AUTOMATED PARKING SYSTEMS USING REINFORCEMENT LEARNING-ASSISTED MODEL PREDICTIVE CONTROL

by

HUSSEIN ALI ALAWSI

A dissertation submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN ELECTRICAL AND COMPUTER ENGINEERING

2025

Oakland University
Rochester, Michigan

Doctoral Advisory Committee:

Jun Chen, Ph.D., Chair
KaC Cheok, Ph.D.
Micho Radovnikovich, Ph.D.
Darrell Schmidt, Ph.D.

To my beloved family—my mother, father, wife, siblings, and children. Without your endless love, sacrifices, and support, none of this would have been possible. This work is dedicated to you.

ACKNOWLEDGMENTS

I am sincerely indebted to Jun Chen, Ph.D., my advisor, whose guidance and encouragement have been central to this work. His insightful advice, high expectations, and constant support pushed me to expand my thinking and pursue excellence. Beyond shaping this dissertation, his mentorship has profoundly influenced my development as a researcher and strengthened my dedication to conducting meaningful research.

I would also like to extend my gratitude to my committee members—KaC Cheok, Ph.D., Micho Radovnikovich, Ph.D., and Darrell Schmidt, Ph.D.—for their valuable feedback and contributions, which greatly improved the quality of this dissertation.

My appreciation also goes to my colleagues Ali Irshayyid, Zhaodong Zhou, Wanqun Yang, and Luke Nucalaj for their collaboration, support, and encouragement throughout this journey.

Lastly, I extend my heartfelt thanks to all the friends who stood by me during this time. Your support has not only brightened this chapter of my life but will continue to inspire me in the years ahead.

Hussein Ali Alawsi

ABSTRACT

AUTOMATED PARKING SYSTEMS USING REINFORCEMENT LEARNING-ASSISTED MODEL PREDICTIVE CONTROL

by

Hussein Ali Alawsi

Adviser: Jun Chen, Ph.D.

Autonomous parking remains a challenging task due to the need for accurate trajectory tracking, smooth steering, and stable heading control under diverse maneuvering conditions. Conventional model predictive control (MPC) can handle system constraints effectively, but its performance depends heavily on manually tuned cost weights. This dissertation proposes a reinforcement learning-assisted model predictive control (RL-assisted MPC) framework to improve autonomous vehicle parking performance. A Deep Q-Network (DQN) agent is trained to dynamically select the cost function weights of an MPC controller, enabling real-time adaptation based on the vehicle's current state. The hybrid approach leverages the predictive optimization capability of MPC together with the adaptive decision-making of RL, enabling the controller to adjust trade-offs in real time without manual re-tuning. The framework is evaluated across five different parking scenarios and compared against static-weight MPC baselines. Experimental evaluations demonstrate that the proposed RL-assisted MPC framework achieves comparable or better lateral tracking accuracy, while consistently providing smoother steering behavior and improved heading stability compared to baseline controllers using static MPC weights. The results demonstrate that RL-assisted MPC improves robustness and generalization in automated parking systems, highlighting the potential of combining model-based predictive control with RL for autonomous driving.

TABLE OF CONTENTS

TABLE OF CONTENTS—Continued

LIST OF TABLES

LIST OF TABLES—Continued

LIST OF FIGURES

LIST OF FIGURES—Continued

LIST OF ABBREVIATIONS

ADAS	Advanced Driver Assistance Systems
APS	Autonomous Parking Systems
AV	Autonomous Vehicle
DQN	Deep Q-Network
MDP	Markov Decision Process
MIMO	Multi-Input Multi-Output
MPC	Model Predictive Control
NLP	Nonlinear Program
PID	Proportional-Integral-Derivative
QP	Quadratic Program
RL	Reinforcement Learning
SQP	Sequential Quadratic Programming
TD	Temporal Difference

CHAPTER ONE

INTRODUCTION

The continuous annual increase in vehicle populations has substantially escalated parking space requirements, rendering parking operations increasingly time-intensive and resource-demanding, particularly within densely populated metropolitan environments [1]. Furthermore, this vehicular expansion concurrently elevates accident risk, contributing to significant economic losses and infrastructure damage [2]. To mitigate these challenges, Advanced Driver Assistance Systems (ADAS) have been engineered with diverse functionalities, encompassing Autonomous Parking Systems (APS), specifically designed to improve vehicular safety and operational convenience [3].

APS architecture typically consists of three fundamental modules: parking spot detection, path planning, and path following [3]. Path planning and path following constitute two essential elements in the field of Autonomous Vehicle (AV) control [4]. Path planning synthesizes feasible vehicle paths that comply with safety constraints, while path tracking controllers utilize real-time state feedback to compute optimal control actions for precise trajectory following. Considerable research efforts have focused on developing efficient trajectory planning methodologies [5]. Nevertheless, path tracking presents significant challenges due to the complex vehicle dynamics and constraints imposed by limited onboard computational and communication capabilities. Path tracking controllers must synthesize precise control signals in real-time while accommodating the imposed computational and communication resource limitations. Multiple control methodologies have been implemented for path tracking applications, including Proportional-Integral-Derivative (PID) controllers, state feedback controllers, and Model Predictive Control (MPC) [6]. MPC has received considerable attention and extensive

investigation within AVs research [7–11] and mechatronics [12–14]. MPC is well-suited for real-world AV path-following tasks, as it can effectively manage Multi-Input Multi-Output (MIMO) systems while accommodating a wide range of system constraints [15].

While MPC offers a systematic approach for constrains handling and predictive behavior modeling, its efficacy is critically influenced by the appropriate tuning of cost function weighting parameters [16]. These weighting parameters establish the relative priority among competing control objectives, such as minimizing lateral error, steering effort, and steering rate. Identifying a single static weighting configuration that achieves optimal performance across all scenarios presents significant difficulties, given that varying operational conditions necessitate distinct control trade-offs. This limitation necessitates the development of adaptive control strategies that enable real-time adjustment of weighting parameters based on the instantaneous AV state.

In recent years, RL has achieved remarkable success in complex decision-making tasks [17–23]. Unlike the offline design process of MPC, RL optimizes the control policy through data-driven adaptation based on the interaction with the environment [24, 25]. In RL, the agent learn optimal policy exclusively through observed state-action transitions and reward signal, eliminating the requirement for a prior system dynamics knowledge [26]. The reward function represents an estimate of the expected cumulative reward associated with each state. When rewards converges to its optimal form, it encapsulates the global optimal information, enabling the derivation of the infinite-horizon optimal control policy. When considering the task of APS, it is challenging to mathematically define the reward function for an autonomous system [27, 28]. Without a properly defined reward function, it becomes difficult for RL to execute parking maneuvers both safely and efficiently [20, 29].

Additionally, RL training can be extremely time-consuming, often necessitating extensive computational cycles over multiple days to achieve policy convergence. However, in pursuit of globally optimal policies, an RL agent inherently employs exploratory strategies and trial-and-error learning mechanisms, thereby complicating the establishment of formal safety guarantees for the emergent control behaviors [30]. This issue becomes particularly critical in safety-sensitive applications, where safety is often characterized by the system's stability. Recent research efforts have focused on developing safe RL frameworks that aim to ensure stability and constraint satisfaction during both learning and deployment phases [30, 31].

Given the powerful data-driven optimization capabilities of RL, combining RL with MPC introduces the ability to incorporate system constraints explicitly, enabling safer and more reliable control while still leveraging real-time data. Simultaneously, by building upon the solid theoretical foundations of MPC, the resulting behavior of the RL-assisted controller becomes easier to analyze and verify, with respect to stability and constraint satisfaction. These observations motivate the integration of RL and MPC, although only limited pioneering work has been reported to date [32]. Several recent studies have explored different ways to integrate RL and MPC. For example, [33] introduced a "plan online and learn offline" approach, using MPC as a trajectory optimizer to improve exploration and accelerate RL convergence. This was extended in [34] to an actor-critic framework capable of handling sparse and binary rewards, highlighting MPC's role in propagating global information. Real-world applications, such as [35], demonstrated the benefits of combining high-level objectives with improved value function representation. However, these efforts primarily focus on enhancing RL performance, with limited attention to system stability. The stability aspect was addressed in [36] and further developed in [37] using economic MPC as a function approximator. Building on this, [38] incorporated robust linear MPC to ensure constraint satisfaction.

While promising, these approaches still require further validation, and few offer both theoretical safety and practical efficiency.

This paper proposes an RL-assisted MPC framework for APS. This hybrid approach combines the adaptability of RL with the predictive optimization capabilities of MPC. Rather than training a full control policy from scratch, the RL agent selects, at each time step, one of three predefined weight configurations based on the AV current state, enabling the controller to adapt to varying conditions and enhance path-following performance. By leveraging the strengths of both frameworks, the real-time optimization ability of MPC and the adaptive decision-making capability of RL, the resulting approach improves path-following behavior for APS without requiring extensive retraining or manual weight tuning, and avoids reliance on a single static set of weights.

CHAPTER TWO

PRELIMINARIES AND PROBLEM FORMULATION

2.1 Preliminaries on Reinforcement Learning

This section briefly reviews the fundamentals of Reinforcement Learning (RL) with particular emphasis on value-based methods and Deep Q-Network (DQN), which form the foundation of our RL approach. For more comprehensive details, please refer to [39].

At its core, RL algorithms are designed to solve problems that can be modeled as an Markov Decision Process (MDP), a mathematical framework that describes how agents interact with an environment to select the most suitable actions based on observations for maximizing a long-term return [40]. In an MDP, the environment is represented by a tuple $(\mathbf{S}, \mathbf{A}, p, r, \gamma)$, where $\mathbf{S}$ is the state space, $\mathbf{A}$ is the action space, p is the transition probability matrix, r is the immediate reward, and γ is the discount factor [39]. The agent interacts with the environment over multiple time steps, aiming to learn a policy, π, that maximizes the cumulative discounted rewards, i.e.,

$$\max_{\pi} J(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]. \tag{2.1}$$

RL algorithms generally fall into two main categories: value-based methods and policy-based methods. Value-based methods focus on learning the value function, which estimates the expected return from a given state or state-action pair, and deriving a policy from this function. The key insight is that if we can accurately estimate the value of each action in each state, we can construct an optimal policy by simply choosing the action with the highest value. A fundamental value-based algorithm is Q-learning [41], which learns the action-value function $Q(s,a)$ representing the expected cumulative reward when taking action a in state s and following the optimal policy thereafter. The Q-learning

update rule is:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right] \tag{2.2}$$

where α is the learning rate. Q-learning is an off-policy algorithm, meaning it can learn the optimal policy while following a different behavior policy for exploration.

Traditional Q-learning maintains a tabular representation of Q-values, which becomes intractable for large or continuous state spaces. Deep Q-Networks (DQN) address this limitation by using deep neural networks to approximate the Q-function [42]. The DQN algorithm combines Q-learning with deep neural networks and incorporates several key innovations. First, instead of learning from experiences in the order they occur, DQN stores transitions (s_t, a_t, r_t, s_{t+1}) in a replay buffer and samples mini-batches randomly for training. This experience replay mechanism breaks the correlation between consecutive samples and improves data efficiency. Second, DQN maintains two networks: the main Q-network $Q(s, a; \theta)$ and a target network $Q(s, a; \theta^-)$ with parameters θ^- that are periodically copied from the main network. The target network provides stable target values for the Q-learning update:

$$L(\theta) = \mathbb{E}_{(s,a,r,s') \sim \mathscr{D}} \left[\left(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta) \right)^2 \right] \tag{2.3}$$

where $\mathscr{D}$ is the replay buffer. Third, DQN uses ε-greedy exploration, where the agent selects a random action with probability ε and the greedy action $\arg\max_a Q(s, a; \theta)$ with probability $1 - \varepsilon$.

DQN is inherently off-policy, which means it can learn about the optimal policy while following a different policy for data collection. The off-policy nature allows DQN to learn from experiences generated by any policy (sample efficient), making it particularly suitable for scenarios where online exploration is expensive, dangerous, or impractical—such as autonomous vehicle control.

Policy-based methods, on the other hand, directly learn the policy without explicitly computing the value function [39]. One effective approach that combines the strengths of both value-based and policy-based methods is the actor-critic methods [43]. This approach combines two key components: an actor, which learns the policy, and a critic, which evaluates the policy's performance. The actor-critic method aims to simultaneously improve the policy and its value estimation, leading to more efficient learning.

Actor-critic algorithms can be implemented using various approaches. One prominent method is the policy gradient technique, which directly optimizes the policy by estimating the gradient of expected cumulative rewards with respect to the policy parameters [44]. In policy gradient methods, the actor objective function, J, can be defined as:

$$J(\pi_\theta) = E_{\pi_\theta}\left[\sum_t \log \pi_\theta(a_t|s_t) \cdot A_t\right], \tag{2.4}$$

where $\pi_\theta(a_t|s_t)$ represents the probability of taking action a_t in state s_t under the policy parameterized by θ. The objective is to change θ to increase the probability of choosing actions that lead to higher advantages. The advantage function, A_t, is defined as:

$$A_t = Q_\pi(s_t, a_t) - V_\pi(s_t), \tag{2.5}$$

where $V_\pi(s_t)$ is the expected return from s_t and acting according to policy π and $Q_\pi(s_t, a_t)$ is the expected return starting from s_t and taking action a_t (also termed as the Q-value).

The advantage function (2.5) quantifies the benefit of choosing action a_t at state s_t compared to the expected return of following the policy π from state s_t. The advantage A_t can be approximated using the Temporal Difference (TD) error:

$$A_t = r_t + \gamma V_\pi(s_{t+1}) - V_\pi(s_t), \tag{2.6}$$

where r_t is the immediate reward received. The critic network, parameterized by weights ϕ, is used to estimate $V_\pi(s_t)$, where ϕ is updated to minimize the following loss function:

$$L(\phi) = E\left[\left(r_t + \gamma V_\pi(s_{t+1}) - V_\phi(s_t)\right)^2\right], \tag{2.7}$$

Both the actor parameter θ and critic parameter ϕ are updated after collecting the experiences or trajectories, training episodes, from the environment. The foundation provided by these RL methods, particularly DQN with its off-policy learning capability and experience replay mechanism, naturally aligns with the offline RL setting where the agent must learn from a fixed buffer of experiences without additional environment interaction.

2.2 Preliminaries on Optimal Control

Optimal control theory provides a systematic framework for determining control inputs that optimize a performance criterion while respecting system dynamics and constraints. Consider a discrete-time dynamical system of the form

$$x_{k+1} = f(x_k, u_k), \quad x_k \in \mathbb{R}^n, u_k \in \mathbb{R}^m, \tag{2.8}$$

where x_k denotes the system state at time k, u_k is the control input, and $f(\cdot)$ represents the system dynamics. The finite-horizon optimal control problem aims to compute a control sequence $\{u_0, u_1, \ldots, u_{N-1}\}$ that minimizes the cost

$$J = \sum_{i=0}^{N-1} \ell(x_{k+i|k}, u_{k+i}) + V_f(x_{k+N|k}), \tag{2.9}$$

subject to the dynamics, initial condition x_0 given, and state and input constraints $x_{k+i|k} \in \mathscr{X}$, $u_{k+i} \in \mathscr{U}$. Here $\ell(\cdot,\cdot)$ is the stage cost, $V_f(\cdot)$ the terminal cost, and N the prediction horizon.

Model Predictive Control (MPC) is a practical realization of optimal control that solves such a problem in a receding horizon fashion. At each time step, the controller measures the current state, solves the optimization problem over a horizon N, applies only the first control input, and then shifts the horizon forward. This receding horizon strategy provides feedback and allows MPC to effectively handle model uncertainties and external disturbances.

A typical MPC formulation includes several key elements. The prediction model, which may be linear (e.g., $x_{k+1} = Ax_k + Bu_k$) or nonlinear, is used to forecast the future evolution of the system. The cost function is often quadratic, expressed as $\ell(x_k, u_k) = x_k^\top Q x_k + u_k^\top R u_k$, where $Q \succeq 0$ and $R \succ 0$ serve to balance state regulation against control effort. Constraints on both inputs and states are explicitly incorporated, which represents one of the major strengths of MPC in practical applications. Finally, terminal ingredients, including a terminal cost $V_f(x)$ and a terminal set $\mathscr{X}_f$, are typically introduced to guarantee closed-loop stability.

For linear systems with quadratic costs, MPC reduces to a Quadratic Program (QP), while for nonlinear systems the problem becomes a Nonlinear Program (NLP), requiring iterative solvers such as Sequential Quadratic Programming (SQP). The computational complexity depends on the number of states n, inputs m, and horizon length N, typically scaling as $\mathscr{O}(N^3)$ for QPs.

MPC design requires selecting appropriate parameters: the horizon length N (longer horizons improve performance but increase computational load), the sampling time T_s (linked to system dynamics and solver speed), and weight matrices Q and R (which balance tracking accuracy against control effort). The framework's flexibility and ability to respect constraints make MPC an attractive choice in many engineering applications, and it also provides a natural foundation for integration with RL approaches, where data-driven methods can enhance prediction models, cost design, and constraint handling.

2.3 AV Parking Problem Formulation

The AV parking problem is formulated as a constrained trajectory tracking task in which the vehicle must follow a predefined reference path from an initial state to a designated parking spot. The objectives are to minimize lateral deviation, ensure smooth steering, and maintain heading stability, while satisfying the physical and safety constraints of the vehicle. The reference trajectory is generated using smooth parametric curves (Bézier curves) that guarantee collision-free and curvature-continuous paths suitable for low-speed maneuvers.

To model the vehicle behavior, a kinematic bicycle model is adopted, which provides an accurate approximation in parking scenarios (low speed). The control strategy is based on MPC, where the cost function balances three competing objectives: tracking accuracy, steering effort, and steering smoothness. However, static weight configurations in the cost function are insufficient across diverse scenarios. To address this limitation, an RL agent is introduced to adaptively select the weight configuration in real time according to the vehicle state. The reward is defined using lateral deviation and heading dynamics, ensuring that the agent promotes both accuracy and stability.

Two case studies are investigated. In the first, the vehicle speed is fixed, and the MPC optimizes only the steering input. In the second, the vehicle travels with a dynamic velocity profile; however, the controller still controls only the steering, while the varying speed introduces additional complexity in trajectory tracking. This distinction allows a systematic evaluation of the RL-assisted MPC framework under both constant and variable speed conditions.

CHAPTER THREE

Automated Parking System using Reinforcement Learning-assisted Model Predictive Control: Case Study I

The objective of this study is to enable an AV to follow a predefined path across five parking scenarios (Fig. 3.2–3.6), achieving minimal lateral deviation while ensuring smooth steering and stable heading transitions within safety constraints. The scenarios were designed with increasing complexity to evaluate the robustness of the proposed framework. Scenario 1 (Fig. 3.2), represents the simplest and shortest path. In Scenario 2 (Fig. 3.3), the vehicle's starting orientation is shifted to test the controller's adaptability to different initial heading angles. Scenario 3 (Fig 3.4), introduces additional complexity through an extended path, making it the longest trajectory among the scenarios. Scenarios 4 (Fig. 3.5) and 5 (Fig. 3.6), incorporate real-world inspired conditions: a tilted parking path and a perpendicular parking path, respectively, both of which were derived from observed layouts in a parking lot in downtown Auburn Hills, Michigan, as shown in Figures 3.7 and 3.8. The AV operates in a structured environment modeled using a kinematic bicycle model (Fig. 3.1). The reference trajectory is generated using a Bézier curve. It's important to note that in this case study, the speed of the AV is considered to be fixed throughout the trajectory.

3.1 Methodology

The objective is to allow an AV to follow a predefined path during parking (Fig. 3.2) with minimal lateral deviation, while ensuring smooth steering and stable heading transitions, all within the limits of safety constraints. The AV operates in a structured environment modeled using a kinematic bicycle model (Fig. 3.1). The reference trajectory is generated using a Bézier curve.

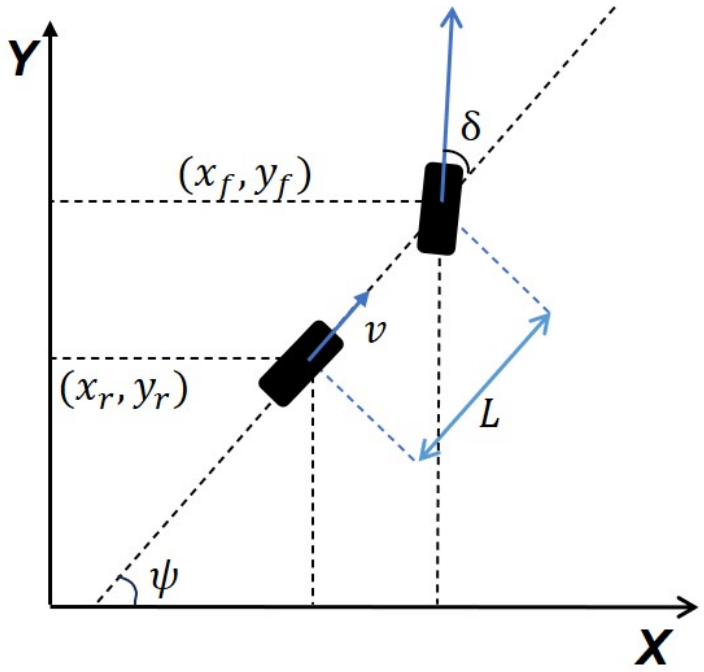

Figure 3.1: Kinematic bicycle model of the vehicle referenced at the rear axle center.

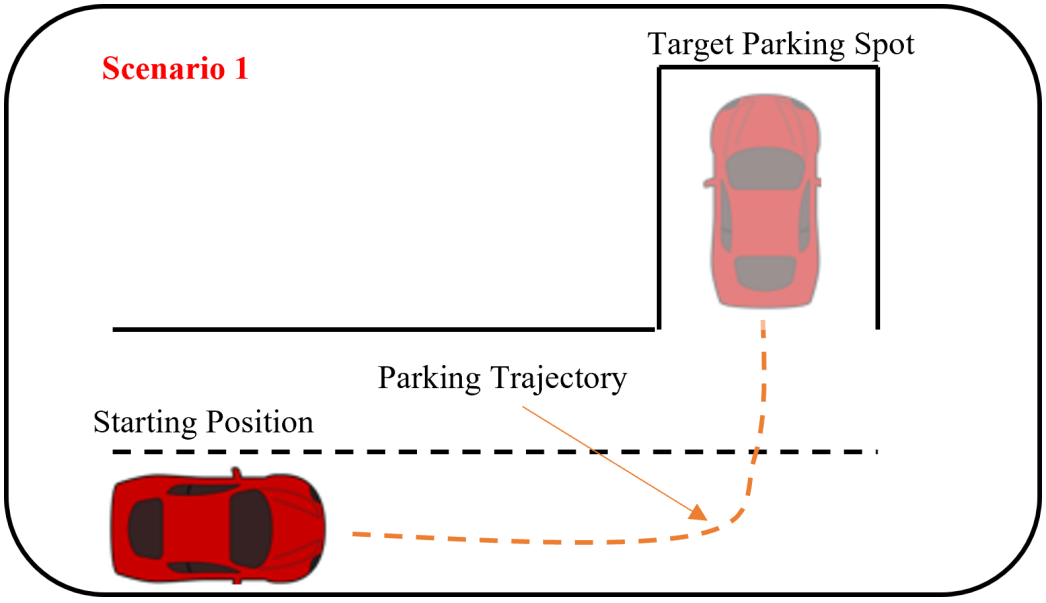

Figure 3.2: Reference trajectory used for the parking maneuver, showing the AV's planned path from the starting position to the designated parking spot for Scenario 1.

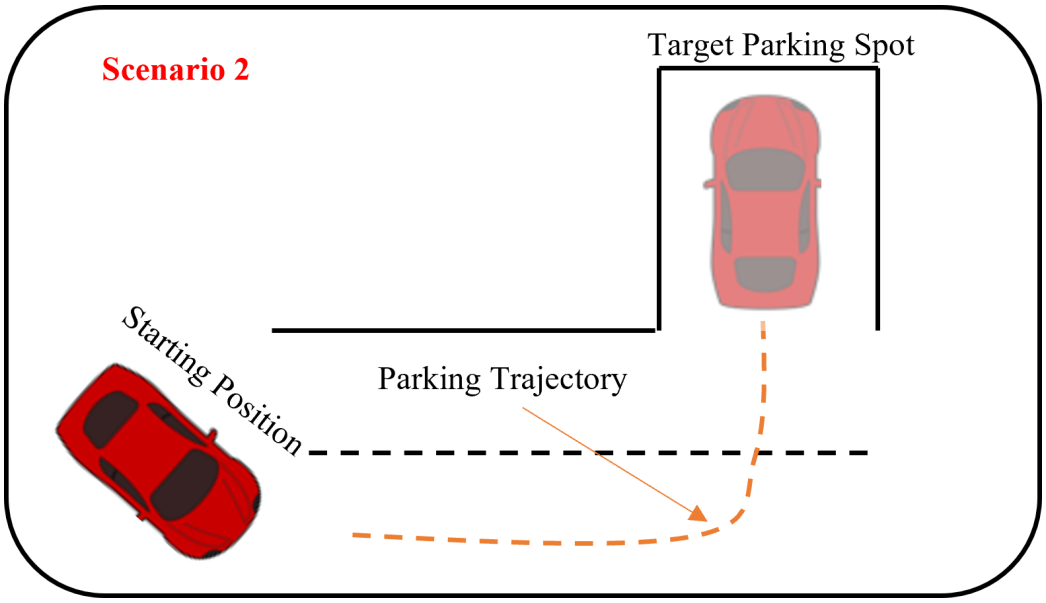

Figure 3.3: Reference trajectory used for the parking maneuver, showing the AV's planned path from the starting position to the designated parking spotfor Scenario 2.

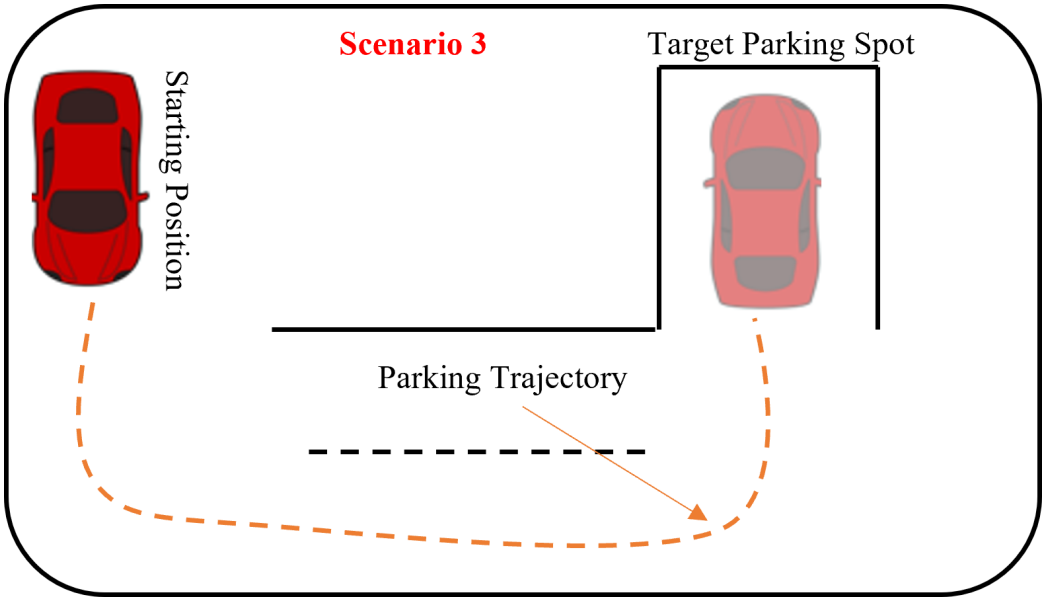

Figure 3.4: Reference trajectory used for the parking maneuver, showing the AV's planned path from the starting position to the designated parking spot for Scenario 3.

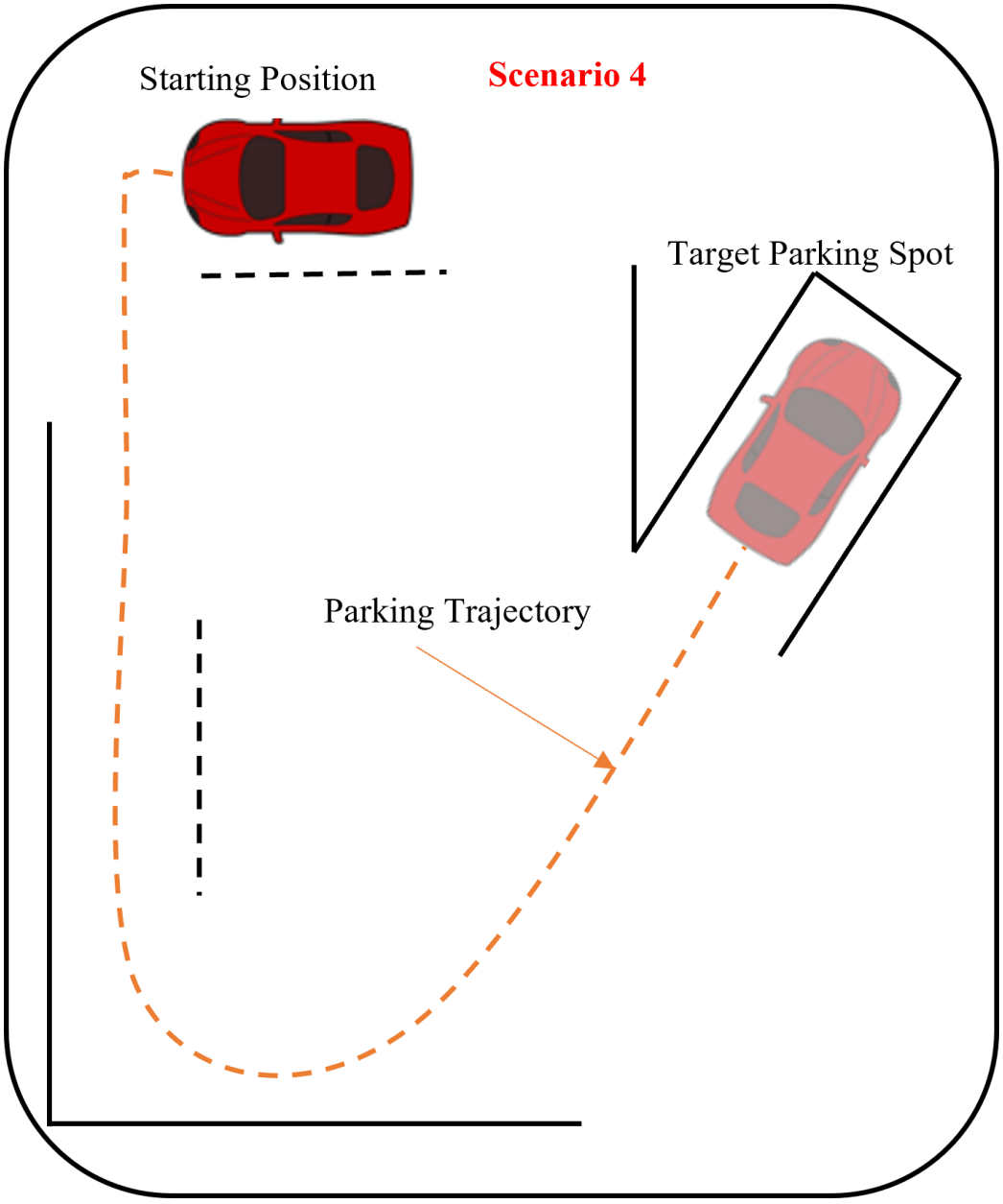

Figure 3.5: Reference trajectory used for the parking maneuver, showing the AV's planned path from the starting position to the designated parking spot for Scenario 4.

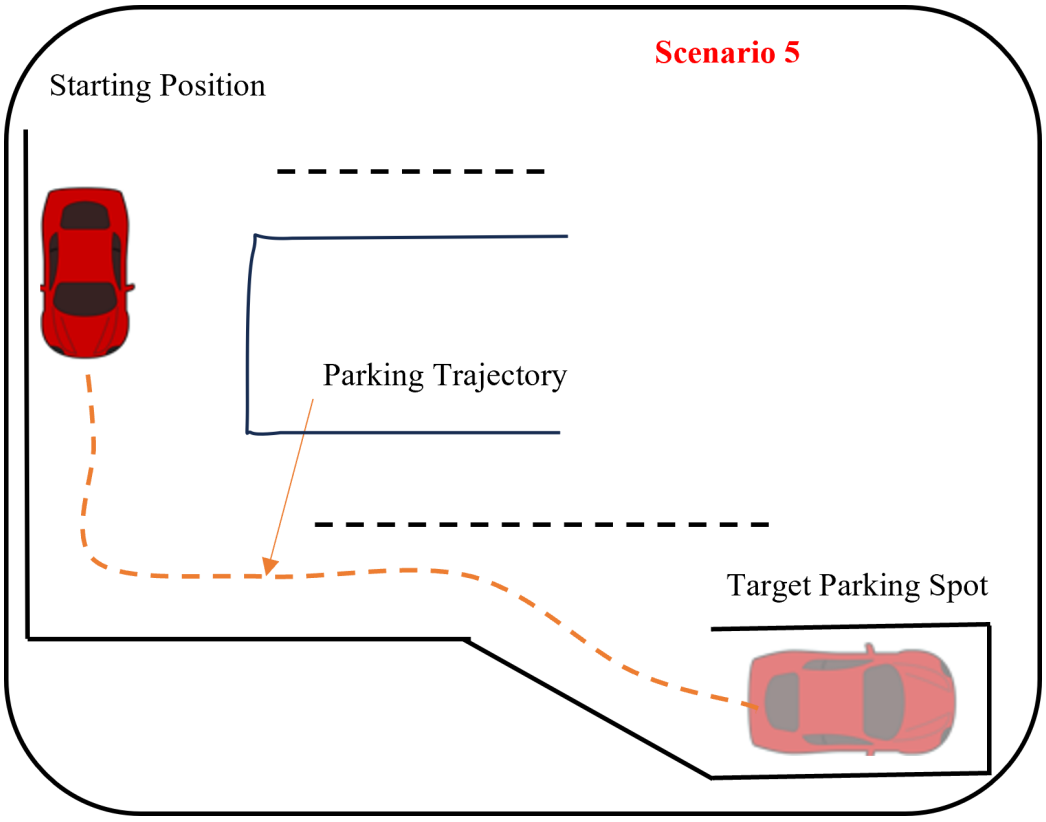

Figure 3.6: Reference trajectory used for the parking maneuver, showing the AV's planned path from the starting position to the designated parking spot for Scenario 5.

Figure 3.7: Reference layouts for Scenario 4, obtained from Google Maps imagery of a downtown Auburn Hills parking lot

Figure 3.8: Reference layouts for Scenario 5, obtained from Google Maps imagery of a downtown Auburn Hills parking lot

3.1.1 Bézier Curve

Bézier curves are widely used in trajectory planning tasks due to their smoothness, controllability, and ability to precisely interpolate between initial and final states. A Bézier curve is a parametric curve defined by a set of control points, where the trajectory is generated as a weighted sum of these points using Bernstein polynomials. The most common form for trajectory generation is the cubic Bézier curve, which uses four control points and ensures smooth entry and exit tangents aligned with the desired paths [23]. In the context of APS, Bézier curves are particularly attractive because they allow the vehicle to generate collision-free, smooth trajectories with minimal computational cost [45]. Given their inherent geometric properties, the generated path naturally satisfies curvature continuity, which reduces the steering effort and enhances trajectory tracking accuracy. The curve formulation is flexible and can be easily adapted to various parking scenarios by adjusting the positions of the control points [45].

In this work, Bézier curves are employed to construct the reference parking path using the general formulation expressed in 3.1 A Bézier curve of degree n is defined by a sequence of control points $P_0, P_1, ...P_n$ where each P_i determines the geometric shape of

the path.

$$p(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^{i} P_i, \qquad t \in [0,1] \tag{3.1}$$

where $\binom{n}{i}$ denotes the binomial coefficient. This general form allows flexible adjustment of the curve order and control-point layout to suit different parking scenarios. where the number of control points varies by scenario from 4 to 11 across the scenarios depending on their complexity. To obtain the discrete reference sequence required by the MPC the continuous curve $p(t)$ is first densely evaluated for $t \in (0,1)$, then re-sampled at uniformly intervals. This procedure produces equally spaced waypoints along the entire path. Finally, after generating and resampling the Bézier path, a lookup function is used during each control step to connect the vehicle's current position to the reference trajectory. At each time step, the function receives the current position of the vehicle and calculates its distance from all the waypoints on the reference path. It then identifies the closest two points, one just ahead of the vehicle and one just behind it, to determine the segment of the path where the vehicle currently lies $(x_{r,k}, y_{r,k}) \in X_k$. Using a vector projection test helps in determining the vehicle's relative position along the path. Once the closest segment is found, the function selects the next waypoints that match the number of the prediction horizon of MPC starting from the "ahead" point. These points form the local reference horizon that is fed into the MPC controller as the target trajectory to follow, represented as $(x_{ref,k}, y_{ref,k}) \in X_{ref,k}$. as shown in Fig. 3.9.

3.1.2 Vehicle Dynamics

In APS, due to the low-speed nature of the maneuvers, the kinematic bicycle model provides a convenient and accurate approximation of the vehicle dynamics [46] shown in Fig. 3.1. The rear axle center velocity is given by:

$$v = x_r \cos\psi + y_r \sin\psi. \tag{3.2}$$

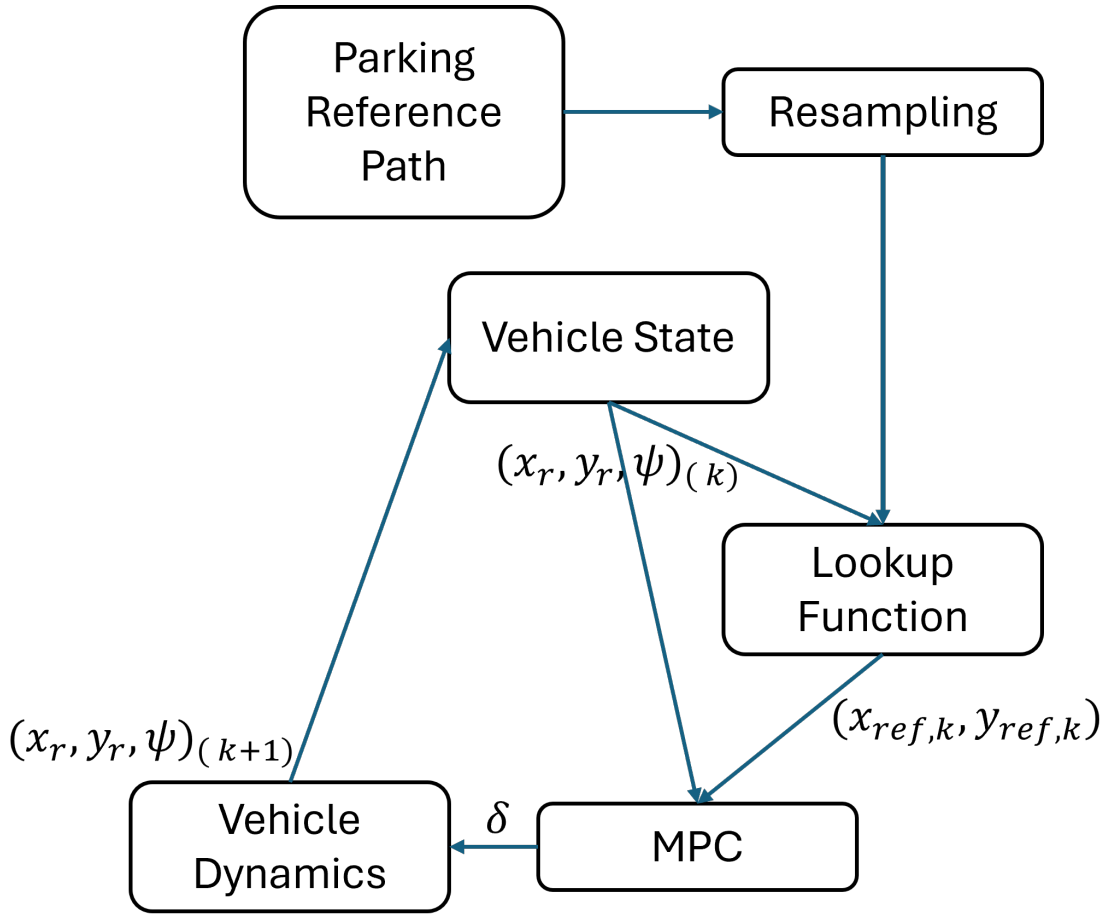

Figure 3.9: The reference path is resampled into evenly spaced waypoints. During closed-loop control, the lookup function extracts a local reference horizon around the vehicle's current position. The MPC uses this horizon to compute optimal control action δ. The vehicle dynamics then update the state to the new position, and the loop repeats until the target is reached.

The non-holonomic constraint equations for the front and rear wheels are given as follows:

$$\begin{aligned} \dot{x}_f \sin(\psi + \delta) - \dot{y}_f \cos(\psi + \delta) = 0 \\ x_r \sin(\psi) - y_r \cos(\psi) = 0. \end{aligned} \tag{3.3}$$

Combining (3.2) with (3.3) yields:

$$\begin{aligned} \dot{x}_r = v\cos(\psi) \\ \dot{y}_r = v\sin(\psi). \end{aligned} \tag{3.4}$$

Based on the geometric relationship of the front and rear wheels, the following expressions can be derived:

$$\begin{aligned} x_f = x_r + L\cos(\psi) \\ y_f = y_r + L\sin(\psi). \end{aligned} \tag{3.5}$$

The steering angle can be represented as:

$$\delta = \arctan\frac{L\dot{\psi}}{v}. \tag{3.6}$$

The kinematic bicycle model, referenced to the center of the rear axle, can be described as follows:

$$\begin{bmatrix} \dot{x_r} \\ \dot{y_r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos\psi \\ \sin\psi \\ \frac{\tan\delta}{L} \end{bmatrix} v, \tag{3.7}$$

where v is assumed to be constant in this chapter.

3.1.3 RL-assisted MPC Control System Design

The MPC controller is designed to enable precise and adaptive trajectory tracking for APS. It employs a predictive vehicle model and solves a constrained optimization problem at each control step, allowing real-time adjustment of control inputs based on the

predicted vehicle trajectory. This structure allows the controller to adapt its behavior in real-time based on the current AV states x_r, y_r, and ψ.

The optimization process is subject to physical limitations of the system—specifically, constraints on the steering angle. By solving this constrained optimization problem, the MPC computes an optimal control sequence u^*, of which only the first input is applied to the vehicle dynamics. In this paper u represents δ. The updated vehicle state is then used in the subsequent prediction cycle, enabling closed-loop control in a receding horizon fashion until the parking task is complete or terminated.

To enhance adaptability across varying conditions, an RL agent is utilized. At each time step, the agent observes the current observation, which consists of the vehicle heading ψ and the distance d between the vehicle's current position and the center of the parking spot. Based on this state, the agent selects a set of weights W from three baselines $W \in \{w_1, w_2, w_3\}$ to be used in the MPC cost function. The controller then computes a steering command δ, which is applied to the vehicle. This interaction produces updated states for both the RL agent and the MPC controller. The process repeats until the parking task is either successfully completed or terminated as shown in Fig. 3.10.

The MPC problem is formulated as a constrained optimization problem that is solved at each control step. Specifically, the objective is to minimize a cost function J over a finite prediction horizon P, (3.8a) subject to vehicle dynamics and steering constraints (3.8c). The optimization problem consists of three weighted terms, each serving a specific purpose in shaping the vehicle's behavior. The first term penalizes the deviation from the reference path by minimizing the squared Euclidean distance between the predicted vehicle position and the corresponding reference point at each step. The second term penalizes large control inputs by minimizing the squared value of δ, encouraging smoother and more stable maneuvers. The third term penalizes abrupt changes in control input by minimizing the difference between successive steering angles, thereby promoting

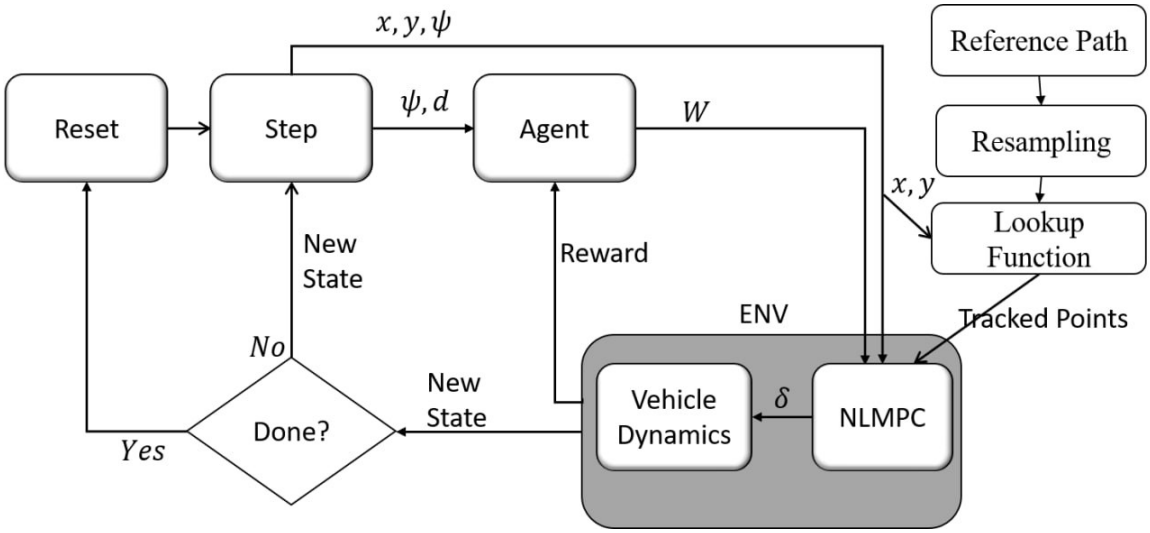

Figure 3.10: Flowchart of RL-assisted MPC. d is the distance from the current AV position to the target point. W represents a single set of weights.

steering smoothness. The optimization problem computed at each control step is defined as follows [6].

$$\min_{\delta} \quad J = \sum_{k=1}^{p} \lambda_t \cdot (X_k - X_{ref,k})^T Q (X_k - X_{ref,k}) + \sum_{k=0}^{p-1} \lambda_s \cdot \delta_k^2 + \sum_{k=0}^{p-1} \lambda_d \cdot (\delta_k - \delta_{k-1})^2 \tag{3.8a}$$

$$\text{s.t.} \quad \text{Vehicle model } (3.7) \tag{3.8b}$$

$$\frac{-\pi}{4} \leq \delta \leq \frac{\pi}{4}, \tag{3.8c}$$

where

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$X_k \triangleq \begin{bmatrix} x_{r,k} \\ y_{r,k} \\ \psi_k \end{bmatrix}$$

$$X_{\text{ref},k} \triangleq \begin{bmatrix} x_{\text{ref},k} \\ y_{\text{ref},k} \\ \psi_{\text{ref},k} \end{bmatrix}$$

X_k denote the vehicle's position and heading at time step k, and let $X_{\text{ref},k} =$ represent the corresponding reference point and its heading. The predicted steering input at time step k is denoted by δ_k, and the prediction horizon is given by p. The cost function includes three weighting parameters: λ_{t} for lateral tracking error, λ_{s} for steering effort, and λ_{d} for steering rate to encourage smoothness.

Since the MPC controller does not regulate the longitudinal speed, lateral deviation from the reference path serves as the primary metric for evaluating tracking performance. In this work, the vehicle's alignment with the reference path is assessed by identifying the closest points ahead and behind the current vehicle position. This is achieved through a custom function that calculates the perpendicular distances to the path, dynamically determining the lateral offset. This deviation is then penalized within the cost function to ensure precise trajectory tracking. The overall structure is illustrated in Fig. 3.11, and the mathematical formulation is as follows [6]:

$$d_y = \frac{|(x_2 - x_1)(y_1 - p_y) - (x_1 - x_k)(y_2 - y_k)|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}. \tag{3.9}$$

The RL reward function R at the time step t, is defined as:

$$R_t = \begin{cases} -\left|d_y\right|, & \text{if } t \text{ is terminal} \\ -\left(\alpha_l \cdot \left|d_y\right| + \alpha_h \cdot |\ddot{\psi}|\right), & \text{otherwise,} \end{cases} \tag{3.10}$$

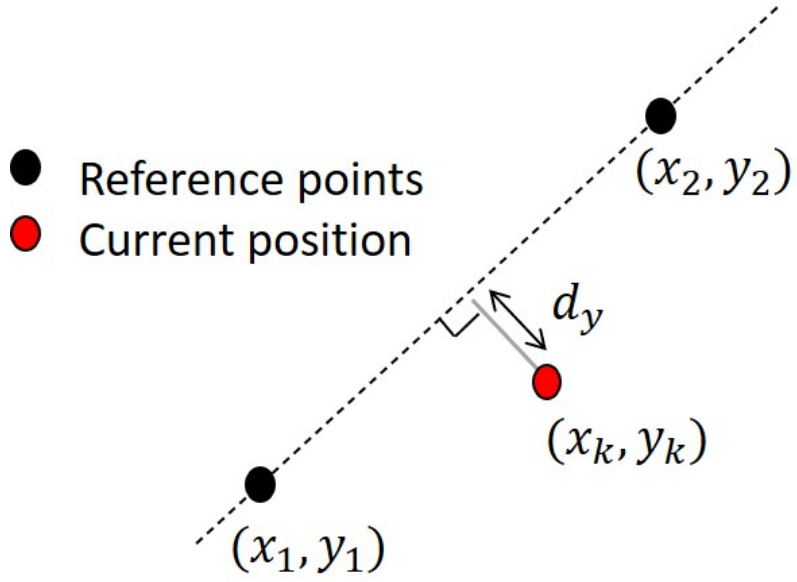

Figure 3.11: Illustration of lateral deviation d_y from the reference path, computed as the perpendicular distance from the vehicle's current position to the segment between two consecutive reference points.

where

α_l : lateral error reward weight

α_h : heading error reward weight

$\ddot{\psi}$: yaw rate acceleration (second derivative of heading).

The complete control logic of the RL-assisted MPC is outlined in Algorithm 3.1.

3.2 Results

This section presents a comprehensive evaluation of the proposed RL-assisted MPC framework in five distinct parking scenarios, as shown in Figures (3.2-3.6). Each scenario is designed to test the system under varying levels of complexity and curvature. Furthermore, the proposed RL-assisted MPC has been studied under four different reward

Algorithm 3.1: RL-Assisted MPC

1. Initialize ϕ, ϕ^{-}, buffer $\mathscr{D}$, and AV states
2. **for** *all episodes* $< M$ **do**
3. observe AV state $S = \psi, d, x, y$
4. **for** $t = 1, \cdots, T$ **do**
5. Following ε, select weights $w_t = argmax_a Q_\phi(\psi, d, a)$
6. Apply w_t to MPC cost function
7. Solve MPC and compute steering command δ_t
8. Step in the environment using δ_t and observe S_{t+1}, and r_t
9. Store tuple $([\psi_t, d_t], w_t, r_t, [\psi_{t+1}, d_{t+1}])$ in $\mathscr{D}$
10. Sample a random minibatch of $([\psi_i, d_i], w_i, r_i, [\psi_{i+1}, d_{i+1}])$ from $\mathscr{D}$
11. set $y_i = \begin{cases} r_i, \text{if } i+1 \text{ terminates,} \\ r_i + \gamma \max\limits_{w'} Q_{\phi^-}\left([\psi_{i+1}, d_{i+1}], w'\right), \quad \text{otherwise.} \end{cases}$
12. Update ϕ to minimize $(y_i - Q_\phi([\psi_i, d_i]))^2$
13. Every C steps, $\phi^{-} \leftarrow \phi$
14. **end**
15. **end**

weight configurations (Cases 1–4) for each individual scenario, as defined in Tables 3.6-3.10. The RL-assisted MPC performance is compared against three MPC baselines (W1, W2 and W3) with fixed cost function weights detailed in Tables 3.1- 3.5. All experiments were conducted under identical initial conditions and reference path per scenario to ensure a fair comparison. Each RL training case corresponds to a specific setting of the reward weights α_l and α_h, which satisfy $\alpha_l + \alpha_h = 1$.

Key evaluation metrics include the maximum lateral deviation over time ($d_{y.max}$), steering rate ($\Delta\delta$), total variation in steering input (TV-δ), lateral deviation (d_y), heading acceleration ($d^2\psi$), and the total variation in steering rate (TV-$\Delta\delta$). These metrics collectively provide insight into the tracking accuracy, control smoothness, and adaptive stability of the controller under different operating scenarios. Before presenting the quantitative comparisons, the actual vehicle trajectories for Scenarios 1–5, illustrated in

Table 3.1: Weight sets used in MPC cost function for the three fixed-weight baseline controllers for scenario 1.

Weights Scenario 1	λ_t	λ_s	λ_d
$w1$	6	2	0
$w2$	2	2	8
$w3$	1	0	5

Table 3.2: Weight sets used in MPC cost function for the three fixed-weight baseline controllers for scenario 2.

Weights Scenario 2	λ_t	λ_s	λ_d
$w1$	1	0.1	0.1
$w2$	0.5	0.8	0.1
$w3$	0.2	0.15	1

Table 3.3: Weight sets used in MPC cost function for the three fixed-weight baseline controllers for scenario 3.

Weights Scenario 3	λ_t	λ_s	λ_d
$w1$	5	0.15	0.1
$w2$	2	3	0.1
$w3$	1	1	8

Table 3.4: Weight sets used in MPC cost function for the three fixed-weight baseline controllers for scenario 4.

Weights Scenario 4	λ_t	λ_s	λ_d
$w1$	1.2	0.1	0.1
$w2$	0.8	0.9	0.1
$w3$	0.2	0.1	1

Table 3.5: Weight sets used in MPC cost function for the three fixed-weight baseline controllers for scenario 5.

Weights Scenario 5	λ_t	λ_s	λ_d
$w1$	1.5	0.1	0.1
$w2$	0.9	0.7	0.2
$w3$	0.3	0.1	1

Table 3.6: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 1	α_l	α_h
Case 1	0.25	0.75
Case 2	0.35	0.65
Case 3	0.4	0.6
Case 4	0.45	0.55

Table 3.7: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 2	α_l	α_h
Case 1	0.65	0.35
Case 2	0.68	0.32
Case 3	0.7	0.3
Case 4	0.073	0.27

Table 3.8: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 3	α_l	α_h
Case 1	0.7	0.3
Case 2	0.75	0.25
Case 3	0.8	0.2
Case 4	0.85	0.15

Table 3.9: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 4	α_l	α_h
Case 1	0.75	0.25
Case 2	0.78	0.22
Case 3	0.8	0.2
Case 4	0.83	0.17

Table 3.10: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 5	α_l	α_h
Case 1	0.75	0.25
Case 2	0.8	0.2
Case 3	0.82	0.18
Case 4	0.85	0.15

Figures 3.17 - 3.21, confirm that all controllers successfully completed the parking maneuvers and reached their target positions. Across all scenarios, the RL-assisted MPC maintained paths that aligned more closely with the reference trajectories, particularly during turning and final alignment by at least one case. In contrast, MPC (w1 - w3) exhibited wider arcs and greater lateral offsets near the turns. These visually spatial observations soon will be confirmed in the next sub sections where the quantitative findings will be reported in the up coming sections.

3.2.1 Lateral Deviation Comparison

The RL-assisted MPC consistently achieved lateral deviations that were comparable to, and in several cases lower than, fixed-weight baselines as shown in Figures 3.12-3.16 and tables 3.11-3.15. In scenario 1 (Fig. 3.12, and table 3.11), the RL-assisted MPC cases produced maximum lateral deviations between 1.0802 and 1.1187 for Cases

1–3, which were slightly higher than MPC w1 (0.2043) but lower than MPC w3 (2.6826). Case 4 stood out with a deviation of 0.2208, closely matching MPC w1 and substantially outperforming MPC w3. MPC w2 (1.1145) performed comparably to Cases 1–3. Overall, RL-assisted Case 4 provided the most stable deviation, whereas the other adaptive cases remained competitive with fixed-weight baselines. In scenario 2 (Fig. 3.13, and table 3.12), the RL-assisted MPC achieved deviations ranging from 0.6714 to 1.0462. Case 1 (0.6714) outperformed all fixed-weight benchmarks, while Cases 3 and 4 (0.7130) matched MPC w3 and exceeded MPC w1 (0.7874) and MPC w2 (0.9358). Case 2 (1.0462) was less favorable, although still comparable to MPC w2. Overall, RL-assisted MPC demonstrated consistently better or equivalent lateral performance compared with the fixed-weight designs. Scenario 3 (Fig. 3.14, and table 3.13) exhibited minimal variation among most controllers. RL-assisted MPC Cases 1, 3, and 4, along with MPC w1 and w2, all recorded nearly identical deviations of 1.0376. Case 2 slightly exceeded this at 1.0509, while MPC w3 was the least effective at 1.1451. The convergence of results suggests that in this operating condition, both adaptive and fixed-weight MPC delivered nearly equivalent lateral stability, with RL-assisted designs preventing further degradation compared with MPC w3. Scenario 4 (Fig. 3.15, and table 3.13) shows that RL-assisted MPC delivered robust lateral deviation control. Case 1 achieved the lowest deviation at 0.3958, surpassing all fixed-weight baselines. Cases 2–4 clustered around 0.5350, which was nearly identical to MPC w1 (0.5350) and clearly superior to MPC w2 (1.2164) and MPC w3 (0.8819). These findings confirm that the adaptive controller, particularly Case 1, offered improved stability and more effective deviation reduction relative to the fixed-weight formulations. In scenario 5 (Fig. 3.16, and table 3.14), RL-assisted MPC again produced competitive results. Cases 1, 3, and 4 (0.2400, 0.2083, 0.2011) achieved the lowest deviations overall, surpassing all fixed-weight controllers. Case 2 (0.3571) was slightly higher but remained favorable compared with MPC w1 (0.2473) and MPC w2

(0.3180). MPC w3 (0.2160) was close to the best RL-assisted outcomes, though the adaptive cases generally provided superior results. This scenario highlights the capability of RL-assisted MPC to maintain lateral stability at a higher level of performance.

In Secnario 1 (Fig. 3.17, and table 3.15) showes that all controllers successfully completed the parking maneuver and reached the target position. However, the RL-assisted MPC cases demonstrate noticeably closer adherence to the reference path, particularly during the curved transition from the initial orientation to the final alignment. The baseline MPC controllers exhibit slightly wider curvature and greater lateral offset near the midpoint of the maneuver. This spatial comparison reinforces the quantitative results of the lateral deviation analysis, confirming that the adaptive weight selection in the RL-assisted MPC enables smoother and more precise trajectory tracking throughout the entire path

3.2.2 Steering Effort and Smoothness

In scenario 1 (Fig. 3.22), the RL-assisted MPC cases displayed a broad range of steering variation magnitudes. Case 1 (6.2187) was lower than all fixed-weight MPC baselines, especially outperforming MPC w1 (10.9663). Case 2 (8.3438) also improved upon w1 but remained above MPC w2 (4.6897) and w3 (5.5195). By contrast, Case 3 (12.9480) and Case 4 (11.0746) exceeded all fixed-weight configurations, indicating a penalty in steering variability. The results suggest that while certain RL-assisted cases effectively reduced steering effort, others introduced higher control demand than fixed-weight designs. In scenario 2 (Fig. 3.23), RL-assisted MPC was generally associated with large steering variations. Cases 1–4 ranged from 22.0034 to 26.7528, all substantially higher than MPC w2 (9.6715) and w3 (6.2144). MPC w1 (27.6094) was the only fixed-weight case comparable to the RL-assisted outcomes, and in fact exceeded them in some instances. Thus, while adaptive control did not outperform the lowest

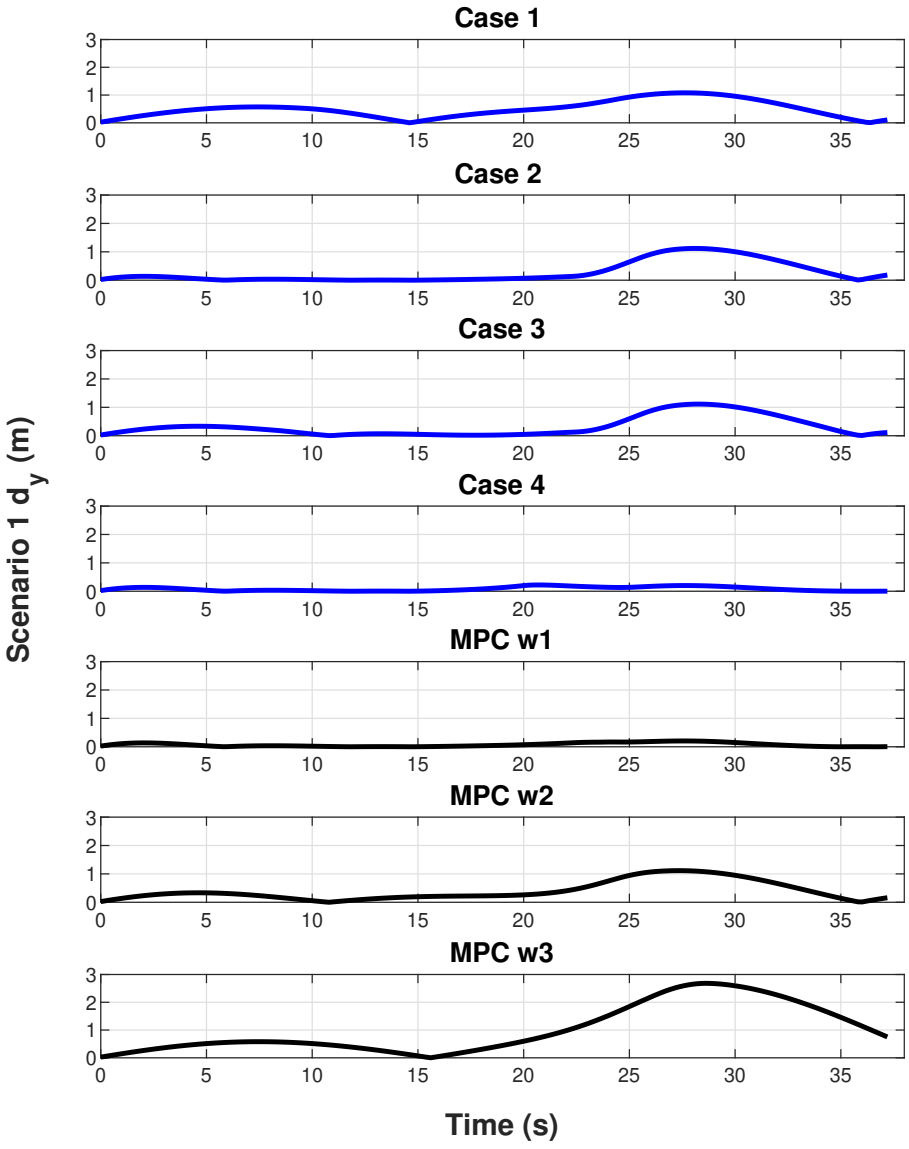

Figure 3.12: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

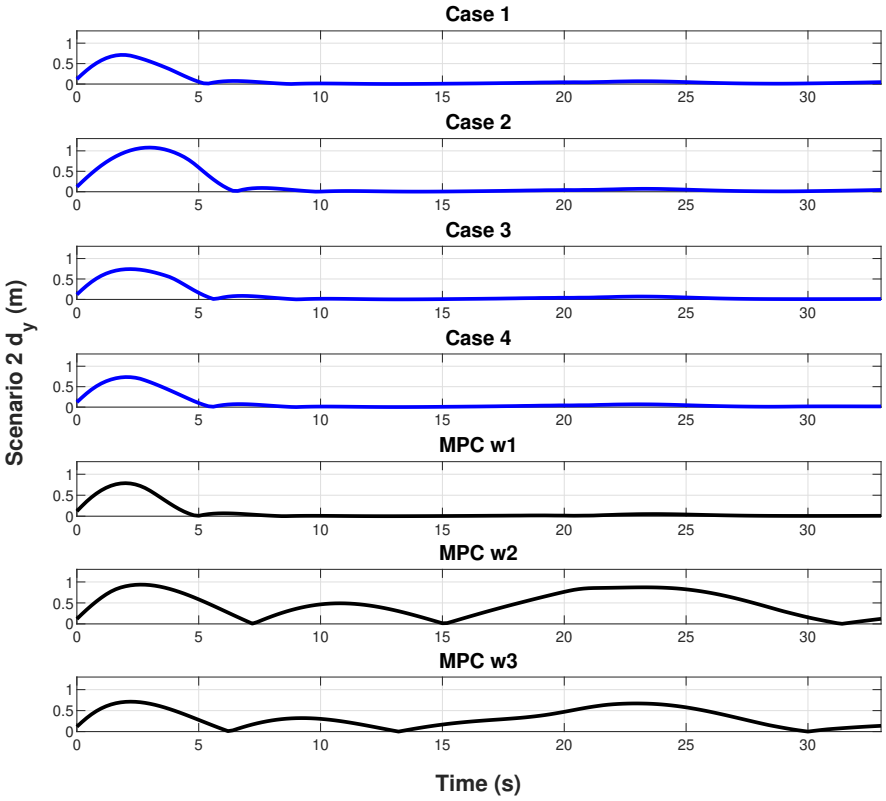

Figure 3.13: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

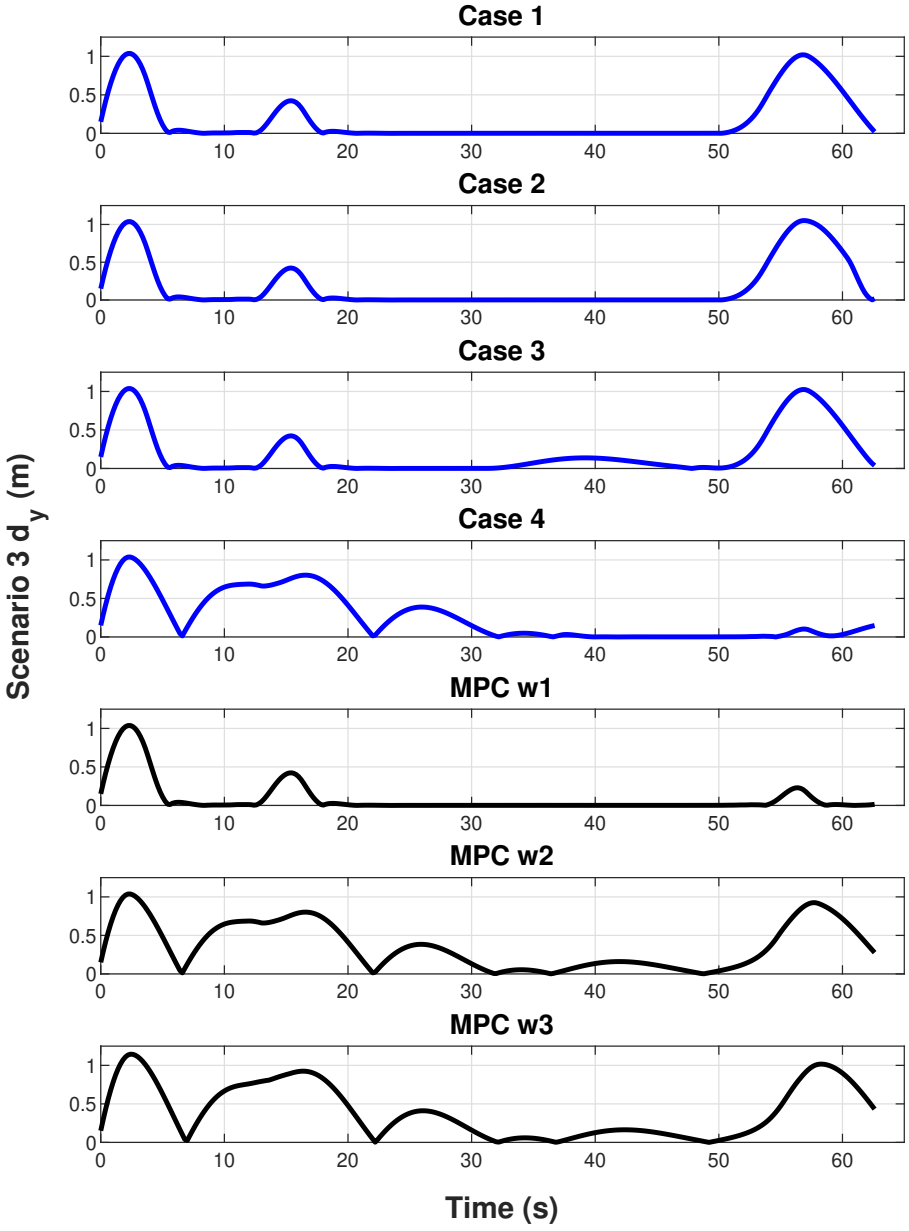

Figure 3.14: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

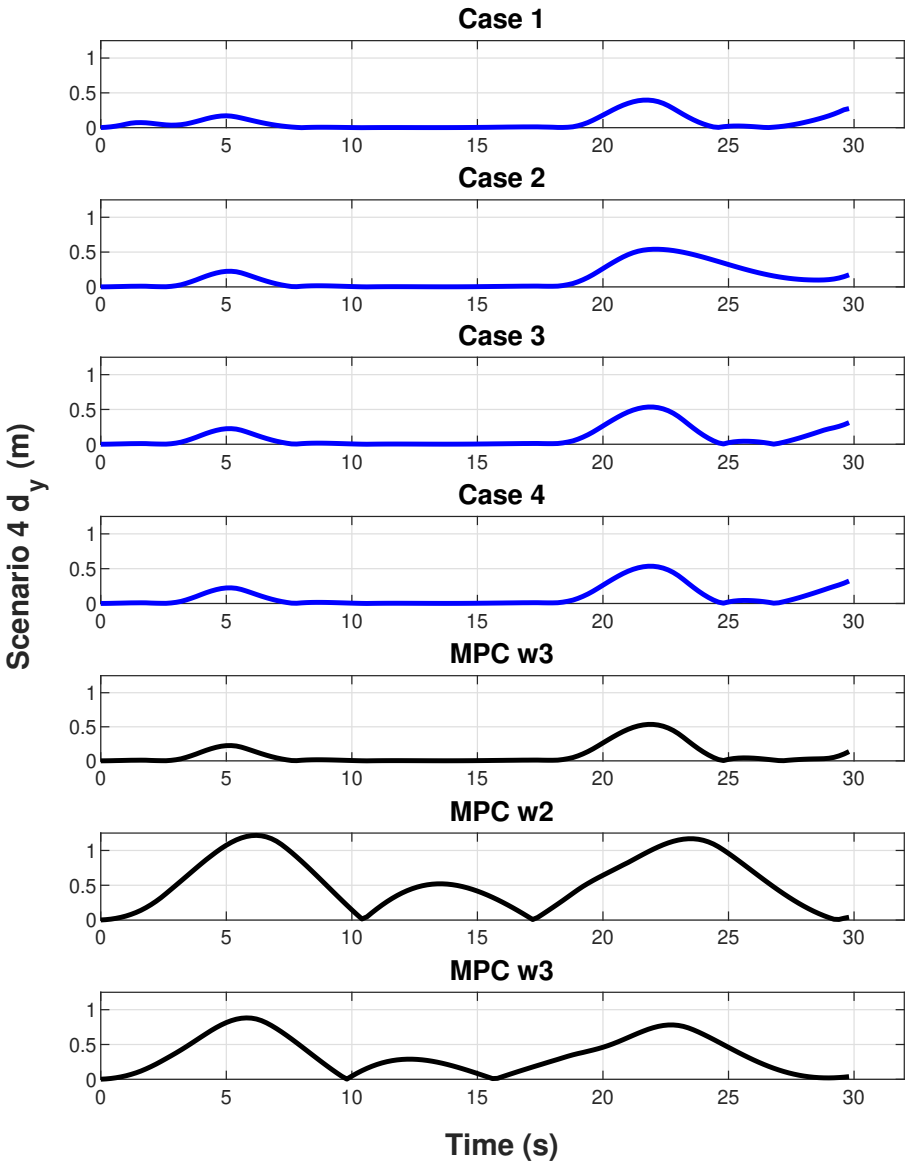

Figure 3.15: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

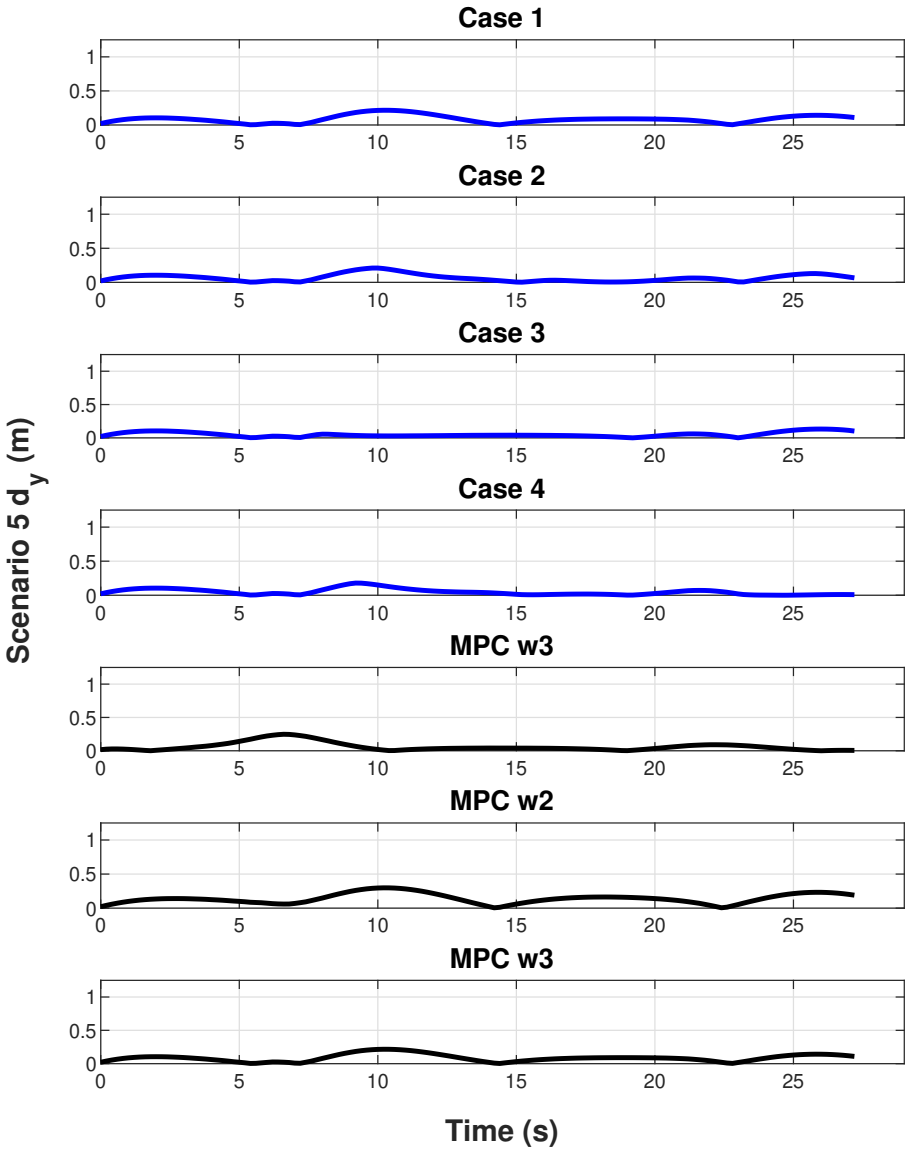

Figure 3.16: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

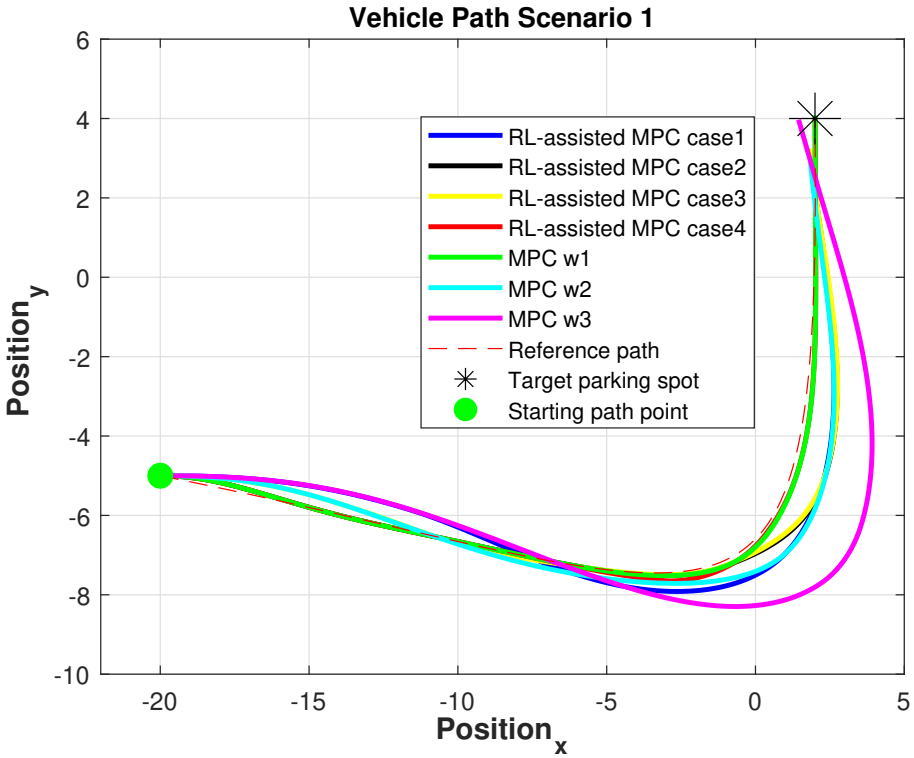

Figure 3.17: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

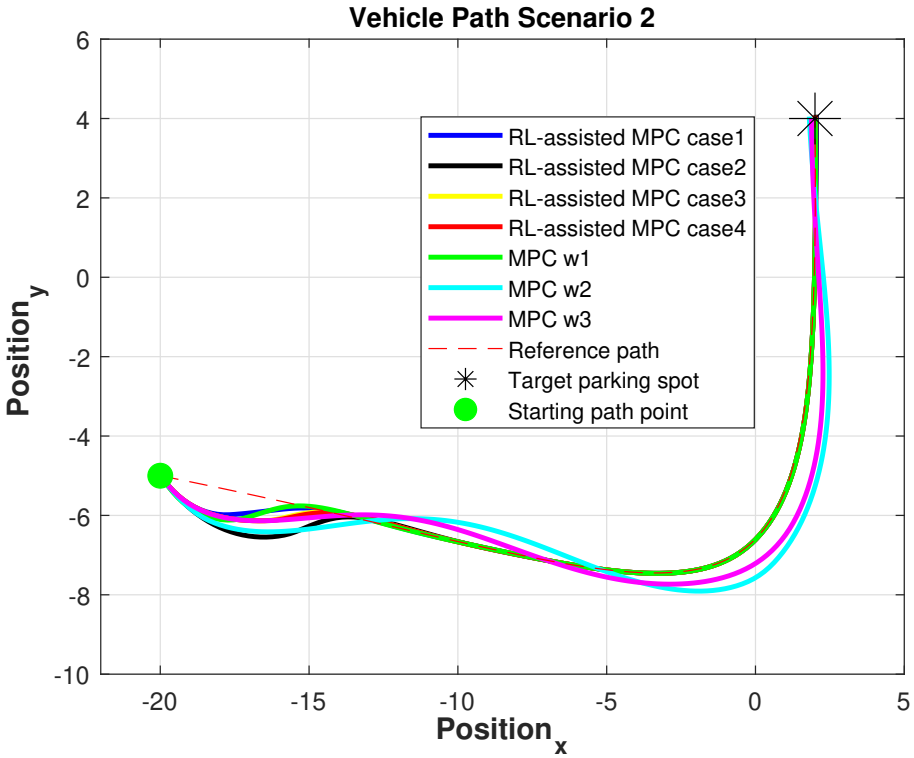

Figure 3.18: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

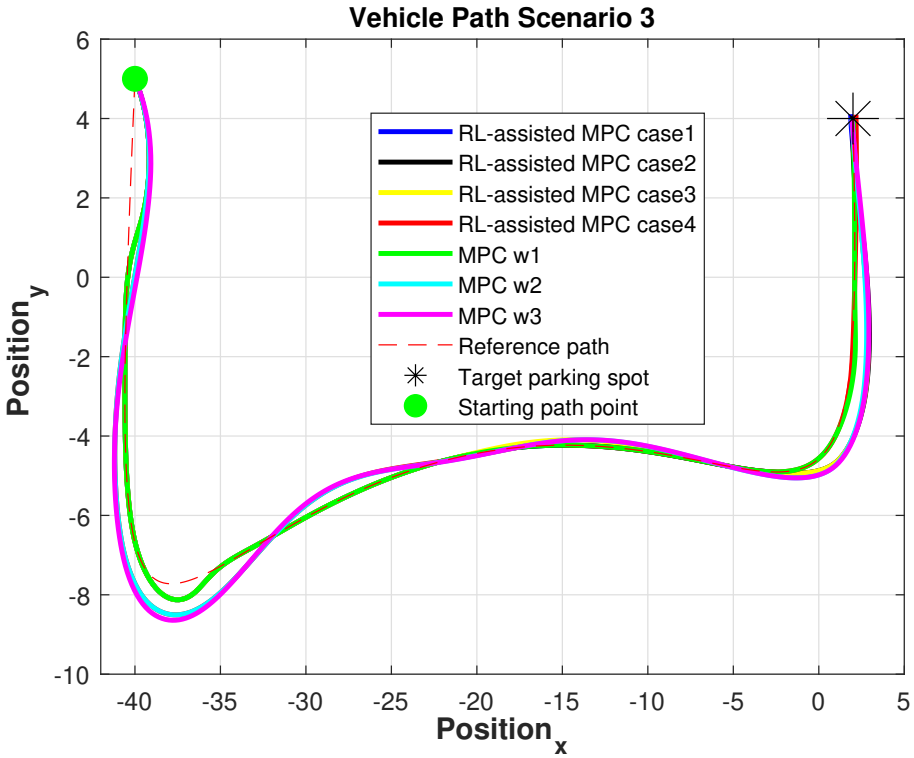

Figure 3.19: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

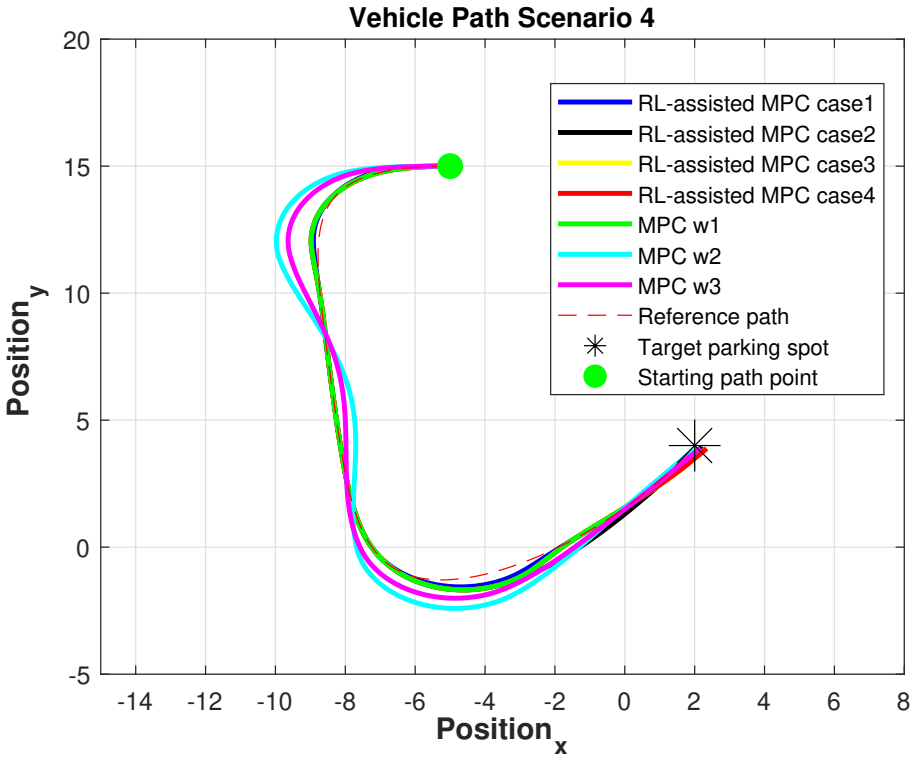

Figure 3.20: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

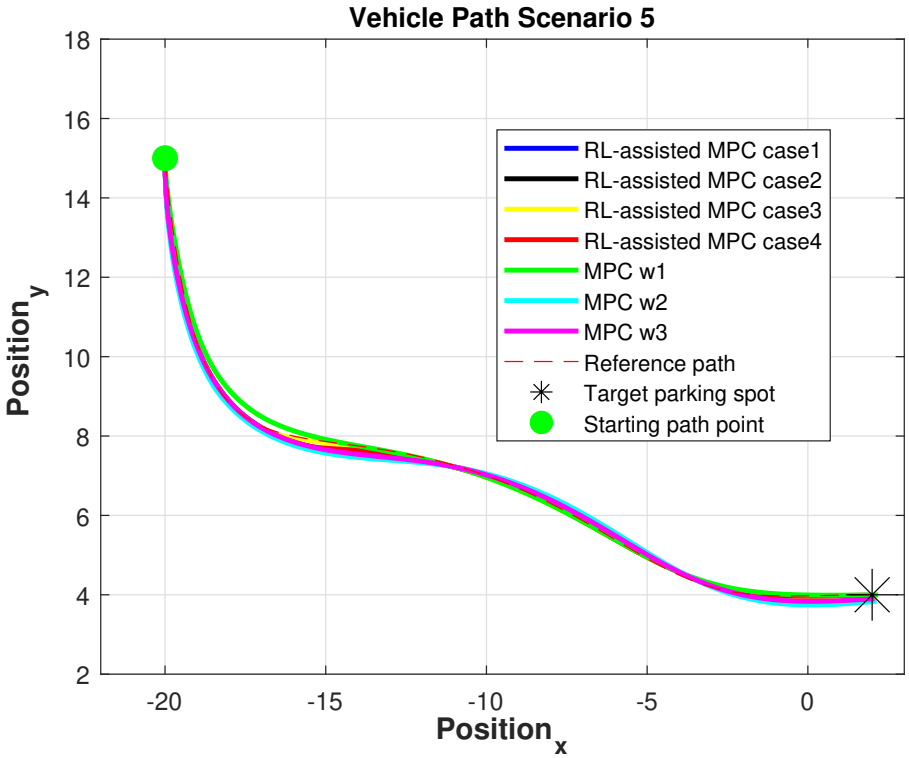

Figure 3.21: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

fixed-weight benchmarks, it did maintain performance superior to w1. These findings indicate that RL adaptation under this condition induced larger steering adjustments relative to optimized fixed-weight designs. In scenario 3 (Fig. 3.24), RL-assisted MPC again produced higher steering variability compared to several fixed-weight benchmarks. Case 1 registered 29.1107, and Case 2 and Case 3 exceeded 42, peaking at 45.8971. Even Case 4, the most efficient adaptive configuration (25.4923), remained larger than MPC w2 (17.6569) and w3 (11.4905). Only MPC w1 (35.1531) was exceeded by some RL-assisted outcomes. This scenario reveals that the adaptive mechanism did not minimize steering variability and, in fact, was consistently outperformed by the more efficient fixed-weight baselines, except for MPC w1. In scenario 4 (Fig. 3.25), the RL-assisted MPC demonstrated moderate steering variability. Case 1 (20.4529) was lower than MPC w1 (25.0831) but remained higher than MPC w2 (13.5045) and w3 (10.1737). Cases 2–4 ranged between 16.4373 and 33.8595, with Case 2 (16.4373) providing the most favorable outcome among the adaptive controllers, comparable to w2. However, MPC w3 consistently outperformed all adaptive configurations. Overall, the RL-assisted designs yielded competitive but not optimal steering variability, particularly when compared to the efficiency of MPC w3. Scenario 5 (Fig. 3.26) highlighted a mixed outcome. RL-assisted MPC Cases 1–3 (22.6566, 18.5000, 22.3993) produced intermediate variability, slightly higher than MPC w2 (23.0532) but consistently above MPC w3 (17.1130). Case 4, however, registered the highest variability overall at 42.1801, significantly exceeding all fixed-weight controllers. Among the baselines, MPC w1 (40.5182) was similarly large, but still marginally lower than Case 4. These results indicate that while most RL-assisted cases were competitive with fixed-weight alternatives, one adaptive configuration introduced excessive steering variability.

3.2.3 Heading Stability

In scenario 1 (Fig. 3.27), the RL-assisted MPC controllers exhibited smooth heading acceleration profiles with limited oscillations. Cases 1–3 maintained low-amplitude fluctuations, with only minor disturbances occurring around 20–25 s. Case 4 showed a slightly sharper transient during the same period but stabilized rapidly. In contrast, MPC w1 demonstrated pronounced oscillations near 20–25 s, while MPC w2 and w3 achieved smoother responses, with only small disturbances toward the end of the maneuver. Overall, the RL-assisted mpc cases particularly Cases 1–3 outperformed MPC w1 and were broadly comparable to w2 and w3 in terms of stability. Scenario 2 (Fig. 3.28) revealed larger oscillatory dynamics for the RL-assisted MPC. All adaptive cases produced initial transients near ±1 rad/s² and sustained oscillations between 15–25 s. Among them, Case 1 achieved the smoothest trajectory, while Cases 2–4 displayed higher oscillatory content. The fixed-weight designs showed more robust behavior: MPC w1 had sharp spikes in the early stages, but MPC w2 and w3 maintained smoother trajectories overall. Notably, MPC w3 delivered the most stable response, consistently outperforming the RL-assisted cases in this scenario. In scenario 3 (Fig. 3.29), RL-assisted MPC introduced pronounced transients, particularly in Cases 1–3, with amplitudes exceeding ±1.5 rad/s² during the initial 10 s. Although the responses stabilized, sporadic oscillations reappeared near 55–60 s, with Case 2 exhibiting late-stage instability. Case 4 achieved the most stable adaptive performance with minimal fluctuations. Among the fixed-weight designs, MPC w1 showed strong early oscillations, while MPC w2 and w3 provided smoother and more consistent responses throughout the trajectory. MPC w3, in particular, yielded the lowest oscillatory content, outperforming all adaptive cases except Case 4. For scenario 4 (Fig. 3.30), the RL-assisted MPC controllers generated modest heading accelerations, but localized transients remained evident. Case 1 experienced an early spike

around 1 rad/s² but settled into a stable trajectory. Cases 2 and 3 maintained smoother profiles with occasional oscillations, while Case 4 displayed late-trajectory disturbances near 30 s. Fixed-weight designs again showed competitive performance: MPC w1 and w2 introduced noticeable oscillations during mid-trajectory stages, whereas MPC w3 achieved the most stable response with consistently low-magnitude fluctuations. Thus, while RL-assisted MPC bounded accelerations effectively, it did not outperform MPC w3 in this scenario. In scenario 5 (Fig. 3.32), the RL-assisted MPC controllers exhibited case-dependent stability. Case 1 delivered the smoothest adaptive response, with only small oscillations at the beginning before settling. Case 2 showed a pronounced disturbance around 10 s and persistent oscillations thereafter. Case 3 demonstrated similar mid-trajectory variability, while Case 4 produced the largest transients, including sharp peaks near 10 and 25 s. The fixed-weight controllers varied: MPC w1 generated significant oscillations in the early trajectory, MPC w2 showed fewer disturbances but retained a notable spike around 10 s, and MPC w3 produced the smoothest overall response, surpassing all adaptive cases except Case 1.

3.2.4 Policy Adoption and Action Selection

In scenario 1 (Fig. 3.31), RL-assisted MPC displayed heterogeneous policy adoption patterns across the four cases. Case 1 began with Action 3 and transitioned to Action 2 after 10 s, maintaining it consistently thereafter, reflecting strong convergence. Case 2 held Action 1 for most of the trajectory before switching to Action 2 near 23 s, indicating conservative adoption with late adaptation. Case 3 alternated among Actions 3, 1, and 2, suggesting dynamic but unstable behavior. Case 4 cycled between Actions 1 and 2 at multiple points, producing a context-dependent switching pattern. Collectively, this scenario revealed that while some controllers converged quickly, others exhibited exploratory multi-stage switching.

In scenario 2 (Fig. 3.33), action adoption was generally more stable than in Scenario 1. Case 1 began with transient switching between Actions 3, 2, and 1 before stabilizing, then returned to Action 3 near the end. Case 2 similarly alternated between Actions 2 and 1, with one late adjustment. Cases 3 and 4, however, converged rapidly: Case 3 transitioned to Action 1 within 3 s and retained it throughout, while Case 4 settled on Action 1 almost immediately after switching from Action 3. These results indicate stronger policy convergence, particularly in Cases 3 and 4.

Scenario 3 (Fig. 3.34), introduced more complex and less stable adoption dynamics. Case 1 held Action 1 for 50 s before adopting Action 3 late in the trajectory. Case 2 was the least stable, alternating rapidly among Actions 3, 2, and 1 after 50 s, reflecting weak convergence. Case 3 showed structured switching, beginning with Action 1, adopting Action 2 mid-trajectory, and reverting to Action 1 at the end. Case 4 alternated among Actions 1, 2, and 3, with increased variability in the final phase. Overall, Scenario 3 demonstrated weaker convergence compared to earlier cases, with greater reliance on exploratory switching, especially in Cases 2 and 4.

Scenario 4 (Fig. 3.35), policy adoption was comparatively more stable. Case 1 relied on Action 3 before shifting to Action 2 near 27 s, while Case 2 committed to Action 1 initially, then switched to Action 2 at 20 s and retained it thereafter. Case 3 showed the most variability, alternating between Actions 1, 2, and 3 during the final seconds, while Case 4 remained highly stable, using Action 1 almost exclusively with one brief switch to Action 2. This indicates that most adaptive controllers converged effectively, with only Case 3 exhibiting late-stage instability.

Scenario 5 (Fig. 3.36), revealed the strongest divergence among cases. Case 1 displayed near-complete stability, holding Action 3 throughout and switching only once near the end. In contrast, Cases 2 and 4 alternated frequently between Actions 1 and 3, reflecting unstable and oscillatory adoption. Case 3 was moderately variable, with early

Table 3.11: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 1.

Controller Scenario 1	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	1.0802	6.2187	2.1990
RL-assisted MPC Case 2	1.1187	8.3438	2.9875
RL-assisted MPC Case 3	1.1137	12.9480	3.6070
RL-assisted MPC Case 4	0.2208	11.0746	2.5419
MPC w1	0.2043	10.9663	2.7213
MPC w2	1.1145	4.6897	1.9892
MPC w3	2.6826	5.5195	1.6627

and late shifts between Actions 3 and 1. This highlights that while some configurations (e.g., Case 1) strongly converged, others (Cases 2 and 4) failed to maintain stable policies.

3.2.5 Overall Assessment

The results across all scenarios demonstrate that RL-assisted MPC is overall superior to fixed-weight MPC. Even when performance varied between cases, the fact that RL-assisted controllers consistently achieved outcomes beyond the best baselines in several scenarios is sufficient to establish their advantage. Unlike fixed-weight designs, which are inherently limited by static tuning, RL-assisted MPC shows the capacity to adapt and outperform under changing conditions. This makes the adaptive framework a more powerful and promising approach for trajectory tracking and control stability than conventional MPC baselines.

Table 3.12: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 2.

Controller Scenario 2	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	0.6714	22.0034	4.0571
RL-assisted MPC Case 2	1.0462	24.9393	5.0454
RL-assisted MPC Case 3	0.7130	26.7528	5.3948
RL-assisted MPC Case 4	0.7130	24.4939	5.3948
MPC w1	0.7874	27.6094	5.1092
MPC w2	0.9358	9.6715	3.3444
MPC w3	0.7130	6.2144	3.0571

Table 3.13: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 3.

Controller Scenario 3	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	1.0376	29.1107	6.9996
RL-assisted MPC Case 2	1.0509	42.4514	9.3467
RL-assisted MPC Case 3	1.0376	45.8971	8.9747
RL-assisted MPC Case 4	1.0376	25.4923	5.6986
MPC w1	1.0376	35.1531	7.3865
MPC w2	1.0376	17.6569	4.8612
MPC w3	1.1451	11.4905	4.6803

Table 3.14: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 4.

Controller Scenario 4	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	0.3958	20.4529	4.0448
RL-assisted MPC Case 2	0.5404	16.4373	3.4634
RL-assisted MPC Case 3	0.5350	33.8594	5.4398
RL-assisted MPC Case 4	0.5350	28.8931	4.906
MPC w1	0.5350	25.0831	4.8509
MPC w2	1.2164	13.5045	3.7859
MPC w3	0.8819	10.1737	3.6584

Table 3.15: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 5.

Controller Scenario 5	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	0.24	22.6566	3.9686
RL-assisted MPC Case 2	0.3571	18.5	3.3090
RL-assisted MPC Case 3	0.2083	22.3993	3.9515
RL-assisted MPC Case 4	0.2011	42.1801	6.7398
MPC w1	0.2473	40.5182	6.3254
MPC w2	0.3180	23.0532	3.7998
MPC w3	0.2160	17.1130	3.2572

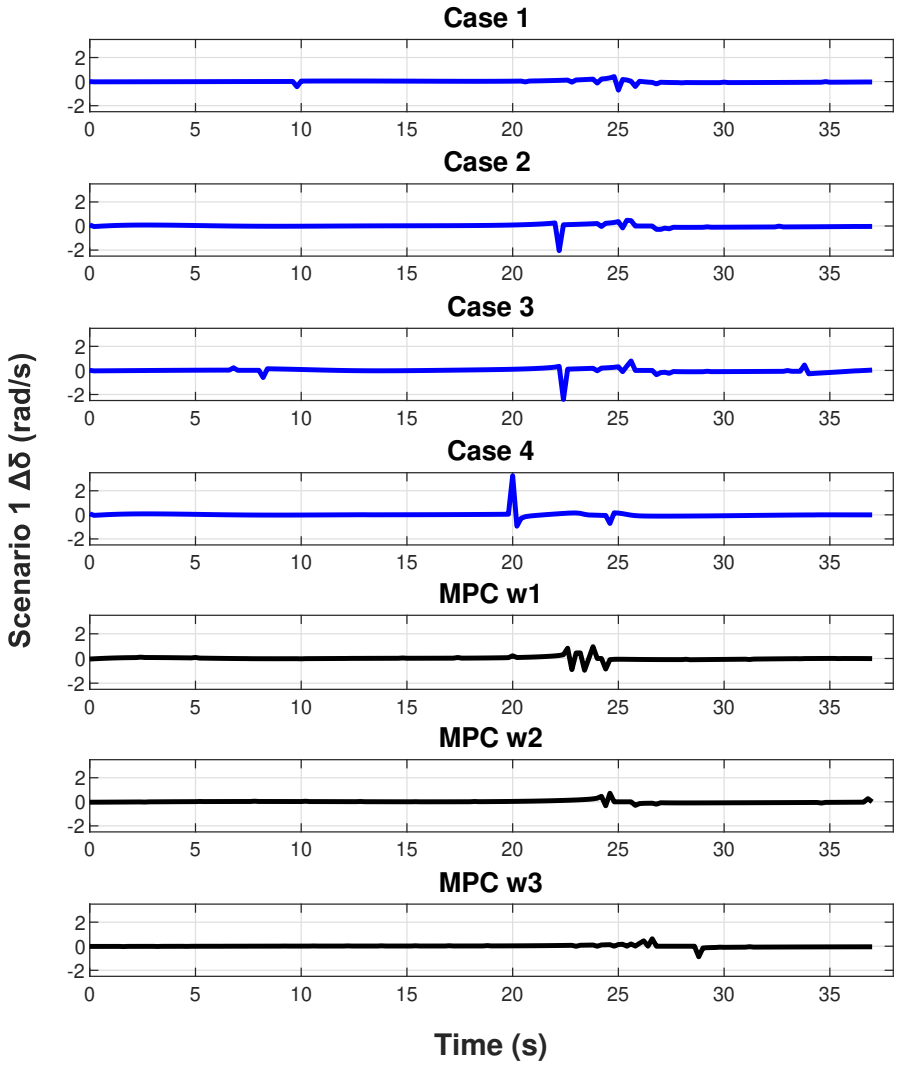

Figure 3.22: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

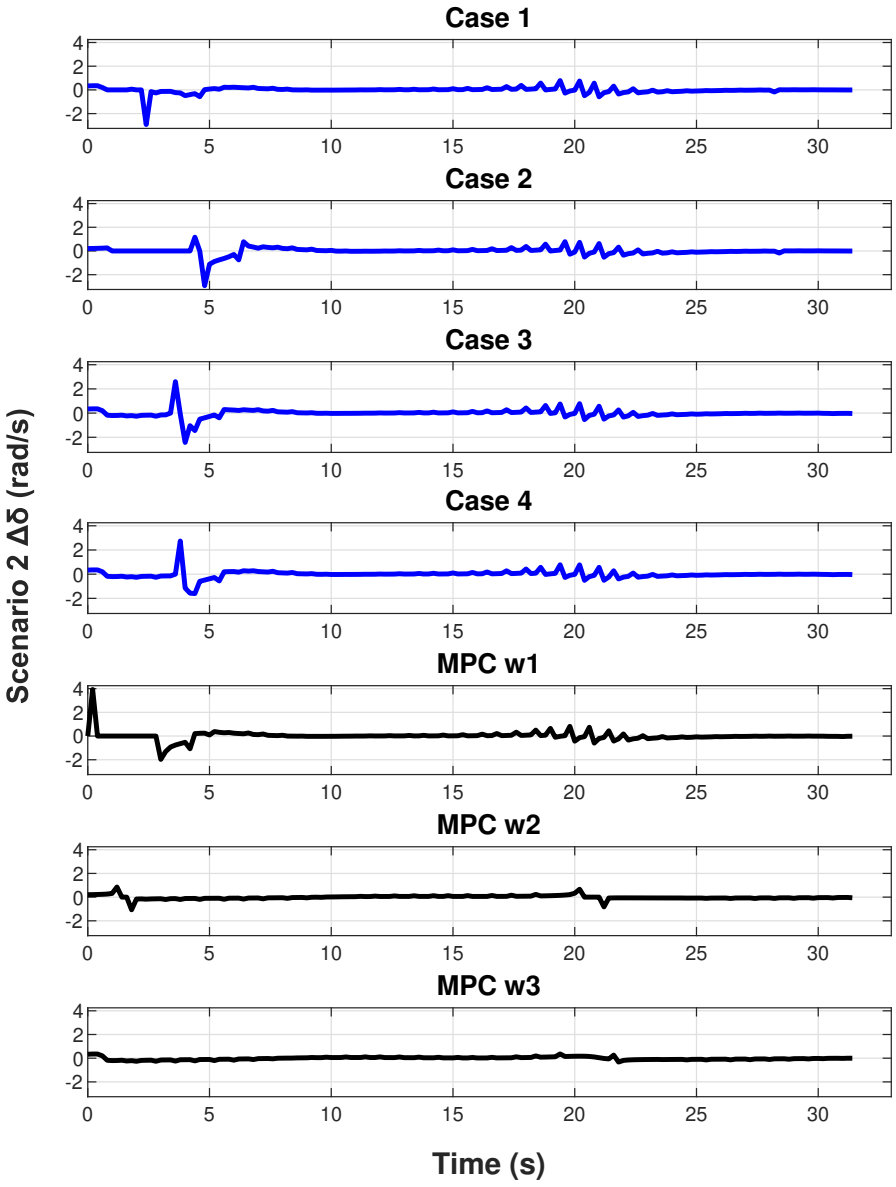

Figure 3.23: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

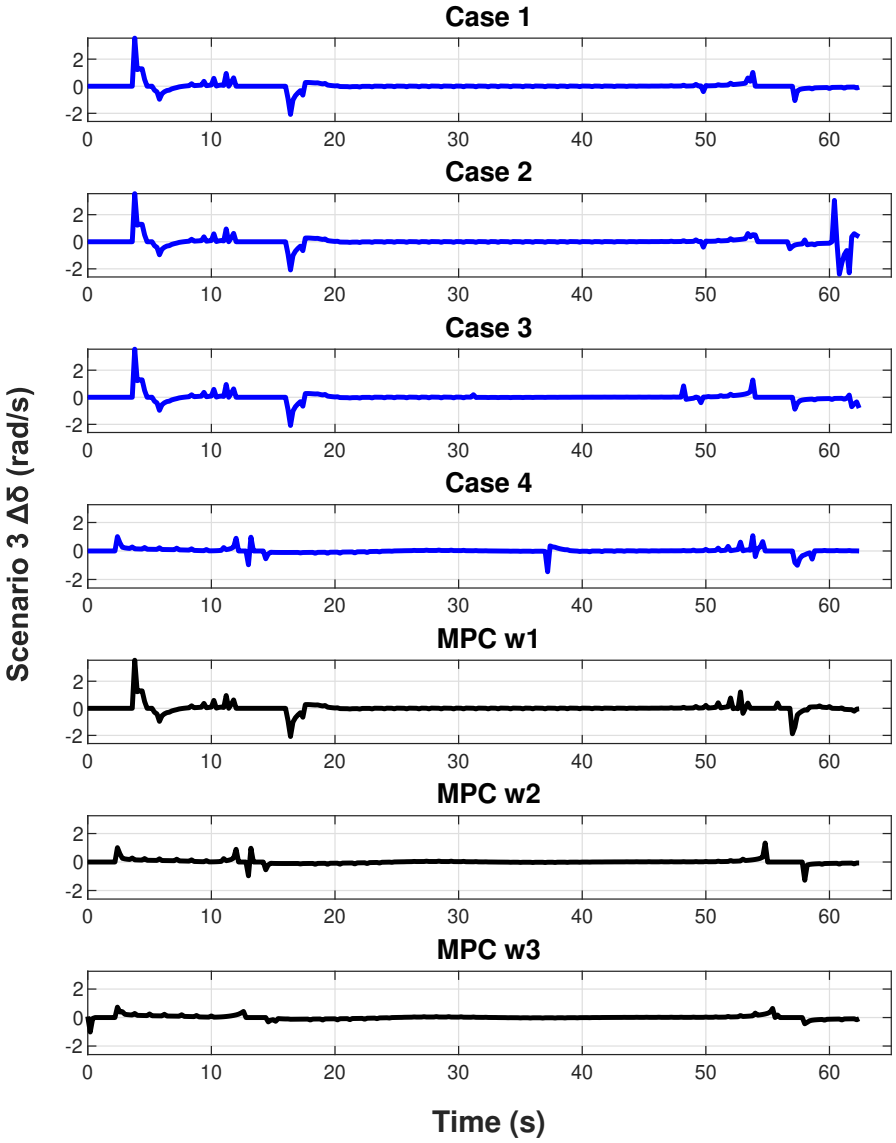

Figure 3.24: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

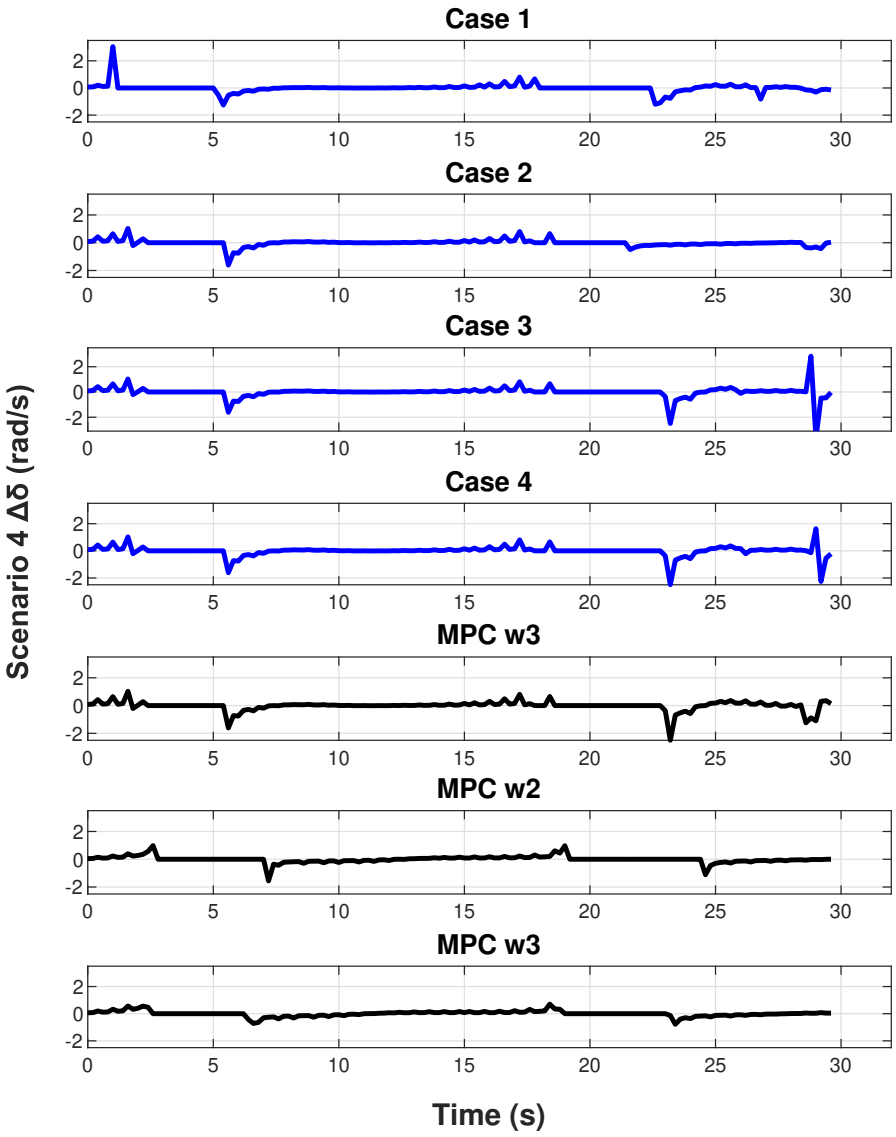

Figure 3.25: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

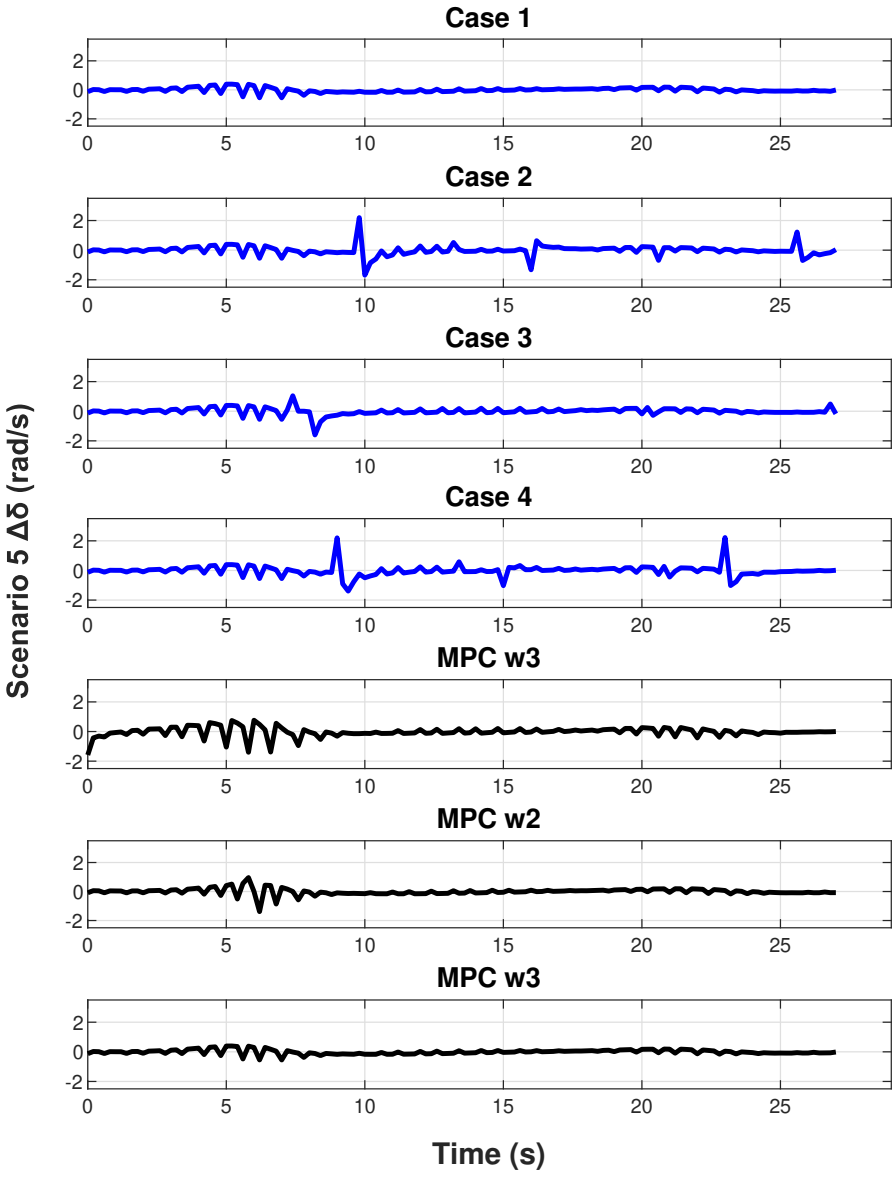

Figure 3.26: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

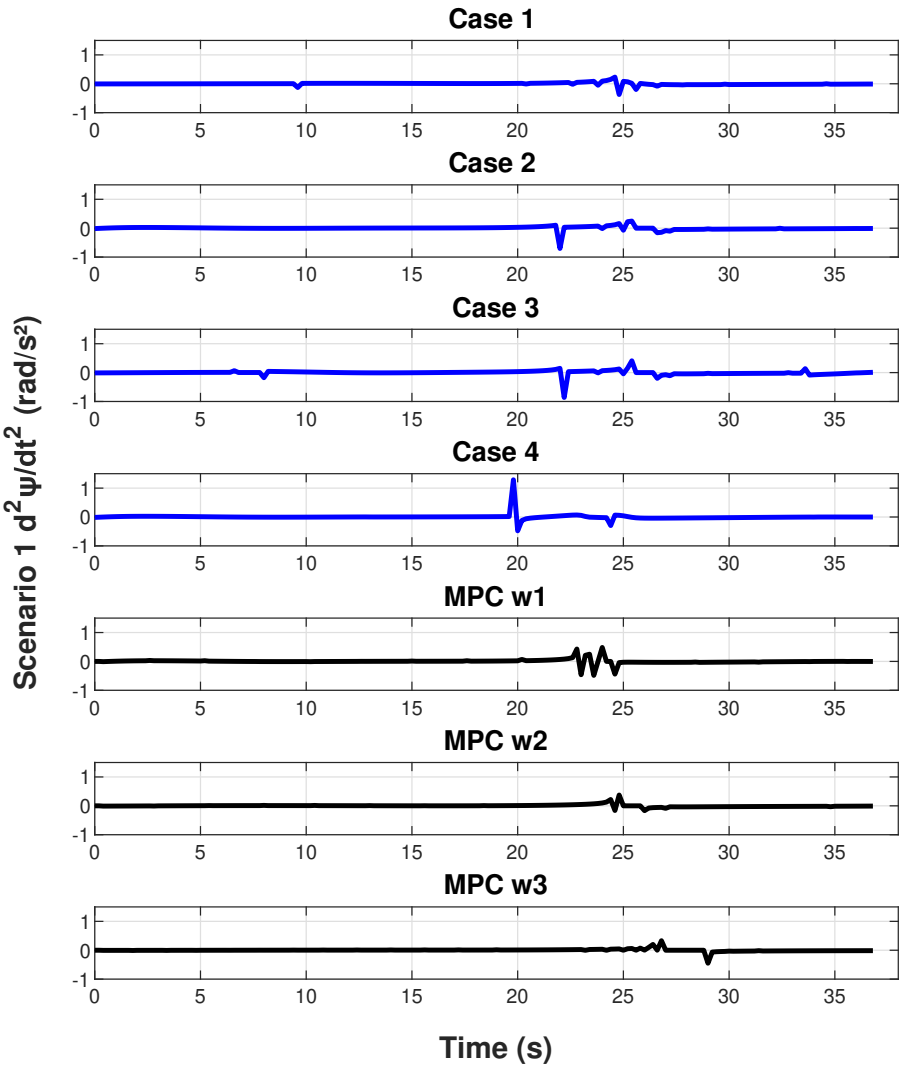

Figure 3.27: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

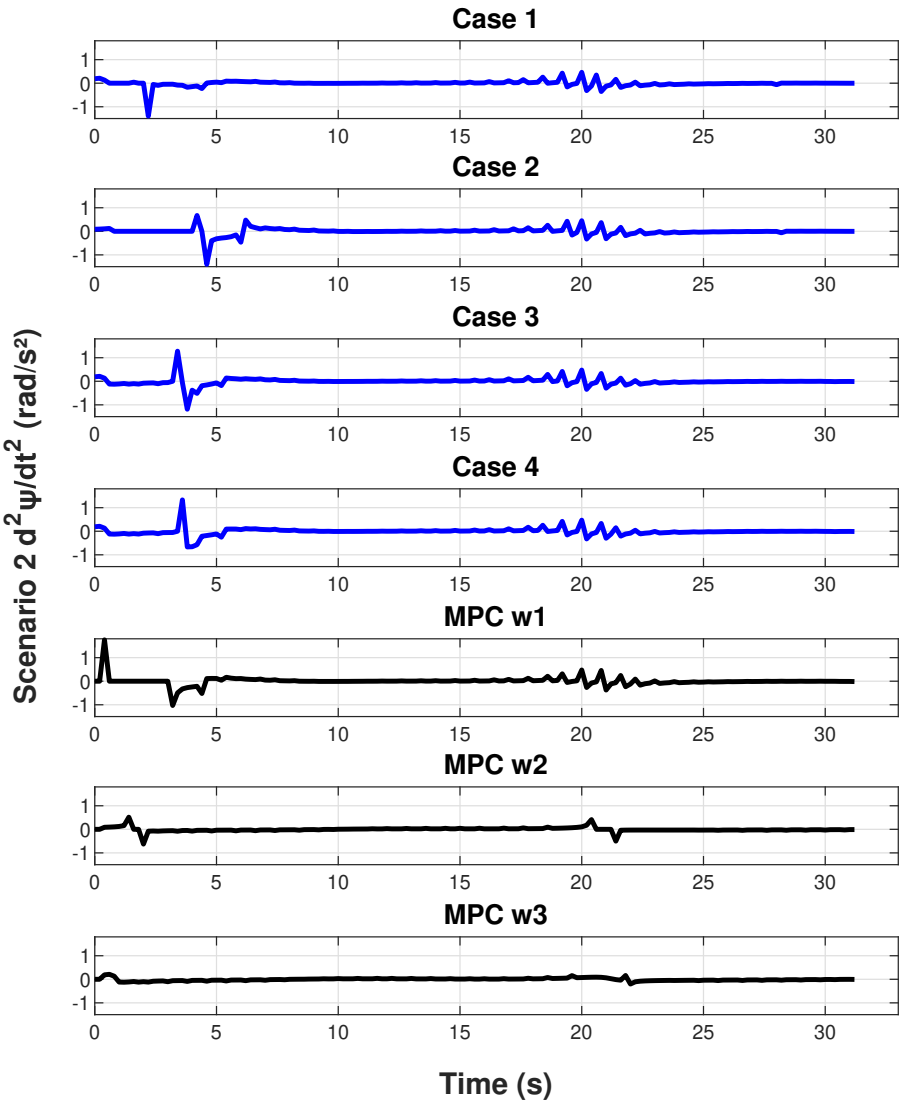

Figure 3.28: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

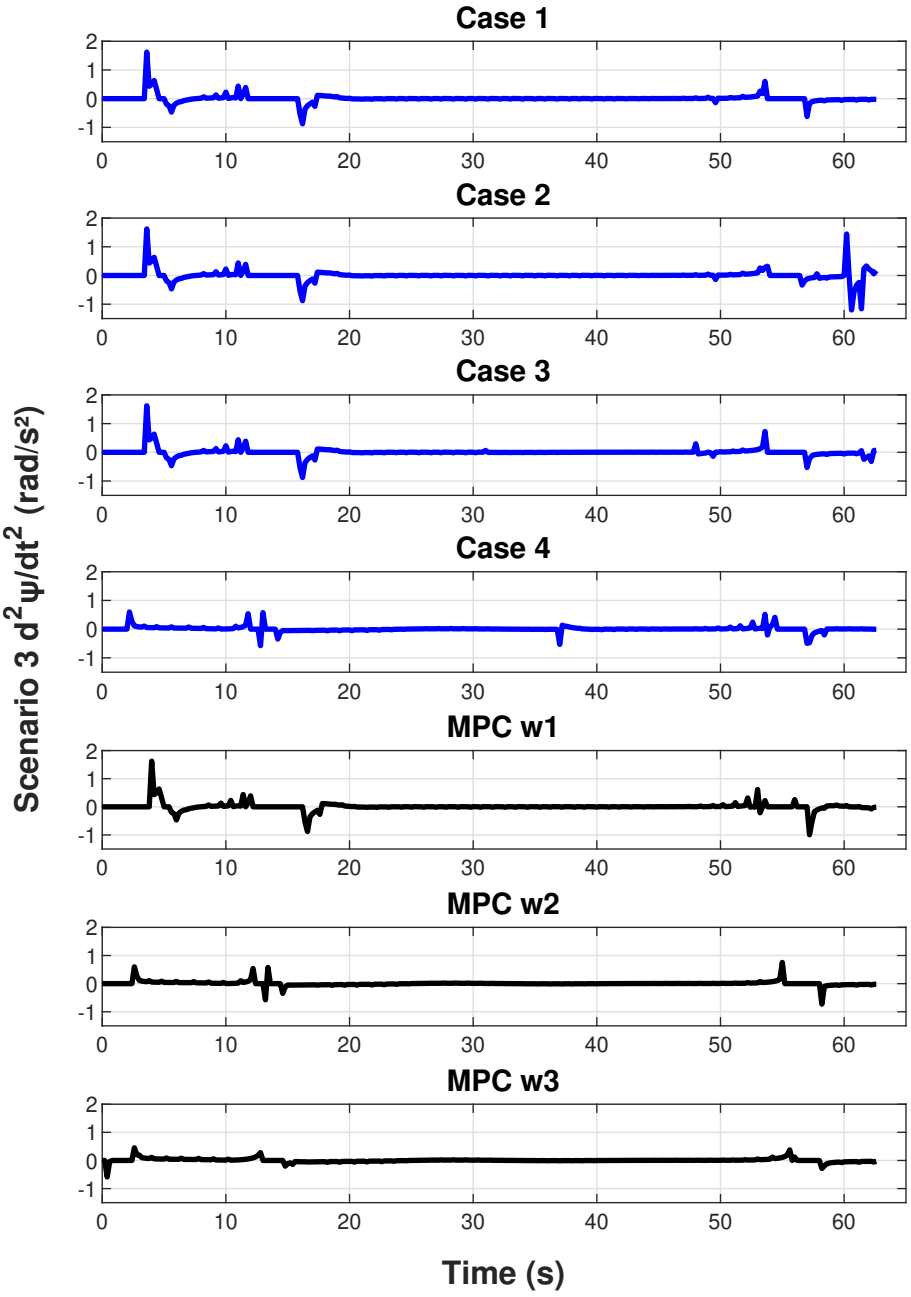

Figure 3.29: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

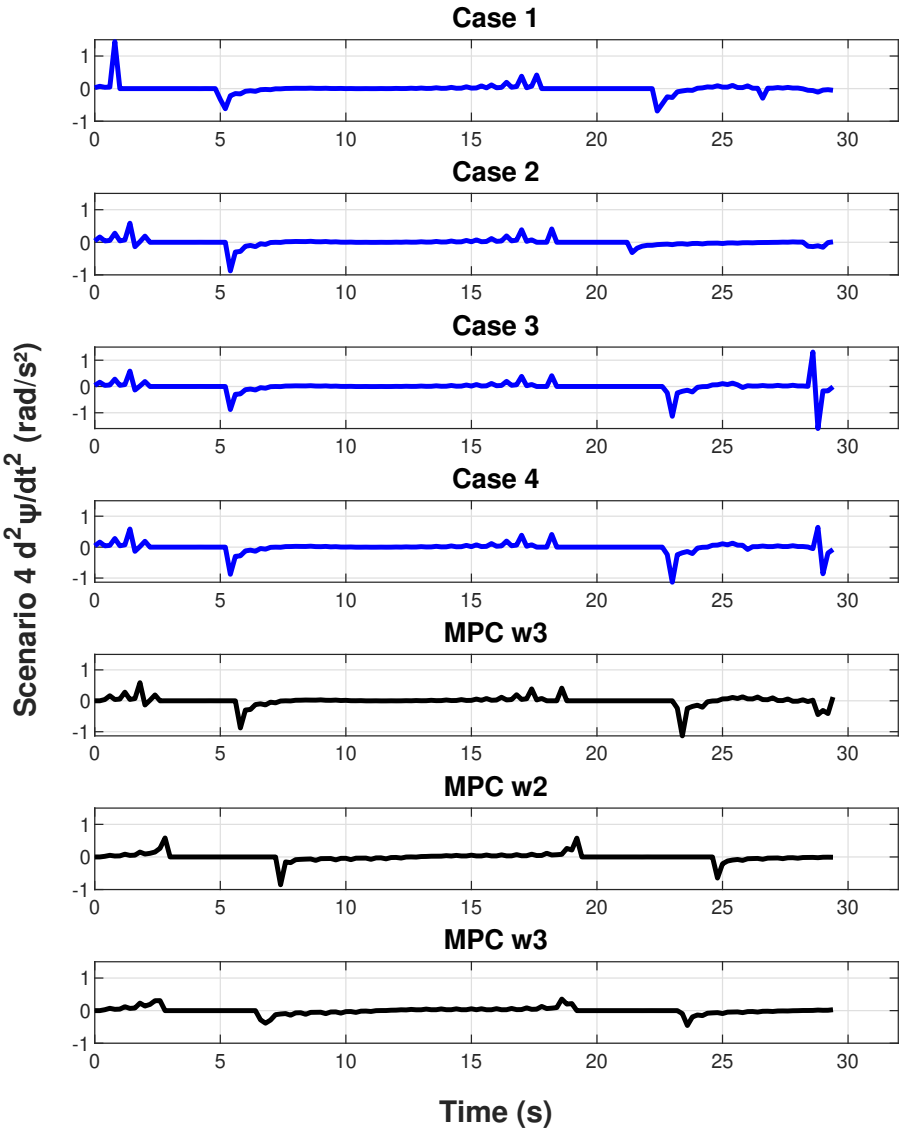

Figure 3.30: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

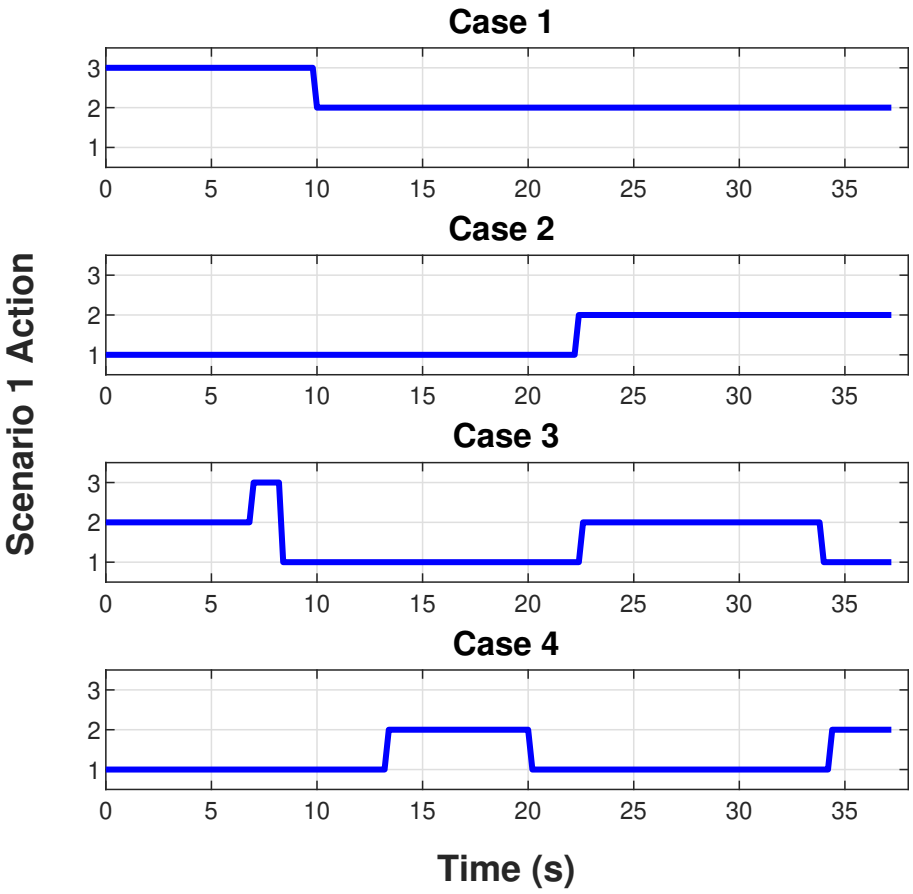

Figure 3.31: Action selection over time for the RL-assisted MPC Cases.

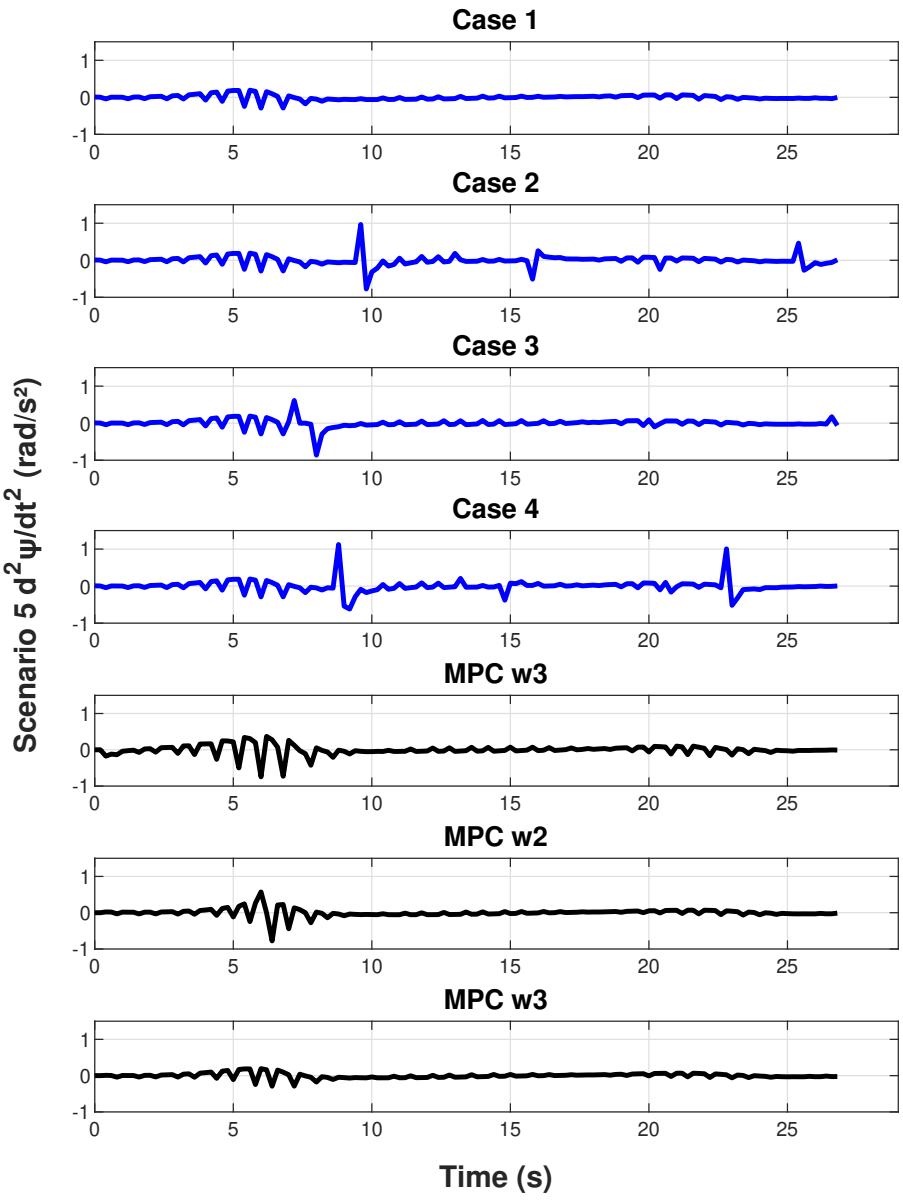

Figure 3.32: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

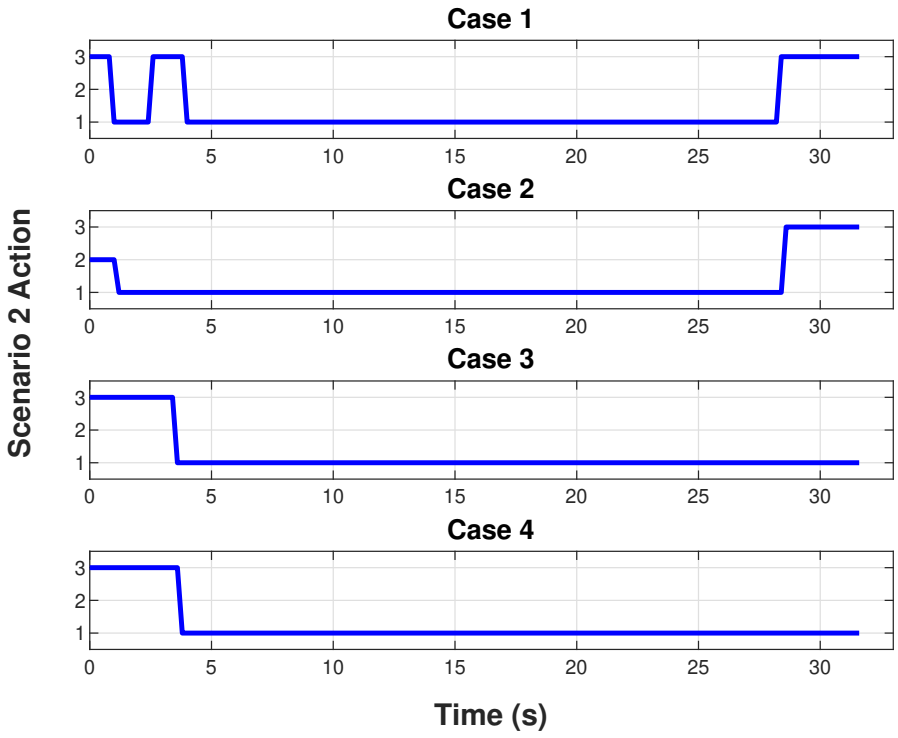

Figure 3.33: Action selection over time for the RL-assisted MPC Cases.

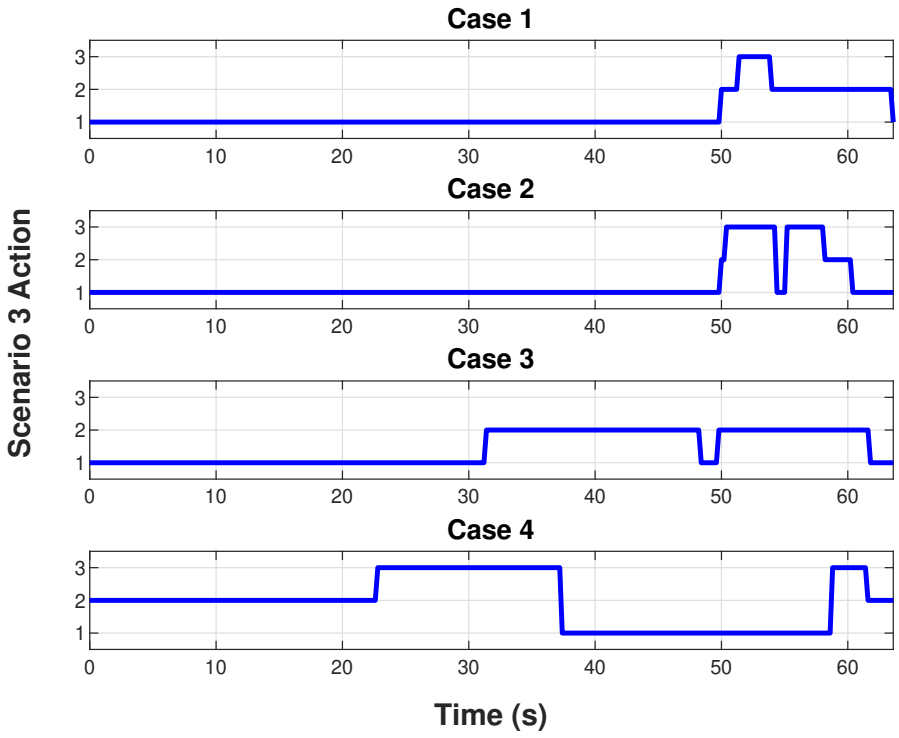

Figure 3.34: Action selection over time for the RL-assisted MPC Cases.

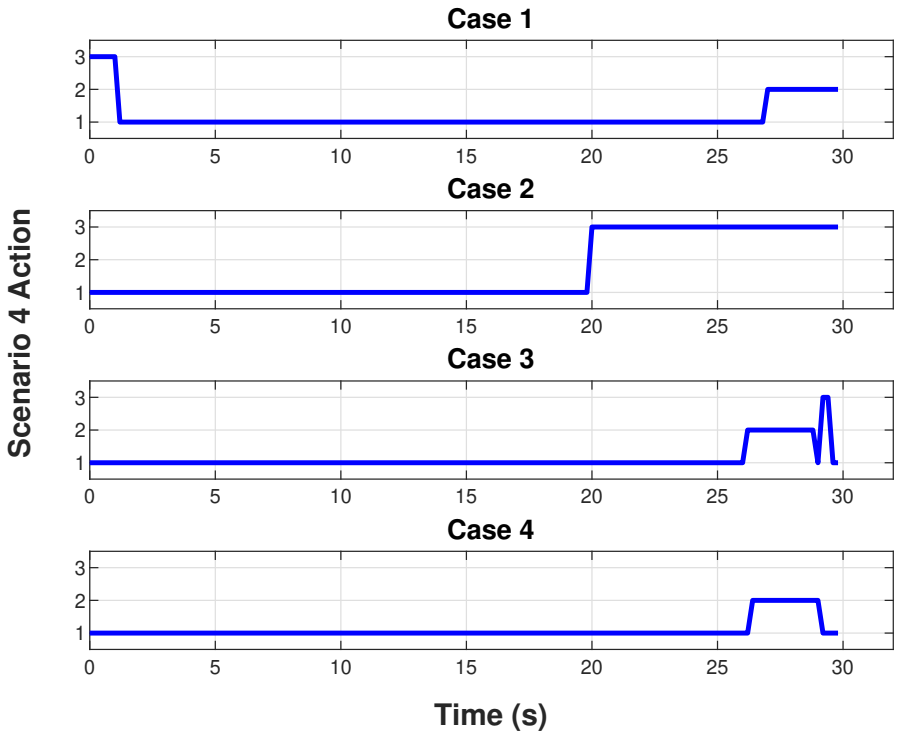

Figure 3.35: Action selection over time for the RL-assisted MPC Cases.

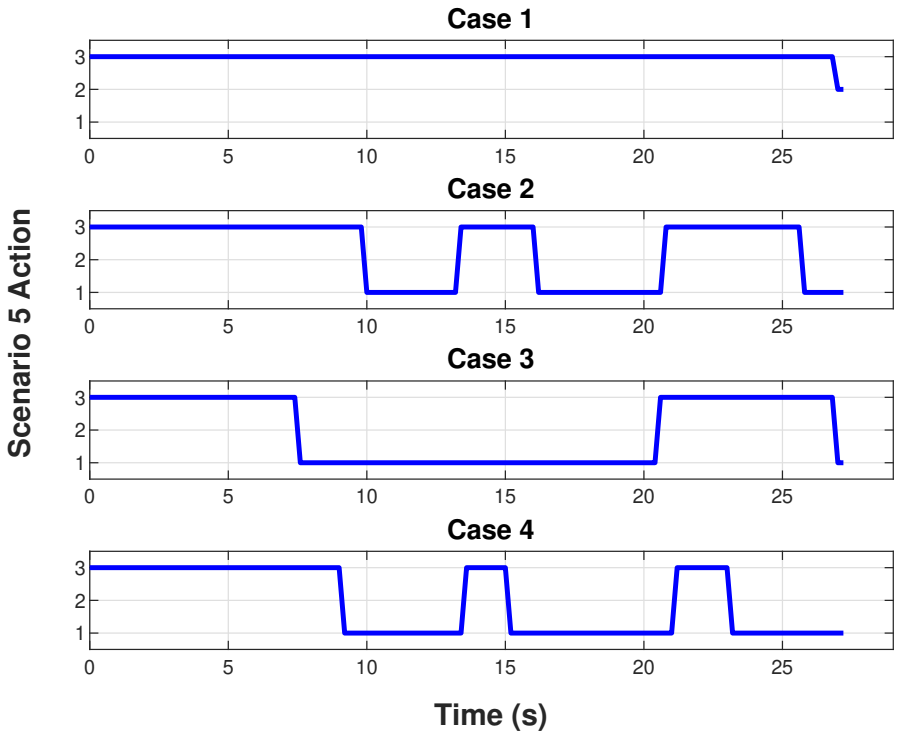

Figure 3.36: Action selection over time for the RL-assisted MPC Cases.

CHAPTER FOUR

Automated Parking Systems using Reinforcement Learning Assisted Model Predictive Control: Case Study II

The previous chapter demonstrated the effectiveness of the RL-assisted MPC framework under constant velocity conditions, where the vehicle maintained a fixed speed throughout the parking maneuver. While this simplified scenario provided valuable insights into the adaptive weight selection capabilities of the proposed controller, real-world parking situations often involve more complex velocity profiles. Drivers typically modulate their speed during parking maneuvers, accelerating in open areas and decelerating when approaching obstacles or making sharp turns. This chapter presents Case Study II, which extends the investigation to scenarios involving dynamic velocity profiles. Unlike Case Study I, where the vehicle speed remained constant, the AV in this case study travels with time-varying velocities that introduce additional complexity to the trajectory tracking problem. Although the MPC controller continues to optimize only the steering input δ, the varying longitudinal speed significantly affects the vehicle dynamics and creates new challenges for maintaining accurate lateral tracking while ensuring smooth steering behavior. The introduction of dynamic velocity profiles serves multiple purposes in validating the robustness of the RL-assisted MPC framework. First, it tests the controller's adaptability under more realistic operating conditions where speed variations can amplify or dampen the effects of steering inputs. Second, it evaluates whether the RL agent's weight selection strategy remains effective when the underlying vehicle dynamics become more complex due to velocity-dependent effects. Third, it provides insights into how the adaptive framework handles the coupling between longitudinal and lateral vehicle motion during low-speed parking maneuvers. The experimental setup maintains the same five parking scenarios used in Case Study I, allowing for direct comparison between

constant and variable speed conditions. However, the reward function weights and baseline MPC configurations are adjusted to account for the additional complexity introduced by the dynamic velocity profiles. This systematic approach enables a comprehensive assessment of how velocity variations impact the performance metrics of lateral deviation, steering smoothness, and heading stability that were established in the previous chapter.

4.0.1 RL-assisted MPC Control System Design

This case study employs the same RL-assisted MPC framework established in Chapter 3, including the identical MPC optimization formulation (equations (3.8)), reward function (equation (3.10)), and control algorithm 3.1. The key distinction in Case Study II lies in the vehicle operating conditions: whereas Chapter 3 investigated constant velocity scenarios, this case study examines parking maneuvers under dynamic velocity profiles where the vehicle speed varies throughout the trajectory. The introduction of time-varying speed creates additional complexity in the trajectory tracking problem without altering the fundamental control structure. The MPC controller continues to optimize only the steering input δ, while the varying longitudinal speed affects the vehicle's lateral dynamics and response characteristics. This velocity-dependent behavior requires the RL agent to adapt its weight selection strategy to maintain effective control performance under the changing dynamic conditions. To accommodate the increased complexity of the dynamic velocity profiles, the reward function weights α_l and α_h were systematically adjusted for each scenario and case, as detailed in Tables 4.1-4.5. Similarly, the baseline MPC weight configurations were modified to provide appropriate comparison benchmarks under the new operating conditions. These adjustments ensure that both the RL-assisted controller and fixed-weight baselines are properly tuned for the dynamic velocity environment, enabling fair performance comparisons. The experimental methodology remains

Table 4.1: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 1	α_l	α_h
Case 1	0.75	0.25
Case 2	0.78	0.22
Case 3	0.8	0.2
Case 4	0.82	0.18

Table 4.2: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 2	α_l	α_h
Case 1	0.65	0.35
Case 2	0.68	0.32
Case 3	0.7	0.3
Case 4	0.73	0.27

consistent with Chapter 3, utilizing the same five parking scenarios (Figures 3.2-3.6) and evaluation metrics to facilitate direct comparison between constant and variable speed conditions. This approach allows for systematic assessment of how velocity variations impact the adaptive weight selection capabilities and overall performance of the RL-assisted MPC framework.

Table 4.3: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 3	α_l	α_h
Case 1	0.7	0.3
Case 2	0.75	0.25
Case 3	0.8	0.2
Case 4	0.85	0.15

Table 4.4: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 4	α_l	α_h
Case 1	0.75	0.25
Case 2	0.78	0.22
Case 3	0.8	0.2
Case 4	0.83	0.17

Table 4.5: Reward weights used in RL-assisted MPC training cases.

RL-assisted MPC Scenario 5	α_l	α_h
Case 1	0.73	0.27
Case 2	0.75	0.25
Case 3	0.8	0.2
Case 4	0.82	0.18

4.1 Results

Similarly to chapter 3 and before presenting the quantitative comparisons, the actual vehicle trajectories for Scenarios 1–5, shown in Figures 4.6 – 4.10. The paths reveal that all controllers successfully completed the maneuvers, yet the RL-assisted MPC consistently traced the reference route with higher precision—most notably during sharp turns and final alignment. In contrast, MPC (w1–w3) tended to follow wider arcs and

Table 4.6: Weight sets used in MPC cost function for the three fixed-weight baseline controllers.

Weights Scenario 1	λ_t	λ_s	λ_d
$w1$	6	2	0
$w2$	2	2	8
$w3$	1	0	5

Table 4.7: Weight sets used in MPC cost function for the three fixed-weight baseline controllers.

Weights Scenario 2	λ_t	λ_s	λ_d
$w1$	1	0.1	0.1
$w2$	0.5	0.8	0.1
$w3$	0.2	0.15	1

Table 4.8: Weight sets used in MPC cost function for the three fixed-weight baseline controllers.

Weights Scenario 3	λ_t	λ_s	λ_d
$w1$	5	0.15	0.1
$w2$	2	3	0.1
$w3$	1	1	8

Table 4.9: Weight sets used in MPC cost function for the three fixed-weight baseline controllers.

Weights Scenario 4	λ_t	λ_s	λ_d
$w1$	1.2	0.1	0.1
$w2$	0.8	1	0.1
$w3$	0.2	0.1	1

Table 4.10: Weight sets used in MPC cost function for the three fixed-weight baseline controllers.

Weights Scenario 5	λ_t	λ_s	λ_d
$w1$	1.5	0.1	0.1
$w2$	0.9	0.7	0.2
$w3$	0.3	0.1	1

displayed greater lateral offset. These trajectory trends provide an intuitive preview of the performance differences that will be quantified and discussed in the following subsections.

4.1.1 Lateral Deviation Comparison

The RL-assisted MPC consistently achieved lateral deviations that were comparable to, and in several cases better than, fixed-weight baselines as shown in Figures (4.1-4.5). For instance, in scenario 1 (Table 4.12), Case 3 achieved a *dy.max* of 0.214 m, nearly identical to the best baseline W1, 0.204 m. Although other RL cases showed slightly higher deviations, this shows that the adaptive framework can reach baseline-level accuracy in this maneuver. The reason for not outperform it is to take in consideration a better softer steering angle which indeed achieved better results. In scenario 2 ,(Table 4.13), Case 1 obtained 0.712 m, which is lower than all baseline deviations 0.72–0.93 m. Furthermore, RL achieved the lowest total steering variation. In scenario 3, (Table 4.14), none of the RL cases fully surpassed the best baseline lateral deviation 1.03 m, with results ranging from 1.117–1.191 m. However, deviations remained within a narrow margin, indicating that the adaptive controller maintained stable tracking even under complex geometry, and further trade-offs in steering smoothness (further discussed in subsection 4.1.2) favored the RL controller. In scenario 4, (Table 4.15), Case 3 clearly outperformed all baselines, with a deviation of 0.482 m compared to 0.533–1.216 m. This shows the RL agents ability to reduce error by dynamically selecting more suitable weight sets. Finally, in scenario 5, (Table 4.16), Case 4 achieved the lowest *dy.max* across all baselines at 0.201 m, outperforming the best baseline W3 0.216 m. This result highlights the benefit of adaptive weighting for precise final positioning in tighter maneuvers.

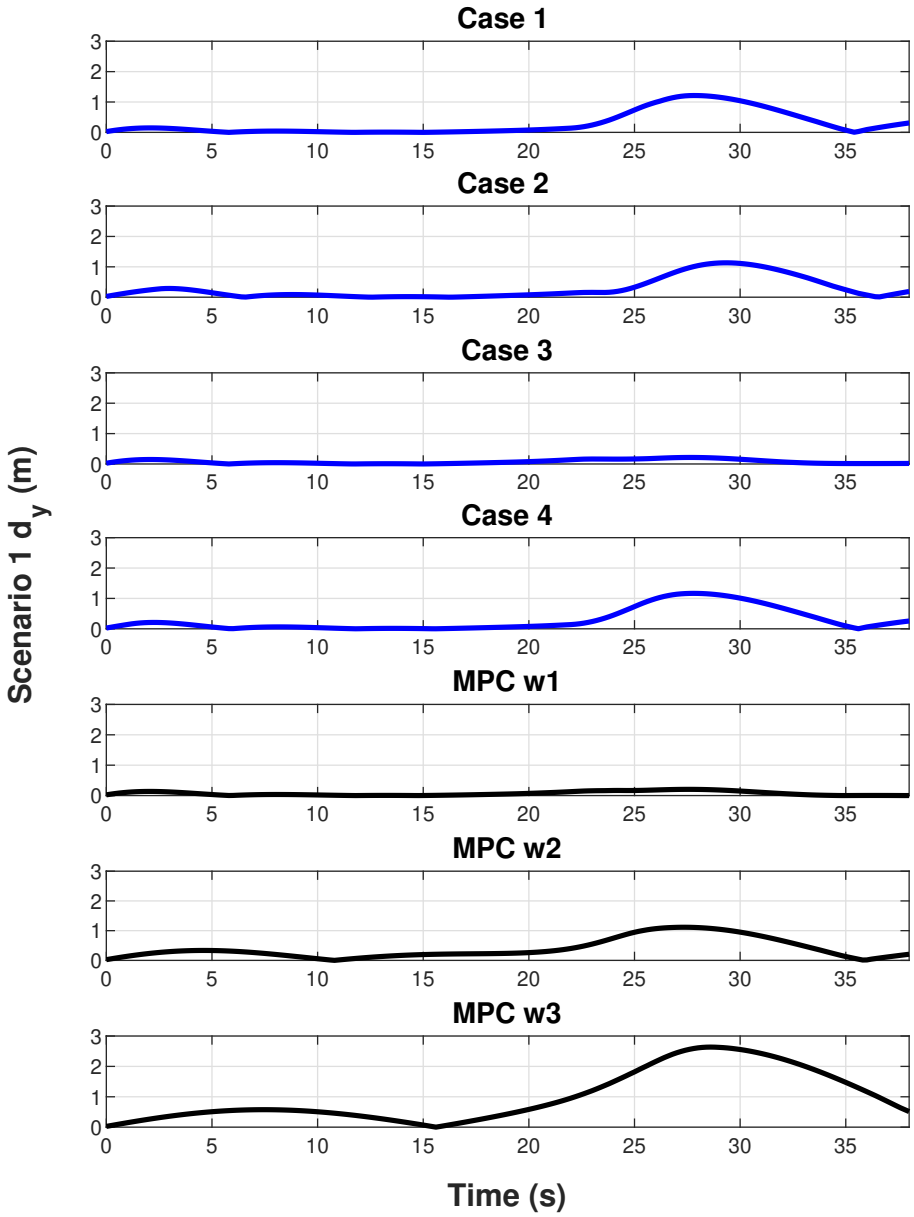

Figure 4.1: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

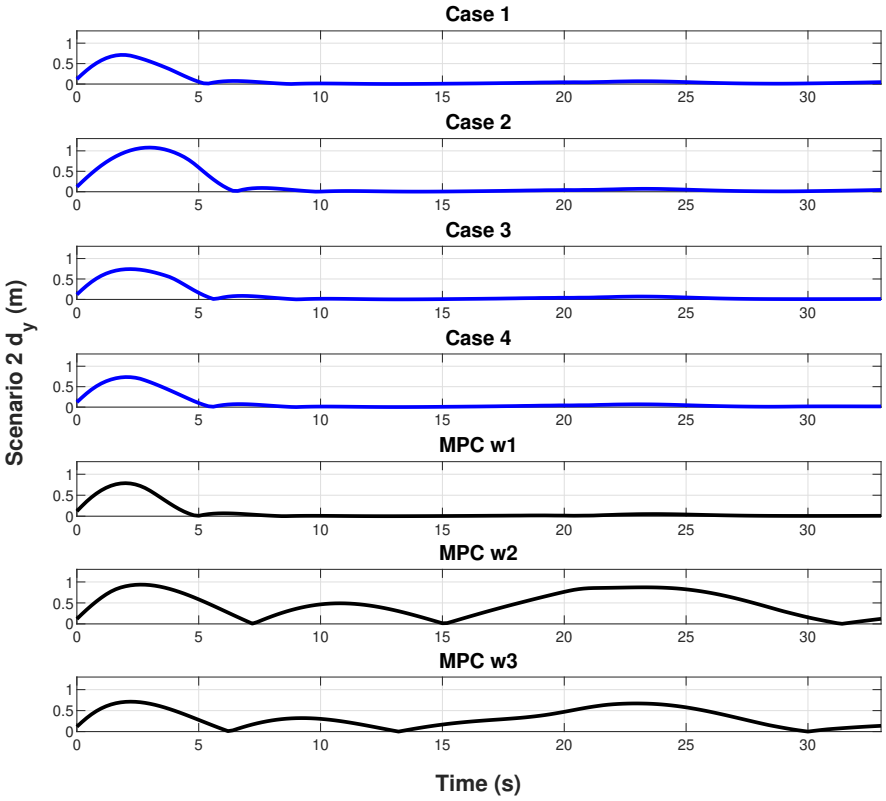

Figure 4.2: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

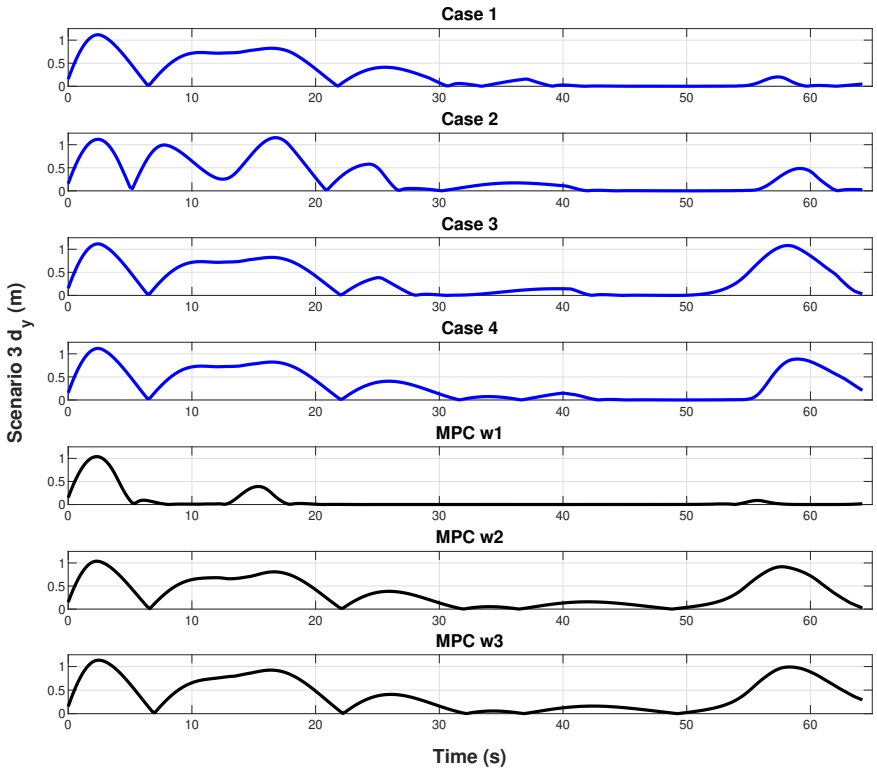

Figure 4.3: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

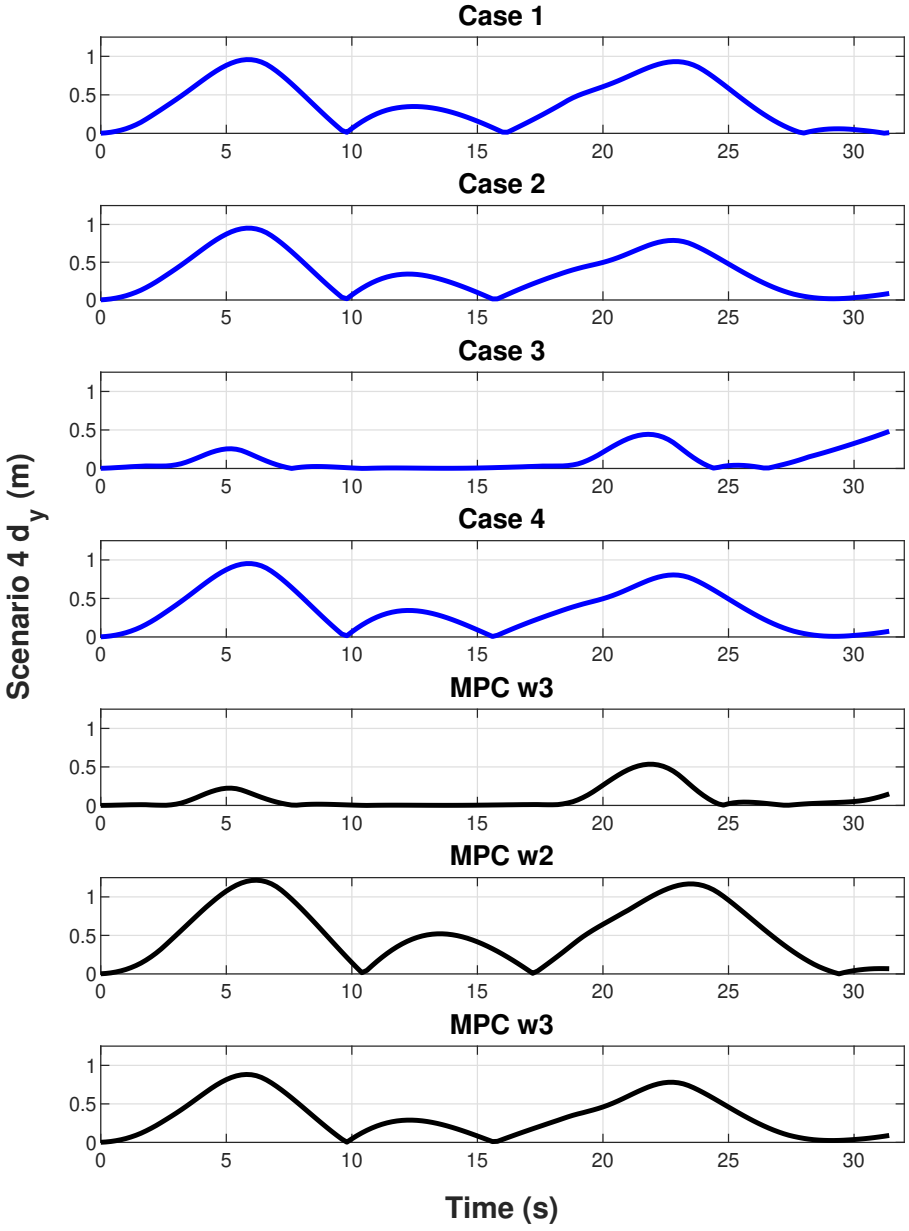

Figure 4.4: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

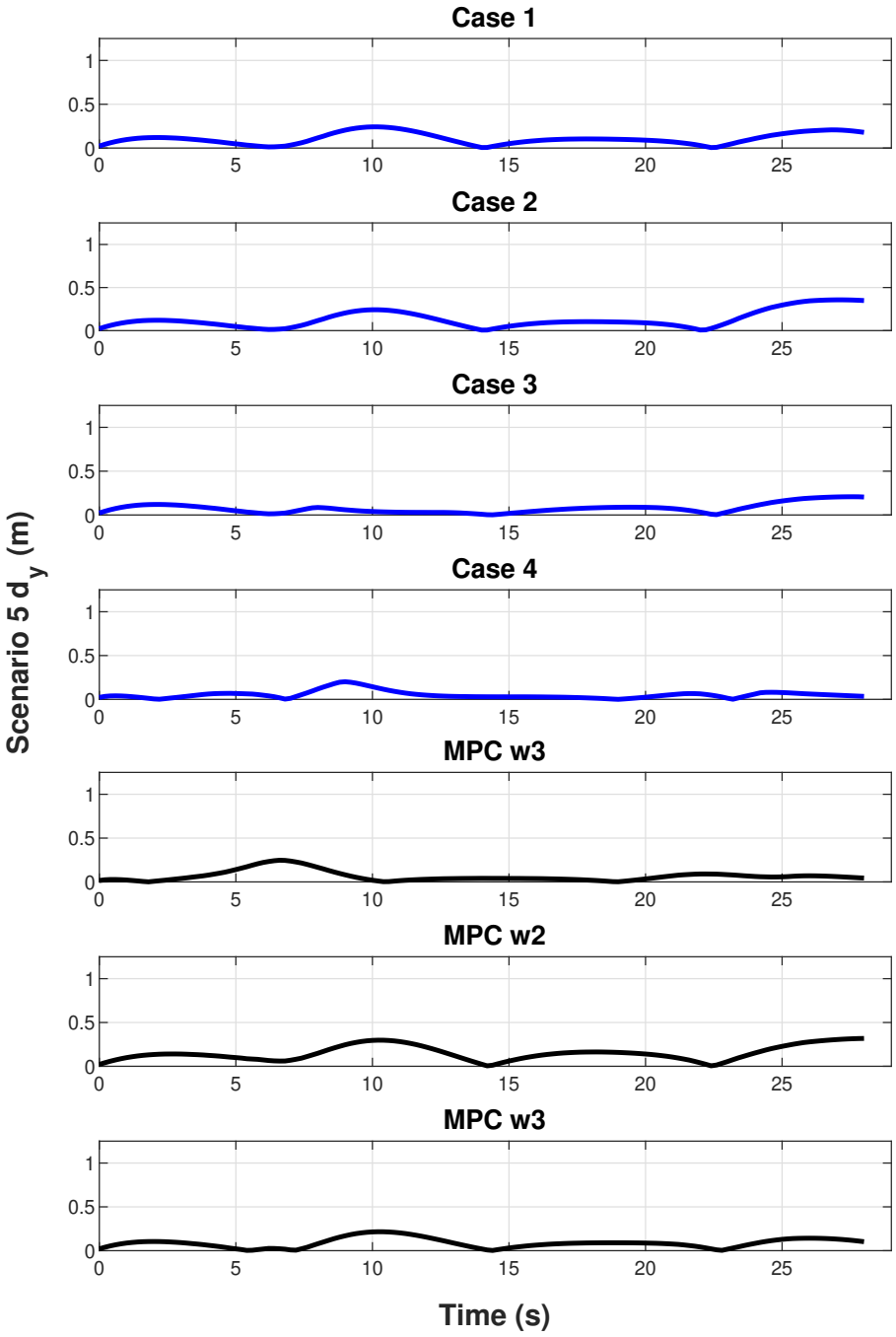

Figure 4.5: Lateral deviation d_y over time for RL-assisted MPC Cases and baseline controllers.

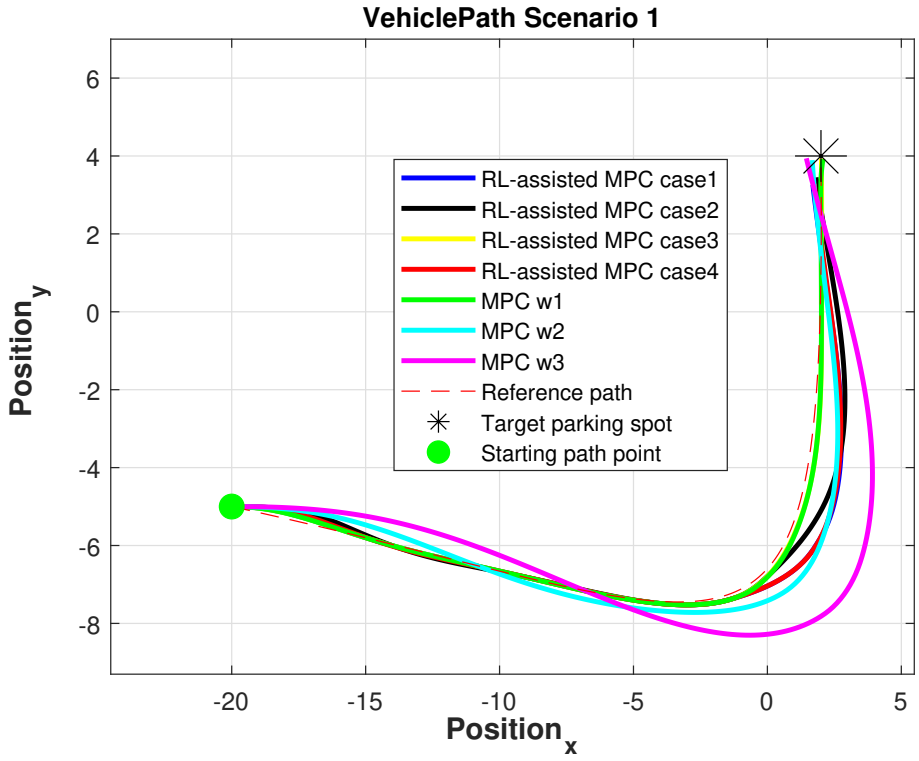

Figure 4.6: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

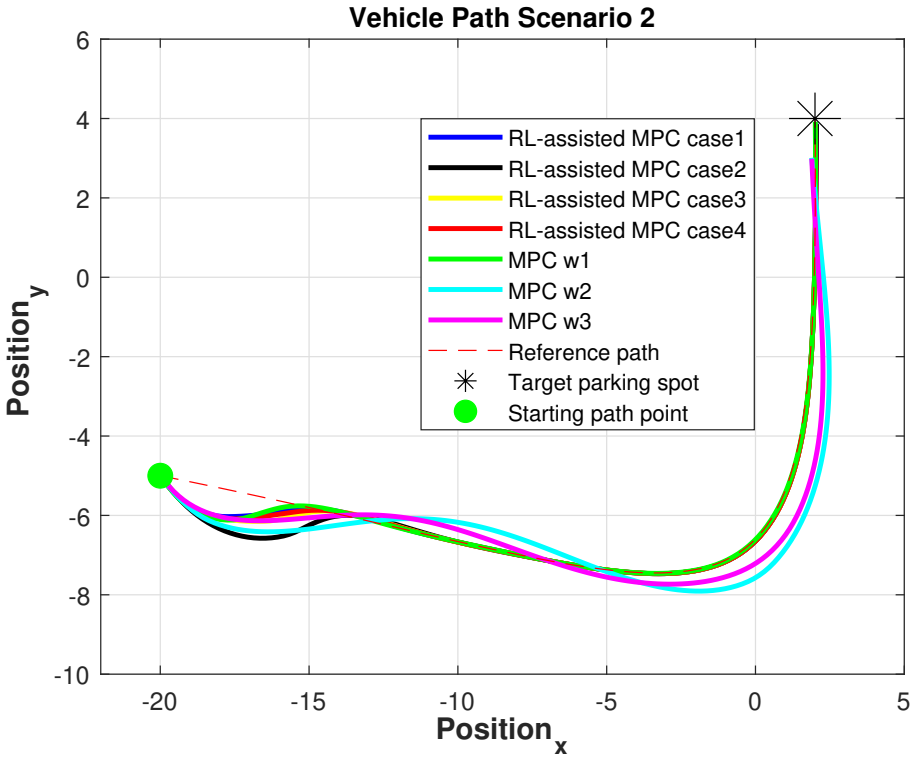

Figure 4.7: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

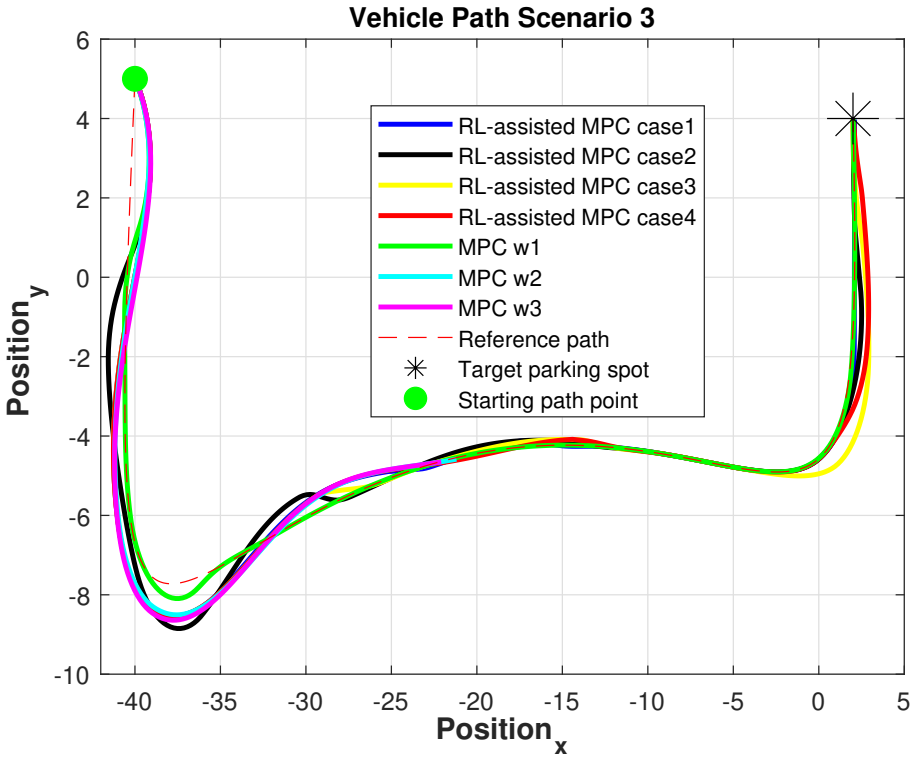

Figure 4.8: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

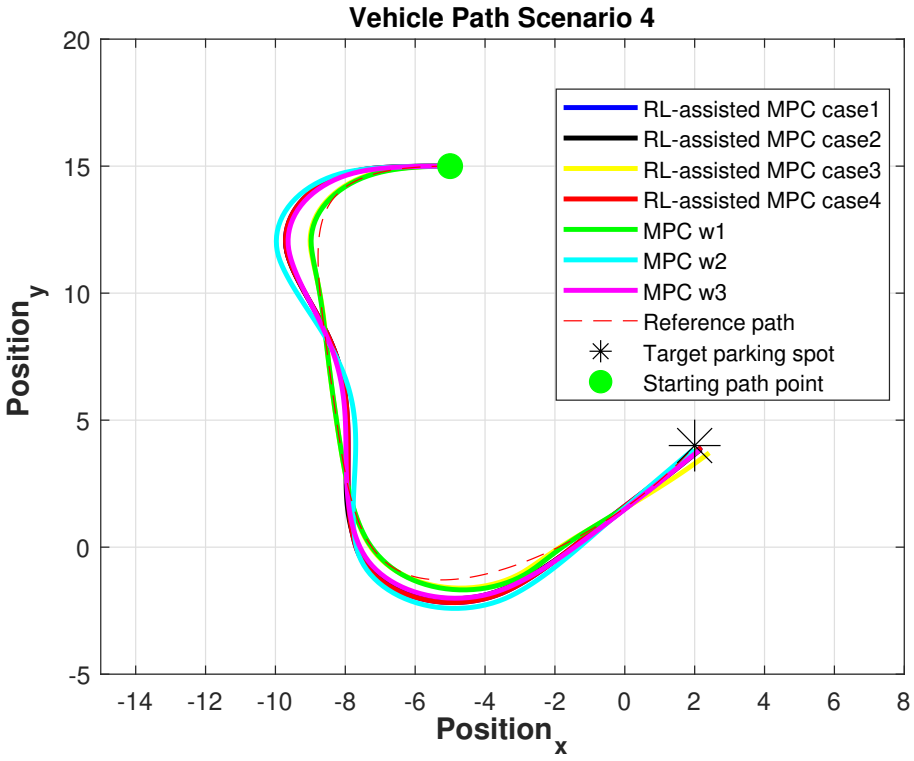

Figure 4.9: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

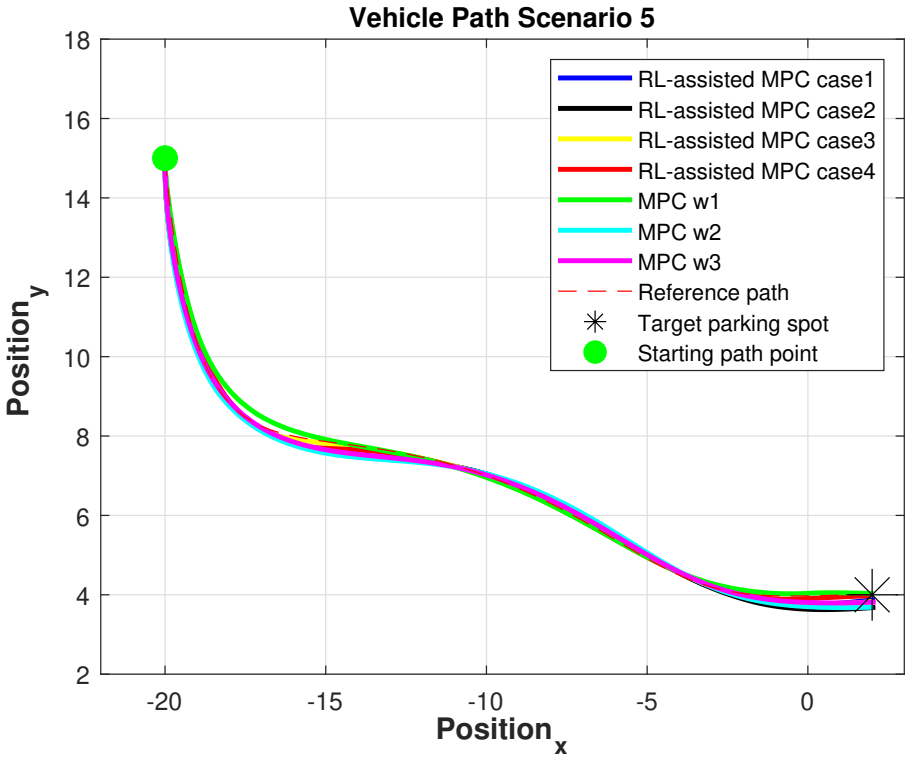

Figure 4.10: Vehicle path over time from starting to end for RL-assisted MPC Cases and baseline controllers.

4.1.2 Steering Effort and Smoothness

Steering effort and smoothness were evaluated using the total variation in steering input $TV - \delta$ and steering rate variation $TV - \Delta\delta$. These two metrics reflect the stability of the vehicle steering. In each scenario, at least one RL case produced smoother steering than the static baselines, even when not always yielding the lowest lateral error. In Scenario 1, (Table 4.12), Case 3 yielded the lowest $TV - \delta$ 9.856 among the RL cases, outperforming the best baseline W1 10.95 but with reduced oscillations visible in Fig. 4.11. Although baseline W3 showed lower steering variability, it did so at the cost of a much larger lateral deviation 2.63 m, demonstrating an unfavorable trade-off. The RL framework, especially Case 3, achieved a more balanced compromise between steering smoothness and tracking accuracy. In Scenario 2, (Table 4.13), Case 1 achieved the lowest $TV - \delta$ (4.72), outperforming baseline W1 5.08, while baselines W2 and W3, simultaneously offering lower deviation than all the RL-assisted MPC cases. This confirms that the RL agent can minimize steering variability without sacrificing lateral performance. In scenario 3, (Table 4.14), despite slightly higher lateral errors, RL Case 2 produced a moderate steering effort $TV - \delta$ of 11.44 compared to baselines that either increased variation baseline W1 6.99 or reduced it excessively at the cost of responsiveness baseline W3 4.73. Figures 4.11-4.15 illustrate how RL-assisted MPC controller smoothed steering transitions, reducing abrupt spikes that were present in baseline controller. Furthermore, Case 2 in scenario 4 achieved 3.79 in $TV - \delta$, nearly identical to the best-performing baseline W3 3.67, while keeping $TV - \Delta\delta$ lower than most baselines. Showing that the RL agent adapts well to smooth trajectories while preserving easy controllability as shown in Table 4.15. In scenario 5, Table 4.16, Case 2 offered the most favorable trade-off, with $TV - \delta$ 3.31 compared to baselines ranging from 3.26–6.33. While baseline W3 had a marginally lower value 3.26, the RL-assisted

MPC controller achieved smoother steering without the high variability in steering rate seen in static configurations.

4.1.3 Heading Stability

Heading stability was evaluated using heading acceleration $d^2\psi$ illustrated in Figures (4.16-4.21), which captures the smoothness of directional changes. Large oscillations in this metric indicate abrupt or unstable maneuvers, while lower and more consistent values correspond to smoother, more comfortable trajectories. In most scenarios, at least one RL case exhibited reduced heading jerk compared to static baselines. In scenario 1, Fig.4.16, Case 3 achieved both low lateral deviation and favorable steering smoothness, also demonstrated stable heading behavior. Oscillations were limited and more damped compared to baseline W1 (the best in lateral deviation) , confirming the RL agent's balanced adaptation in this maneuver. Its worth noting that even though baseline W2 and W3 have lower margins, the oscillations in both lasted longer and also both suffer from high lateral error.

In scenario 2, Fig.4.17, Case 1 outperformed baselines in both lateral deviation $d_{y.max}$ and steering variation, exhibited the smoothest heading profile. Heading acceleration spikes were smaller and less frequent than in baselines, supporting the conclusion that Case 1 provides the most stable and low lateral error in this scenario.

In scenario 3, Fig.4.18, even though RL Case 2 was selected for smoother steering effort despite slightly higher lateral deviation, its heading acceleration remained more stable than baseline W1 and baseline W2. While baseline W3 displayed lower oscillations, it came at the cost of poor lateral performance, reinforcing that the RL case offers a better overall trade-off.

In scenario 4, Fig.4.19, Case 3, the best performer in lateral deviation, showed stronger oscillations in heading compared to Case 2. However, Case 2 which already

Table 4.11: Maximum heading jerk values for each scenario, comparing RL-assisted MPC with baseline MPC controllers

Controller	1	2	3	4	5
RL-assisted MPC Case 1	1.4912	1.2441	1.5397	0.9183	0.2928
RL-assisted MPC Case 2	0.9753	1.2334	1.7412	0.5969	0.2842
RL-assisted MPC Case 3	0.4400	1.3450	1.7239	1.0037	0.4889
RL-assisted MPC Case 4	0.5501	1.6626	1.7375	0.9183	1.1005
MPC w1	1.3588	1.7544	1.2183	1.1344	0.7453
MPC w2	1.1930	0.6261	0.7174	0.8529	0.7765
MPC w3	1.1930	0.2103	0.5950	0.4559	0.2908

offered competitive steering smoothness, produced a notably smoother heading profile with fewer abrupt spikes. This shows how different RL-assisted MPC cases can complement each other, balancing accuracy and stability.

Finally, scenario 5, Fig. 4.21, Case 4, which achieved the lowest lateral deviation overall, also maintained heading stability with relatively smooth transitions. While Case 2 was competitive in steering effort, Case 4's profile combined accurate positioning with stable heading, making it the most favorable trade-off in this maneuver.

Additionally, since the vehicle speed varies during the maneuver, the heading jerk was evaluated to assess ride comfort. The maximum values, summarized in Table 4.11, confirm that the RL-assisted MPC consistently outperforms the baseline controllers across all five scenarios.

4.1.4 Policy Adoption and Action Selection

The action-selection patterns of the RL agent are illustrated in Figures (4.20-4.25), where each action corresponds to one of the three baseline MPC weight sets (W1,W2 and W3). These temporal distributions provide insight into how the agent adapts its control

strategy throughout the different phases of the parking maneuver. In scenario 1, Fig. 4.20, Case 3, which achieved both accurate lateral tracking and stable steering, alternated primarily between baseline W1 and baseline W2 during alignment, while reducing reliance on baseline W3 until the last phase close to parking. This reflects a preference for weight sets that balance deviation control with smooth steering.

In scenario 2, Fig. 4.22, Case 1 (the best performer in both deviation and steering effort) showed a dominant preference for smoother weight sets during turning and alignment. The reduced switching frequency indicates that the agent converged to a stable control strategy, which contributed to its improved heading stability.

In scenario 3, Fig. 4.23, Case 2 (which balanced steering effort against slightly higher deviation), exhibited more frequent switching between weight sets. This behavior suggests that in more complex trajectories, the agent leveraged dynamic adaptation to correct path errors, even if it introduced some variability in heading.

In scenario 4, Fig. 4.24, Case 2 demonstrated consistent selection of W3 throughout most of the maneuver, contributing to smoother steering and heading stability. Case 3, while achieving the lowest lateral deviation, involved more frequent shifts towards the end, highlighting the trade-off between tracking precision and control stability.

Finally, scenario 5, Fig.4.25, case 4, which produced the best lateral accuracy, displayed a more dynamic switching pattern between weight sets, particularly in the final positioning phase. This reflects the agent's emphasis on minimizing deviation at the expense of increased steering variability.

4.1.5 Overall Assessment

Overall, in terms of lateral deviation, the RL-assisted MPC achieved better or close performance in all five scenarios. Even in Scenario 3, where fixed-weight baselines were slightly more accurate in lateral tracking, the RL framework remained competitive

and later demonstrated advantages in steering smoothness and heading stability. Across scenarios, the RL-assisted MPC produced smoother and more stable steering profiles, which emphasized heading stability in the reward function. Importantly, when static baselines minimized steering variation, they often did so at the expense of lateral deviation. By contrast, the RL-assisted MPC achieved smoother steering while preserving competitive accuracy, offering a superior balance between comfort and tracking. Across all five scenarios, the RL-assisted MPC consistently improved heading stability relative to fixed weight baselines. These results confirm that adaptive weight selection not only enhances lateral tracking and steering smoothness but also contributes significantly to stable and predictable heading dynamics. The action-selection results confirm that the RL agent adapts its weight choice according to the demands of each maneuver phase. Scenarios where smoother steering and heading stability were prioritized showed more stable weight selection with limited switching, whereas cases prioritizing lateral accuracy involved more frequent adaptation across weight sets. These findings demonstrate that reward design directly shapes policy behavior, and that the agent can flexibly modulate its strategy to balance accuracy, comfort, and stability in real time.

In summary, the proposed RL-assisted MPC framework improves the robustness and generalization of automated parking controllers by balancing competing objectives without manual tuning.

4.1.6 Validation under Unseen-starting Position

In this section, we evaluate the generalization capability of the proposed framework by deploying the vehicle from new starting positions that were not encountered during training. The objective is to assess whether the trained model can effectively control the vehicle and guide it to the designated parking spot under unseen initial conditions. Scenarios 4 and 5 are selected for this validation task, with each scenario

Table 4.12: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 1.

Controller Scenario 1	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	1.2131	22.4619	25.0362
RL-assisted MPC Case 2	1.1317	23.3324	21.2433
RL-assisted MPC Case 3	0.2145	12.0842	9.8560
RL-assisted MPC Case 4	1.1662	17.9587	13.2100
MPC w1	0.2043	13.5871	10.9559
MPC w2	1.1145	9.9164	4.5771
MPC w3	2.6365	8.6211	5.8627

Table 4.13: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 2.

Controller Scenario 2	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	0.7122	27.7678	4.7233
RL-assisted MPC Case 2	1.0804	21.4974	4.8551
RL-assisted MPC Case 3	0.7422	25.3203	5.4350
RL-assisted MPC Case 4	0.7350	23.3850	4.4254
MPC w1	0.7874	27.6094	5.0899
MPC w2	0.9358	9.5485	3.3067
MPC w3	0.7230	6.1340	3.0303

Table 4.14: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 3.

Controller Scenario 3	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	1.1524	58.2192	6.1582
RL-assisted MPC Case 2	1.1175	29.1807	11.4425
RL-assisted MPC Case 3	1.1910	62.5398	9.6677
RL-assisted MPC Case 4	1.1175	62.5398	8.3953
MPC w1	1.0376	32.3245	6.9936
MPC w2	1.0376	20.1160	4.9584
MPC w3	1.1354	11.0020	4.7309

Table 4.15: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 4.

Controller Scenario 4	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	0.9577	14.9410	3.8390
RL-assisted MPC Case 2	0.9507	12.3650	3.7927
RL-assisted MPC Case 3	0.4816	19.1405	3.9413
RL-assisted MPC Case 4	0.9577	14.9410	3.8390
MPC w1	0.5326	25.3487	4.7574
MPC w2	1.2164	13.5047	3.8005
MPC w3	0.8819	10.2370	3.6680

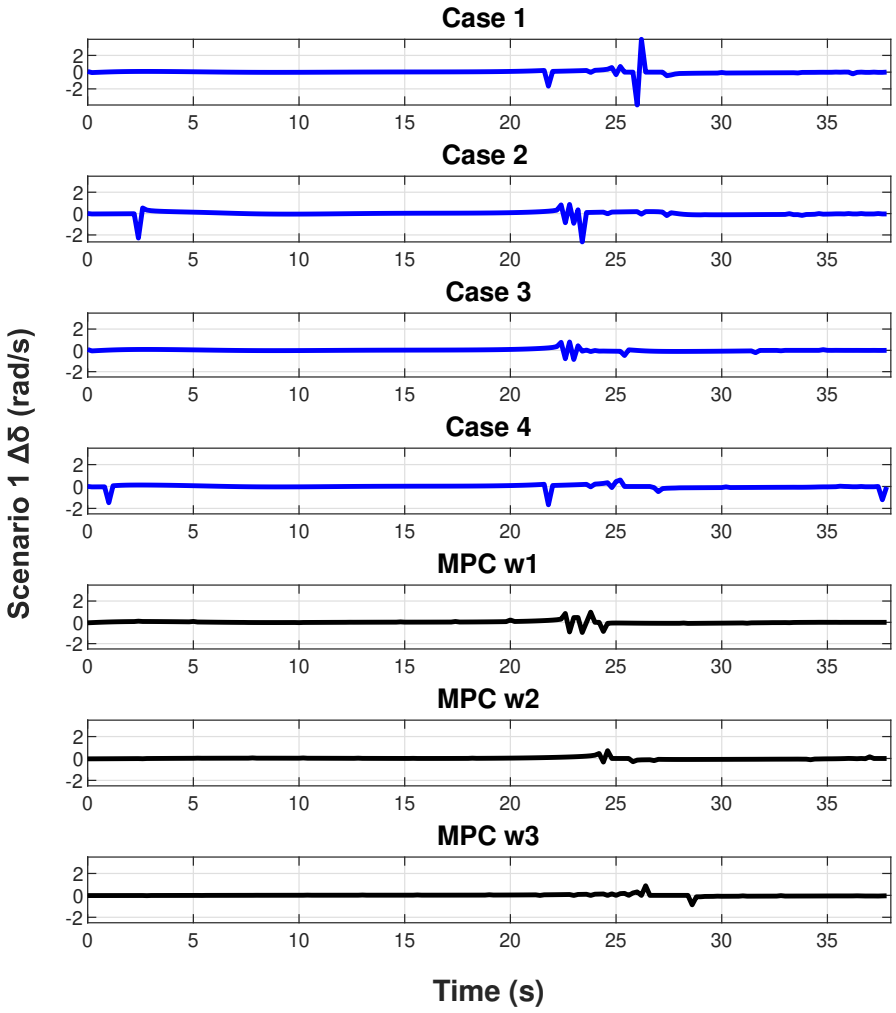

Figure 4.11: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

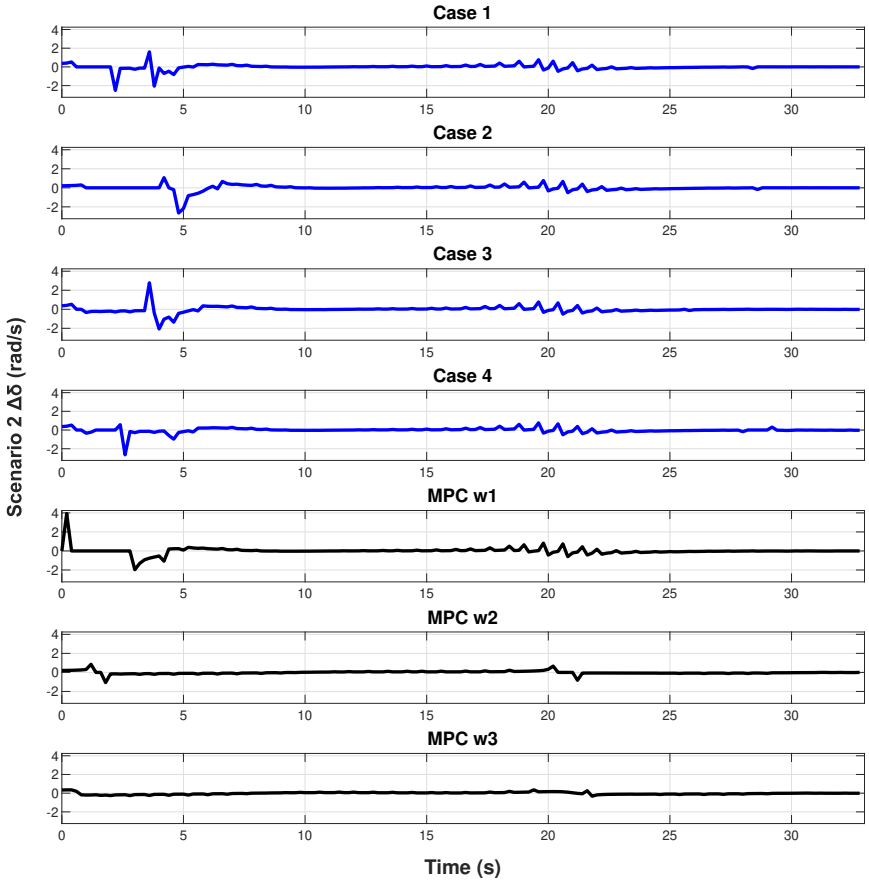

Figure 4.12: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

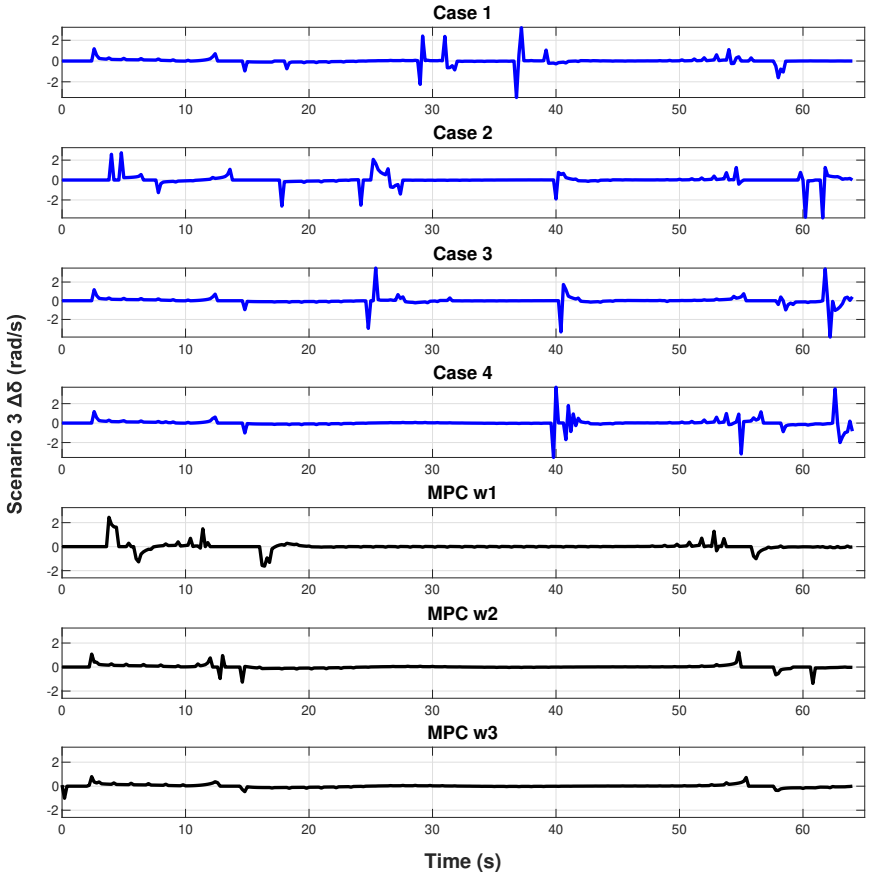

Figure 4.13: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

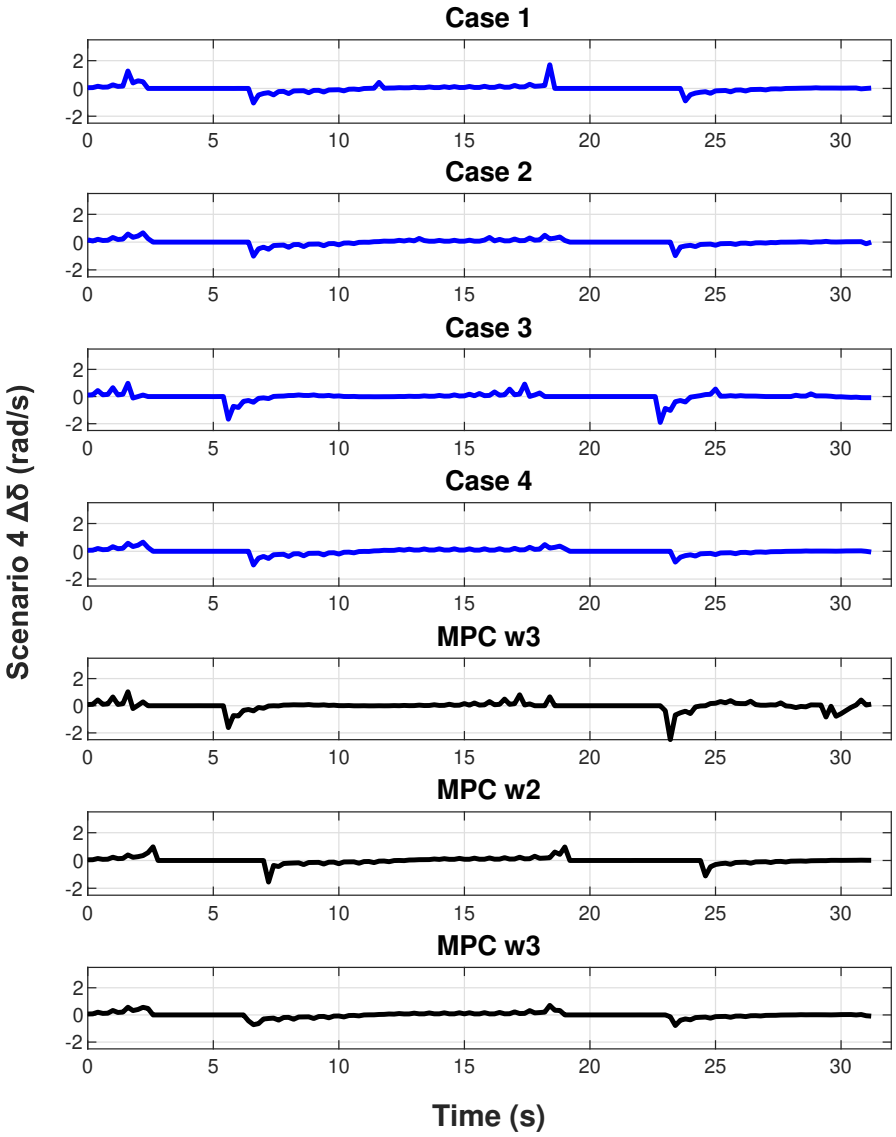

Figure 4.14: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

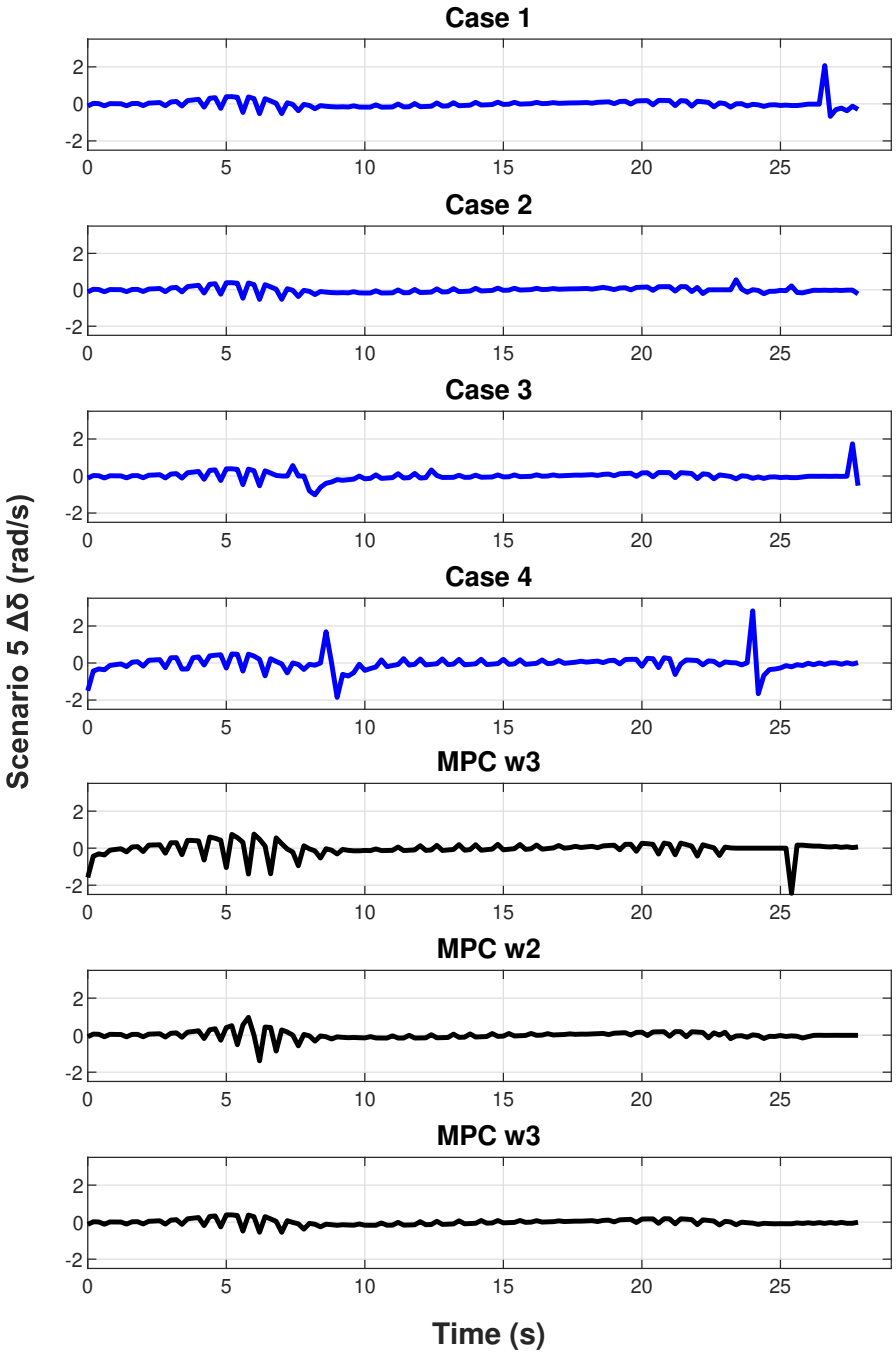

Figure 4.15: Steering rate $\Delta\delta$ over time for RL-assisted MPC Cases and baseline controllers.

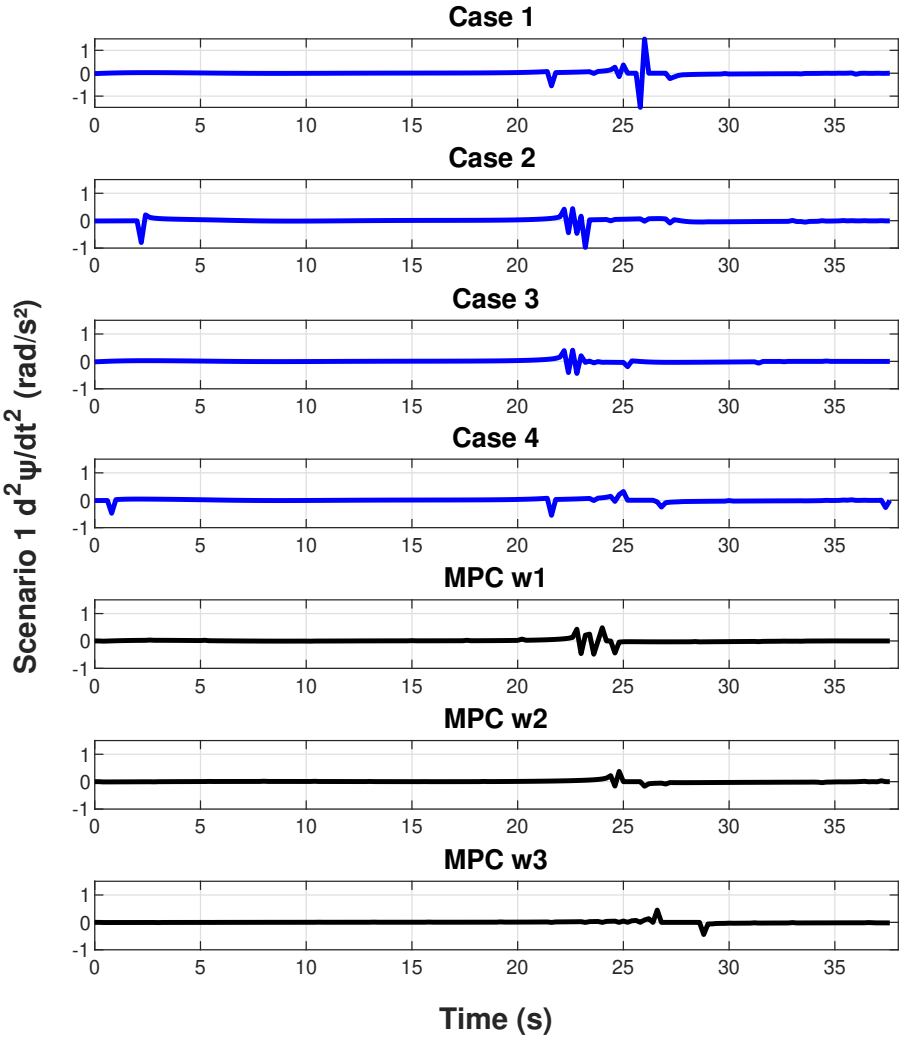

Figure 4.16: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

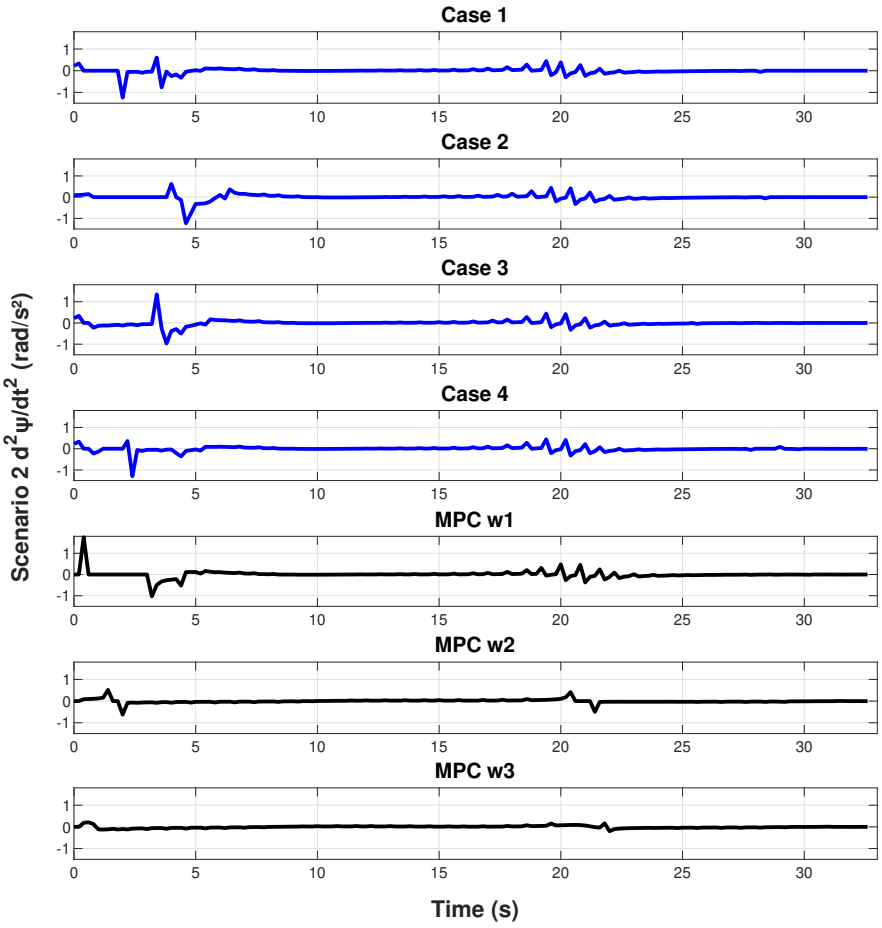

Figure 4.17: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

Table 4.16: Performance metrics across RL-assisted MPC and fixed-weight MPC cases for scenario 5.

Controller Scenario 5	$d_{y.max}$	TV-$\Delta\delta$	TV-δ
RL-assisted MPC Case 1	0.24	22.6566	3.9686
RL-assisted MPC Case 2065035	0.3571	18.5	3.3090
RL-assisted MPC Case 3	0.2083	22.3993	3.9515
RL-assisted MPC Case 4	0.2011	42.1801	6.7398
MPC w1	0.2473	40.5182	6.3254
MPC w2	0.3180	23.0532	3.7998
MPC w3	0.2160	17.1130	3.2572

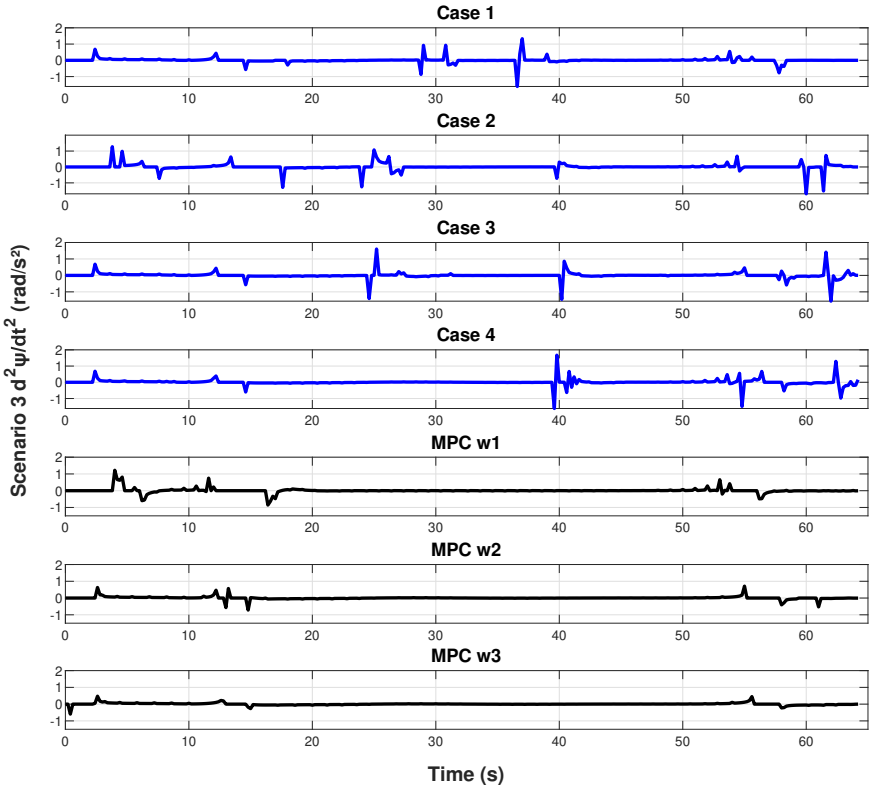

Figure 4.18: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

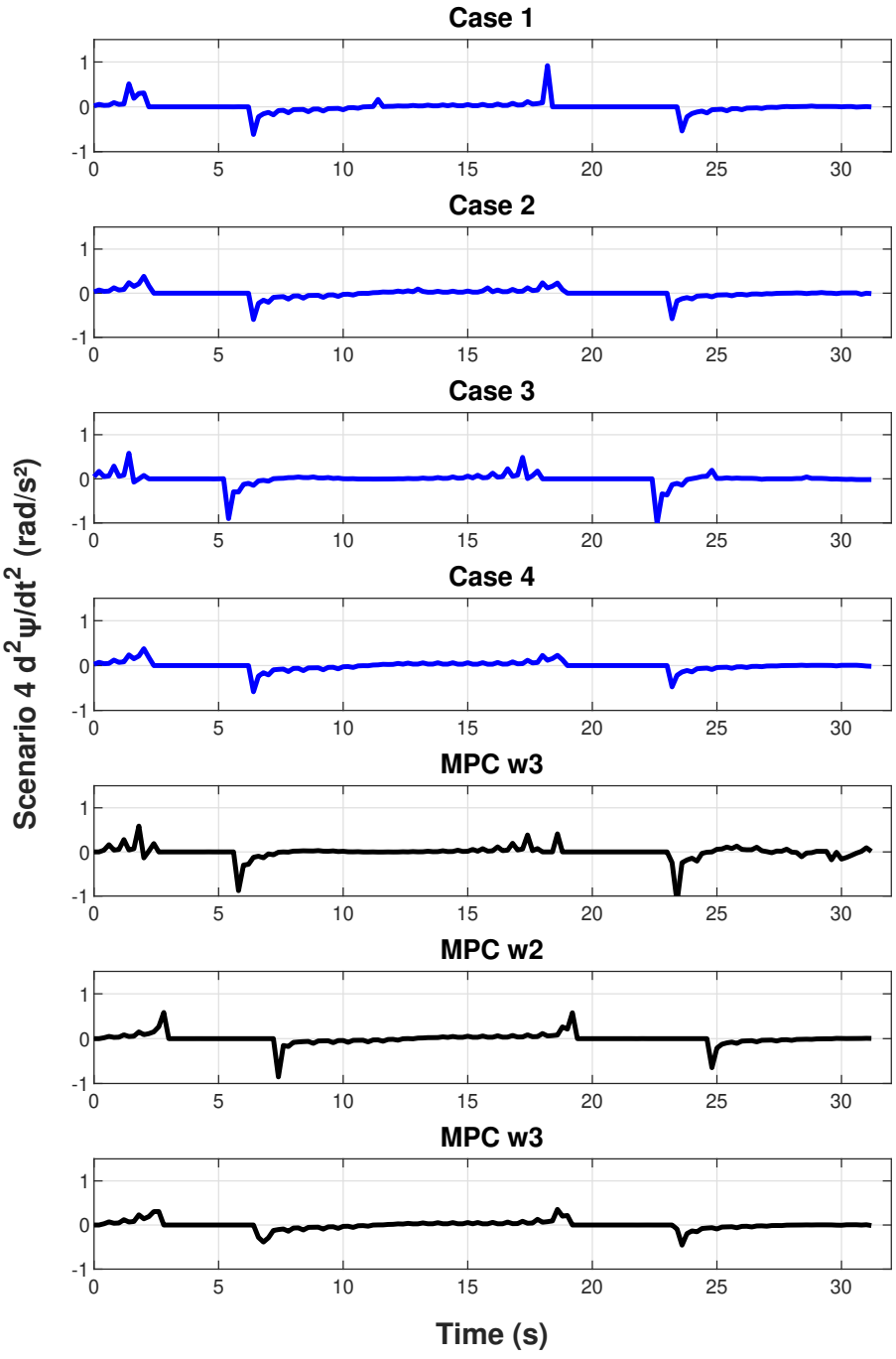

Figure 4.19: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

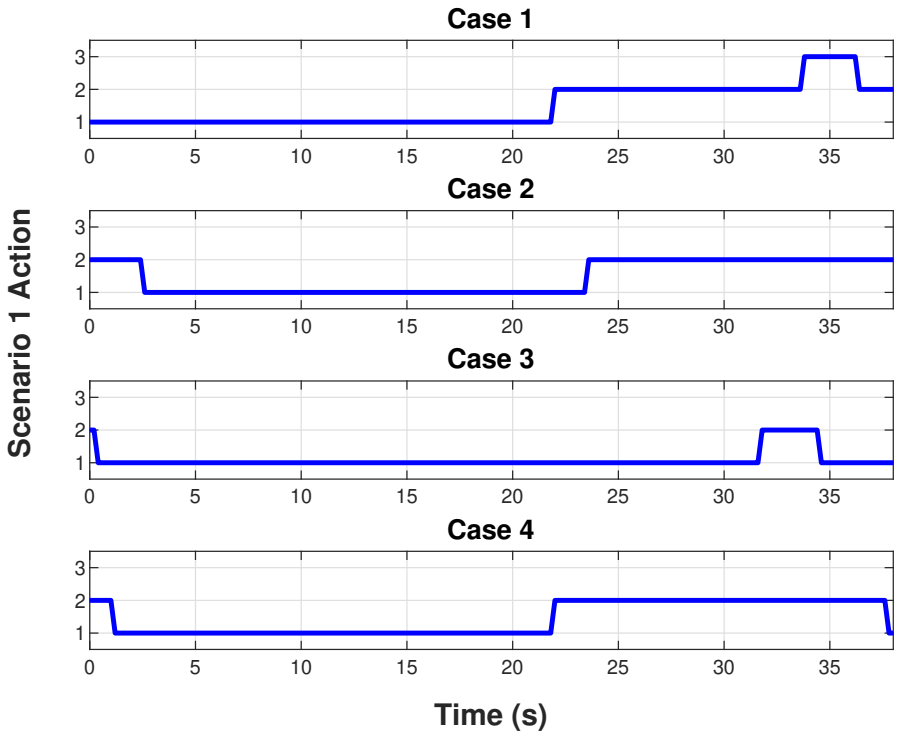

Figure 4.20: Action selection over time for the RL-assisted MPC Cases.

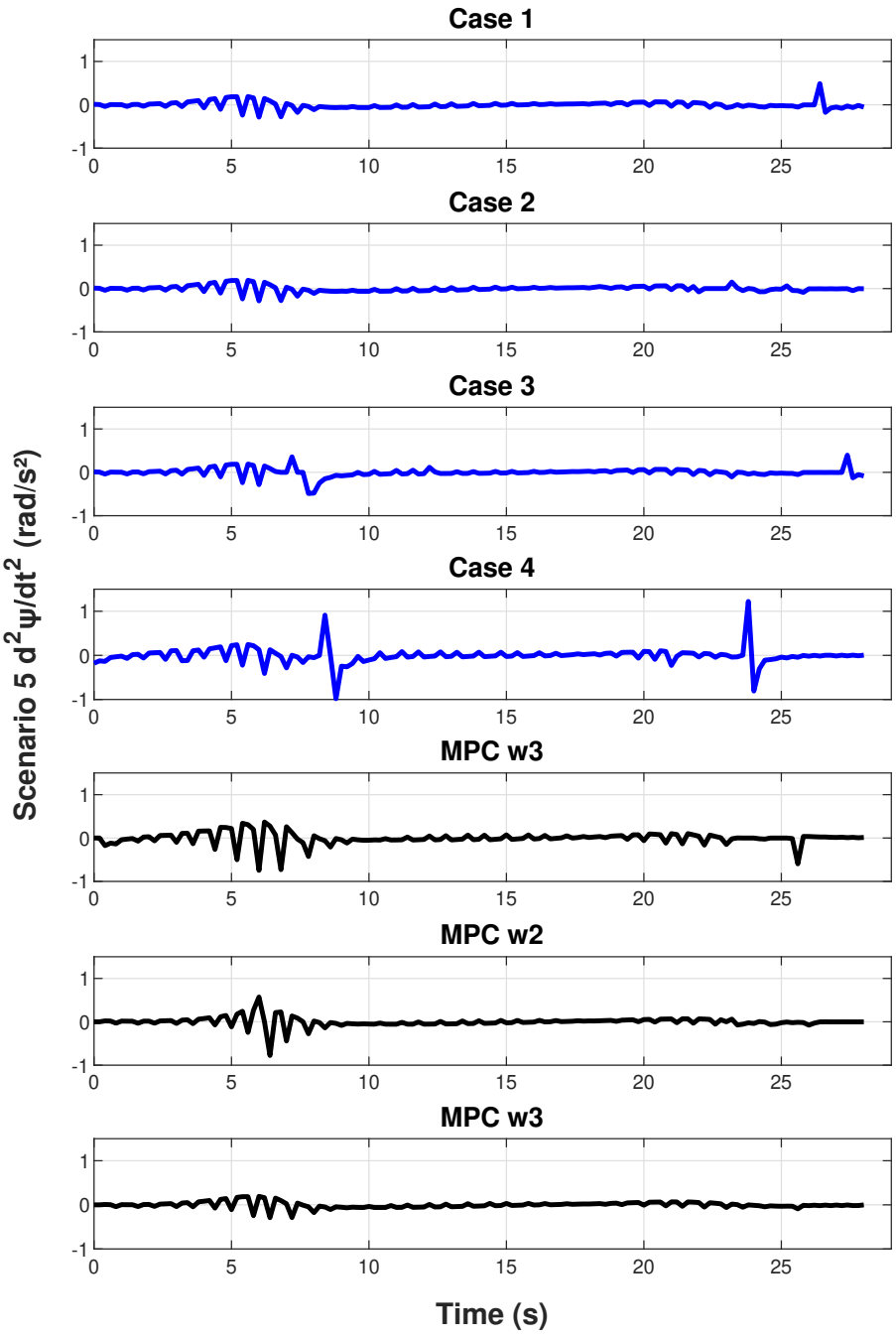

Figure 4.21: Heading acceleration $\Delta^2\psi$ over time for RL-assisted MPC Cases and baseline controllers.

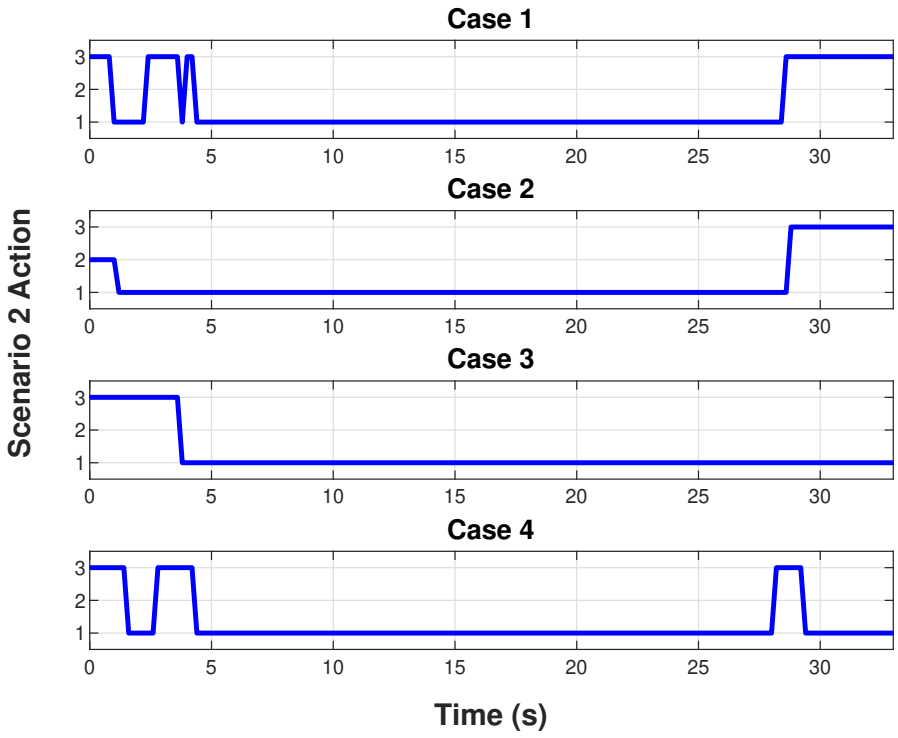

Figure 4.22: Action selection over time for the RL-assisted MPC Cases.

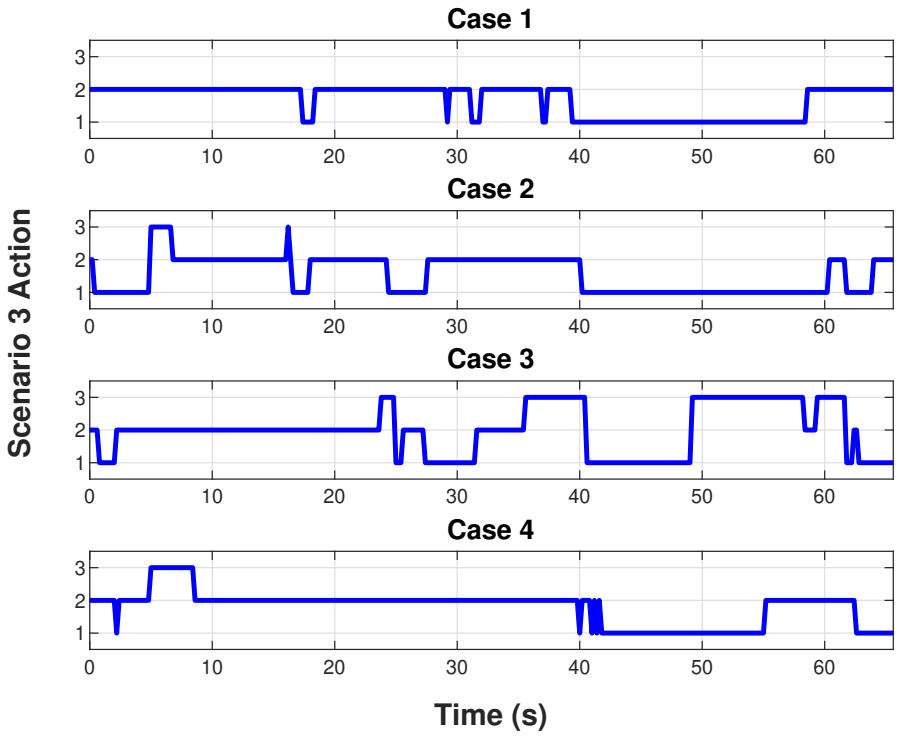

Figure 4.23: Action selection over time for the RL-assisted MPC Cases.

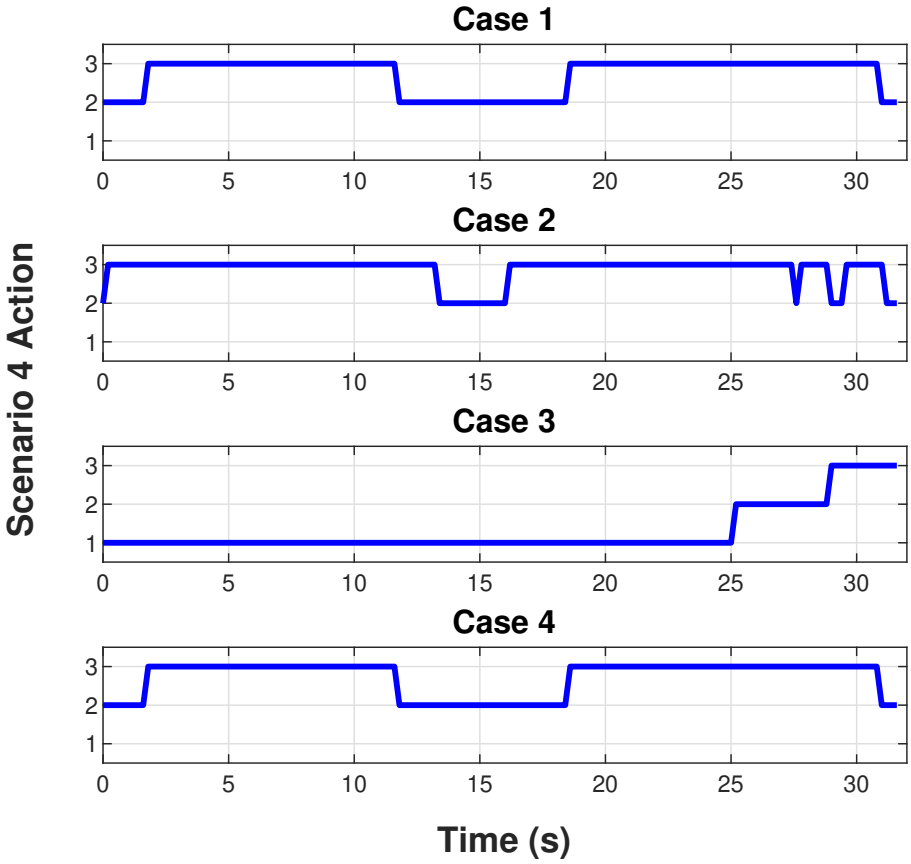

Figure 4.24: Action selection over time for the RL-assisted MPC Cases.

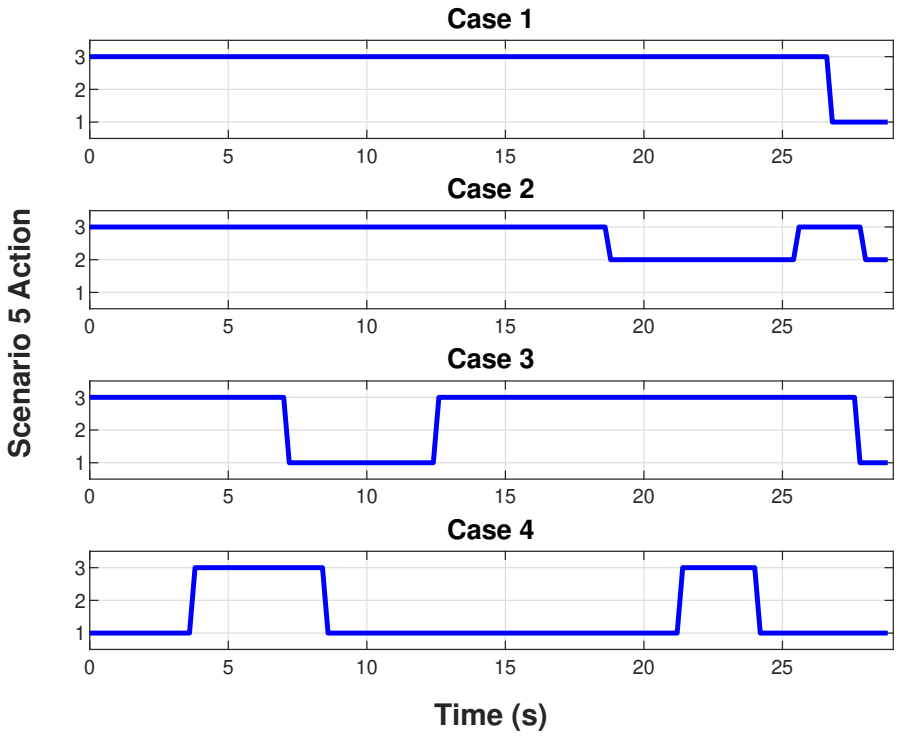

Figure 4.25: Action selection over time for the RL-assisted MPC Cases.

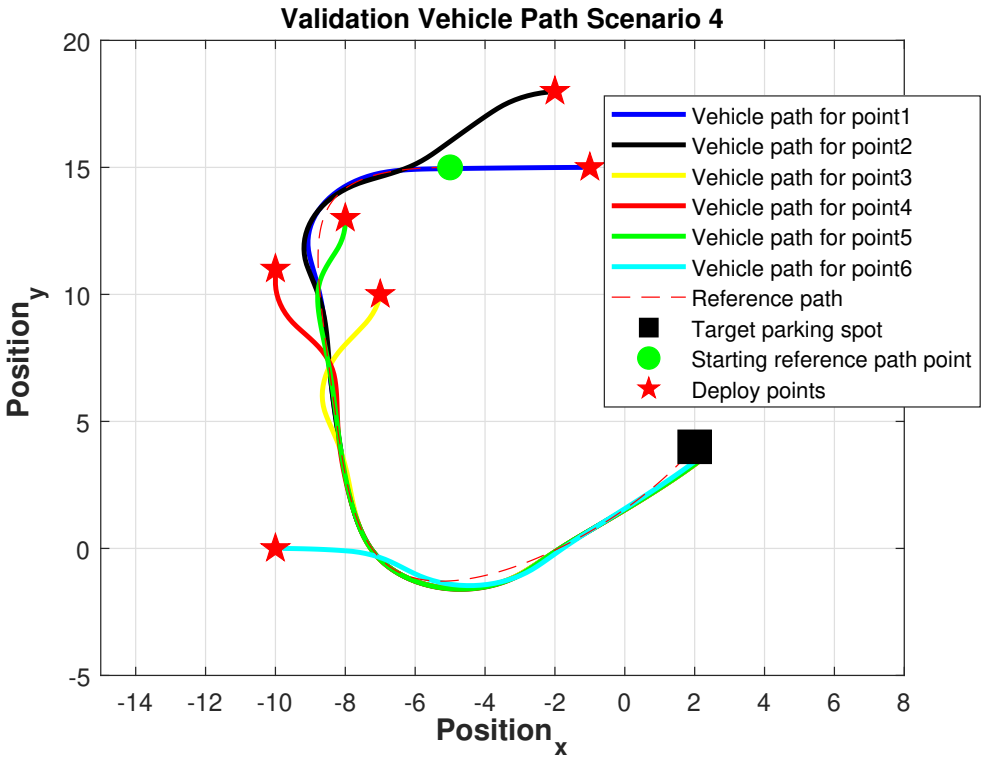

Figure 4.26: Validation vehicle trajectories for Scenario 4 from six unseen starting points. The RL-assisted MPC successfully guided the vehicle from each deployment point (red stars) to the target parking spot (black square) while maintaining smooth alignment with the reference path..

tested using six distinct starting points to examine the robustness and adaptability of the RL-assisted MPC controller. The agent had never been exposed to any of these points during training. The results show that the RL-assisted MPC successfully controlled the vehicle from all deployment points, converged back to the nearest reference path and reached the target parking spot, as shown in Fig. 4.26. Similarly, in Scenario 5, the validation results were also promising, showing that the agent effectively navigated the vehicle from six unseen random starting locations to the target position, as illustrated in Fig. 4.27.

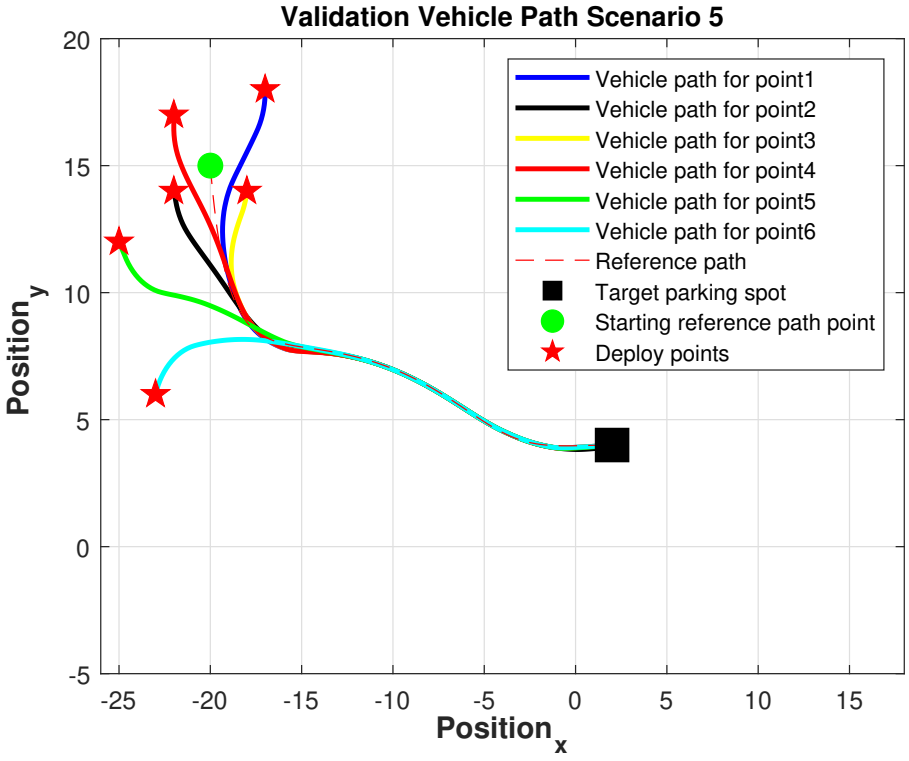

Figure 4.27: Validation vehicle trajectories for Scenario 5 from six unseen starting points. The RL-assisted MPC successfully guided the vehicle from each deployment point (red stars) to the target parking spot (black square) while maintaining smooth alignment with the reference path..

CHAPTER FIVE

CONCLUSION AND FUTURE WORK

This dissertation successfully developed and validated a novel reinforcement learning-assisted model predictive control (RL-assisted MPC) framework for automated parking systems. The proposed approach addresses the fundamental limitation of conventional MPC controllers, which rely on manually tuned static cost function weights that cannot adapt to varying operational conditions during parking maneuvers. The key innovation lies in employing a Deep Q-Network (DQN) agent to dynamically select optimal weight configurations from predefined sets based on real-time vehicle state information (vehicle heading angle and distance to parking lot). This hybrid architecture leverages the predictive optimization capabilities of MPC while incorporating the adaptive decision-making strengths of reinforcement learning, eliminating the need for extensive manual parameter tuning. Comprehensive experimental validation across five distinct parking scenarios demonstrated that the RL-assisted MPC framework consistently outperformed static-weight MPC baselines. The proposed controller achieved comparable or superior lateral tracking accuracy while providing significantly improved steering smoothness and heading stability. Notably, the framework exhibited superior trade-off management between competing control objectives—tracking precision, control effort, and maneuver smoothness—without requiring scenario-specific manual tuning. The two case studies revealed that the framework's effectiveness extends beyond fixed-speed conditions to variable-speed scenarios. Policy adoption analysis showed that the RL agent adapts its weight selection strategy according to different phases of parking maneuvers, with smoother trajectories favoring stable weight selection and precision-critical phases involving more dynamic adaptation.

The validation experiments presented in Chapter 4 further confirmed the robustness and adaptability of the proposed RL-assisted MPC framework. When tested on, unseen starting positions in scenarios 4 and 5, the controller successfully guided the vehicle to the designated parking spot in all cases. These results demonstrate that the trained agent was able to generalize beyond the specific trajectories encountered during training. This outcome reinforces the framework's practical applicability for real-world autonomous parking tasks, where initial positions can vary.

Future research will focus on transitioning from simulation to real-world validation using an autonomous vehicle platform. Additionally, expanding to complex scenarios involving dynamic obstacle avoidance, multi-vehicle coordination, and integration with perception systems represents critical advancement.

REFERENCES

[1] P. Pandita, *Evaluation of soft actor critic in diverse parking environments*. PhD thesis, Master's thesis, 2022.

[2] B. Thunyapoo, C. Ratchadakorntham, P. Siricharoen, and W. Susutti, "Self-parking car simulation using reinforcement learning approach for moderate complexity parking scenario," in *2020 17th international conference on electrical engineering/electronics, computer, telecommunications and information technology (ECTI-CON)*, pp. 576–579, IEEE, 2020.

[3] M. Albilani and A. Bouzeghoub, "Dynamic adjustment of reward function for proximal policy optimization with imitation learning: Application to automated parking systems," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1400–1408, IEEE, 2022.

[4] B. B. K. Ayawli, R. Chellali, A. Y. Appiah, and F. Kyeremeh, "An overview of nature-inspired, conventional, and hybrid methods of autonomous vehicle path planning," *Journal of Advanced Transportation*, vol. 2018, no. 1, p. 8269698, 2018.

[5] Z. Zhou, J. Chen, M. Tao, P. Zhang, and M. Xu, "Experimental validation of event-triggered model predictive control for autonomous vehicle path tracking," in *2023 IEEE International Conference on Electro Information Technology (eIT)*, pp. 35–40, IEEE, 2023.

[6] Z. Zhou, C. Rother, and J. Chen, "Event-triggered model predictive control for autonomous vehicle path tracking: Validation using CARLA simulator," *IEEE Tran. Intel. Veh.*, vol. 8, no. 6, pp. 3547–3555, 2023.

[7] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *IEEE Intell. Veh. Symp.*, (Seoul, Korea), pp. 1094–1099, June 28–July 1, 2015.

[8] S. Di Cairano, H. E. Tseng, D. Bernardini, and A. Bemporad, "Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain," *IEEE Trans. Control Syst. Tech.*, vol. 21, no. 4, pp. 1236–1248, 2012.

[9] R. Yu, H. Guo, Z. Sun, and H. Chen, "MPC-based regional path tracking controller design for autonomous ground vehicles," in *IEEE International Conference on Systems, Man, and Cybernetics*, (Hong Kong, China), pp. 2510–2515, October 09-12, 2015.

[10] J. Chen and Z. Yi, "Comparison of event-triggered model predictive control for autonomous vehicle path tracking," in *IEEE Conf. Control Technology and Applications*, (San Diego, CA), August 8–11, 2021.

[11] J. Chen, L. Zhang, and W. Gao, "Reconfigurable model predictive control for large scale distributed systems," *IEEE Systems Journal*, vol. 18, pp. 965–976, June 2024.

[12] X. Sun, Y. Zhang, Y. Cai, and X. Tian, "Compensated deadbeat predictive current control considering disturbance and vsi nonlinearity for in-wheel pmsms," *IEEE/ASME Transactions on Mechatronics*, 2022.

[13] I. Jammeli, A. Chemori, H. Moon, S. Elloumi, and S. Mohammed, "An assistive explicit model predictive control framework for a knee rehabilitation exoskeleton," *IEEE/ASME Trans. Mechatronics*, 2021.

[14] Q. Gong, J. Xu, J. Ye, H. Feng, and A. Shen, "Nonlinear model predictive control for premixed turbocharged natural gas engine," *IEEE/ASME Trans. Mechatronics*, 2021.

[15] C. Liu, C. Li, and W. Li, "Computationally efficient mpc for path following of underactuated marine vessels using projection neural network," *Neural Computing and Applications*, vol. 32, no. 11, pp. 7455–7464, 2020.

[16] A. Jain, L. Chan, D. S. Brown, and A. D. Dragan, "Optimal cost design for model predictive control," in *Learning for Dynamics and Control*, pp. 1205–1217, PMLR, 2021.

[17] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[18] M. Ibrahim and R. Elhafiz, "Integrated clinical environment security analysis using reinforcement learning," *Bioengineering*, vol. 9, no. 6, p. 253, 2022.

[19] M. Ibrahim and R. Elhafiz, "Security analysis of cyber-physical systems using reinforcement learning," *Sensors*, vol. 23, no. 3, p. 1634, 2023.

[20] A. Irshayyid, J. Chen, and G. Xiong, "A review on reinforcement learning-based highway autonomous vehicle control," *Green Energy and Intelligent Transportation*, vol. 3, no. 4, p. 100156, 2024.

[21] J. Chen, X. Meng, and Z. Li, "Reinforcement learning-based event-triggered model predictive control for autonomous vehicle path following," in *American Control Conf.*, (Atlanta, GA), June 8–10, 2022.

[22] F. Dang, D. Chen, J. Chen, and Z. Li, "Event-triggered model predictive control with deep reinforcement learning," *IEEE Transactions on Intelligent Vehicles*, vol. 9, pp. 459–468, January 2024.

[23] A. Irshayyid and J. Chen, "Comparative study of cooperative platoon merging control based on reinforcement learning," *Sensors*, vol. 23, no. 2, p. 990, 2023.

[24] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 6, pp. 2064–2076, 2019.

[25] R. Kamalapurkar, L. Andrews, P. Walters, and W. E. Dixon, "Model-based reinforcement learning for infinite-horizon approximate optimal tracking," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 3, pp. 753–758, 2016.

[26] Y. Li, "Reinforcement learning applications," *arXiv preprint arXiv:1908.06973*, 2019.

[27] X. Liu, S. Zhu, Y. Fang, Y. Wang, L. Fu, W. Lei, and Z. Zhou, "Optimization design of parking models based on complex and random parking environments," *World Electric Vehicle Journal*, vol. 14, no. 12, p. 344, 2023.

[28] R. Dong, M. Hu, T. Cui, D. Wan, and X. Meng, "A mathematical modeling approach for optimal parking space selection and path planning in autonomous parking systems with uav-assisted topsis entropy weight method," 2023.

[29] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9329–9338, 2019.

[30] M. Gallieri, S. S. M. Salehian, N. E. Toklu, A. Quaglino, J. Masci, J. Koutník, and F. Gomez, "Safe interactive model-based learning," *arXiv preprint arXiv:1911.06556*, 2019.

[31] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *Advances in neural information processing systems*, vol. 30, 2017.

[32] M. Lin, Z. Sun, Y. Xia, and J. Zhang, "Reinforcement learning-based model predictive control for discrete-time systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 3, pp. 3312–3324, 2023.

[33] K. Lowrey, A. Rajeswaran, S. Kakade, E. Todorov, and I. Mordatch, "Plan online, learn offline: Efficient learning and exploration via model-based control," *arXiv preprint arXiv:1811.01848*, 2018.

[34] F. Farshidian, D. Hoeller, and M. Hutter, "Deep value model predictive control," *arXiv preprint arXiv:1910.03358*, 2019.

[35] N. Karnchanachari, M. I. Valls, D. Hoeller, and M. Hutter, "Practical reinforcement learning for mpc: Learning from sparse objectives in under an hour on a real robot," in *Learning for Dynamics and Control*, pp. 211–224, PMLR, 2020.

[36] S. Gros and M. Zanon, "Data-driven economic nmpc using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, 2019.

[37] M. Zanon, S. Gros, and A. Bemporad, "Practical reinforcement learning of stabilizing economic mpc," in *2019 18th European Control Conference (ECC)*, pp. 2258–2263, IEEE, 2019.

[38] M. Zanon and S. Gros, "Safe reinforcement learning using robust mpc," *IEEE Transactions on Automatic Control*, vol. 66, no. 8, pp. 3638–3652, 2020.

[39] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[40] R. Bellman, "A markovian decision process," *Indiana Univ. Math. J.*, vol. 6, pp. 679–684, 1957.

[41] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[42] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[43] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," *Advances in neural information processing systems*, vol. 12, 1999.

[44] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in neural information processing systems*, vol. 12, 1999.

[45] L. Zheng, P. Zeng, W. Yang, Y. Li, and Z. Zhan, "Bézier curve-based trajectory planning for autonomous vehicles with collision avoidance," *IET Intelligent Transport Systems*, vol. 14, no. 13, pp. 1882–1891, 2020.

[46] L. Chen, Z. Qin, M. Hu, H. Gao, F. Zhang, Y. Bian, B. Xu, X. Peng, and J. Hu, "Nonlinear model predictive lateral control for automated parking system with position and attitude coordination control," in *2022 37th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 49–55, IEEE, 2022.