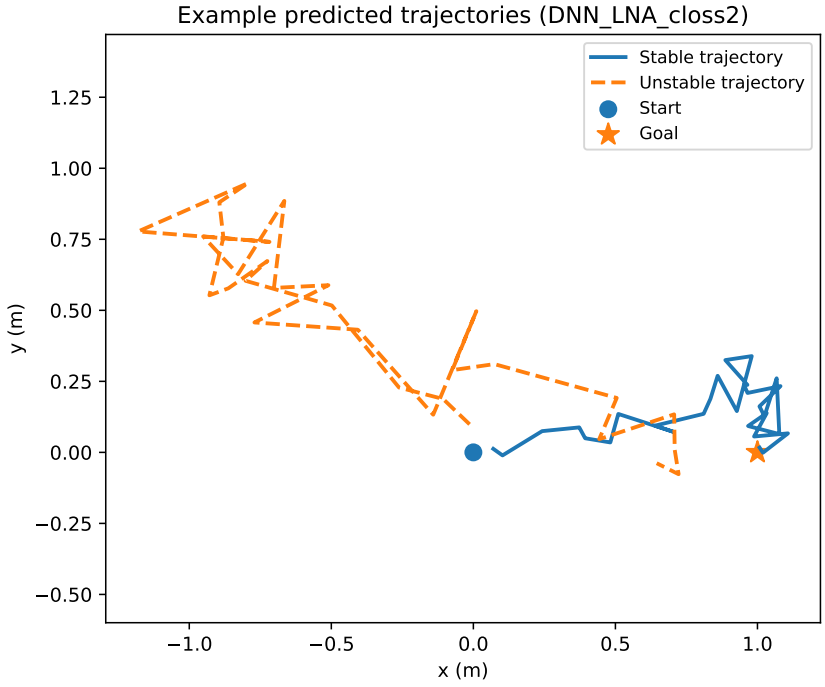

Figure 3.2: Workspace, initial state, and example predicted trajectories for a representative goal. The figure illustrates two qualitatively different trajectory behaviours produced by `DNN_LNA_closs2` under different input commands, highlighting the sensitivity of the predicted motion to variations in the input space.

3.3.4 Neural Network Architectures

Five pretrained architectures are analysed: `DNN_LNA_closs1`, `DNN_LNA_closs2`, `DNN_LNA_mse`, `DNN_LNA_on_wheel1`, and `DNN_LNA_sinu` [17]. Although their internal structures differ — in the number of layers, loss functions, and input parametrisation — they share the same high-level interface defined in Equation (3.1) and are all used as black-box predictors in this thesis. All models are implemented in Python 3.9 using PyTorch for inference, and NumPy and Matplotlib for trajectory processing and visualisation. The same pre-processing and normalisation steps are applied to all models, so that differences in behaviour can be attributed to the model architectures and loss functions rather than to inconsistencies in the input representation.

3.4 Input Space and Test Set Generation

3.4.1 Operational Ranges

The wheelchair controller operates within predefined admissible ranges for the three scalar command inputs used in the sensitivity analysis:

- absolute orientation $\theta \in [-\pi, \pi]$ rad;
- linear velocity $v \in [v_{\min}, v_{\max}]$ m/s;
- angular velocity $\omega \in [\omega_{\min}, \omega_{\max}]$ rad/s.

The numerical bounds are fixed according to the specifications provided by the REXASI-PRO project team and are reported explicitly in Chapter 4 to ensure reproducibility.
A total of $N = 200$ input commands are generated by uniform random sampling over these ranges. This sample size provides sufficient coverage of the operational envelope while remaining computationally tractable for five models and three goal configurations. Uniform sampling is preferred over structured grid sampling because it avoids aliasing effects at regular grid boundaries and provides a more natural representation of the variability that might be encountered in real operational conditions.

3.4.2 Grid-Based Input Coverage

The input set is constructed by sampling uniformly and independently across the three command dimensions (θ, v, ω) within their operational ranges. This defines a structured coverage of feasible model inputs used to systematically map how

trajectory outcomes vary across the command space. No perturbation around a nominal point is applied; the goal is to provide broad, uniform coverage of the operational envelope, including both typical commands close to those observed during training and more extreme configurations that may reveal model weaknesses.
The generated input sets are used in two complementary experimental settings. In Experiment 1 (Exp1), the focus is on analysing how trajectory outcomes vary across the full command space (θ, v, ω), independently of a specific goal. In Experiment 2 (Exp2), the same inputs are evaluated across three reference goal positions to study goal-dependent behaviour. This distinction underlies the experimental design presented in Chapter 4.

3.5 Dataset Construction and Labelling

For each sampled input $(\theta_i, v_i, \omega_i)$ and each goal configuration (x_g, y_g), the corresponding pretrained model is queried to produce a predicted trajectory $\hat{\mathbf{X}}_i$ as defined in Equation (3.1). The final predicted position $(\hat{x}_T, \hat{y}_T)$ is extracted and the Euclidean distance to the goal is computed:

$$d_i = \sqrt{(\hat{x}_T - x_g)^2 + (\hat{y}_T - y_g)^2}. \tag{3.3}$$

Each trajectory is then assigned a binary label based on the strict success criterion (Section 3.6):

$$y_i = \begin{cases} 1 \quad \text{(Success)} & \text{if } d_i < d_{\text{thresh}}, \\ 0 \quad \text{(Failure)} & \text{if } d_i \geq d_{\text{thresh}}. \end{cases} \tag{3.4}$$

Three reference goal configurations are evaluated, chosen to represent different levels of navigational complexity in terms of distance and lateral offset from the starting position. This procedure is repeated for each of the $G = 3$ goals, yielding a labelled dataset of $N \times G = 600$ observations per model. Each observation consists of the input features $(\theta_i, v_i, \omega_i, x_g, y_g)$ and the binary outcome label y_i. This dataset is used directly as input to the decision tree classifier described in Section 3.7.

3.6 Success Criteria

Two levels of success are defined to evaluate trajectory outcomes:

- **Strict success:** $d < 0.30$ m. The trajectory is considered successful if the final predicted position falls within 0.30 m of the target goal. The threshold

$d_{\text{thresh}} = 0.30$ m is grounded in the physical tolerances of the Rolland platform and aligned with the operational requirements of the REXASI-PRO project. This criterion is used to construct the binary labels for the decision tree analysis.

- **Soft success:** trajectories are grouped into three categories — Success ($d < 0.30$ m), Near-success ($0.30 \leq d < 0.50$ m), and Failure ($d \geq 0.50$ m). This graded criterion provides a more informative view of borderline cases, allowing distinction between models that barely miss the target and those that fail completely. The near-success category is particularly useful for identifying models that are close to the performance boundary and could potentially be improved with minor adjustments to the input commands.

3.7 Decision Tree Interpretability Framework

Risk maps and aggregated statistics provide a global view of model behaviour across the command space, but do not directly express explicit decision rules. To obtain interpretable conditions linking input commands to trajectory outcomes, a decision tree classifier is fitted independently for each model on the labelled dataset described in Section 3.5.

Decision trees are chosen as the interpretability tool for several reasons. First, they produce explicit if-then rules that directly map input conditions to predicted outcomes, making them immediately actionable for run-time safety applications such as command filtering or fallback planning. Second, unlike black-box models such as random forests or support vector machines, a shallow decision tree can be inspected and understood by a human operator without requiring any additional explanation layer. Third, the tree structure naturally reveals which input variables are the primary drivers of failure through the choice of root and internal split nodes.

The input features to the decision tree are $(\theta, v, \omega, x_g, y_g)$ and the target label is the binary Success/Failure outcome defined in Section 3.6. Tree depth is limited to four levels (`max_depth` $= 4$) and a minimum of ten samples per leaf is enforced (`min_samples_leaf` $= 10$) to prevent overfitting and to ensure that each leaf node is supported by a sufficient number of observations. These hyperparameters produce compact and interpretable rule sets that generalise well within the evaluated input range.

For each model, the fitted tree is evaluated on the full labelled dataset ($N \times G = 600$ samples) and the training accuracy is reported as a consistency check between the tree rules and the observed trajectory outcomes. The resulting trees are presented in Chapter 4 for the two most informative models, and the remaining three trees are reported in Appendix A for completeness.

3.8 Implementation Details

All experiments are implemented in Python 3.9. Model loading and inference are performed using PyTorch, while trajectory processing and statistical analysis rely on NumPy and SciPy. Figures and visualisations are produced with Matplotlib. Decision trees are implemented using `scikit-learn` with `DecisionTreeClassifier` and the Gini impurity criterion.
Input sampling, trajectory generation, labelling, and decision tree fitting are organised into modular scripts so that each step of the evaluation pipeline can be inspected and repeated independently. Results for different models and input sets are stored in structured directories with consistent naming conventions, facilitating reproducibility and systematic comparison of model behaviour.

Chapter 4

Experimental Results

This chapter reports the experimental evaluation of the proposed analysis pipeline on smart wheelchair trajectory predictors. The results are organised into two primary experiments: (i) an **input-space sensitivity analysis (Exp1)** and (ii) a **goal-based difficulty analysis (Exp2)**. For each experiment, we report the main trends observed across the neural models and discuss risk regions and representative failure modes.

4.1 Experimental Setup

4.1.1 Models Under Study

All experiments evaluate the following five pretrained trajectory prediction models: `DNN_LNA_closs1`, `DNN_LNA_closs2`, `DNN_LNA_mse`, `DNN_LNA_on_wheel1`, `DNN_LNA_sinu`. The models are treated as black-box predictors and are not retrained. Their architectures and training procedures are described in the project documentation [17].

4.1.2 Input Space and Sampling

Each test input is defined by the absolute orientation θ, the linear velocity v, and the angular velocity ω. The admissible operational ranges are:

- $\theta \in [-\pi, \pi]$ rad,
- $v \in [-1.05, 2.88]$ m/s,
- $\omega \in [-1.99, 1.99]$ rad/s.

For Exp1, $N = 200$ inputs are generated by uniform sampling over these ranges. For each sampled input, each model produces a predicted trajectory of $T = 30$

steps with a time step of $\Delta t = 0.1$ s, corresponding to a total prediction horizon of 3 s.

4.1.3 Trajectory Outputs

For each input and each model, the predicted trajectory is stored as a sequence of planar poses $\hat{\mathbf{x}}_t = (x_t, y_t, \theta_t)$ together with the predicted linear and angular speeds $(\hat{v}_t, \hat{\omega}_t)$, for $t = 1, \ldots, T$. The final predicted position $(\hat{x}_T, \hat{y}_T)$ is extracted to compute the goal-reach metric defined in Section 3.6.

4.1.4 Success Criteria and Labels

Each trajectory is evaluated based on the final Euclidean distance d from the predicted endpoint to the target goal. As defined in Section 3.6, two criteria are applied:

- **Strict success:** $d < d_{\text{thresh}} = 0.30$ m.
- **Soft success:** trajectories are grouped into *Success* ($d < 0.30$ m), *Near-success* ($0.30 \leq d < 0.50$ m), and *Failure* ($d \geq 0.50$ m).

The threshold $d_{\text{thresh}} = 0.30$ m is grounded in the physical tolerances of the Rolland platform and is aligned with the operational requirements of the REXASI-PRO project, as discussed in Section 3.6.

4.2 Experiment 1: Input-Space Sensitivity Analysis

4.2.1 Objective

The objective of Exp1 is to characterise how predicted trajectories vary across the operational command space and to identify input regions that trigger failure, providing an empirical mapping between user commands (θ, v, ω) and trajectory-level outcomes.

4.2.2 Representative Trajectories

To provide an interpretable qualitative comparison, representative trajectories are reported for the worst-performing (`DNN_LNA_closs1`) and best-performing (`DNN_LNA_closs2`) models under a reference goal (1.0, 0.0).

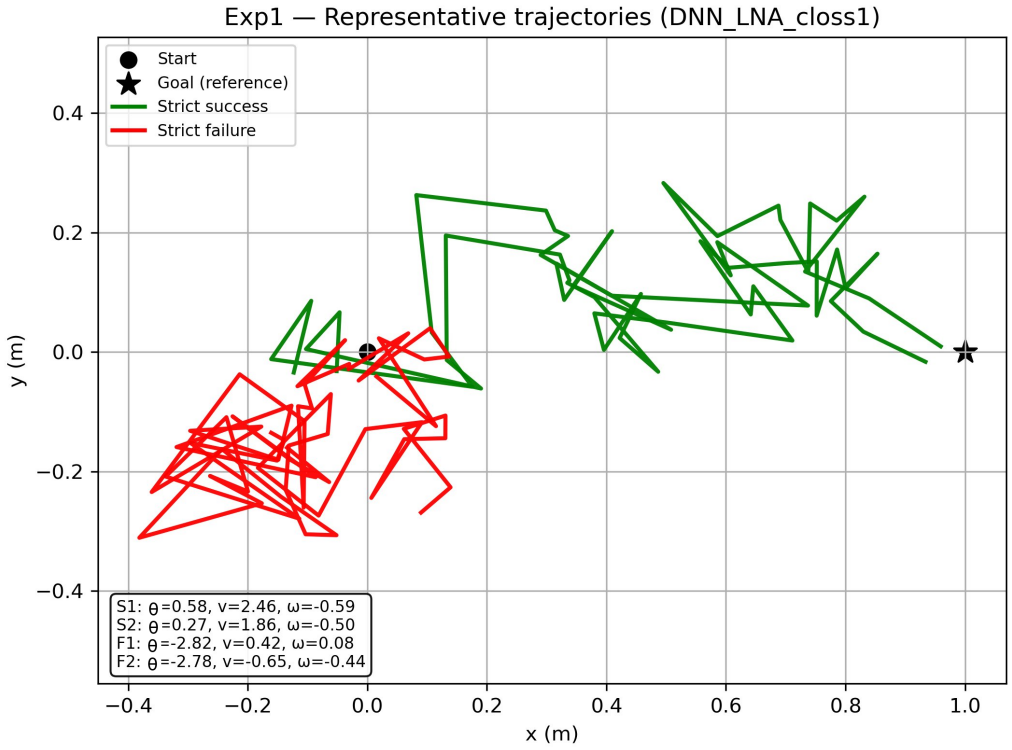

Figure 4.1: Exp1: Representative predicted trajectories for `DNN_LNA_closs1` (reference goal (1.0, 0.0)). Green indicates strict success ($d < 0.30$ m); red indicates failure ($d \geq 0.30$ m).

Figure 4.1 illustrates the characteristic failure pattern of `DNN_LNA_closs1`: failed trajectories (red) are predominantly associated with extreme initial orientations near $\theta \approx \pm\pi$, where the wheelchair starts facing away from the goal. Successful trajectories (green) originate from orientations closer to zero, confirming that initial orientation is the primary determinant of goal-reaching for this model.

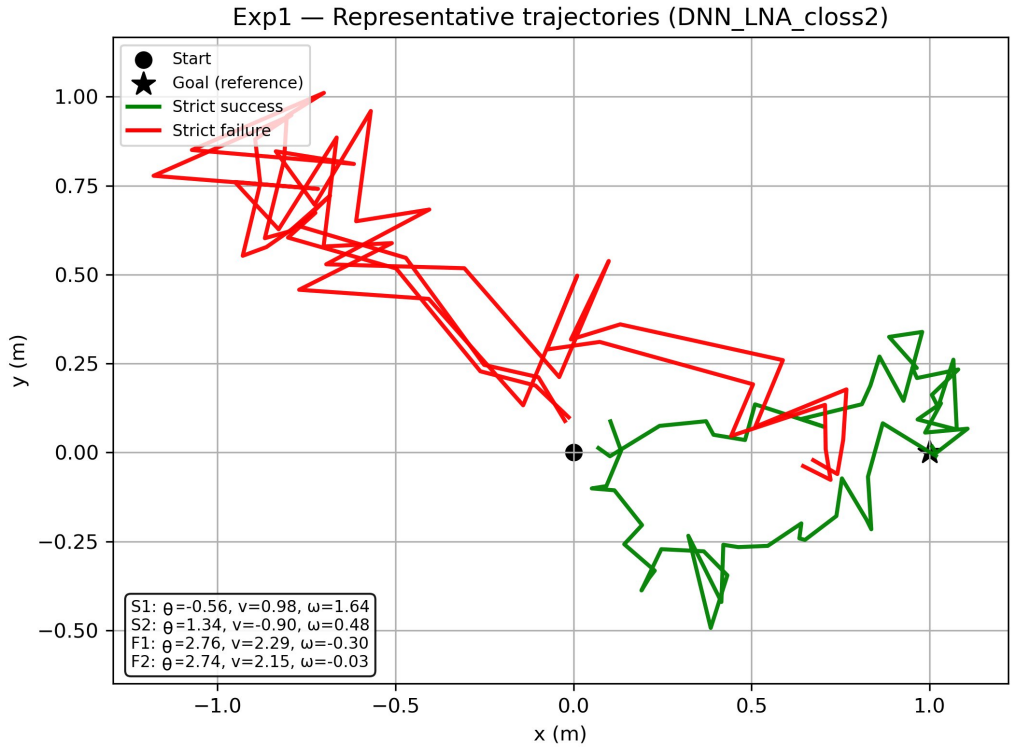

Figure 4.2: Exp1: Representative predicted trajectories for `DNN_LNA_closs2` (reference goal (1.0, 0.0)), using the same colour convention as Figure 4.1.

In contrast, Figure 4.2 shows that `DNN_LNA_closs2` produces reliable goal-directed trajectories across a much wider range of input conditions. Even under challenging orientations, the model consistently generates paths that converge towards the target, with only a small number of failures visible at the most extreme orientations.

4.2.3 Risk Map

To provide a compact overview, trajectory outcomes are aggregated over sampled inputs and visualised as a risk map in the command space. **Risk** is defined as the empirical failure rate within a specific input bin — the fraction of trajectories in that bin that fail to reach the goal within the strict threshold.
Formally, the input space (θ, ω) is partitioned into B uniform bins of equal width along each axis. For each bin $\mathcal{U}_b$, the risk is:

$$\mathcal{R}(\mathcal{U}_b) = \frac{1}{N_b} \sum_{j\,:\,(\theta_j,\omega_j)\in\mathcal{U}_b} \mathbf{1}[d_j \geq d_{\text{thresh}}]\,, \tag{4.1}$$

where N_b is the number of samples falling in bin $\mathcal{U}_b$, d_j is the final distance to goal for sample j, $d_{\text{thresh}} = 0.30$ m (Section 3.6), and $\mathbf{1}[\cdot]$ is the indicator function:

$$\mathbf{1}[d_j \geq d_{\text{thresh}}] = \begin{cases} 1 & \text{if } d_j \geq d_{\text{thresh}}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

The bins are constructed by partitioning $\theta \in [-\pi, \pi]$ and $\omega \in [-1.99, 1.99]$ into uniform intervals. Each bin contains all samples whose (θ, ω) values fall within its boundaries; the risk value assigned to the bin is the fraction of those samples that are failures.

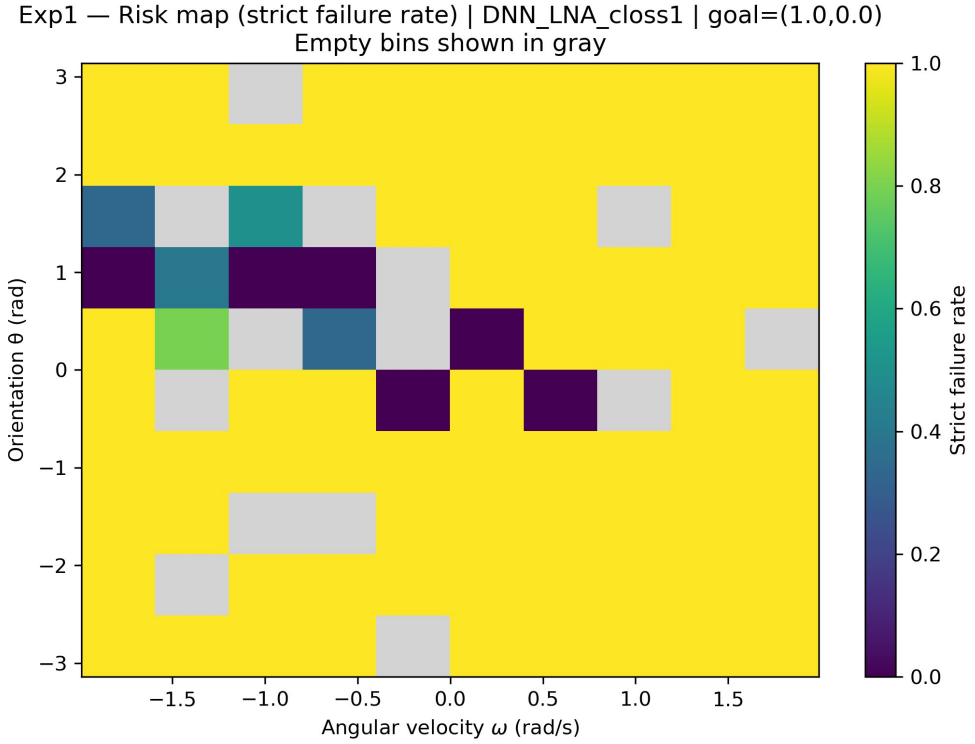

Figure 4.3: Exp1: Risk map for `DNN_LNA_closs1`. Colour intensity represents the failure rate $\mathcal{R}$ (Equation 4.1), highlighting regions where the model consistently generates trajectories that fail to reach the goal.

The risk map in Figure 4.3 reveals a clear structure: high failure rates are concentrated in the bands $\theta \approx \pm\pi$, corresponding to backward-facing orientations, and persist across the full range of angular velocities ω. The central orientation range ($|\theta| < 1.5$ rad) shows consistently lower failure rates regardless of ω, identifying it as a more reliable operational region for `DNN_LNA_closs1`. Gray bins indicate sparsely sampled regions where the failure rate estimate is less reliable.

4.2.4 Decision Tree Analysis

While risk maps provide a global visual overview of failure-prone command regions, they do not express explicit conditions. To obtain human-readable rules linking input commands to trajectory outcomes, a decision tree classifier is fitted independently for each model on the labelled dataset described in Section 3.5. The input features are $(\theta, v, \omega, \text{GoalX}, \text{GoalY})$ and the target label is binary Success/Failure based on the strict criterion. The same $N = 200$ samples are repeated across $G = 3$ goals, yielding 600 labelled observations per model. Tree depth is limited to four levels (`max_depth` $= 4$) with a minimum of ten samples per leaf (`min_samples_leaf` $= 10$).
The extracted if-then rules for the two most informative models are reported below. In each rule set, leaf nodes labelled **Success** correspond to input regions where the model reliably reaches the goal; leaf nodes labelled **Failure** correspond to regions of consistent goal miss. In the tree visualisations, **blue nodes indicate regions predicted as Success** and **orange nodes indicate regions predicted as Failure**.

DNN_LNA_closs2 (99.3% strict success, accuracy = 0.99)

The tree contains only 5 nodes (2 internal, 3 leaves). All three leaves predict Success, confirming near-universal reliability across the input space. The only identified risk-adjacent region is:

- **IF** $\theta \leq 2.27$ rad **THEN** Success (480 samples, 100% success)
- **IF** $\theta > 2.27$ rad **AND** $v \leq 1.78$ m/s **THEN** Success (97 samples, 97% success)
- **IF** $\theta > 2.27$ rad **AND** $v > 1.78$ m/s **THEN** Success (23 samples, 87% success)

The tree identifies a narrow risk-adjacent region: orientations near $\theta \approx \pi$ (wheelchair facing nearly backward) combined with high forward speed. Even in this region the model mostly succeeds, explaining the near-perfect overall rate.

Figure 4.4: Decision tree for `DNN_LNA_closs2` ($N = 600$, train accuracy = 0.99, threshold = 0.30 m). The tree contains only 5 nodes (2 internal, 3 leaves), and all three leaf nodes predict Success, confirming near-universal reliability across the input space. The single relevant split at $\theta = 2.27$ rad identifies the only region of marginally reduced confidence.

DNN_LNA_closs1 (25.3% strict success, accuracy = 0.90)

The tree contains 27 nodes (13 internal, 14 leaves), reflecting that this model only succeeds under a narrow combination of conditions. The root splits on `GoalX`, meaning goal distance is the primary determinant. The main extracted rules are:

- **IF** GoalX ≤ 0.75 m **THEN** Failure (majority, consistent failure for near-lateral goals)
- **IF** GoalX > 0.75 m **AND** $\theta \leq -2.36$ rad **THEN** Failure (backward-facing orientation, any velocity)
- **IF** GoalX > 0.75 m **AND** $\theta > -2.36$ rad **AND** $v \leq 0.58$ m/s **THEN** Failure (low velocity prevents goal reach)
- **IF** GoalX > 0.75 m **AND** $-2.36 < \theta \leq 1.80$ rad **AND** $v > 0.58$ m/s **THEN** Success (narrow forward-facing corridor with sufficient speed)

These rules confirm that `DNN_LNA_closs1` relies on a narrow combination of favourable conditions to achieve success. In contrast to `DNN_LNA_closs2`, which succeeds for nearly any input, success here requires both a forward-facing orientation and sufficient velocity, while near-lateral goals (GoalX ≤ 0.75 m) are almost universally associated with failure regardless of other inputs. This joint dependency on goal distance, orientation, and velocity directly explains the low overall success rate of 25.3% observed in Exp2: the majority of sampled inputs fall outside this narrow success corridor.

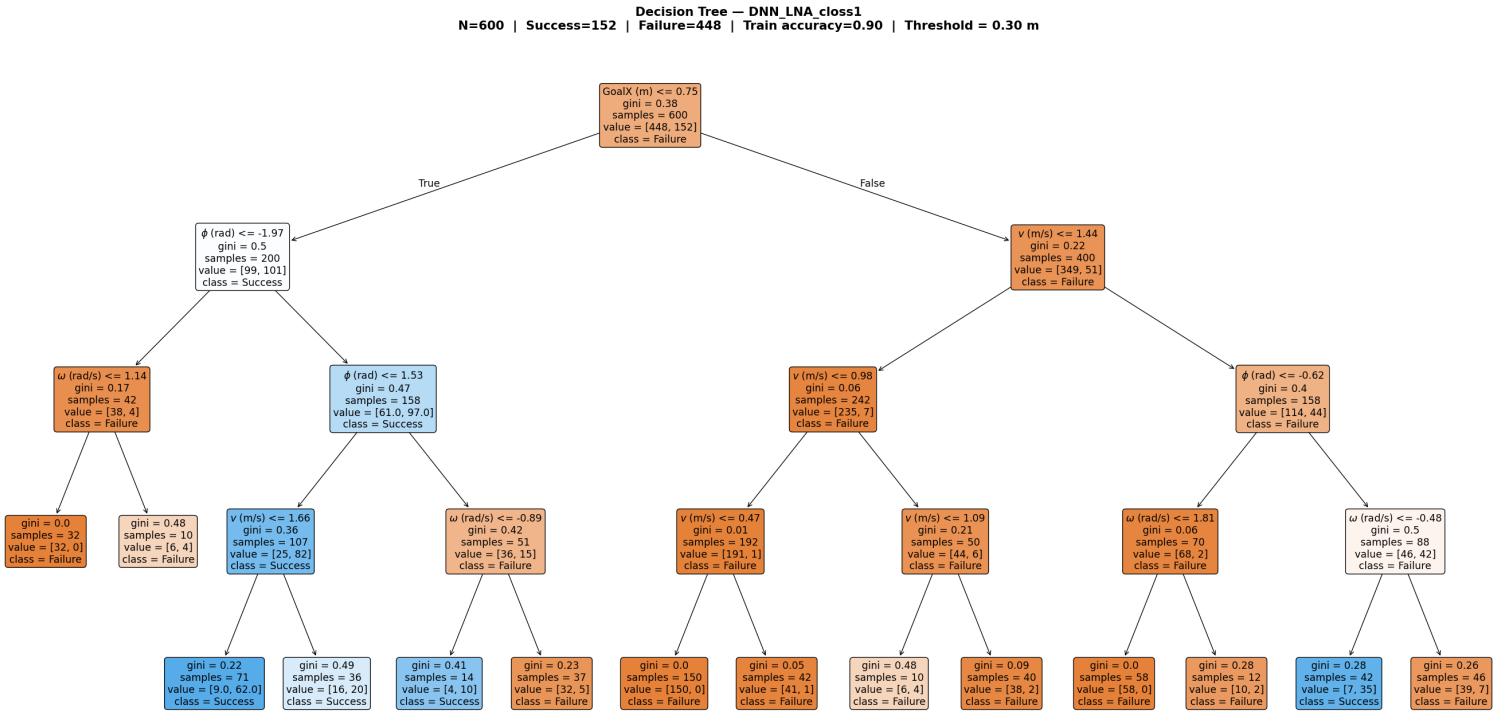

Figure 4.5: Decision tree for `DNN_LNA_closs1` ($N = 600$, train accuracy = 0.90, threshold = 0.30 m). The tree contains 27 nodes (13 internal, 14 leaves). The root splits on `GoalX`, meaning goal distance is the primary determinant of success. Most leaf nodes predict Failure, confirming the low overall success rate of 25.3%.

The sharp contrast between the two trees directly illustrates the impact of the loss function on model robustness. `DNN_LNA_closs2` requires only a single condition ($\theta \leq 2.27$ rad) to predict Success for 80% of observations, while `DNN_LNA_closs1` requires a conjunction of three conditions — goal distance, orientation, and velocity — to identify even a narrow success corridor. The full decision trees for all five models are provided in Appendix A.4.

4.2.5 Discussion of Interpretable Failure Patterns

The decision trees fitted for each model reveal interpretable structural differences that reflect deeper properties of the underlying training procedures. Several observations deserve closer analysis.

Why does initial orientation θ dominate the splits? For most models, the first or second split in the tree involves θ, particularly around $\theta \approx \pm\pi$. This is consistent with the physical interpretation: when the wheelchair faces directly away from the goal, the required turning manoeuvre is maximal. The DNN-LNA models, trained on demonstrations of nominal navigation, have limited coverage of such extreme configurations, leading to systematic prediction errors. The orientation θ is therefore both the most physically meaningful and the most statistically informative feature for predicting failure.

Why does `GoalX` appear as the root split for `DNN_LNA_closs1`? Unlike the other models, `DNN_LNA_closs1` has a root split on the goal coordinate `GoalX` rather than on an input command. This indicates that, for this model, the target position is a stronger predictor of failure than the initial orientation or velocity. Specifically, near-lateral goals (GoalX ≤ 0.75 m) cause almost universal failure regardless of the command inputs. This suggests that `DNN_LNA_closs1` has a systematic bias towards forward-directed navigation and is poorly calibrated for lateral displacement, a limitation not visible from aggregate success rates alone.

Why does `DNN_LNA_closs2` produce such a small tree? The 5-node tree for `DNN_LNA_closs2` (all leaves predicting Success) reflects the near-absence of failure in the labelled dataset. When the model succeeds on 596 out of 600 observations, a decision tree has almost no signal to split on — any partition produces a leaf that is almost entirely Success. The compactness of the tree is therefore not a limitation of the fitting procedure but a direct consequence of the model's exceptional robustness. The single split at $\theta = 2.27$ rad identifies the only region of marginally reduced confidence, which still predicts Success.

What is the relationship between tree complexity and model robustness? Across all five models, there is a clear inverse relationship between tree complexity (number of nodes) and overall success rate. `DNN_LNA_closs2` achieves 99.3% success with a 5-node tree; `DNN_LNA_on_wheel1` achieves 90.7% with 19 nodes; `DNN_LNA_mse` achieves 74.7% with 27 nodes; `DNN_LNA_closs1` achieves

25.3% with 27 nodes. A more robust model requires fewer conditions to describe its behaviour because its success region is broad and structurally simple. A fragile model requires many conditions because its success corridor is narrow and depends on a precise alignment of multiple input variables. Tree complexity is therefore a direct proxy for model fragility.

4.2.6 Summary of Exp1 Findings

Exp1 shows that trajectory failure is concentrated in specific command regions, particularly near $\theta \approx \pm\pi$ (backward orientation) and at low velocities. The risk maps and decision tree rules consistently identify the same failure-prone zones, confirming the validity of both analysis approaches. `DNN_LNA_closs2` operates reliably across almost the entire input space, while `DNN_LNA_closs1` succeeds only within a narrow forward-facing corridor — a difference that is directly attributable to their respective loss functions.

4.3 Experiment 2: Goal-Based Difficulty Analysis

4.3.1 Objective

The objective of Exp2 is to analyse how model performance changes as the navigation goal varies across the workspace, evaluating whether predicted trajectories reach a specified goal and how failure severity evolves across different goal configurations.

4.3.2 Goal Set and Protocol

A set of $G = 3$ goal positions is defined: (0.5, 0.5), (1.0, 0.0), and (1.5, −0.5). These represent different navigational complexities, ranging from lateral near-field displacement to long-range off-axis navigation. For each goal, the same $N = 200$ test inputs from Exp1 are reused.

4.3.3 Strict Success Rates and Average Distance

Table 4.1 reports strict success rates per model and per goal, and Table 4.2 reports the average final distance to goal.

Table 4.1 — Strict Success Rate (%) per Model and Goal
($d < 0.30$ m, $N = 200$ per cell, $G = 3$ goals — Rank 1 = best)

Model	Goal (0.5,0.5)	Goal (1.0,0.0)	Goal (1.5,−0.5)	Overall	Rank
DNN_LNA_closs1	50.5%	10.5%	15.0%	25.3%	5
DNN_LNA_closs2	100.0%	98.0%	100.0%	99.3%	1
DNN_LNA_mse	83.5%	76.5%	64.0%	74.7%	3
DNN_LNA_on_wheel1	91.0%	88.0%	93.0%	90.7%	2
DNN_LNA_sinu	79.5%	64.5%	63.5%	69.2%	4

Table 4.1: Exp2: Strict success rate (%) per model and goal ($d < 0.30$ m, $N = 200$ per cell, $G = 3$ goals). Overall = mean across all goals. Green cells indicate high success; red cells indicate low success.

Table 4.1 reveals substantial variability across models and goals. `DNN_LNA_closs2` achieves 99.3% strict success consistently across all three goals, while `DNN_LNA_closs1` reaches only 25.3% overall. Intermediate models (`DNN_LNA_mse`, `DNN_LNA_on_wheel1`, `DNN_LNA_sinu`) show success rates of 74.7%, 90.7%, and 69.2% respectively, with performance generally decreasing for Goal C (1.5, −0.5) — the farthest off-axis configuration.

Table 4.2 — Average Final Distance to Goal (m) per Model
($N = 200$ per cell, $G = 3$ goals — lower is better)

Model	Goal (0.5,0.5)	Goal (1.0,0.0)	Goal (1.5,−0.5)	Overall avg
DNN_LNA_closs1	0.305 m	0.603 m	0.538 m	0.482 m
DNN_LNA_closs2	0.123 m	0.137 m	0.129 m	0.130 m
DNN_LNA_mse	0.200 m	0.228 m	0.283 m	0.237 m
DNN_LNA_on_wheel1	0.188 m	0.190 m	0.150 m	0.176 m
DNN_LNA_sinu	0.216 m	0.261 m	0.255 m	0.244 m

Table 4.2: Exp2: Average final distance to goal (m) per model ($N = 200$ per cell, $G = 3$ goals). Lower is better. Green cells indicate small distance; red cells indicate large distance.

Table 4.2 provides a complementary view: even among trajectories that fail the strict threshold, `DNN_LNA_closs2` ends at an average distance of only 0.130 m from the goal, compared to 0.482 m for `DNN_LNA_closs1`. This indicates qualitatively

different failure modes — `DNN_LNA_closs2` produces near-miss trajectories, while `DNN_LNA_closs1` often diverges far from the target.

4.3.4 Soft Success Distribution

Figure 4.6 summarises the soft outcome distribution across models using $d_{\text{success}} = 0.30$ m and $d_{\text{near}} = 0.50$ m.

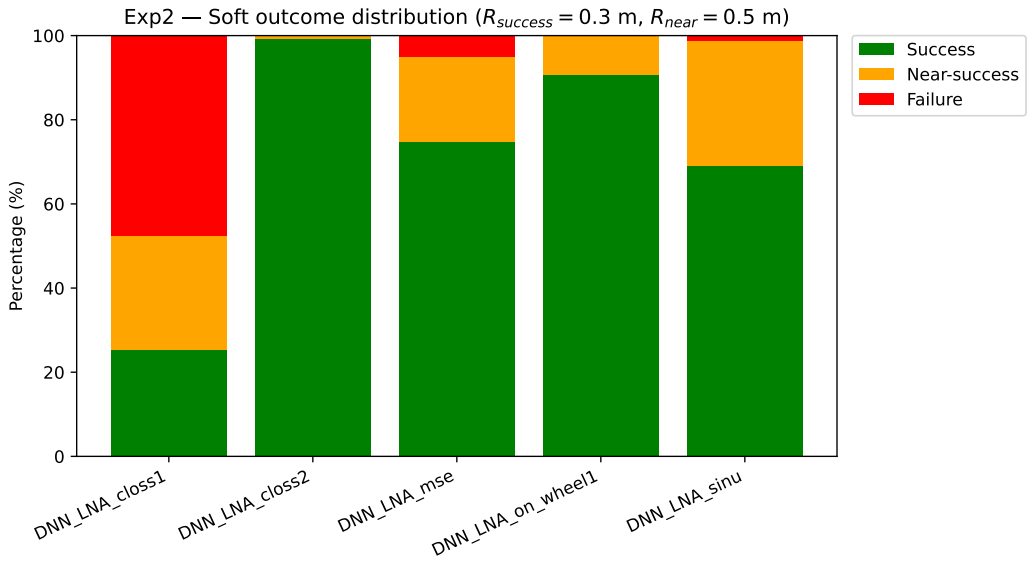

Figure 4.6: Exp2: Soft outcome distribution across models. Bars show the percentage of Success, Near-success, and Failure trajectories aggregated over all evaluated goals ($d_{\text{success}} = 0.30$ m, $d_{\text{near}} = 0.50$ m).

Figure 4.6 shows that the soft criterion reveals important distinctions invisible to binary metrics. For `DNN_LNA_closs2`, nearly all trajectories fall in the Success category with a negligible Near-success fraction, confirming consistent goal-reaching behaviour. For `DNN_LNA_closs1`, the majority of trajectories are classified as Failure, with a small Near-success fraction indicating that some trajectories approach but do not reach the goal. Models such as `DNN_LNA_mse` and `DNN_LNA_sinu` show a visible Near-success band, suggesting that minor input adjustments could improve their performance.

4.3.5 Goal Success Patterns

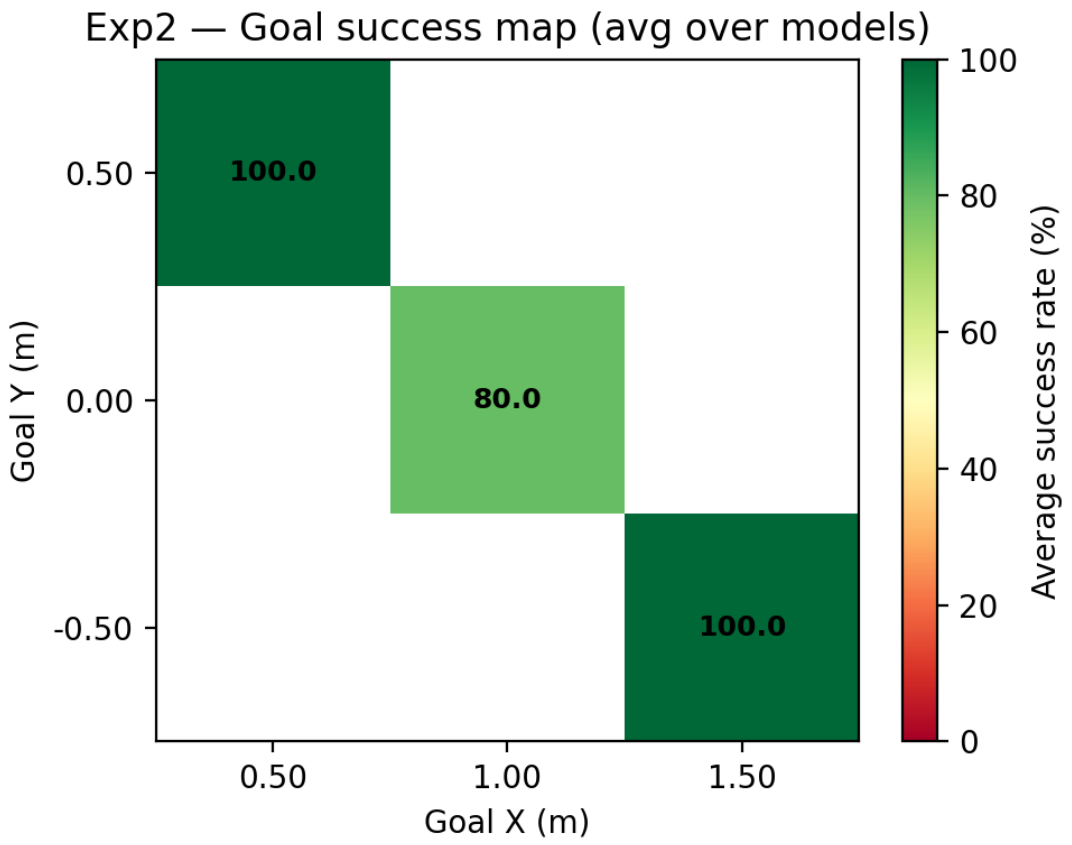

Figure 4.7: Exp2: Goal success map (average strict success rate over all five models). Darker cells indicate higher success rates. The central goal (1.0, 0.0) achieves a lower average success rate than the remaining two targets, suggesting that it is comparatively more challenging under the strict success criterion.

Figure 4.7 indicates that success rates remain high across all evaluated goals, ranging from 80% to 100%. While the central target $(1.0, 0.0)$ achieves the lowest average success rate, the difference remains modest compared with the other two goals. This suggests that goal position has a relatively limited influence on model performance. In comparison, the input-space analysis revealed substantially larger performance variations across different motion commands, indicating that orientation and velocity conditions play a more important role than goal location in determining prediction success.

4.3.6 Summary of Exp2 Findings

Exp2 confirms that goal reachability is strongly goal-dependent and that model differences become more pronounced as goals require larger displacement and lateral offset. The average distance metric (Table 4.2) shows that even in failure cases, `DNN_LNA_closs2` ends much closer to the target than `DNN_LNA_closs1`, indicating qualitatively different failure modes. The soft outcome analysis further separates borderline cases from clear failures, providing a more informative interpretation than a strict binary metric alone.

4.4 Cross-Experiment Comparison and Practical Implications

Combining Exp1 and Exp2 provides complementary views of risk. Exp1 identifies command regions that trigger failures, while Exp2 links these behaviours to goal-reaching performance. The decision tree rules in Section 4.2.4 translate these findings into explicit conditions that can directly inform run-time control decisions.
The consistency between the risk maps, success-rate analysis, and decision tree rules indicates that model failures are not random but arise from identifiable and structured regions of the operational space. This predictability is a key finding: it implies that failure-prone inputs can be detected in advance from the command vector alone, without requiring trajectory execution, opening the door to the following concrete collision-avoidance applications:

- **Dynamic Command Filtering:** Commands falling in high-risk regions — for example $\theta > 2.27$ rad with $v > 1.78$ m/s for `DNN_LNA_closs2`, or outside the narrow success corridor of `DNN_LNA_closs1` — can be proactively scaled or redirected before execution, keeping the wheelchair within a reliable operational envelope.

- **Hybrid Navigation Fallback:** When the current goal lies in a historically high-failure region (Table 4.1), the system can automatically switch from the neural predictor to a conservative model-based planner, ensuring collision-avoidance guarantees are maintained.

- **Predictive Failure Monitoring:** The distinction between soft and strict success, combined with the average distance metric (Table 4.2), enables real-time monitoring of navigation confidence. Consistent drift into the Near-success zone can trigger haptic or visual feedback to prompt a timely transition to manual control.

4.5 Chapter Summary

This chapter presented experimental results for two complementary analyses: input-space sensitivity (Exp1) and goal-based difficulty (Exp2), supplemented by decision tree analysis that translates empirical risk regions into explicit if-then rules. The findings show that failure is concentrated in specific command regions, that goal configurations significantly affect failure severity, and that the choice of loss function is the dominant factor determining model robustness across all experiments.

Chapter 5

Conclusion and Future Work

5.1 Discussion

5.1.1 Interpretation of Failure Regions

The experimental results consistently identify initial orientation θ near $\pm\pi$ as the primary failure trigger across all five models. This finding has a clear physical interpretation: when the wheelchair faces directly away from the goal, the required turning angle is maximal, and the model must produce a sharp reversal manoeuvre that is likely underrepresented in the training distribution. The concentration of failures in this region — visible in the risk maps, the decision tree splits, and the representative trajectory plots — suggests that all five DNN-LNA models share a common distributional bias towards forward-facing configurations. This is not a coincidence but a natural consequence of training on demonstrations of nominal navigation, where extreme backward orientations are rare.

Beyond orientation, the experiments reveal model-specific failure mechanisms. For `DNN_LNA_closs1`, the root split on `GoalX` indicates a systematic bias: the model fails for near-lateral goals regardless of the initial command, suggesting that its training did not adequately cover the lateral displacement regime. For `DNN_LNA_mse`, the root split on ω indicates sensitivity to high rotational commands, pointing to a different distributional gap. For `DNN_LNA_sinu`, the root split on v (specifically, negative velocity) reveals that backward motion is almost universally associated with failure. These model-specific patterns would not be visible from aggregate performance metrics alone — they emerge only through the combination of risk maps and decision tree analysis.

5.1.2 Impact of Training Loss Functions

One of the most striking findings of this thesis is the large performance gap between models trained with different loss functions. `DNN_LNA_closs2` achieves 99.3% strict success and a 5-node decision tree, while `DNN_LNA_closs1` achieves only 25.3% with a 27-node tree — a 74-percentage-point difference between models that share the same architecture and training data. This gap cannot be attributed to differences in input coverage or evaluation methodology, since all five models were evaluated under identical conditions.

The implication is that the loss function is the dominant factor determining model robustness. A loss function that penalises goal-reaching errors more directly (as in `DNN_LNA_closs2`) produces a model that generalises across a much wider range of input conditions than one trained with a mean squared error criterion on trajectory poses. This finding has practical consequences for model selection in safety-critical applications: aggregate evaluation metrics such as average trajectory error are insufficient to predict operational robustness, and trajectory-level evaluation under a diverse input sample is necessary to reveal the true performance envelope of each model.

5.1.3 Practical Implications for Smart Wheelchair Navigation

The risk maps and decision tree rules produced in this thesis are directly applicable to run-time collision-avoidance in smart wheelchair systems. The key insight is that failure-prone inputs can be detected from the command vector alone, before trajectory execution, using the extracted rules as a pre-execution filter. For `DNN_LNA_closs2`, the filter is simple: flag commands with $\theta > 2.27$ rad and $v > 1.78$ m/s as potentially risky. For `DNN_LNA_closs1`, the filter is more restrictive: reject commands outside the narrow success corridor defined by the four extracted rules.

Such a filter could be integrated into the wheelchair's shared autonomy framework as a lightweight safety layer that operates in parallel with the neural predictor. When a command is flagged as high-risk, the system can either prompt the user to adjust their input, automatically re-orient the wheelchair to a safer initial orientation, or switch to a classical model-based planner for that manoeuvre. This hybrid architecture would preserve the responsiveness of neural prediction for the majority of inputs while guaranteeing safe behaviour in the identified failure regions.

5.1.4 Limitations of the Study

Several limitations of this work should be explicitly acknowledged.

Black-box evaluation. The models were evaluated as fixed black-box predictors. While this reflects realistic deployment conditions where model weights are proprietary, the root causes of failure within the neural architectures remain partially opaque. Internal representations, weight distributions, and layer-level activation patterns were not analysed, limiting the causal interpretation of the observed failure patterns.

Offline setting. The analysis was conducted entirely offline using uniformly sampled inputs. Dynamic environmental factors — moving obstacles, varying floor friction, sensor noise, or real user command distributions — were not accounted for. The sampled inputs may therefore not fully represent the distribution of commands that would arise in real deployment.

Limited model and goal coverage. The evaluation used five pretrained models, $N = 200$ sampled inputs per goal configuration, and $G = 3$ goal positions. While sufficient for identifying systematic trends, a larger and more diverse sample — including real user command distributions and a wider range of goal configurations — would strengthen the generalisability of the findings.

Fixed success threshold. The strict success criterion ($d < 0.30$ m) was fixed throughout all experiments. In practice, the acceptable tolerance may vary depending on the environment and task, and results could differ under alternative threshold definitions.

Decision tree as surrogate. The decision trees are shallow surrogates of the true failure boundary in the input space. With max_depth = 4, they cannot capture all interaction effects between input variables and may misclassify a small fraction of inputs near the true boundary. The reported training accuracies (85%–99%) confirm that the trees are good approximations but not exact characterisations of model behaviour.

Threats to validity. The main threat to internal validity is that the decision tree rules are derived from the same dataset used for evaluation, which may lead to overfitting of the interpretability layer. However, the consistency between the tree rules and the independently computed risk maps and success rate tables provides strong evidence that the identified patterns are genuine properties of the models rather than artefacts of the fitting procedure.

5.2 Conclusion

This thesis investigated the operational reliability of neural trajectory prediction models for smart wheelchair navigation within the REXASI-PRO project framework. By developing a systematic trajectory-level evaluation pipeline, we mapped

the relationship between input command spaces and model performance, treating all five DNN-LNA models as black-box predictors without requiring access to their internal weights or training procedures.
The experimental results from Exp1 and Exp2 provided a clear characterisation of failure zones in the (θ, v, ω) command space. The input-space sensitivity analysis (Exp1) revealed that failures are not uniformly distributed but concentrated in identifiable regions, as formalised by the empirical failure probability measure $\mathcal{R}(\mathcal{U}_{\text{bin}})$ introduced in Equation 4.1. The goal-based analysis (Exp2) further demonstrated that while some models (e.g., `DNN_LNA_closs2`) exhibit near-perfect goal-reaching across all tested configurations, others are highly sensitive to target positions requiring sharp manoeuvres, with success rates as low as 25.3% for `DNN_LNA_closs1`.
A key contribution of this work is the integration of decision tree analysis as an interpretability layer on top of the quantitative risk maps. By fitting a compact decision tree classifier for each model, explicit and human-readable rules were extracted that identify the precise input conditions under which each model is likely to fail. These rules are fully consistent with the quantitative performance metrics: models with complex, failure-dominated trees also perform worst in the goal-reaching evaluation, while models with minimal trees achieve near-perfect performance. The analysis further revealed that the choice of loss function during training is the dominant factor determining model robustness, with a 74-percentage-point success-rate gap between the best and worst models.
The proposed risk maps, goal Success patterns, and decision tree rules together serve as a foundation for implementing run-time collision-avoidance monitors and hybrid control strategies, ensuring that the wheelchair remains within a reliable operational envelope.

5.3 Future Work

The findings of this thesis open several avenues for future research:

- **Real-time Collision-Avoidance Filtering.** Integrating the risk maps and decision tree rules into a live control loop to proactively filter user commands that fall into high-risk regions, preventing the wheelchair from entering unsafe operational states.
- **Hybrid Navigation Fallback.** Developing a switching mechanism that automatically reverts to a classical, model-based planner when the neural predictor signals low confidence or the current command falls within a known failure region identified by the decision tree rules.

- **Advanced Explainability (XAI).** Extending the decision tree analysis with more powerful feature attribution methods such as SHAP or LIME, which can provide finer-grained explanations of why specific input combinations trigger failed trajectories, including interaction effects between features that a single decision tree cannot capture.
- **On-platform Validation.** Transitioning from offline simulation to physical testing on the smart wheelchair platform to validate the predictive power of the identified risk zones and confirm that the extracted decision rules generalise to real sensor and actuator conditions.
- **Adaptive Threshold Selection.** Investigating whether the success threshold ($d < 0.30$ m) can be adapted dynamically based on the environment or user profile, and studying how the risk maps and decision tree boundaries shift under different threshold definitions.

In conclusion, this research demonstrates that while deep learning offers powerful predictive capabilities for smart wheelchair navigation, systematic evaluation is essential to ensure user safety and reliable collision-avoidance behaviour. The interpretability tools developed in this thesis — risk maps, goal success patterns, and decision tree rules — provide a practical and model-agnostic framework that can be applied to other neural navigation systems beyond the specific models evaluated here.

Appendix A

Extracted Features and Dataset Details

This appendix provides the technical specification of the features used for trajectory analysis and the numerical ranges observed in the experimental datasets, as referenced in Chapters 3 and 4. These parameters represent the operational envelope defined within the *REXASI-PRO* project framework [17].

A.1 Feature Characterization

To assess the reliability of the `DNN_LNA` models, a set of key navigation features was extracted. These features, summarised in Table A.1, represent the state-space and command-space used for risk mapping and goal-difficulty analysis.

A.2 Trajectory Prediction Format

The trajectory prediction models generate a temporal sequence of $T = 30$ steps with time step $\Delta t = 0.1$ s (total horizon 3 s). Each pose is defined in the $SE(2)$ planar space as $\hat{\mathbf{x}}_t = (x_t, y_t, \theta_t)$, together with predicted linear and angular speeds $(\hat{v}_t, \hat{\omega}_t)$. The following numerical sample illustrates the output format for a stable trajectory:

- **Initial State ($t = 1$):** (x: 0.000, y: 0.000, θ: 0.000)
- **Mid-horizon ($t = 15$):** (x: 1.052, y: 0.221, θ: 0.064)
- **Terminal State ($t = 30$):** (x: 2.104, y: 0.450, θ: 0.122)

Feature	Description	Operational Range
θ (Orientation)	Initial absolute orientation of the wheelchair in the robot-centric reference frame.	$[-\pi, \pi]$ rad
v (Linear)	Commanded linear velocity (includes reverse and forward motion).	$[-1.05, 2.88]$ m/s
ω (Angular)	Commanded angular velocity (rotational rate).	$[-1.99, 1.99]$ rad/s
d_{goal}	Initial Euclidean distance from the starting pose to the target.	$[0, 5.0]$ m
d_{final}	Residual Euclidean distance to goal at the final prediction step $T = 30$.	≥ 0 m

Table A.1: Summary of extracted features and their operational ranges as defined by the controller specifications provided by the project partners.

A.3 Reference Goal Configurations (Exp2)

For the goal-based difficulty analysis in Experiment 2, three specific target positions were selected in the local coordinate frame to represent different manoeuvre complexities:

1. **Goal A — Near lateral:** $(0.5,\ 0.5)$ — Short-range diagonal navigation requiring moderate turning (low-to-medium complexity).
2. **Goal B — Straight ahead:** $(1.0,\ 0.0)$ — Long-range forward navigation along the heading axis (medium complexity).
3. **Goal C — Far off-axis:** $(1.5,\ -0.5)$ — Long-range navigation with lateral offset, requiring combined forward and rotational commands (high complexity).

The strict success rates and average distances across these three goals are reported in Tables 4.1 and 4.2 of Chapter 4. The results confirm that Goal C consistently produces lower success rates across most models, with the performance gap between models becoming more pronounced at longer distances and larger lateral offsets.

A.4 Decision Trees for Remaining Models

The following figures report the decision trees for the three remaining `DNN_LNA` models not shown in Chapter 4. The trees for `DNN_LNA_closs1` and `DNN_LNA_closs2` are presented and discussed in Figures 4.5 and 4.4 of Chapter 4. All trees share the same configuration: input features (θ, v, ω, GoalX, GoalY), maximum depth 4, minimum samples per leaf 10, strict success threshold $d < 0.30$ m, and $N = 600$ labelled observations per model (200 inputs $\times$ 3 goals). **Blue nodes indicate regions predicted as Success; orange nodes indicate regions predicted as Failure.**

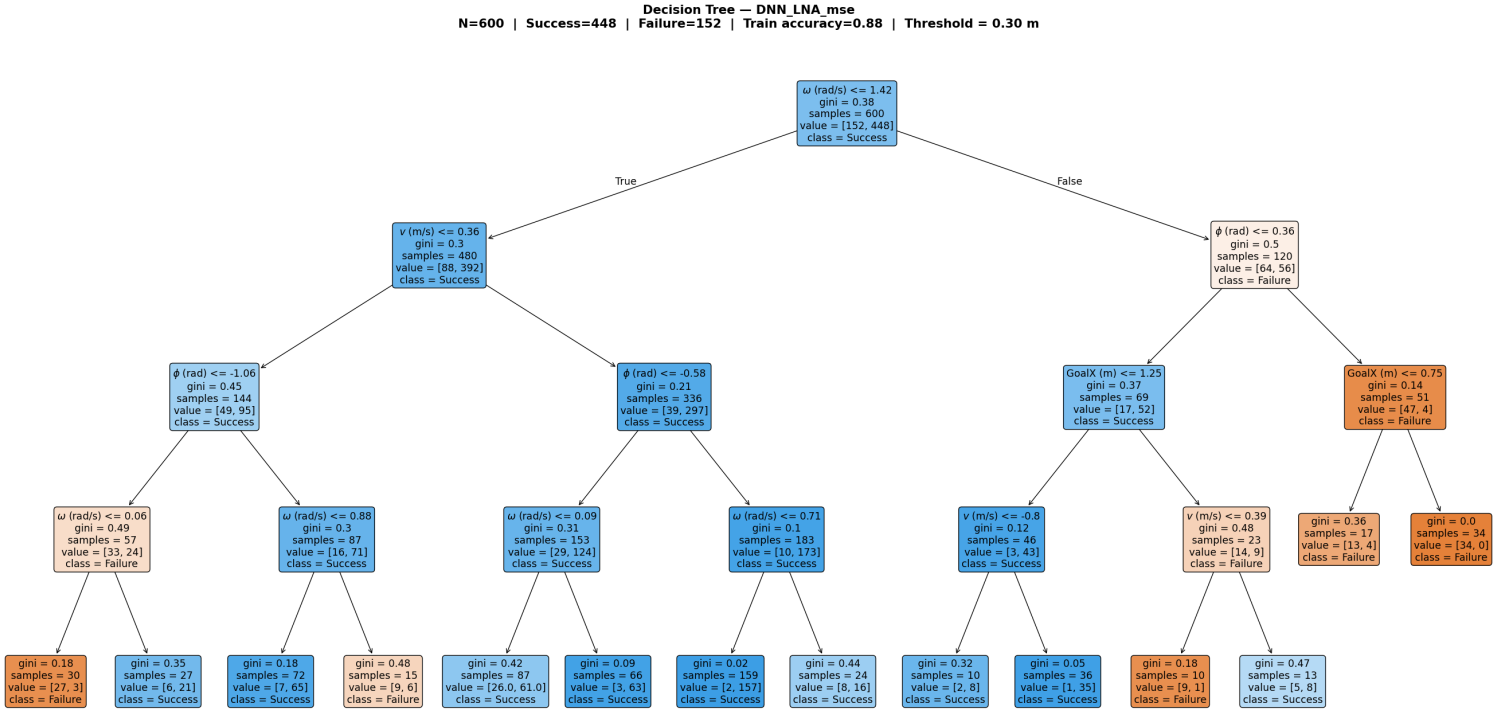

Figure A.1: Decision tree for `DNN_LNA_mse` ($N = 600$, train accuracy $= 0.88$, overall success rate $= 74.7\%$). The tree contains 27 nodes (13 internal, 14 leaves). The root splits on ω (angular velocity), indicating that high rotational speed is the primary trigger of failure for this model.

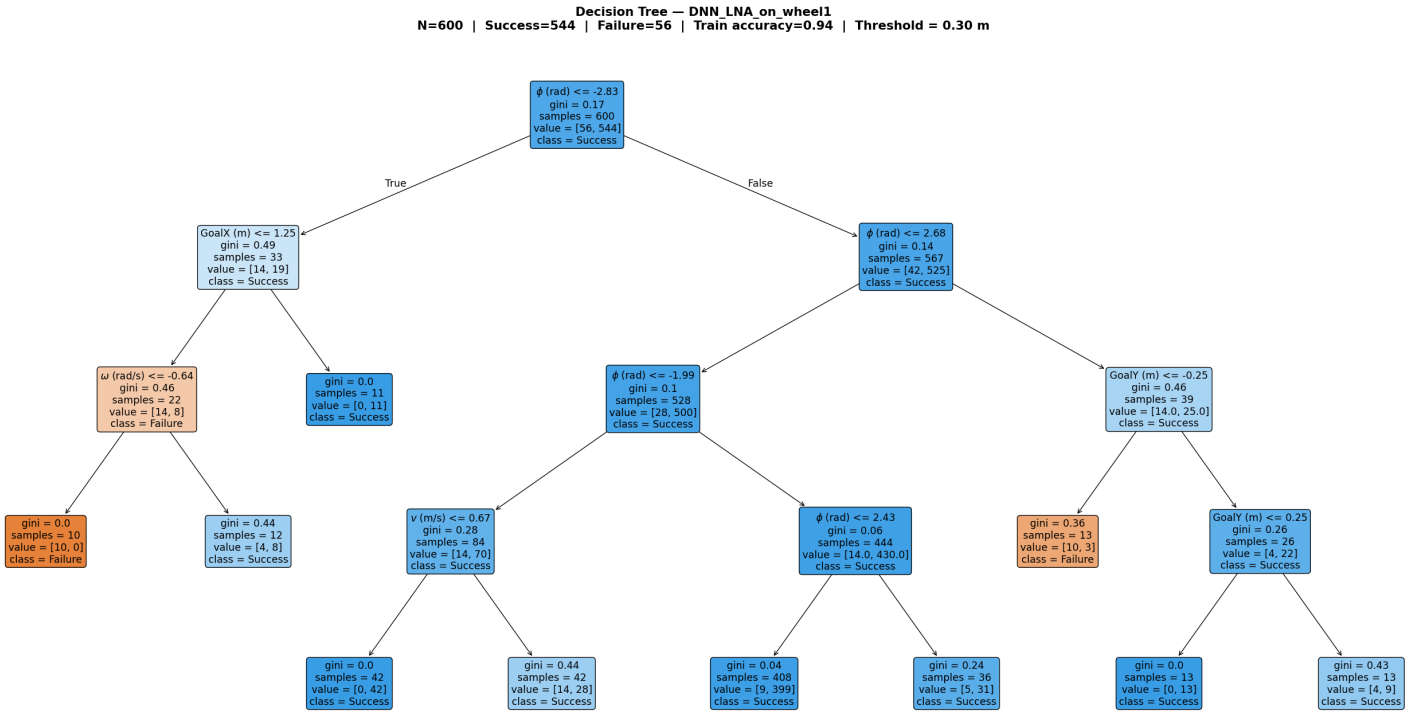

Figure A.2: Decision tree for `DNN_LNA_on_wheel1` ($N = 600$, train accuracy = 0.94, overall success rate = 90.7%). The tree contains 19 nodes (9 internal, 10 leaves). Failure leaf nodes are concentrated at extreme orientations near $\pm\pi$ rad, confirming the high overall reliability of this model across most input conditions.

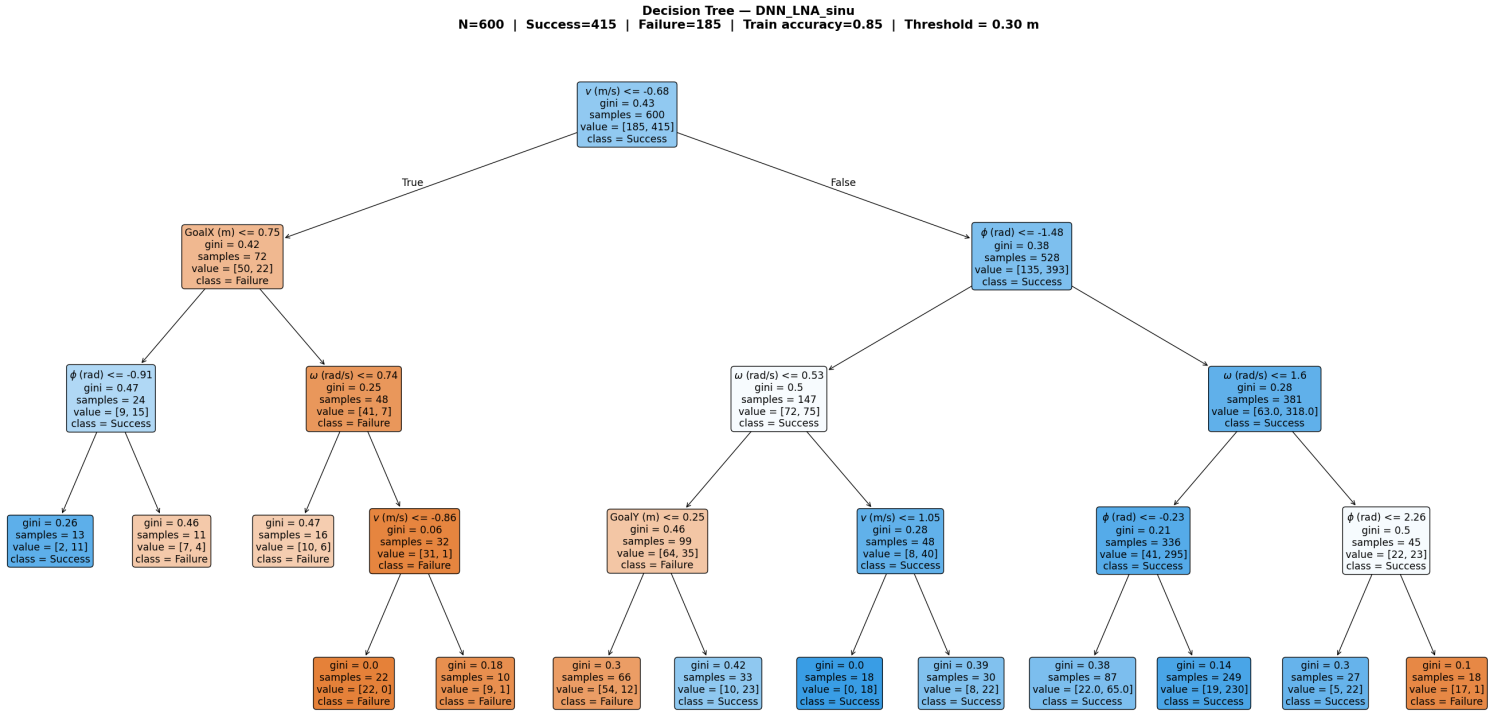

Figure A.3: Decision tree for `DNN_LNA_sinu` ($N = 600$, train accuracy = 0.85, overall success rate = 69.2%). The tree contains 25 nodes (12 internal, 13 leaves). The root splits on v (linear velocity), showing that backwards motion (negative velocity) is the primary failure trigger for this model.

Bibliography

[1] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016. URL https://www.jmlr.org/papers/volume17/15-522/15-522.pdf. 1, 5, 6

[2] Arun Salvador et al. Shared control for intelligent mobility assistance in wheelchairs. *Robotics and Autonomous Systems*, 2018. URL https://www.argallab.northwestern.edu/files/2016/05/13icra_ws_argall.pdf. Representative related work on shared control in assistive robotics. 1, 5

[3] S. Udupa, V. Kamat, and C. Menassa. Shared autonomy in assistive mobile robots: a review. *Disability and Rehabilitation: Assistive Technology*, 2021. doi: 10.1080/17483107.2021.1928778. 1, 5

[4] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1):33–55, 2016. doi: 10.1109/TIV.2016.2578706. 2, 5, 6

[5] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:187–210, 2018. doi: 10.1146/annurev-control-060117-105157. 2, 5, 6

[6] L. Liu et al. Path planning techniques for mobile robots: Review and future research directions. *Expert Systems with Applications*, 2023. doi: 10.1016/j.eswa.2023.121821. 2, 6

[7] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016. URL https://arxiv.org/abs/1604.07316. 3, 5, 6

[8] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. *arXiv preprint arXiv:1710.02410*, 2017. URL https://arxiv.org/abs/1710.02410. 3, 5, 6

[9] Markus Kuderer et al. Learning driving styles for autonomous vehicles. *ICRA*, 2015. 3, 5

[10] Andrey Rudenko et al. Human motion trajectory prediction: A survey. *IJRR*, 2020. 3, 5

[11] Alexandre Alahi et al. Social lstm: Human trajectory prediction in crowded spaces. *CVPR*, 2016. 5

[12] Agrim Gupta et al. Social gan: Socially acceptable trajectories with gans. *CVPR*, 2018. 5

[13] A. Erdogan and B. Argall. The effect of robotic wheelchair control paradigm and interface on user performance, perception and preference: An experimental assessment. *Robotics and Autonomous Systems*, 94:282–297, 2017. doi: 10.1016/j.robot.2017.05.006. 5

[14] N. D. Muñoz et al. Evaluation of navigation of an autonomous mobile robot. In *Proceedings of the 5th International Conference on Information Technology and Applications in Biomedicine*, 2007. 6

[15] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html. 7

[16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016. doi: 10.1145/2939672.2939778. 7

[17] Zeming Duan, Serge Autexier, Jan Janssen, and Christian Mandel. Tensorflow model description for dnn-lna. REXASIPRO Project Deliverable D3.2, DFKI – German Research Center for Artificial Intelligence, 2024. EU Horizon Europe Project REXASIPRO, Grant Agreement No. 101070028-2. 10, 12, 16, 37