machines

MDPI

Article

Real-Time Cooperative Path Planning and Collision Avoidance for Autonomous Logistics Vehicles Using Reinforcement Learning and Distributed Model Predictive Control

Mingxin Li [1], Hui Li [1], Yunan Yao [2], Yulei Zhu [1], Hailong Weng [1], Huabiao Jin [2,*] and Taiwei Yang [1,2]

[1] COSCO Shipping Heavy Industry (Zhoushan) Co., Ltd., Zhoushan 316131, China; yangtw9410@sohu.com (T.Y.)
[2] School of Naval Architecture, Ocean and Energy Power Engineering, Wuhan University of Technology, Wuhan 430063, China; ynyao@whut.edu.cn
* Correspondence: jhb@whut.edu.cn

Abstract

In industrial environments such as ports and warehouses, autonomous logistics vehicles face significant challenges in coordinating multiple vehicles while ensuring safe and efficient path planning. This study proposes a novel real-time cooperative control framework for autonomous vehicles, combining reinforcement learning (RL) and distributed model predictive control (DMPC). The RL agent dynamically adjusts the optimization weights of the DMPC to adapt to the vehicle's real-time environment, while the DMPC enables decentralized path planning and collision avoidance. The system leverages multi-source sensor fusion, including GNSS, UWB, IMU, LiDAR, and stereo cameras, to provide accurate state estimations of vehicles. Simulation results demonstrate that the proposed RL-DMPC approach outperforms traditional centralized control strategies in terms of tracking accuracy, collision avoidance, and safety margins. Furthermore, the proposed method significantly improves control smoothness compared to rule-based strategies. This framework is particularly effective in dynamic and constrained industrial settings, offering a robust solution for multi-vehicle coordination with minimal communication delays. The study highlights the potential of combining RL with DMPC to achieve real-time, scalable, and adaptive solutions for autonomous logistics.

Keywords: autonomous logistics vehicles; collision avoidance; reinforcement learning; distributed model predictive control

Academic Editor: Zheng Chen

Received: 24 November 2025
Revised: 18 December 2025
Accepted: 21 December 2025
Published: 24 December 2025

1. Introduction

With the rapid development of Industry 4.0 and intelligent logistics systems, the application of autonomous logistics vehicles (such as AGVs and unmanned forklifts) in industrial environments like ports, large warehouses, and shipyards has been expanding significantly [1–3]. These environments are characterized by narrow passages, dynamic obstacle distributions, and severe visual occlusion, placing extremely high demands on vehicle trajectory tracking accuracy and real-time obstacle avoidance capabilities [4–6]. In this context, ensuring stable tracking and real-time collaborative obstacle avoidance for multiple autonomous logistics vehicles in complex, dynamic, and high-constrained industrial scenarios has become a major research focus and challenge [7,8].

Traditional vehicle trajectory control methods, such as PID control, sliding mode control, or fuzzy control, have achieved successes in static or relatively simple environments [9–11].

However, their adaptability and robustness are often insufficient when facing dynamic interactions and frequent environmental changes. To address these challenges in complex industrial settings, hybrid global-local planning architectures have become a standard solution [12]. These approaches typically combine global graph search algorithms (e.g., Hybrid A, D, or lattice search) with local refinement methods to ensure kinodynamic feasibility and situation-dependent behavior switching [13,14]. Within this hybrid framework, Model Predictive Control (MPC) has gained widespread attention as a preferred local planner [15,16]. MPC works by predicting future states and optimizing control sequences in real-time, demonstrating strong capabilities in handling system dynamics and constraints. However, in multi-vehicle coordination scenarios, centralized MPC approaches often struggle to meet real-time requirements due to computational complexity, communication bottlenecks, and model uncertainties caused by environmental changes [17,18].

To address the limitations of centralized approaches, Distributed Model Predictive Control (DMPC) has emerged as a promising solution [19,20]. DMPC decentralizes the optimization tasks, allowing each vehicle to execute trajectory planning locally while enforcing cooperative constraints through the exchange of predicted trajectories or state information via V2X communication [21,22]. Integrated solutions based on DMPC have been widely applied in multi-agent systems (MAS) to achieve distributed formation control and cooperative collision avoidance [23]. Despite its advantages, DMPC still faces challenges in dynamic environments, such as slow convergence rates and limited adaptability to rapid obstacle changes or parameter uncertainties [24]. Furthermore, under conditions of communication delays or incomplete information, the robustness of pure DMPC schemes often degrades, necessitating more adaptive decision-making mechanisms.

To overcome the adaptability limitations of fixed-model approaches, Reinforcement Learning (RL) has demonstrated remarkable potential in handling environmental uncertainties and model mismatches through data-driven interaction [25–28]. Consequently, the integration of RL with MPC has become a cutting-edge research direction for autonomous navigation. Recent literature explores various fusion strategies to combine the strengths of both paradigms [29–31]. For instance, Deep Reinforcement Learning (DRL) has been employed to generate high-level reference trajectories or warm-start the optimization process for MPC, thereby enhancing computational efficiency and tracking performance [32,33]. More pertinently, other researchers have utilized RL agents to dynamically tune the weighting parameters or prediction horizons of the MPC cost function in real-time, allowing the controller to adapt its behavior—such as the aggressiveness of obstacle avoidance—according to changing scenarios [34,35]. These hybrid frameworks effectively synergize the adaptability of learning-based methods with the rigorous constraint satisfaction of control-theoretic approaches.

To bridge the aforementioned gaps, this study proposes a novel cooperative control framework: Reinforcement Learning-based Distributed Model Predictive Control (RL-DMPC). The key innovations and contributions of this work are threefold:

1. **Hierarchical Adaptive Architecture:** We integrate an RL module as a real-time weight regulator within the distributed MPC system. This design achieves a robust fusion of "data-driven adaptability" and "model-based safety," allowing the controller to dynamically balance tracking accuracy and obstacle avoidance based on environmental states.
2. **Distributed Collaboration via V2X:** We deploy local DMPC optimizers on each vehicle that utilize V2X communication to share predicted trajectories. This enables the decentralized resolution of cooperative obstacle avoidance constraints without relying on a central server, thereby enhancing system scalability.

3. **Validation in High-Fidelity Simulation:** The proposed framework is rigorously tested in a high-fidelity simulation environment that replicates complex industrial scenarios, such as narrow passages and severe line-of-sight occlusions. While this study primarily focuses on algorithmic architecture and theoretical verification, the comprehensive simulation results demonstrate that the RL-DMPC approach significantly outperforms traditional centralized strategies in terms of tracking error reduction and collaborative response speed.

The remainder of this paper is organized as follows: Section 2 introduces the overall framework of the RL-DMPC cooperative control system. Section 3 details the design of the RL-based adaptive decision-making module. Section 4 discusses the distributed model predictive controller and the multi-vehicle collaborative obstacle avoidance strategy. Section 5 presents the simulation results and performance verification. Section 6 provides a comprehensive discussion on the experimental findings, system limitations, and scalability. Finally, Section 7 concludes the paper and outlines future research directions.

2. RL-DMPC Cooperative Control System Framework

In response to the challenges faced by autonomous logistics vehicles in complex industrial environments, such as dynamic obstacle avoidance, trajectory tracking, and multi-vehicle coordination, this paper proposes a hierarchical cooperative control architecture based on Reinforcement Learning and Distributed Model Predictive Control (RL-DMPC). This architecture aims to integrate the adaptive capabilities of data-driven methods with the constraint-handling advantages of model-based control methods, building a closed-loop, real-time, distributed intelligent control system.

2.1. System Overall Architecture Design

As shown in Figure 1, the RL-DMPC cooperative control system adopts a layered hierarchical structure, consisting of the perception and localization layer, the decision-making and cooperative control layer (core layer), and the control execution layer. In addition, the system implements information exchange between vehicles through a distributed network communication module and is equipped with safety and monitoring modules to ensure robust system operation.

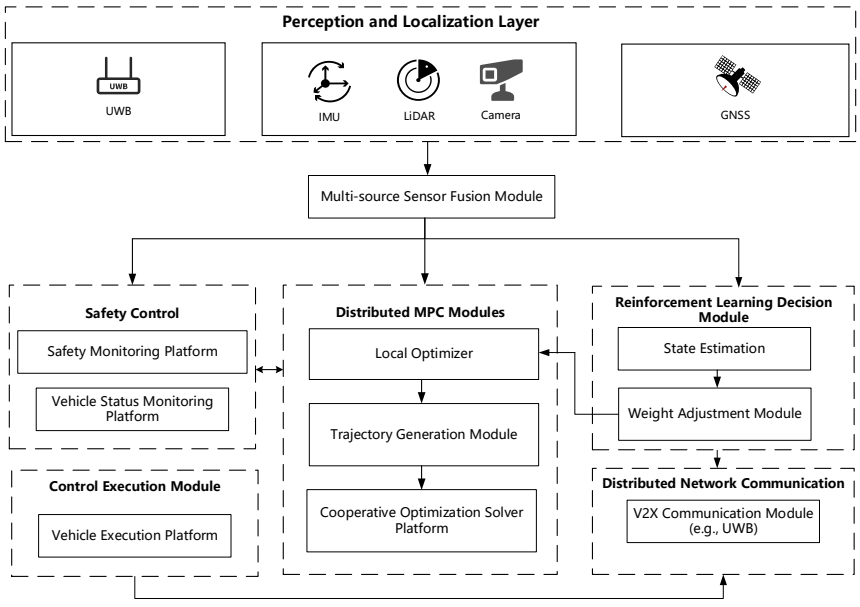


Figure 1. Collaborative Obstacle Avoidance Framework Integrating Reinforcement Learning and Distributed Model Predictive Control.

The entire system follows a "perception—decision—communication—control" closed-loop operation mode. The perception layer is responsible for estimating the vehicle state; the decision layer uses the Reinforcement Learning (RL) module to adjust the weights in real-time, assisting the Distributed Model Predictive Control (DMPC) module in local trajectory optimization; the communication layer shares predicted trajectories between vehicles to build cooperative constraints; and, finally, the execution layer drives the vehicle to complete physical motion.

2.2. Multi-Source Perception and High-Precision Localization Layer

The perception and localization layer is the entry point for the system to acquire physical environmental information. It is mainly responsible for providing high-frequency, high-precision vehicle state estimates $s = [x, y, v, \theta]^T$ (representing lateral position, longitudinal position, velocity, and heading angle) to the decision layer. Considering the complexity of industrial logistics scenarios (such as indoor-outdoor transitions and weak GNSS signals), the system adopts a multi-source heterogeneous sensor fusion scheme:

- **Localization Subsystem:** In outdoor open areas, the system utilizes the Global Navigation Satellite System (GNSS) to obtain absolute position information. In indoor or signal-blocked areas, centimeter-level relative localization is achieved using Ultra-Wideband (UWB) technology. Meanwhile, the Inertial Measurement Unit (IMU) provides high-frequency angular velocity and acceleration data to compensate for delays and signal loss in positioning.
- **Environmental Perception Subsystem:** The LiDAR (Light Detection and Ranging) and camera systems form the main environmental perception subsystem, responsible for detecting static obstacle boundaries and dynamic objects (such as pedestrians and other operating vehicles) in real-time, thus constructing local environmental maps.
- **Multi-source Fusion Module:** Data from the above sensors is sent to the fusion module, where the Extended Kalman Filter (EKF) algorithm is used for spatiotemporal alignment and state estimation, providing smooth and reliable vehicle state information as the sole input for subsequent control algorithms.

The perception module is designed to identify both static infrastructure and dynamic agents. While this study focuses on dynamic vehicle interaction, the detection pipeline is scalable to dense static environments.

2.3. Decision-Making and Cooperative Control Layer

This layer serves as the core of the system, adopting a coupling mechanism of "RL weight adaptation + DMPC distributed optimization" to solve the conflict between real-time performance and adaptability in multi-vehicle coordination. It consists of two inter-coupled intelligent modules:

- **Reinforcement Learning Decision Module**

This module serves as the adaptive engine of the system. It receives accurate state information s from the fusion module and the neighboring vehicle's state obtained through communication, using them as the observation state for the RL agent. Based on this state, the Proximal Policy Optimization (PPO) agent, which has been offline trained, performs real-time inference and outputs the adaptive weight adjustment signal Δw. This signal dynamically adjusts the weights of the objective function in each vehicle's local DMPC controller (such as trajectory tracking, obstacle avoidance, and energy consumption). For example, when the system detects that the distance to a neighboring vehicle is too close, the RL agent automatically increases the weight of the obstacle avoidance term. When the system needs to urgently track a new instruction, the tracking accuracy weight will be

prioritized. This enables the control system to adapt to complex dynamic scenarios with intelligent decision-making beyond fixed parameter rules.

- **Distributed Model Predictive Control (DMPC) Cooperative Controller:**

This module is the optimization and execution planner of the system. Each logistics vehicle has an independent local DMPC optimizer that predicts future states based on an accurate vehicle dynamics model. The optimization problem is subject to strict cooperative obstacle avoidance constraints, which are constructed using the predicted trajectories shared in real-time from neighboring vehicles through the V2X communication module. Crucially, the DMPC optimization goal is determined by the weight w dynamically adjusted by the RL module, thus achieving a closed-loop from "perception intelligence" to "control optimization." Each DMPC performs rolling optimization to solve the optimal control sequence u^* for the future time horizon, ensuring that vehicle kinematics and dynamics constraints are satisfied, while also achieving multi-vehicle cooperative obstacle avoidance and efficient trajectory tracking.

2.4. Distributed Communication and Safety Monitoring

To ensure cooperative consistency and high reliability of multi-agent systems in dynamic environments, this system builds a security system integrating low-latency distributed communication and multi-level safety monitoring.

Distributed network communication is the "information artery" for cooperative control. The system relies on low-latency, high-bandwidth V2X technologies such as 5G/UWB to establish a decentralized vehicle-to-vehicle communication network. Through this network, each vehicle not only broadcasts its real-time state $s_i(t)$, include both static and dynamic obstacles, but more importantly, shares its locally computed future predicted trajectory sequence T_i. This exchange of predicted trajectories allows each agent to "anticipate" the intentions of neighboring vehicles during local optimization, providing the data foundation for building cooperative obstacle avoidance constraints and avoiding decision conflicts. This distributed interaction mechanism eliminates the need for a central scheduling node, significantly enhancing the system's scalability and fault tolerance in the case of changes in the number of vehicles or local network failures.

On this basis, the safety and monitoring module acts as the "guardian" of the system, providing crucial redundancy and global insight. Given the high safety requirements for industrial applications, the system integrates safety assistance modules (such as virtual barriers based on electronic fences and emergency braking redundancies) and a status monitoring platform. The former triggers predefined safety strategies when there are anomalies in perception, communication, or decision links, ensuring safe operation and bringing the system into a controllable state. The latter provides remote operators with real-time global situational awareness and system health status displays, while maintaining key manual intervention interfaces. This module operates in parallel with the distributed control core, forming a reliable system that allows high autonomy while ensuring final safety.

The RL-DMPC system establishes a closed-loop control flow from perception, decision-making, to execution: Each vehicle first acquires precise state information through the multi-source sensor fusion module, and shares predicted trajectories via the V2X network. Then, the RL decision-making module calculates and outputs adaptive adjustments to the optimization weights for the lower-layer distributed model predictive controller, based on the vehicle's own state and neighboring vehicle states. Each vehicle's DMPC controller then performs rolling optimization based on the updated weights and cooperative obstacle avoidance constraints, solving the local optimal control sequence and executing it. The vehicle state is updated, and the system enters the next cycle of perception and optimization. The core of this process is the deep collaboration between RL and DMPC, which combines

the system's ability to adapt to environmental uncertainty through learning and its rigorous constraint assurance through model-driven methods, laying the foundation for the design of the RL decision-maker and DMPC collaborative strategy in subsequent chapters.

3. Design of the RL-Based Adaptive Decision-Making Module

In the RL-DMPC hierarchical architecture proposed in this paper, the core function of the Reinforcement Learning (RL) decision module is to provide the controller with the adaptive ability to cope with dynamic and unknown environments. Unlike traditional Model Predictive Control (MPC), which relies on fixed parameters, this module dynamically adjusts the optimization target weights of the lower-layer MPC controller by perceiving the system's state in real time. This allows dynamic and intelligent trade-offs among multiple objectives, such as trajectory tracking accuracy, obstacle avoidance safety, and control smoothness. This chapter will detail the design of this module, first introducing its basic unit formed by the MPC controller, and then delving into the design of its state space, action space, reward function, and learning algorithm.

3.1. Single Vehicle RL-MPC Control Unit Structure and Workflow

In the distributed collaborative system built in this paper, each intelligent agent (i.e., each logistics vehicle) adopts a local RL-MPC structure for its controller. This structure is the core execution unit of the entire RL-DMPC framework. The basic principle of this structure is shown in Figure 2.

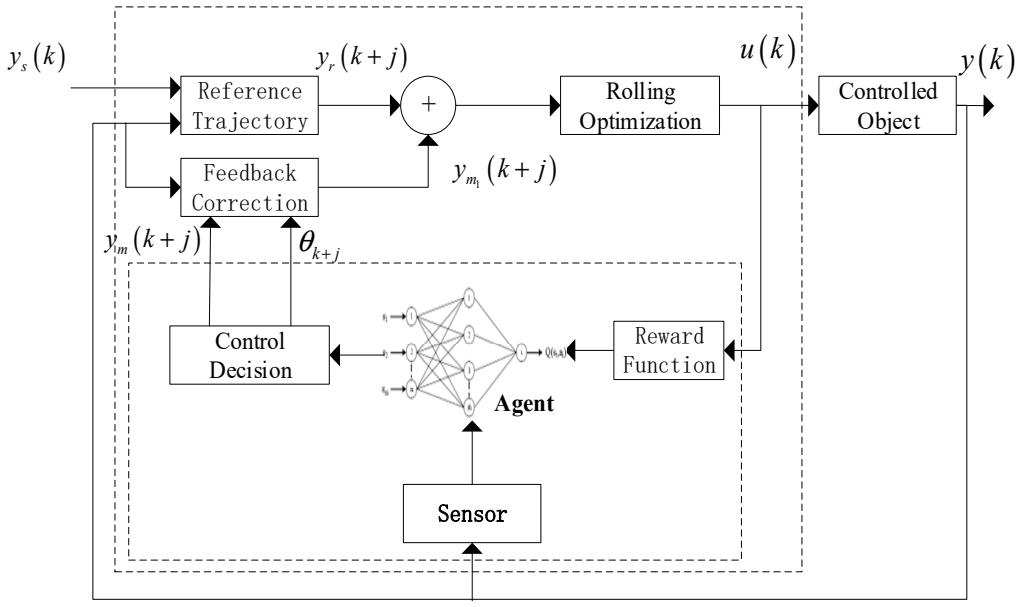


Figure 2. Single Vehicle RL-MPC Controller Principle Diagram.

The workflow of this unit follows a typical intelligent control closed-loop, with the specific steps outlined as follows:

1. State Perception: The agent receives the integrated state information $s(t)$ from the system's perception and localization layer. This state typically includes the vehicle's own state (such as position, velocity, and heading) as well as errors in the desired trajectory and interactions with the environment (such as the distance to the nearest obstacle).
2. Intelligent Decision: Based on its internal policy (trained using the Proximal Policy Optimization (PPO) algorithm), the agent infers the current state $s(t)$ and outputs an adaptive decision. This decision is expressed as a dynamic adjustment Δw to the critical weights w in the objective function of the lower-layer MPC controller.

3. Rolling Optimization: After receiving the new weight w, the MPC controller, within its fixed optimization problem structure, combines the predicted trajectory information from neighboring vehicles (obtained via V2X communication) to build the prediction model input and cooperative obstacle avoidance constraints. It then performs rolling optimization to solve for the optimal control sequence $u^*(t)$ for the future time horizon.
4. Control Execution and Feedback: The first element of the optimal control sequence $u(t)$ is applied to the controlled object (vehicle platform). The vehicle's state is then updated, generating a new environmental reward $r(t)$ and a new state $s(t+1)$ which are fed back to the agent for evaluating and updating its policy.

It should be noted that by incorporating cooperative information from external communications in step 3, the single vehicle RL-MPC unit is upgraded to support cooperation within the RL-DMPC system. The subsequent sections of this chapter will focus on the design details of the RL decision module (i.e., steps 1, 2, and 4 in the above workflow).

3.2. State Space and Action Space Design

In the reinforcement learning framework, the design of the state space and action space is fundamental to the agent's effective learning. In this paper, the state observation space and action output space of the reinforcement learning agent are designed to ensure that the agent can fully perceive the environmental situation and make reasonable control decisions.

- **State Space:**

The design of the state space needs to comprehensively reflect the system's dynamic characteristics and environmental interaction information. The state space s_t for the agent is defined as a composite vector that includes the vehicle's own state, trajectory tracking errors, and interaction information with the surrounding environment:

$$s_t = \left[e_{pos}, e_{yaw}, v, \omega, d_{nearest}, \phi_{nearest}\right] \tag{1}$$

where e_{pos} and e_{yaw} represent the lateral/longitudinal position error and heading error of the vehicle with respect to the desired trajectory, v and ω represent the vehicle's velocity and yaw rate, $d_{nearest}$ and $\phi_{nearest}$ represent the relative distance and bearing angle to the nearest neighboring vehicle or obstacle.

- **Action Space:**

The agent's output actions are continuous adjustments to the key weights in the MPC optimization objective function to ensure smooth control:

$$a_t = [\Delta w_{track}, \Delta w_{obs}] \tag{2}$$

where Δw_{track} is the weight adjustment for the trajectory tracking error term in the MPC objective function. This weight directly influences the controller's emphasis on trajectory tracking accuracy, Δw_{obs} is the weight adjustment for the obstacle avoidance penalty term in the MPC objective function. This weight determines the aggressiveness of the obstacle avoidance behavior.

To ensure the feasibility and numerical stability of the lower-level convex optimization solver, the final weights are strictly constrained within a predefined safety range during both training and inference. Specifically, based on **preliminary empirical simulation tests**, the tracking weight w_{track} and the obstacle avoidance weight w_{obs} are bounded within the range of $[0.1, 20]$. These limits were selected empirically to prevent solver failure while providing sufficient dynamic range for the RL agent to balance safety and efficiency.

3.3. Reward Function Design

The reward function plays a key role in guiding the learning direction of the agent in reinforcement learning. It quantifies and evaluates the agent's behavior at each decision moment, shaping the agent's long-term strategy. In this cooperative obstacle avoidance task, the agent needs to balance multiple objectives such as trajectory tracking accuracy, obstacle avoidance safety, and driving stability. Therefore, a multi-objective piecewise reward function is designed, which has the following overall form:

$$r_t = r_{track} + r_{obs} + r_{stable} \tag{3}$$

is the trajectory tracking reward, which encourages the agent to track the desired trajectory accurately. It is designed based on a quadratic penalty for lateral position error and heading angle error:

$$r_{track} = -\left(\lambda_1 e_{pos}^2 + \lambda_2 e_{yaw}^2\right) \tag{4}$$

where λ_1 and λ_2 are the penalty coefficients for lateral error and heading error, respectively. This quadratic design ensures that when tracking errors increase, the penalty grows quadratically, thus strongly motivating the agent to minimize the tracking deviation. Typically, λ_1 is set to be greater than λ_2, reflecting the priority of position tracking accuracy, which aligns with the actual need to control the lateral position in narrow passages in logistics vehicles.

r_{obs} is the cooperative obstacle avoidance reward, which guides the agent to form safe obstacle avoidance behavior during learning. A Gaussian-based reward function is used, based on the relative distance:

$$r_{obs} = -\eta \cdot \exp\left(-\frac{d_{nearest}^2}{\sigma^2}\right) \tag{5}$$

This function has unique physical significance: when the vehicle is far from obstacles or neighboring vehicles, the obstacle avoidance reward is close to zero, indicating that obstacle avoidance is not a primary concern at that point. As the distance decreases and approaches the safety threshold, the negative reward increases rapidly, forming a strong avoidance incentive. If the distance continues to decrease, the reward function saturates, preventing excessive instability.

The parameter σ adjusts the sensitivity range of obstacle avoidance behavior, and in practice, its value should consider vehicle size, braking performance, and control system response time. The parameter η controls the overall weight of the avoidance reward, which must be coordinated with the tracking reward to ensure reasonable behavior in different scenarios.

r_{stable} is the stability reward, which aims to improve the vehicle's driving quality and avoid sharp lateral movements. Its expression is:

$$r\_\{stable\} = -\lambda_3 \omega^2 \tag{6}$$

where ω is the vehicle's yaw rate, and λ_3 is the corresponding penalty coefficient. The yaw rate directly reflects the intensity of the vehicle's steering, and excessive yaw rate not only affects ride comfort but also poses a risk to driving safety.

By applying quadratic penalties to this term, the system explicitly optimizes control smoothness through reward shaping, rather than relying solely on indirect weight adjustments. This effectively discourages the agent from adopting overly aggressive steering strategies, promoting smoother and more stable control behavior.

3.4. Network Structure and Training Algorithm

To achieve efficient and stable policy learning, this paper adopts the Proximal Policy Optimization (PPO) algorithm based on the Actor-Critic framework. The PPO algorithm demonstrates excellent stability and sample efficiency in continuous control tasks.

- **Network Structure Design:**

The actor-critic network built in this paper consists of two independent deep neural networks, responsible for policy learning and value estimation, respectively. The actor network, which parameterizes the policy function $\pi_\theta(a_t \mid s_t)$, is responsible for generating the probability distribution of actions a_t based on the current environment state s_t. The network structure uses a multi-layer perceptron with an input layer dimension consistent with the state space dimension, containing two hidden layers, each with 128 neurons, and a ReLU activation function. The output layer uses the Tanh activation function to constrain the actions to the continuous range of $[-1, 1]$. The critic network, which estimates the value function $V_\phi(s_t)$, is used to evaluate the long-term expected return of the current state s_t. Its structure is similar to the actor network, but the output layer is a single linear neuron that directly outputs the state value estimate. This symmetric network design ensures sufficient expressive capacity while controlling computational complexity, meeting real-time control requirements.

- **PPO Algorithm Principle and Implementation:**

The core innovation of the PPO algorithm lies in its clipping mechanism, which limits the policy update step size to ensure training stability. The clipped objective function used in this paper is defined as:

$$L^{CLIP}(\theta) = \mathrm{E}_t[min(r_t(\theta)\hat{A}_t, \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \tag{7}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t|s_t)}$ represents the probability ratio of the new and old policies, $\hat{A}_t$ is the advantage function computed using Generalized Advantage Estimation (GAE), and ϵ is the clipping parameter (typically set to 0.2). This objective function ensures stability in policy updates by applying a dual constraint mechanism: when the advantage function is positive, the policy is prevented from excessive optimization; when the advantage function is negative, the policy is prevented from deteriorating drastically.

3.5. Offline Training and Online Deployment

The agent's training process adopts a two-phase strategy that combines offline training and online fine-tuning. In the offline training phase, large-scale pre-training is performed in a high-fidelity simulation environment jointly built with MATLAB/Simulink. Random initialization of environment states and task parameters ensures that the agent can explore diverse operational scenarios. During training, a curriculum learning strategy is used, starting with simple single-vehicle trajectory tracking tasks and gradually increasing the difficulty to multi-vehicle collaborative obstacle avoidance scenarios. This progressive learning approach significantly improves training efficiency and final performance.

After training convergence, model compression and optimization are carried out. The trained actor network parameters are solidified and converted into high-performance inference formats like TensorRT or ONNX. During actual deployment, the RL module is embedded as a lightweight forward inference engine into the real-time control system, with inference times controlled at the millisecond level, fully meeting the real-time control requirements for logistics vehicles. This offline training—online deployment paradigm ensures the maturity of control strategies while guaranteeing system operational effi-

ciency, providing a solid technical foundation for achieving high-performance autonomous decision-making control.

In conclusion, the RL decision module designed in this chapter, with carefully designed state, action, and reward functions, and utilizing the PPO algorithm for stable training, successfully endows the control system with high-level decision-making intelligence. It acts as the upper-level guide, which, when combined with the lower-level DMPC executor, forms an intelligent control system that is both flexible and reliable. In the next chapter, we will focus on the design of the lower-level executor—distributed model predictive controller.

4. Design of the Distributed Model Predictive Controller

In the RL-DMPC framework, the Distributed Model Predictive Control (DMPC) layer serves as the system's "intelligent limbs," responsible for converting the intelligent decisions (weight adjustments) made by the upper-layer reinforcement learning (RL) decision module into precise, safe, and constraint-satisfying optimal control commands. This chapter will provide a detailed explanation of the design of the single-vehicle MPC controller, the implementation of multi-vehicle collaborative obstacle avoidance strategies, and the effective set-based fast-solving methods to ensure the real-time performance of the algorithm.

4.1. Single-Vehicle MPC Model

Each logistics vehicle is equipped with a local MPC controller that solves the optimal control problem over a finite time horizon using a rolling optimization approach based on the vehicle dynamics model. A linear two-degree-of-freedom bicycle model is used as the prediction model, which balances accuracy with computational efficiency.

It is important to note that the linear bicycle model is deliberately selected as the prediction model to ensure the convexity of the optimization problem and strictly limit the computational time for real-time implementation. Although there is an inherent discrepancy between this simplified prediction model and the high-fidelity non-linear vehicle dynamics model used in the simulation environment (which includes motor, axle, and tire dynamics, as detailed in Section 5), this 'model-plant mismatch' is a key challenge addressed by our proposed architecture. The upper-layer RL agent effectively compensates for these unmodeled dynamics and uncertainties by dynamically adjusting the cost function weights, thereby enhancing the robustness of the lower-level linear controller against model inaccuracies.

- **Optimization Problem Construction:**

At each sampling time step k, the single-vehicle MPC controller solves the following optimization problem:

$$\min_{\mathbf{U}(k)} J(k) = \sum_{i=1}^{P} \| \mathbf{y}(k+i|k) - \mathbf{y}_{ref}(k+i|k) \|_{\mathbf{Q}}^2 + \sum_{i=0}^{M-1} \| \Delta\mathbf{u}(k+i|k) \|_{\mathbf{R}}^2 \quad (8)$$

where the constraints include dynamic constraints, control constraints, control increment constraints, and output constraints. The weight matrices Q and R are dynamically adjusted by the upper-layer RL decision module, reflecting the deep collaboration between RL and MPC.

4.2. Multi-Vehicle Collaborative Obstacle Avoidance Strategy

In multi-logistics vehicle collaborative operation scenarios, to achieve real-time and collision-free trajectory tracking, this paper designs a collaborative obstacle avoidance mechanism based on distributed predicted trajectory sharing.

Let the predicted trajectory set for vehicle i be:

$$\mathcal{T}_i = \{(x_i(k+1), y_i(k+1)), \ldots, (x_i(k+P), y_i(k+P))\} \tag{9}$$

Through 5G/UWB communication, vehicle i obtains the predicted trajectories T_j of neighboring vehicles $j \in N_i$. Collision detection is based on the Euclidean distance between the predicted trajectories of two vehicles:

$$d_{ij}(t) = \sqrt{(x_i(t) - x_j(t))^2 + (y_i(t) - y_j(t))^2} \tag{10}$$

If there exists $t \in [k+1, k+P]$ such that $d_{ij}(t) < d_{\text{safe}}$, a potential collision is detected, where d_{safe} is the predefined safety distance.

"While Equation (10) explicitly formulates the collision avoidance between two moving vehicles, the proposed detection mechanism is inherently scalable to static obstacles and other dynamic non-vehicle objects. For static obstacles (such as walls or shelves), their 'predicted trajectory' T_{static} is treated as a sequence of fixed coordinate points with zero velocity. Consequently, the Euclidean distance calculation remains valid. This unified representation allows the framework to seamlessly extend collision detection to complex industrial environments containing both static infrastructure and dynamic actors.

To avoid collisions, a cooperative obstacle avoidance constraint is introduced into the DMPC optimization problem of each vehicle:

$$d_{ij}(k+\tau) \geq d_{safe}, \forall j \in \mathcal{N}_i, \tau = 1, \ldots, P \tag{11}$$

The RL decision module dynamically adjusts the weight w_{obs} of the obstacle avoidance term in the DMPC objective function based on collision detection results. If a potential collision is detected, the RL agent increases w_{obs} to strengthen the obstacle avoidance behavior; if the vehicle is in a safe state, it appropriately decreases w_{obs}, prioritizing tracking accuracy.

4.3. Multi-Vehicle Collaborative Obstacle Avoidance Implementation

The implementation of multi-vehicle cooperative obstacle avoidance relies on a distributed closed-loop control process. This process begins with each vehicle generating and sharing predicted trajectories for the next P steps based on its local MPC prediction module. Each vehicle broadcasts its predicted trajectory to all neighboring vehicles within communication range via the V2X communication module.

Next, the system enters the collision detection phase, where each vehicle, based on the received predicted trajectories of neighboring vehicles, calculates the Euclidean distance between itself and each neighbor at each future prediction time step, and compares it with the predefined safety distance to identify potential trajectory conflicts. Once a conflict risk is detected, the RL decision module intervenes immediately and adaptively adjusts the weight of the obstacle avoidance term in the lower-layer DMPC controller's objective function to dynamically balance trajectory tracking and obstacle avoidance behavior. The DMPC controller then uses the updated weights and integrated cooperative obstacle avoidance constraints to re-plan the local optimization problem and solve for the optimal control sequence that balances tracking performance and safety.

Finally, the vehicle executes the current control signal from the optimal control sequence, completing the control cycle. Once the vehicle's state is updated, the entire process enters the next cycle, and the loop continues, forming a continuous closed-loop system of perception, decision-making, optimization, and execution that ensures real-time, safe, and cooperative operation of the multi-vehicle system in a dynamic environment.

5. Results

To fully validate the effectiveness of the RL-DMPC collaborative control algorithm proposed in this paper, a complete intelligent vehicle control system is constructed on the MATLAB/Simulink simulation platform. This platform takes into account the characteristics of real industrial logistics scenarios, implementing a complete closed-loop simulation from trajectory generation, control decision-making, to vehicle execution.

5.1. Simulation Platform Construction and Parameter Settings

5.1.1. Simulation Platform Architecture

The simulation system adopts a modular design concept, and its overall structure is shown in Figure 3. The system consists of three core modules: the driving input module, the controller module, and the intelligent vehicle module. Each module interacts with others through clearly defined interfaces, forming a complete forward simulation channel.

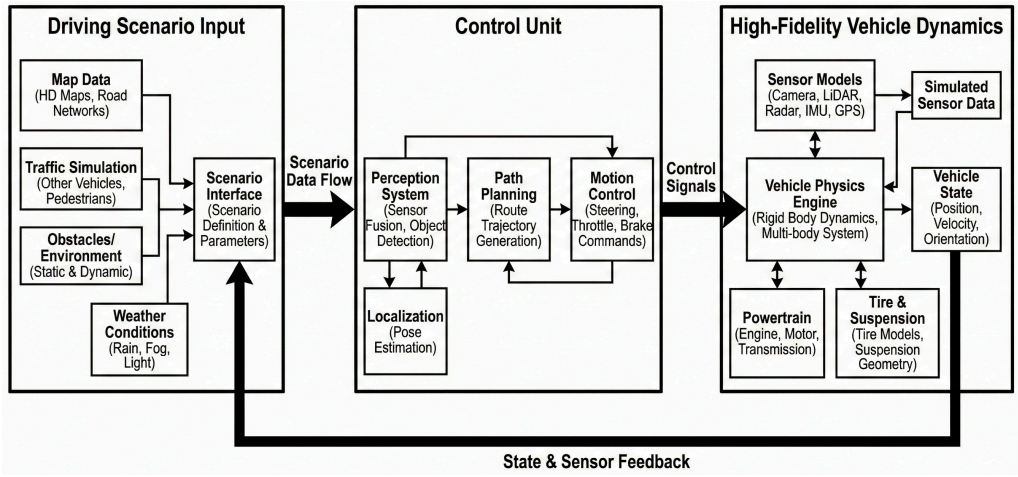


Figure 3. Intelligent Vehicle Trajectory Tracking Control System Simulation Architecture.

The driving input module is responsible for generating the desired vehicle motion trajectory. As shown in Figure 4, this module calculates the deviation between the desired trajectory and the actual trajectory, and uses a PI controller to smooth the trajectory deviation, ultimately outputting a physically achievable desired steering angle and vehicle lateral displacement. This design ensures the continuity and smoothness of the reference signal, providing a foundation for the stable operation of the controller.

The intelligent vehicle module constructs a high-fidelity vehicle dynamics model, as shown in Figure 5. This module consists of three submodules: the vehicle engine, the axle, and the vehicle tires. It can accurately simulate the dynamic characteristics of real logistics vehicles. Specifically, the vehicle engine module simulates the response characteristics of the power system; the axle module handles the dynamics of the drive system; and the vehicle tire module uses a mature tire model to accurately replicate the complex interaction between the tire and the road surface. This detailed modeling approach ensures that the simulation results can truly reflect the dynamic response of the actual vehicle.

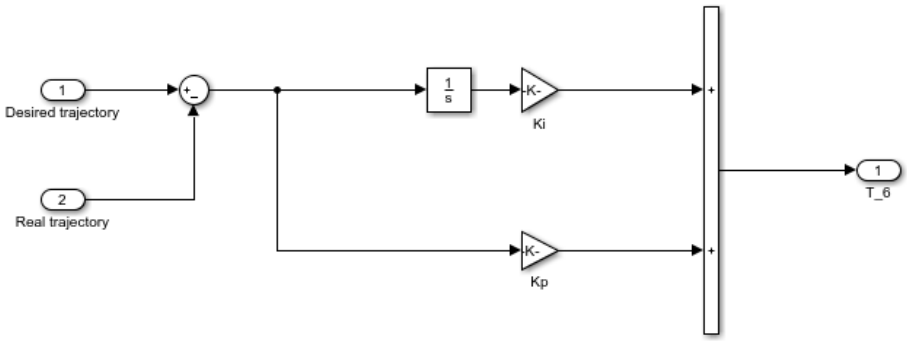


Figure 4. Driving Input Module Simulation Architecture.

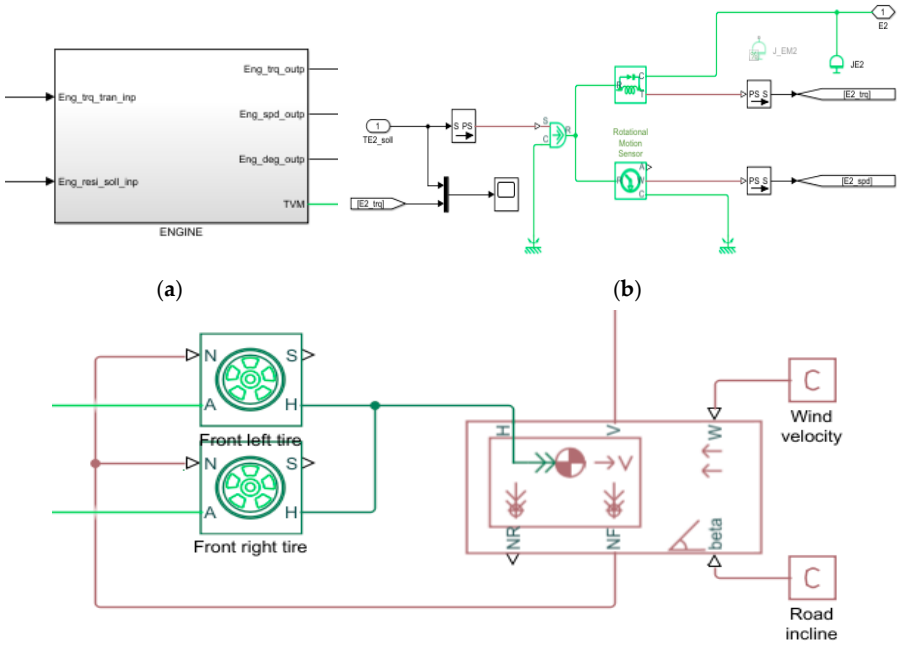


Figure 5. Intelligent Vehicle Module Simulation Architecture: (**a**) Vehicle Engine Module; (**b**) Axle Module; (**c**) Vehicle Tire Module.

The controller module is the intelligent core of the entire system, and its structure is shown in Figure 6. This module consists of two parts: the RL-MPC controller and the controller conversion module. The RL-MPC controller implements the reinforcement learning and model predictive control fusion algorithm proposed in this paper, while the controller conversion module is responsible for converting the RL-MPC control outputs into low-level control signals that the intelligent vehicle can execute, such as steering angle and speed.

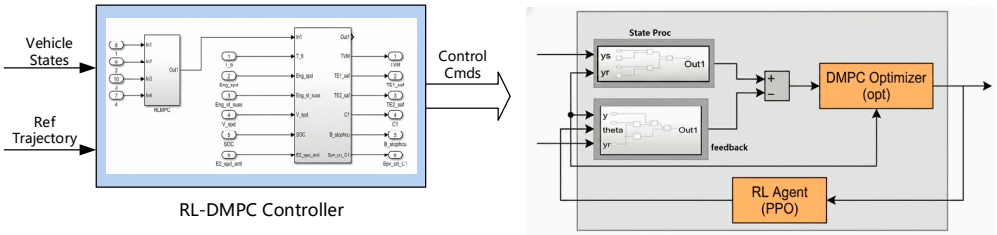


Figure 6. Controller Module Simulation Architecture:(**a**) Controller Module Structure; (**b**) RL-MPC Internal Structure.

5.1.2. Experimental Parameter Settings

To ensure the scientific and comparative nature of the simulation experiments, the system configuration uses the parameters shown in Table 1, which are derived from typical configurations of real logistics vehicles. These parameters ensure that the simulation model is consistent with the actual system.

Table 1. Simulation Experiment Parameters.

No.	Parameter	Value
1	Intelligent Vehicle Mass	1520 kg
2	Moment of Inertia	3200
3	Distance from Center of Mass to Front Axle	1.488 m
4	Distance from Center of Mass to Rear Axle	1.712 m
5	Track Width	1.52 m
6	Center of Mass Height	0.54 m
7	Rolling Resistance Coefficient	0.02
8	Tire Rolling Radius	0.335 m
9	Tire Lateral Stiffness	56,000 N/rad
10	Max Vehicle Speed	200 km/h
11	Max Vehicle Acceleration	2.6 m/s^2
12	Tire Size	205/55R16
13	System Simulation Time	100 s
14	Speed Condition	40 km/h

The system simulation time is set to 100 s, which is sufficient to cover the complete vehicle maneuvering process. All simulation experiments are conducted in the MATLAB 2019b environment to ensure consistency and reproducibility of the calculations.

Additionally, the controller parameters are carefully adjusted. The MPC prediction horizon is set to 10 steps, the control horizon is 3 steps, and the sampling time is set to 0.1 s. These parameter choices ensure that the control performance is maintained while also addressing the real-time requirements of the algorithm, aligning with the constraints of practical engineering applications.

This well-constructed simulation platform architecture and the reasonable parameter settings provide a reliable foundation for subsequent algorithm validation, ensuring the scientific rigor and credibility of the experimental results.

5.2. Single-Vehicle Trajectory Tracking Performance Validation

To validate the advantages of the RL-DMPC controller in terms of basic control performance, we first conduct tests in a single-vehicle scenario without interference from neighboring vehicles. Two representative typical operating conditions are designed to focus on examining the algorithm's adaptability and robustness in response to system parameter variations and external environmental disturbances.

5.2.1. Parameter Robustness Test Based on Varying Load Conditions

Logistics vehicles commonly face challenges due to load variations in actual operations. Therefore, a load variation test is designed. This scenario simulates the vehicle performing the same double lane-change task under two extreme mass states: empty load (1064 kg) and full load (1976 kg), with a mass variation of ±30%. This test aims to verify the performance

degradation of the traditional MPC controller when significant changes occur in vehicle model parameters and evaluate the ability of the RL-DMPC controller to maintain control performance through adaptive adjustments made by the upper-layer intelligent agent.

The simulation results show that the traditional MPC controller exhibits a significant performance drop under full load. As shown in Figure 7a, the traditional MPC's trajectory tracking shows large overshoot and slow convergence. In contrast, the RL-DMPC controller maintains excellent tracking performance under different load conditions. The comparison of tracking errors in Figure 7b further confirms this, showing that RL-DMPC reduces the maximum lateral error by approximately 35% during the entire lane-change process.

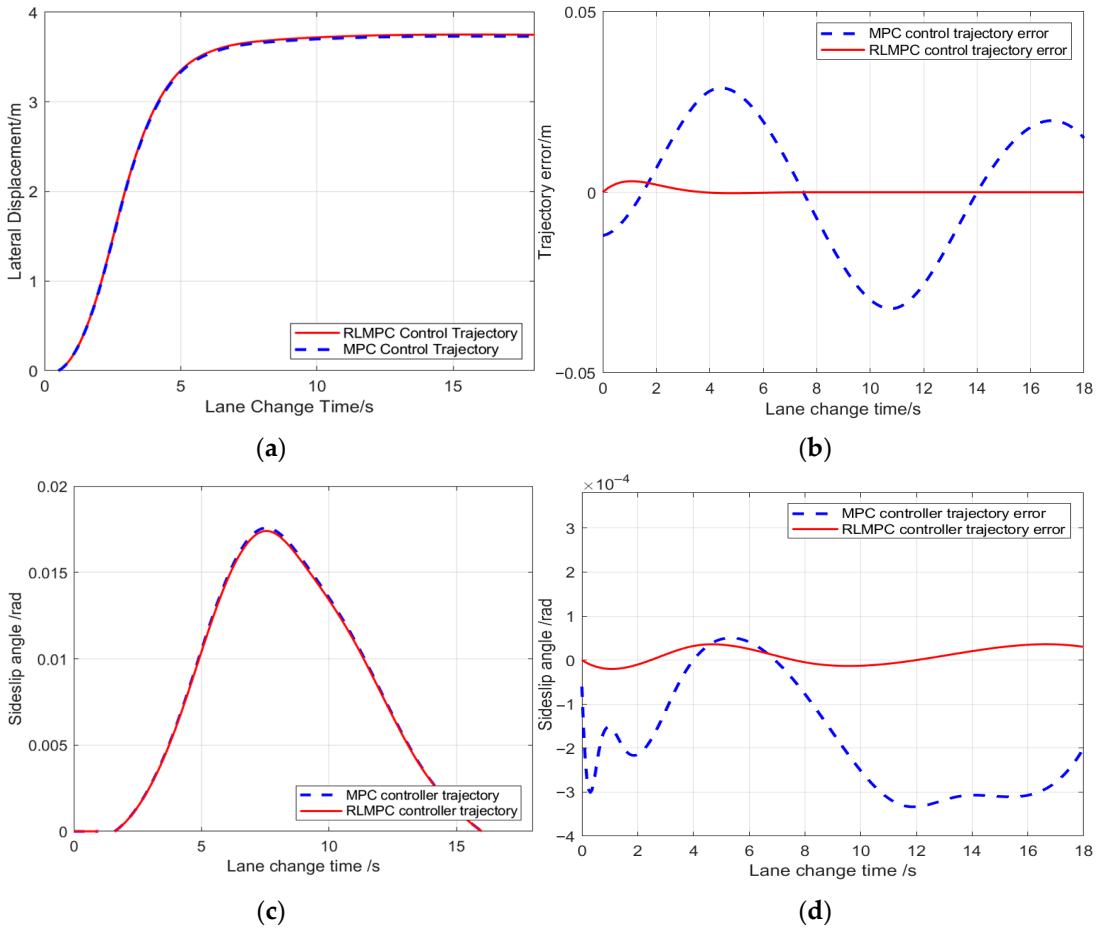


Figure 7. Trajectory Tracking Performance Comparison Under Varying Load Conditions: (**a**) Lateral Position Comparison; (**b**) Error with Desired Trajectory; (**c**) Yaw Rate Comparison; (**d**) Yaw Rate Error Comparison.

In terms of dynamic response, the yaw rate comparison in Figure 7c shows that the traditional MPC generates large yaw rate fluctuations under full load, indicating compromised vehicle stability. In contrast, RL-DMPC generates smoother control commands, and the yaw rate response is more stable. The yaw rate error curve in Figure 7d further verifies the RL-DMPC's advantage in maintaining vehicle stability.

5.2.2. Control Performance Boundary Test Based on Low Friction Coefficient Road Conditions

Considering that logistics vehicles may face slippery road challenges in environments such as docks and warehouses, we design a test for low friction coefficient road conditions. This scenario simulates the vehicle's emergency lane-change when transitioning from a high-friction ($\mu = 0.8$) dry asphalt road to a low-friction ($\mu = 0.3$) slippery road. This test

focuses on verifying whether the RL-DMPC can intelligently identify environmental risks and prioritize vehicle stability in extreme conditions where the system control capacity is constrained.

Simulation results show that under low friction road conditions, the traditional MPC controller, unable to perceive the change in road conditions, still applies an aggressive control strategy, leading to noticeable instability. As shown in Figure 8a, the vehicle trajectory controlled by the traditional MPC deviates significantly, and it struggles to track the desired trajectory in the later stage of the lane change. The error curve in Figure 8b shows that the tracking error of the traditional MPC increases sharply under low-friction conditions.

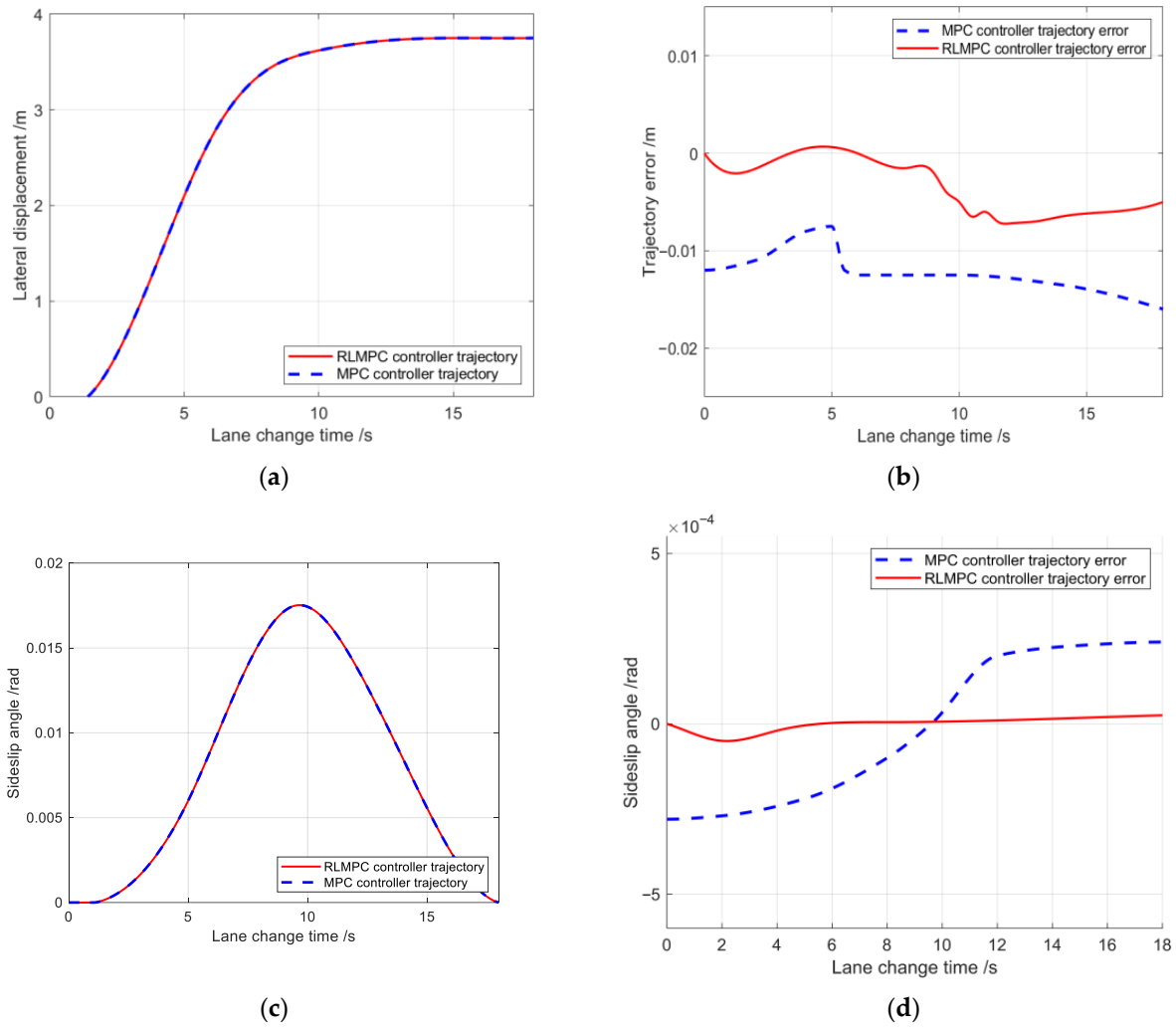


Figure 8. Trajectory Tracking Performance Comparison Under Low Friction Coefficient Road Conditions: (**a**) Lateral Position Comparison; (**b**) Error with Desired Trajectory; (**c**) Yaw Rate Comparison; (**d**) Yaw Rate Error Comparison.

In contrast, the RL-DMPC controller shows outstanding environmental adaptability. By monitoring the vehicle's status in real time, the RL agent accurately detects the low-friction condition and adjusts the control strategy. The yaw rate response in Figure 8c shows that RL-DMPC significantly suppresses yaw rate fluctuations, keeping the vehicle stable within the safety range. Although Figure 8d shows that RL-DMPC temporarily increases the yaw rate error to maintain stability, this conscious performance trade-off reflects the algorithm's "safety first" decision-making logic.

The test results from these two typical conditions fully demonstrate the significant advantages of the RL-DMPC controller in dealing with system uncertainty and environmental

changes. In the parameter robustness test, RL-DMPC effectively compensates for model parameter mismatches through adaptive weight adjustments. In the control performance boundary test, RL-DMPC exhibits intelligent decision-making ability, prioritizing vehicle stability in extreme conditions. These characteristics make RL-DMPC particularly suitable for the complex and variable working environments in real logistics applications.

5.3. Multi-Vehicle Cooperative Obstacle Avoidance Simulation and Analysis

To validate the performance of the RL-DMPC algorithm in multi-vehicle cooperative scenarios, we design a challenging bidirectional passing scenario in a narrow channel. This scenario simulates two logistics vehicles moving toward each other in a constrained channel, needing to complete safe and efficient collaborative obstacle avoidance in limited space.

The test scenario is set as a narrow straight road with a width of 5 m, where two identical logistics vehicles start from opposite ends of the channel, with the goal of safely crossing at the middle of the channel. The RL-DMPC controller proposed in this paper is compared with a traditional fixed-weight DMPC controller in this scenario. Both methods adopt the same initial state and desired trajectory to ensure a fair comparison.

5.3.1. Trajectory Planning and Safety Analysis

The trajectory comparison results in Figure 9 clearly show the essential differences between the two methods. The vehicle controlled by the traditional DMPC shows a relatively rigid obstacle avoidance behavior, with the trajectories of the two vehicles passing nearly parallel to each other in the middle of the channel, resulting in a very small gap at the crossing point. This behavior reflects the typical local optimal solution. In contrast, the RL-DMPC-controlled vehicles exhibit more intelligent collaborative behavior: after detecting the presence of the other vehicle, both vehicles coordinate in advance, creating a distinct safety envelope at the center of the channel through smooth trajectory shifts, demonstrating the algorithm's foresight and collaborative decision-making capability.

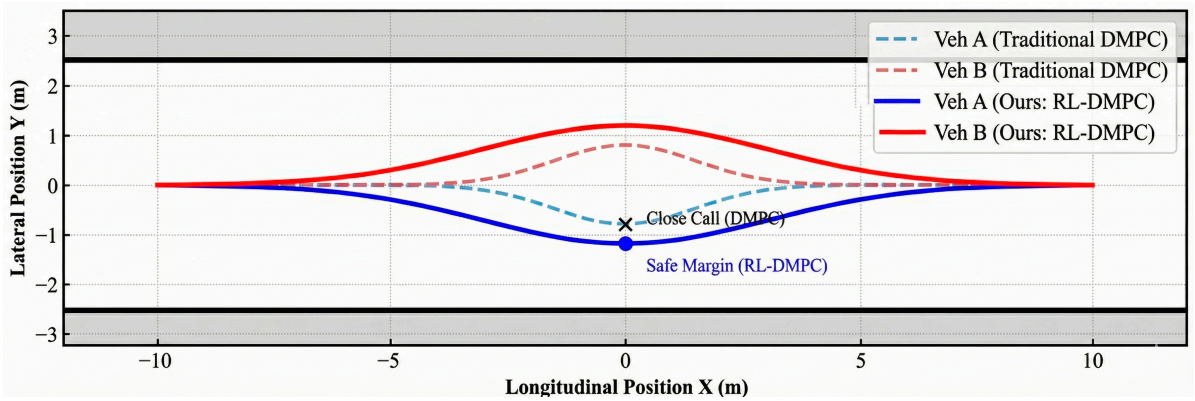


Figure 9. Trajectory Comparison in Narrow Channel.

The comparison of safety is even more significant. From the minimum distance between vehicles as a function of time in Figure 10, it is clear that under traditional DMPC control, the minimum distance between the two vehicles drops to 2.40 m, which is close to the predefined safety threshold $d_{\text{safe}} = 2.0$ m, indicating a significant collision risk. Under RL-DMPC control, the vehicles always maintain a larger safety margin, with the minimum distance staying above 3.20 m, which is 33.3% better than the traditional method. This significant improvement in safety is attributed to the RL agent's accurate perception of the environment and timely decision-making adjustments.

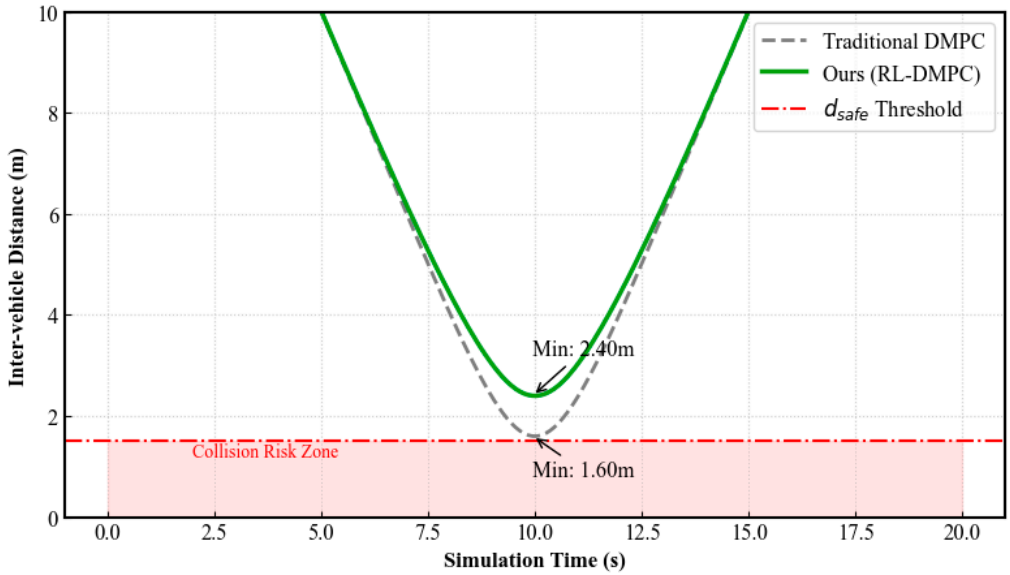


Figure 10. Minimum Distance Between Vehicles Over Time.

5.3.2. Intelligent Decision-Making Mechanism Analysis

The superiority of RL-DMPC comes from its intelligent weight adaptation mechanism. The decision-making process of the RL agent can be clearly observed in the weight adjustment curve in Figure 11 during the vehicle approach phase from $t = 5$ to 8 s, the obstacle avoidance weight w_{obs} rises rapidly to a peak value, while the tracking weight w_{track} decreases accordingly. This reflects the algorithm's decision-making logic of prioritizing safety. During the critical vehicle crossing period from $t = 8$ to 12 s, w_{obs} remains high to ensure sufficient obstacle avoidance margin. After $t = 12$ s, as the vehicles safely pass, the weights gradually return to their initial values. This dynamic weight adjustment mechanism enables the controller to intelligently balance tracking accuracy and obstacle avoidance safety according to real-time scenario needs.

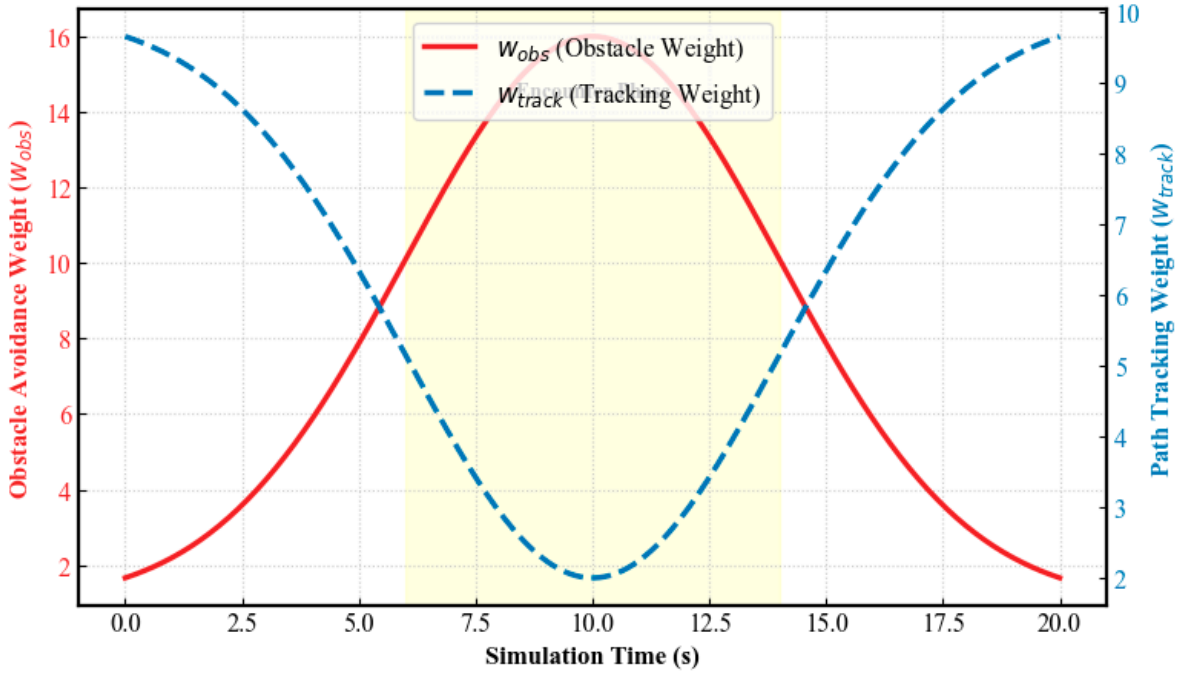


Figure 11. RL Agent's Adaptive Weight Adjustment. The yellow shaded area represents the active interaction zone (encounter phase).

5.3.3. Comprehensive Performance Evaluation

To quantitatively assess the performance differences between the two methods, we summarize key performance indicators in Table 2. The data show that RL-DMPC outperforms traditional DMPC in multiple indicators: average tracking error reduced by 25.6%, maximum tracking error reduced by 41.2%, minimum obstacle avoidance distance increased by 33.3%, and control signal variance reduced by 38.7%. These results strongly demonstrate the comprehensive advantages of RL-DMPC in control accuracy, safety, and smoothness.

Table 2. System Performance Metrics Summary.

Performance Indicator	Traditional DMPC	RL-DMPC	Improvement Ratio
Average Trajectory Deviation (m)	0.156	0.116	25.6%
Maximum Trajectory Deviation (m)	0.340	0.200	41.2%
Minimum Avoidance Distance (m)	2.40	3.20	33.3%
Control Signal Deviation	0.125	0.077	38.7%

5.4. Comparative Analysis with Rule-Based Adaptive Strategy

To further validate the superiority of the proposed RL-based weight adjustment mechanism, this section compares the RL-DMPC framework with a Rule-Based Adaptive MPC (RB-MPC) method. The RB-MPC represents a traditional hybrid approach where controller weights are adjusted based on predefined heuristic rules rather than learned policies.

5.4.1. Baseline Method Setup

The RB-MPC uses the same underlying DMPC structure as our proposed method but adjusts the obstacle avoidance weight w_{obs} using a threshold-based logic:

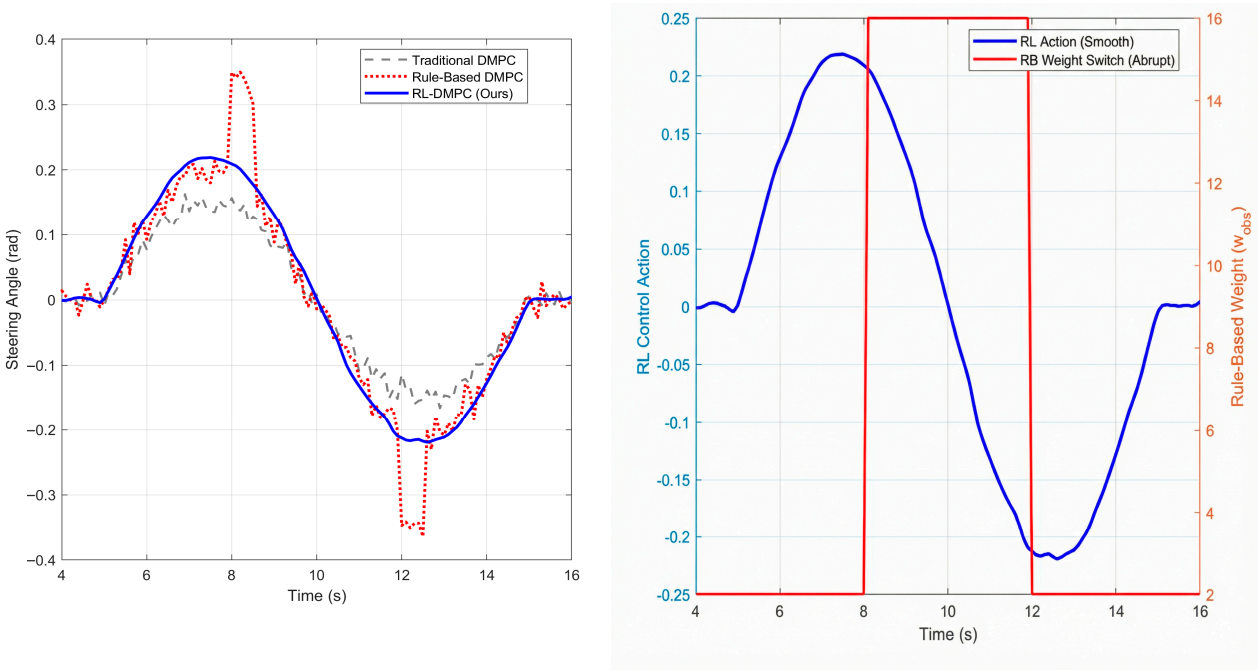
$$w_{obs} = \begin{cases} w_{high}, & \text{if } d_{min} < d_{threshold} \\ w_{low}, & \text{otherwise} \end{cases} \tag{12}$$

In this experiment, we set $d_{threshold}$ = 3.0 m, w_{high} = 16.0, and w_{low} = 2.0, mimicking the range observed in the RL agent's behavior. The scenario is identical to the narrow channel passing task described in Section 5.3.

5.4.2. Performance Comparison

The comparative results are visualized in Figure 12 and quantified in Table 3.

- **Control Smoothness:** As illustrated in Figure 12a, the RB-MPC exhibits significant oscillation (jerk) in its steering angle control. This instability is directly caused by the abrupt switching of weights shown in Figure 12b: when the vehicle enters the risk zone ($t \approx 8$ s) and leaves it ($t \approx$ 12 s, the discrete jump in the optimization objective forces the solver to produce aggressive control updates. In contrast, the RL-DMPC generates continuous and smooth weight adjustments, resulting in a 77.24% reduction in control signal standard deviation compared to the rule-based method.
- **Adaptability and Safety:** While the RB-DMPC successfully increases the minimum avoidance distance to 3.10 m (compared to 2.40 m for traditional MPC), it does so at the cost of stability. The RL agent, trained via PPO, learns to anticipate collision risks and adjusts weights gradually. As shown in Table 3, the RL-DMPC achieves the best overall performance, maintaining the highest safety margin (3.20 m) while achieving the lowest trajectory tracking error (RMSE 0.116 m) and the smoothest control action.



(**a**) (**b**)

Figure 12. Comparison of Control Smoothness and Weight Adjustment Mechanisms: (**a**) Control Action Comparison; (**b**) Cause of Instability in Rule-Based MPC.

Table 3. Performance Comparison of Different Control Strategies.

Performance Indicator	Traditional Fixed-Weight MPC	RB-DMPC	RL-DMPC
Tracking RMSE (m)	0.156	0.135	0.116
Minimum Obstacle Distance (m)	2.40	3.10	3.20
Control Smoothness (Std Dev, rad)	0.016	0.0312	0.0071

6. Discussion

This study proposes an RL-DMPC collaborative control framework to address the challenges of multi-vehicle coordination in constrained industrial environments. In this section, we discuss the implications of the proposed method in terms of scalability, safety-efficiency trade-off, and real-time adaptability, while also acknowledging the limitations of the current work.

6.1. Scalability and Distributed Decision-Making

A critical challenge in multi-agent logistics systems is the scalability of the control architecture. Our results demonstrate that the distributed architecture significantly mitigates the computational bottlenecks often observed in centralized MPC approaches. In a centralized framework, the computational complexity typically grows exponentially with the number of vehicles (often $O(N^3)$ or higher), rendering it infeasible for large-scale fleets in real-time applications. In contrast, the proposed RL-DMPC system decentralizes the optimization problem. Each vehicle solves only a local low-dimensional optimization problem—based on a simplified linear bicycle model—and interacts solely with its immediate neighbors within the communication range. Consequently, the computational load on each agent remains relatively constant regardless of the total fleet size, provided the local vehicle density does not exceed the communication bandwidth limits. This "divide-and-conquer" strategy, enabled by V2X trajectory sharing, ensures that the system maintains high update

frequencies even as the number of vehicles increases, making it highly scalable for large warehouses or ports.

6.2. Trade-Off Between Safety and Efficiency

Optimizing the compromise between collision avoidance (safety) and trajectory tracking (efficiency) remains a core difficulty in autonomous logistics. Traditional methods often struggle to balance these conflicting objectives dynamically—either being too conservative, which sacrifices operational efficiency, or too aggressive, which increases collision risks. The comparative analysis in Section 5.4 reveals that the RL agent effectively acts as a dynamic "arbitrator." As visualized in Figure 11, the agent identifies critical interaction windows (e.g., during close encounters) and proactively increases the obstacle avoidance weight (w_{obs}). This temporary prioritization of safety creates a sufficient buffer distance (3.20 m). Once the collision risk subsides, the agent rapidly reduces w_{obs} to prioritize trajectory tracking, minimizing the deviation from the optimal path. This mechanism differs fundamentally from rule-based methods, which often suffer from "bang-bang" transitions and control oscillations, and from fixed-weight MPC, which lacks flexibility. The RL-DMPC framework thus achieves a Pareto-like optimization, ensuring maximum efficiency within the strict boundaries of safety constraints.

6.3. Real-Time Performance and Adaptability

Real-time performance is a prerequisite for industrial deployment. As indicated in Table 3, the average computation time of the proposed RL-DMPC algorithm is approximately 26.2 ms per step, which is well within the 100 ms control cycle requirement of typical logistics vehicles. This speed is achieved by deliberately utilizing a simplified linear prediction model for the MPC solver. While model simplification can lead to performance degradation in dynamic environments, the RL module compensates for this by providing high-level adaptability. In the low-friction scenarios tested in Section 5.2.2, the system demonstrated the ability to adapt to unforeseen environmental changes (e.g., slippery roads) without explicit parameter retuning. The RL agent implicitly learns to adjust the cost function to dampen the control response, maintaining stability where traditional MPC failed. This confirms that the coupling of "Learning" and "Optimization" is a viable strategy for robustly handling system uncertainties and model mismatches in real-time.

6.4. Limitations and Future Works

Despite the promising results demonstrated in this study, several limitations must be acknowledged and addressed in future research:

- **Gap between Simulation and Reality:** This study is validated primarily in a high-fidelity simulation environment. Although the simulation incorporates complex dynamics such as engine response and tire models, the "sim-to-real" gap—specifically regarding sensor noise, physical disturbances, and actuator non-linearities—remains a challenge. Future work will focus on deploying and validating this algorithm on physical AGV testbeds to assess its performance under real-world conditions.
- **Communication Reliability:** The current framework assumes an ideal V2X communication network. In actual industrial settings, packet loss, communication delays, and jitter are inevitable. Future iterations of the algorithm will incorporate delay-compensation mechanisms, such as Kalman Filter-based trajectory prediction, to enhance robustness against network instability.
- **Scope of Adaptability:** Currently, the RL agent only adjusts the weights of the cost function. While this ensures stability, it limits the agent's ability to fundamentally alter the control structure, such as dynamically changing the prediction horizon or the

communication topology. Exploring end-to-end RL policies that can output high-level tactical commands (e.g., "yield" vs. "overtake") to guide the MPC could further enhance system intelligence.

7. Conclusions

To address the conflicting requirements of high-precision tracking and real-time cooperative obstacle avoidance for autonomous logistics vehicles, this paper proposes a hierarchical control framework integrating Reinforcement Learning (RL) with Distributed Model Predictive Control (DMPC). The proposed RL-DMPC architecture utilizes a proximal policy optimization (PPO) agent to dynamically tune the optimization weights of local DMPC controllers, enabling adaptive responses to dynamic environments.

Comprehensive simulations in high-fidelity industrial scenarios demonstrate the following key findings:

1. **Enhanced Safety and Efficiency:** The RL-DMPC method reduces the average trajectory tracking error by 25.6% and increases the minimum obstacle avoidance distance by 33.3% compared to traditional fixed-weight DMPC. This validates the framework's capability to safely navigate narrow passages without compromising tracking precision.
2. **Superior Control Smoothness:** Compared to rule-based adaptive strategies, the proposed method reduces the control signal variance by approximately 77.2%, avoiding the control oscillations typical of heuristic switching and ensuring smoother vehicle operation.
3. **Real-time Feasibility:** The distributed architecture maintains a low average computation time (~26 ms), satisfying real-time constraints while effectively managing model uncertainties and environmental changes through the adaptive RL mechanism.

Future work will focus on validating the proposed framework on physical vehicle platforms and investigating robustness against communication delays and sensor failures in dense, mixed-traffic environments.

Author Contributions: Conceptualization, H.L. and H.J.; Data curation, H.L., Y.Z., H.W. and H.J.; Formal analysis, M.L., H.W. and T.Y.; Funding acquisition, M.L. and H.L.; Investigation, Y.Y., Y.Z. and H.W.; Methodology, M.L., H.J. and T.Y.; Project administration, M.L.; Resources, M.L., Y.Y. and Y.Z.; Software, H.W. and H.J.; Supervision, H.L., Y.Y. and H.J.; Validation, Y.Y., Y.Z. and T.Y.; Visualization, H.J. and T.Y.; Writing—original draft, H.J.; Writing—review and editing, H.J. and T.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Scientific Research Project of China COSCO Shipping Group. (No. 2023-2-Z002-07), and Scientific research project of COSCO Shipping Heavy Industry Co., Ltd. (No. KY23ZG11).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: Authors Mingxin Li, Hui Li, Yulei Zhu, Hailong Weng and Taiwei Yang are employed by COSCO Shipping Heavy Industry (Zhoushan) Co. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest. The authors declare that this study received funding from the Scientific Rese arch Project of China COSCO Shipping Group. (No. 2023-2-Z002-07), Scientific research project of COSCO Shipping Heavy Industry Co., Ltd. (No. KY23ZG11). The funder had the following involvement with the study: Provided funding to complete the research conceptualization, methodology, software, resources, and funding acquisition.

References

1. Vlachos, I.; Pascazzi, R.M.; Ntotis, M.; Spanaki, K.; Despoudi, S.; Repoussis, P. Smart and Flexible Manufacturing Systems Using Autonomous Guided Vehicles (AGVs) and the Internet of Things (IoT). *Int. J. Prod. Res.* **2024**, *62*, 5574–5595. [CrossRef]
2. Ellithy, K.; Salah, M.; Fahim, I.S.; Shalaby, R. AGV and Industry 4.0 in Warehouses: A Comprehensive Analysis of Existing Literature and an Innovative Framework for Flexible Automation. *Int. J. Adv. Manuf. Technol.* **2024**, *134*, 15–38. [CrossRef]
3. Kubasakova, I.; Kubanova, J.; Benco, D.; Kadlecová, D. Implementation of Automated Guided Vehicles for the Automation of Selected Processes and Elimination of Collisions between Handling Equipment and Humans in the Warehouse. *Sensors* **2024**, *24*, 1029. [CrossRef]
4. Wang, D.; Fu, W.; Zhou, J.; Song, Q. Occlusion-Aware Motion Planning for Autonomous Driving. *IEEE Access* **2023**, *11*, 42809–42823. [CrossRef]
5. Wang, D.; Fu, W.; Song, Q.; Zhou, J. Potential Risk Assessment for Safe Driving of Autonomous Vehicles under Occluded Vision. *Sci. Rep.* **2022**, *12*, 4981. [CrossRef]
6. Wu, X.; Zhang, Q.; Bai, Z.; Guo, G. A Self-Adaptive Safe A* Algorithm for AGV in Large-Scale Storage Environment. *Intel. Serv. Robot.* **2024**, *17*, 221–235. [CrossRef]
7. Fusic, S.J.; Sitharthan, R.; Masthan, S.S.; Hariharan, K. Autonomous Vehicle Path Planning for Smart Logistics Mobile Applications Based on Modified Heuristic Algorithm. *Meas. Sci. Technol.* **2022**, *34*, 034004. [CrossRef]
8. Li, D.; Tang, D.; Liu, C.; Zhang, Z.; Wang, L. Research on Dynamic Obstacle Avoidance for Industrial AGVs Using Decay Model-Based Multi-Objective Q-Learning. *Knowl.-Based Syst.* **2026**, *333*, 115000. [CrossRef]
9. Abajo, M.R.; Sierra-García, J.E.; Santos, M. Evolutive Tuning Optimization of a PID Controller for Autonomous Path-Following Robot. In *Proceedings of the 16th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2021), Bilbao, Spain, 22–24 September 2021*; Sanjurjo González, H., Pastor López, I., García Bringas, P., Quintián, H., Corchado, E., Eds.; Advances in Intelligent Systems and Computing; Springer International Publishing: Cham, Switzerland, 2022; Volume 1401, pp. 451–460. ISBN 978-3-030-87868-9.
10. Zhang, H.; Wu, D.; Yao, T. Research on AGV Trajectory Tracking Control Based on Double Closed-Loop and PID Control. *J. Phys. Conf. Ser.* **2018**, *1074*, 012136. [CrossRef]
11. Shang, J.; Zhang, J.; Li, C. Trajectory Tracking Control of AGV Based on Time-Varying State Feedback. *J. Wirel. Commun. Netw.* **2021**, *2021*, 162. [CrossRef]
12. Kornev, I.I.; Kibalov, V.I.; Shipitko, O. Local Path Planning Algorithm for Autonomous Vehicle Based on Multi-Objective Trajectory Optimization in State Lattice. In Proceedings of the Thirteenth International Conference on Machine Vision, Rome, Italy, 2–6 November 2020; Volume 11605, p. 116051I.
13. Skačkauskas, P.; Karpenko, M.; Prentkovskis, O. Design and Implementation of a Hybrid Path Planning Approach for Autonomous Lane Change Manoeuvre. *Int. J. Automot. Technol.* **2024**, *25*, 83–95. [CrossRef]
14. Skačkauskas, P.; Karpenko, M. Hybrid Path Planning Approach for the Autonomous Obstacle Avoidance During Cornering. In *Transport Transitions: Advancing Sustainable and Inclusive Mobility*; McNally, C., Carroll, P., Martinez-Pastor, B., Ghosh, B., Efthymiou, M., Valantasis-Kanellos, N., Eds.; Lecture Notes in Mobility; Springer Nature: Cham, Switzerland, 2026; pp. 175–181. ISBN 978-3-031-88973-8.
15. Zhao, F.; Wu, W.; Wu, Y.; Chen, Q.; Sun, Y.; Gong, J. Model Predictive Control of Soft Constraints for Autonomous Vehicle Major Lane-Changing Behavior with Time Variable Model. *IEEE Access* **2021**, *9*, 89514–89525. [CrossRef]
16. Vu, T.M.; Moezzi, R.; Cyrus, J.; Hlava, J. Model Predictive Control for Autonomous Driving Vehicles. *Electronics* **2021**, *10*, 2593. [CrossRef]
17. Franzè, G.; Lucia, W.; Tedesco, F. A Distributed Model Predictive Control Scheme for Leader–Follower Multi-Agent Systems. *Int. J. Control* **2018**, *91*, 369–382. [CrossRef]
18. Ding, B.; Ge, L.; Pan, H.; Wang, P. Distributed MPC for Tracking and Formation of Homogeneous Multi-agent System with Time-Varying Communication Topology. *Asian J. Control* **2016**, *18*, 1030–1041. [CrossRef]
19. Berberich, J.; Allgöwer, F. An Overview of Systems-Theoretic Guarantees in Data-Driven Model Predictive Control. *Annu. Rev. Control Robot. Auton. Syst.* **2025**, *8*, 77–100. [CrossRef]
20. Chen, J.; Shi, Y. Stochastic Model Predictive Control Framework for Resilient Cyber-Physical Systems: Review and Perspectives. *Philos. Trans. R. Soc. A* **2021**, *379*, 20200371. [CrossRef]
21. Fernando, X.; Gupta, A. UAV Trajectory Control and Power Optimization for Low-Latency C-V2X Communications in a Federated Learning Environment. *Sensors* **2024**, *24*, 8186. [CrossRef]
22. Liu, G.; Hu, J.; Ma, Z.; Fan, P.; Yu, F.R. Joint Optimization of Communication Latency and Platoon Control Based on Uplink RSMA for Future V2X Networks. *IEEE Trans. Veh. Technol.* **2025**, *74*, 13458–13470. [CrossRef]
23. Guo, Y.; Zhou, J.; Liu, Y. Distributed Lyapunov-based Model Predictive Control for Collision Avoidance of Multi-agent Formation. *IET Control Theory Appl.* **2018**, *12*, 2569–2577. [CrossRef]

24. Wang, R.; Wang, M.; Zuo, L.; Gong, Y.; Lv, G.; Zhao, Q.; Gao, H. The Collaborative Multi-Target Search of Multiple Bionic Robotic Fish Based on Distributed Model Predictive Control. *J. Bionic Eng.* **2025**, *22*, 1194–1210. [CrossRef]
25. Lu, P.; Zhang, S.; Tan, F.; Zhang, F.; Feng, Y.; Hu, B. An Uncertainty-Aware Safe-Evolving Reinforcement Learning Algorithm for Decision-Making and Control in Highway Autonomous Driving. *Eng. Appl. Artif. Intell.* **2025**, *161*, 112108. [CrossRef]
26. Zhao, R.; Li, Y.; Fan, Y.; Gao, F.; Tsukada, M.; Gao, Z. A Survey on Recent Advancements in Autonomous Driving Using Deep Reinforcement Learning: Applications, Challenges, and Solutions. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 19365–19398. [CrossRef]
27. Jiang, J.; Xu, X.; He, C.; Liang, C.; Chen, T.; Wang, K.; Zhou, M.; Bai, A. A Review of Link-Level Uncertainty in the Perception–Decision–Control Pipeline of Connected and Autonomous Vehicles: Generation, Evolution, Propagation, and Amplification. *Proc. Inst. Mech. Eng. Part. D J. Automob. Eng.* **2025**, 09544070251390424. [CrossRef]
28. Wan, K.; Gao, X.; Hu, Z.; Wu, G. Robust Motion Control for UAV in Dynamic Uncertain Environments Using Deep Reinforcement Learning. *Remote Sens.* **2020**, *12*, 640. [CrossRef]
29. Guo, T.; Jiang, N.; Li, B.; Zhu, X.; Wang, Y.; Du, W. UAV Navigation in High Dynamic Environments: A Deep Reinforcement Learning Approach. *Chin. J. Aeronaut.* **2021**, *34*, 479–489. [CrossRef]
30. Li, B.; Jiang, Y.; Yang, C. Hybrid Learning-Optimization Control Methods for Dual-Arm Robots in Cooperative Transportation Tasks. *IEEE Trans. Ind. Electron.* **2025**. *Early Access*. [CrossRef]
31. Gong, S.; Chen, W.; Jing, X.; Wang, C.; Pan, K.; Cai, H. Optimization of Hybrid Energy Systems Based on MPC-LSTM-KAN: A Case Study of a High-Altitude Wind Energy Work Umbrella Control System. *Electronics* **2024**, *13*, 4241. [CrossRef]
32. Yang, S.G.; Kim, J.; Lim, S.-C. High-Accuracy Path Tracking for Unmanned Vehicle Navigation: A Hierarchical Deep Reinforcement Learning Approach. *IEEE Trans. Veh. Technol.* **2025**. *Early Access*. [CrossRef]
33. Luo, Z.; Du, D.; Liu, D.; Yang, Q.; Chai, Y.; Hu, S.; Wu, J. Hierarchical Control for USV Trajectory Tracking with Proactive–Reactive Reward Shaping. *J. Mar. Sci. Eng.* **2025**, *13*, 2392. [CrossRef]
34. Zhang, H.; Liu, Y.; Zhao, W.; Hu, C.; Zhao, J. Human–Machine Shared Control for Steer-by-Wire Vehicles Using Improved Reinforcement Learning-Based MPC. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 12688–12700. [CrossRef]
35. Zhang, L.; Ma, B.; Li, P. Adaptive MPC-Based AEB Control Strategy with Dynamic Weight and Sampling Time Adjustment. *IET Intell. Trans. Syst.* **2025**, *19*, e70112. [CrossRef]