

EPiC Series in Computing

Volume 114, 2026, Pages 206–223

Proceedings of the Workshop on Mathematical Modeling and Scientific Computing



Model-Predictive-Control-Based Trajectory Generation Enhanced with Reinforcement Learning for Formation Maneuvering

Alessandro Rossi[1], Patrick Piprek[2], and Antonio Genova[1]

[1] Sapienza University of Rome - Department of Mechanical and Aerospace Engineering, Rome, Italy
rossi.1798981@studenti.uniroma1.it, antonio.genova@uniroma1.it
[2] Airbus Defence and Space GmbH, Manching, Germany
patrick.piprek@airbus.com

Abstract

Autonomous formation maneuvering requires guidance strategies that ensure real-time operation, robustness to modeling errors, and strict safety guarantees. Linear Model Predictive Control (MPC) is a widely used approach for constrained trajectory tracking, but its performance may degrade under model mismatch and linearization errors. This paper proposes a decentralized guidance architecture in which linear MPC is augmented with a fully online, model-free Reinforcement Learning (RL) layer. The RL component acts as an external adaptive feedback correction, compensating for model inaccuracies without altering the internal MPC optimization problem or its constraint-handling structure. Simulation results on dual-agent formation scenarios based on a planar kinematic point-mass model show tracking performance improvements exceeding 90% compared to pure linear MPC. Moreover, the proposed approach enables successful resolution of safety-critical scenarios involving simultaneous obstacle avoidance and inter-agent collision constraints, while preserving the safety-oriented structure of the MPC framework.

1 Introduction

Multi-agent autonomous systems are increasingly adopted in aeronautical applications requiring cooperative maneuvering, such as formation flight and collision-aware trajectory tracking [10]. While formation-based architectures offer improved mission efficiency and robustness, they impose strict requirements on real-time feasibility, safety, and reliability under uncertainty.

Model Predictive Control (MPC) is an attractive solution for constrained trajectory tracking due to its predictive structure and explicit constraint handling [8]. However, real-time requirements often lead to simplification and linearization of the MPC internal model, which may introduce significant tracking errors and limit the recovery capability in the presence of disturbances or unmodeled dynamics [6].

Reinforcement Learning (RL) offers adaptive capabilities that can mitigate model mismatch, but its direct application in safety-critical multi-agent systems remains challenging due to exploration risks and lack of predictability [1].

Following recent trends in MPC-guided learning [4,5], this work proposes a hybrid MPC+RL architecture in which RL is embedded as an external adaptive feedback correction to a linear MPC controller. The MPC controller deploys the baseline constrained feedback and constitutes the primary mechanism for constraint handling and safety-oriented decision making, while the RL module learns online to compensate for the modeling and linearization errors. The adopted RL strategy, inspired by learning feedback linearization approaches [2], enables deterministic and safety-oriented adaptation during mission execution.

Structure: The paper presents (i) a decentralized MPC framework for multi-agent formation maneuvering with collision avoidance (presented in Sections 2 and 3), (ii) a fully online model-free RL correction guided by MPC decisions (presented in Section 4), and (iii) a simulation-based validation demonstrating significant tracking improvements in constrained multi-agent scenarios (presented in Sections 5 and 6).

2 Problem formulation

2.1 Agent model

Each agent of the formation is modeled as a planar point-mass system with kinematic dynamics. The state is defined as $\mathbf{X} = [x, y, V, \chi]^\top$, where (x, y) denotes the position, V the speed, and χ the heading angle. The control input $\mathbf{u} = [u_1, u_2]^\top$ consists of longitudinal acceleration u_1 and heading rate u_2.

The system dynamics are given by:

$$\begin{aligned} \dot{x} &= V \cos\chi, \\ \dot{y} &= V \sin\chi, \\ \dot{V} &= u_1, \\ \dot{\chi} &= u_2. \end{aligned}$$

Bounds on speed and control inputs are enforced to reflect actuator limitations and ensure feasible maneuvers.

2.2 Multi-agent configuration

A decentralized dual-agent formation scenario is considered, in which each agent is tasked to track a predefined reference state–control trajectory while avoiding static obstacles and maintaining collision avoidance with the other agent.

Each agent operates independently using onboard guidance, while limited information exchange of predicted states and control sequences enables coordination and collision avoidance. The decision-making logic is governed by a leader–follower structure, where at each formation maneuver Agent 1 has priority in generating the control input, and the second agent (Agent 2) adapts its trajectory accordingly to ensure collision avoidance.

Reference trajectories are minimum-time paths assigned independently to each agent of the formation. The generation of each sequence of reference state–control waypoints is performed offline in the operational environment free of obstacles and conflicts with other agents, which are instead handled online by the guidance algorithm deployed on the actual mission scenario.

The overall problem is therefore formulated as a constrained trajectory tracking task with safety requirements, including obstacle avoidance and inter-agent separation.

3 Linear model predictive control formulation

3.1 Predictive model

At each execution, the MPC onboard each agent computes an internal prediction based on a linear model. The model is obtained by linearizing the nonlinear agent dynamics along the portion of the desired trajectory corresponding to the current prediction horizon, starting from the latest estimated projection of the agent state over the global mission reference.

Defining $\mathbf{z} = \mathbf{X} - \mathbf{X}_d$ and $\mathbf{v} = \mathbf{u} - \mathbf{u}_d$ as the state and control deviations with respect to the reference trajectory $(\mathbf{X}_d, \mathbf{u}_d)$ within the current prediction window, the linearized dynamics are described by:

$$\dot{\mathbf{z}} = \mathbf{A}(t)\mathbf{z} + \mathbf{B}(t)\mathbf{v},$$

where $\mathbf{A}(t)$ and $\mathbf{B}(t)$ are the Jacobians of the nonlinear dynamics evaluated along the reference trajectory.

3.2 Cost function

At each control step, the MPC computes the optimal control sequence by minimizing a finite-horizon quadratic cost function:

$$J = \int_0^{T_f} \frac{1}{2}\left(\mathbf{z}^\top \mathbf{Q}\mathbf{z} + \mathbf{v}^\top \mathbf{R}\mathbf{v}\right) \mathrm{d}t + \frac{1}{2}\mathbf{z}(T_f)^\top \mathbf{P}\mathbf{z}(T_f), \tag{1}$$

where $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{P}$ are positive definite weighting matrices. The cost penalizes tracking errors and control effort, while the terminal term is included to improve convergence at the end of the prediction horizon.

3.3 Constraints

The formulation of the MPC internal optimization problem is completed by the following constraints, enforced over the entire prediction horizon:

- *Model constraints.* Upper bounds (UB) and lower bounds (LB) on speed and scalar controls to ensure physically feasible maneuvers:

$$\begin{aligned} 0.1 \le z_3(t) + V_d(t) \le 7, \\ -0.1 \le v_1(t) + u_{d1}(t) \le 0.1, \\ -\frac{1}{2\big(z_3(t) + V_d(t)\big)} \le v_2(t) + u_{d2}(t) \le \frac{1}{2\big(z_3(t) + V_d(t)\big)}. \end{aligned}$$

- *Obstacle avoidance.* Positional constraints preventing motion within forbidden regions. The obstacles in the mission environment are collected in the set $\mathcal{O} = \{obs_1, \ldots, obs_E\}$. Each detected obstacle obs_e is shaped as a conservative elliptic envelope with position $\big(x_c^{(e)}, y_c^{(e)}\big)^\top$, major and minor semi-axis $\big(a_e, b_e\big)$, and orientation angle φ_e. The geometric envelopes are assumed to enclose the actual obstacles, allowing tangential motion along

their boundaries while ensuring safe separation:

$$
\begin{aligned}
&- \frac{\left[\left(z_1(t) + x_d(t) - x_c^{(e)}\right)\cos\varphi_e + \left(z_2(t) + y_d(t) - y_c^{(e)}\right)\sin\varphi_e\right]^2}{{a_e}^2} + \\
&- \frac{\left[\left(z_1(t) + x_d(t) - x_c^{(e)}\right)\sin\varphi_e - \left(z_2(t) + y_d(t) - y_c^{(e)}\right)\cos\varphi_e\right]^2}{{b_e}^2} + \\
&\qquad\qquad 1 \leq 0, \\
&\qquad\qquad e = 1, \ldots, E.
\end{aligned}
$$

- *Collision avoidance.* Positional constraints requiring minimum separation of 2 meters between agents based on the predicted trajectories:

$$4 - \left(z_1(t) + x_d(t) - \bar{\xi}_x(t)\right)^2 - \left(z_2(t) + y_d(t) - \bar{\xi}_y(t)\right)^2 \leq 0,$$

where $\left(\bar{\xi}_x(t), \bar{\xi}_y(t)\right)^\top$ represents the available prediction of the other agent's positional trajectory.

3.4 Numerical implementation

By means of direct collocation, the optimal control problem is discretized over a finite prediction horizon using a fixed number of internal time steps. The resulting nonlinear program is solved at each MPC update using the Interior-Point Optimizer (IPOPT) solver [12], providing a control sequence over the prediction horizon.

3.5 Receding horizon implementation

The MPC operates in a receding-horizon configuration. At each execution, the optimal control sequence is computed over the prediction horizon, but only the first control input is applied to the system. At the next time step, the optimization is repeated using updated state estimates, thus providing feedback and enabling adaptation to disturbances and model mismatch.

4 Enhancement with reinforcement learning

4.1 Enhanced controller architecture

Each agent is equipped with a decentralized guidance module composed of a linear MPC and an external Reinforcement Learning (RL) correction layer.

At each controller update, the MPC computes the baseline control action by solving a finite-horizon constrained tracking problem based on the internal linearized prediction model. This provides handling of state, control, obstacle avoidance, and inter-agent collision constraints within the MPC optimization problem and establishes the baseline safety-oriented behavior of the controller.

To compensate for the inaccuracies of the internal MPC model, the MPC-based feedback $\mathbf{u}_{\mathrm{MPC}}(t)$ is augmented by an additive RL-based correction $\Delta\mathbf{u}_\theta(t)$, thus defining the overall deployed control input $\mathbf{u}(t)$:

$$\mathbf{u}(t) = \mathbf{u}_{\mathrm{MPC}}(t) + \Delta\mathbf{u}_\theta(t). \tag{2}$$

The RL module operates externally to the MPC and does not modify its internal model, cost function, or constraints. Instead, it learns online a corrective deterministic feedback that minimizes the mismatch between the MPC-predicted state transition and the observed response of the real dynamics.

Consistent with the global feedback policy formulation in (2), the RL correction is allowed to slightly relax the nominal MPC control bounds enforced by the MPC, within margins of violation Δ_1 for acceleration and Δ_2 for the heading rate. This extends the admissible action space, enabling compensation of model inaccuracies beyond the nominal MPC limits. In practice, the MPC baseline remains the dominant component of the control action, providing the primary safety-oriented mechanism for constraint handling, particularly in critical situations such as obstacle and collision avoidance. The relaxation margins are conservatively selected assuming that the nominal MPC constraints are tightened with respect to the actual system limits.

While the MPC enforces hard constraints within its optimization problem, the RL correction acts at the policy level, allowing small bounded deviations from the nominal control limits without altering the internal constraint formulation of the MPC, which remains the primary mechanism ensuring safety.

The learning process is performed fully online using model-free temporal-difference (TD) Q-learning [11, 13] with linear function approximation, while exploration and policy updates are implicitly bounded by the MPC-based control, supporting controlled online adaptation during mission execution. The resulting learning behavior is assessed empirically through the simulation scenarios presented in Section 6.

The overall hybrid controller preserves the safety-oriented structure of MPC while enabling adaptive compensation of linearization and numerical integration errors. The architecture of the proposed MPC(+RL) controller is illustrated in Fig. 1.

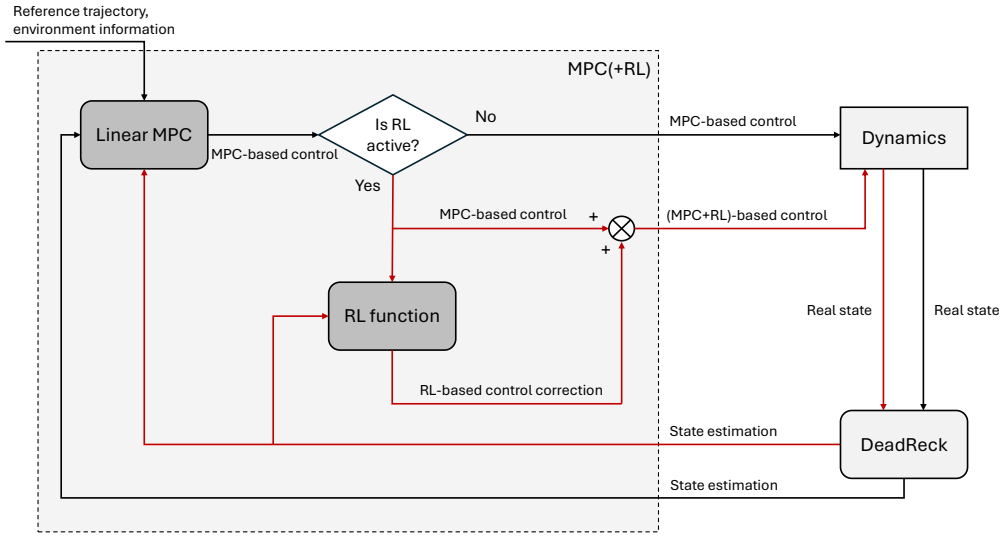


Figure 1: Architecture of the `MPC(+RL)` controller. In black, the data flow without RL. In red, the data flow with RL enabled.

4.2 Learning formulation

The learning problem is formulated as a model-free online reinforcement learning task, where the current state $\mathbf{s}$ corresponds to the latest estimated system state $\mathbf{X}_{\text{obs}}$ and the current deployed action $\mathbf{a}$ corresponds to the applied control input $\mathbf{u}$ defined in (2).

At each MPC update step, the reward function r, defined in (3), penalizes the squared prediction error between the MPC-predicted state $\mathbf{X}_{\text{pred}}$ and the observed system state $\mathbf{X}_{\text{obs}}$ at the end of the receding-horizon control interval:

$$r = -k_L \left\| \mathbf{X}_{\text{pred}} - \mathbf{X}_{\text{obs}} \right\|^2, \tag{3}$$

where the internal prediction is obtained by propagating the MPC-based feedback through the linearized model, while the observed state results from applying the RL-corrected policy in (2). The gain k_L controls the sensitivity of the learning process to prediction errors. Larger values increase the reactivity of the correction but may induce oscillations or instability, whereas smaller values lead to smoother but slower adaptation. A suitable trade-off is therefore required to ensure stable and effective learning.

The action-value function approximation $\hat{Q}(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta})$ is constructed via the linear combination of two sets of parametric features $\left(\boldsymbol{\phi}_1(\mathbf{s}, a_1), \boldsymbol{\phi}_2(\mathbf{s}, a_2)\right)$ defined along the N_w state-control reference waypoints. The sets of features and the related parameters $\left(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\right)$ are separately assigned to the acceleration and heading rate scalar controls:

$$\hat{Q}(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}) = \sum_{k=1}^{N_w} \left[\theta_1^{(k)} \phi_1^{(k)}(\mathbf{s}, a_1) \right] \Big|_{\mathbf{s}=\mathbf{X}_0, a_1=u_1} + \sum_{k=1}^{N_w} \left[\theta_2^{(k)} \phi_2^{(k)}(\mathbf{s}, a_2) \right] \Big|_{\mathbf{s}=\mathbf{X}_0, a_2=u_2},$$

where $\mathbf{X}_0$ is the latest estimated state of the agent, which coincides with the initial state of the current MPC execution. The adopted approximating structure decouples the computation of acceleration and heading rate features at each input state–action pair, allowing the agent to independently adjust each scalar control input. As a result, the learning process becomes more responsive to observed real-time transitions, improving tracking accuracy and robustness to internal model errors.

At each controller execution, the control correction $\Delta\mathbf{u}_\theta$ is computed as:

$$\Delta\mathbf{u}_\theta = \nabla_{\mathbf{a}} \hat{Q}(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}) \big|_{\mathbf{s}=\mathbf{X}_0, \mathbf{a}=\mathbf{u}_{\text{MPC}}}. \tag{4}$$

According to (4), the RL-based correction $\Delta\mathbf{u}_\theta$ adjusts the MPC feedback in the direction of increasing the expected reward.

4.3 Online learning

For each deployed controller, parameters $\boldsymbol{\theta}$ are initialized with zero values at the start of the mission. The policy is updated online using a temporal-difference Q-learning scheme based on batches of recently observed transitions collected over consecutive receding-horizon control executions. This enables continuous adaptation of the feedback policy as new regions of the state-action space are explored during tracking, without requiring prior training.

4.4 Discussion of the hybrid architecture

The proposed hybrid controller preserves the constraint-handling capabilities of MPC while improving tracking performance through data-driven compensation of internal model inaccuracies. The MPC remains the primary decision-making component, while the RL module acts as a lightweight corrective layer operating in parallel.

4.5 Considerations on formal guarantees

The proposed controller is designed with a safety-oriented MPC architecture. However, the present formulation does not provide formal closed-loop stability guarantees for the complete feedback controller. To promote tracking convergence, the internal formulation of the MPC includes a terminal cost, as defined in (1). However, the MPC formulation does not employ additional stabilization strategies such as a dedicated stabilizing gain matrix within a policy-based MPC formulation or terminal constraints and associated terminal ingredients commonly used to establish asymptotic stability guarantees [6, 7]. Therefore, the positive definiteness of the weighting matrices in (1), including the terminal weighting matrix $\mathbf{P}$, although beneficial for penalizing tracking errors and control effort, is not by itself sufficient to establish asymptotic stability of the closed-loop system.

Moreover, the RL correction is applied externally to the MPC-generated control action and operates in an extended action space, according to (2). In particular, the RL correction can operate outside the nominal MPC action bounds within the predefined relaxation margins. Consequently, any stability property established for the baseline MPC cannot be directly extended to the complete MPC+RL controller without an additional analysis of the learned correction and its interaction with the MPC feedback.

A formal proof of stability and robustness is therefore outside the scope of the present work. Instead, the behavior of the proposed controller is assessed empirically through numerical simulations, including nonlinear trajectory tracking, obstacle avoidance, inter-agent collision avoidance, and online adaptation to model mismatch. The observed evolution of the control corrections and performance metrics provides an empirical assessment of the learning behavior, but should not be interpreted as a formal proof of closed-loop stability.

Formal stability and robustness analysis, including the design of additional stabilizing terminal ingredients and the derivation of guarantees for the MPC+RL closed loop, is left for future work.

5 Simulation setup

5.1 Multi-agent operational assumptions

The presented simulation scenarios involve a decentralized dual-agent architecture, in which each agent is assigned a specific set of reference state-control waypoints that are generated offline without taking into account formation constraints and obstacles. In each simulated scenario, the setup of the onboard controller is equal among the agents. During the mission, each agent of the formation performs online tracking to independently compute its control action while exchanging limited information on their predicted trajectories to ensure coordination and collision avoidance.

Each agent has access to a local representation of the environment within its prediction horizon, including obstacle geometry, while the state estimation is updated after completion of each tracking maneuver by means of nonlinear dead reckoning.

A leader–follower structure is adopted to resolve inter-agent conflicts. Agent 1 has priority in computing the control action, while Agent 2 adapts its behavior based on shared predicted information. This hierarchical logic does not affect synchronization, as both agents deploy their feedback commands simultaneously.

Information exchange between agents includes predicted states and control sequences, enabling consistent collision avoidance while preserving decentralized operation.

Computations and communication are assumed to be instantaneous, so that control updates are applied immediately after state estimation.

5.2 MPC configuration

The MPC is configured with a prediction horizon of $T_f = 7.5\,\mathrm{s}$ and discretized using $H = 23$ equidistant internal epochs, resulting in a control update interval $\delta = T_f/(H-1)$.

At each iteration, a constrained finite-horizon tracking problem is solved and the first control input is applied according to the receding-horizon strategy described in Section 3. The control input is held constant over each time interval δ using a zero-order-hold implementation.

The quadratic cost function weights in (1) are selected uniformly to balance tracking accuracy and control effort. In particular, the weighting matrices are defined as:

$$\mathbf{Q} = \mathbf{P} = 10^{-2}\mathbf{I}_4, \quad \mathbf{R} = 10^{-2}\mathbf{I}_2,$$

assigning equal priority to state and control tracking while keeping the overall cost magnitude moderate. The adopted configuration provides a suitable trade-off between tracking performance and real-time computational efficiency.

The control inputs are subject to bounded constraints consistent with the agent model, namely $u_1 \in [-0.1,\, 0.1]$ and $u_2 \in \left[-\frac{1}{2V},\, \frac{1}{2V}\right]$.

For collision avoidance, in each presented scenario the minimum distance between the agents is set to 2 meters.

5.3 Reinforcement learning configuration

The RL module operates fully online using model-free temporal-difference Q-learning with linear function approximation (Section 4).

The action-value function is approximated using Gaussian radial basis function (RBF) features [9] distributed along the reference trajectory. Separate feature sets are defined for each control input, allowing independent adaptation of acceleration and heading rate corrections.

In the results, a softmax transformation of the RBF features is adopted to improve generalization and stabilize the learning process, consistently with common practices in reinforcement learning and function approximation [11]. The softmax temperature is scheduled according to an increasing law:

$$T_i = \min\left\{T_{\max},\; T_0 \exp(\rho i)\right\},$$

where i is the learning update index, $T_0 \in (0, 1]$ is the initial (cold) temperature, and ρ is the growth rate. This strategy enables real-time tuning of the activation of the visited RBF features as tracking evolves, yielding marked, localized learning in early stages and progressively smoother policy adaptation as experience accumulates.

Policy updates are performed using batches of $B = 10$ transitions collected over consecutive receding-horizon executions. The discount factor is set to $\gamma = 1$.

A decreasing learning rate schedule is adopted:

$$\alpha_i = \frac{\alpha_0}{1 + \beta i},$$

where α_0 is the initial learning rate and β is a null or positive coefficient tuning the rate of decrease. The learning rate schedule enables fast initial adaptation while improving stability in later stages of the mission.

5.4 Case studies

Two representative dual-agent formation scenarios are considered to evaluate the performance of the proposed MPC+RL framework under increasing levels of complexity.

5.4.1 Parallel trajectory tracking with obstacle avoidance

This scenario considers two agents tracking parallel reference trajectories with multiple consecutive curvilinear segments in the presence of static obstacles.

The configuration is designed to stress tracking performance under repeated nonlinear maneuvers, where linearization errors and control saturation may degrade the accuracy of the baseline MPC. No inter-agent conflicts occur in this case, allowing the analysis to focus on tracking improvement, learning convergence, and obstacle avoidance.

5.4.2 Crossing trajectories with double obstacle avoidance and conflict negotiation

In this scenario, the agents follow intersecting reference trajectories, leading to a conflict region where simultaneous inter-agent collision avoidance and obstacle avoidance must be ensured.

Two symmetric obstacles are placed downstream of the crossing point, requiring coordinated maneuvers under constrained conditions. This setup represents a safety-critical scenario in which the controllers must balance tracking performance with strict constraint satisfaction.

5.4.3 RL parameter selection

The RL hyperparameters are tuned separately for each scenario through empirical testing, in order to achieve a suitable trade-off between learning stability, adaptation speed, and constraint compliance. The selected configurations are summarized in Table 1.

The reported values result from scenario-specific tuning campaigns, in which the learning rate, reward scaling, and softmax temperature scheduling were adjusted to ensure both significant tracking improvement and stable online adaptation.

An empirical relation was consistently observed between the initial learning rate α_0 and the linearization error penalty gain k_L, which appeared to hold within each scenario:

$$\alpha_0 \propto \frac{1}{k_L}.$$

This relation is not claimed to be universal, but it provided a useful guideline to balance learning responsiveness and stability in the considered configurations.

Table 1: RL configuration for the considered scenarios.

Scenario	α_0	β	T_0	ρ	k_L	Δ_1	Δ_2
Parallel trajectories	$5 \cdot 10^{-3}$	10^{-1}	0.6	$\frac{1}{55}$	$5 \cdot 10^{3}$	~ 0.05	~ 0.02
Crossing trajectories	10^{-3}	10^{-3}	0.5	$\frac{1}{15}$	$5 \cdot 10^{2}$	~ 0.17	~ 0.08

5.5 Performance metrics

The performance of the proposed MPC+RL architecture is evaluated against the baseline linear MPC using tracking accuracy, safety, and learning behavior indicators.

Tracking performance is assessed through the evolution of the internal MPC cost function at each control step. The relative improvement introduced by the RL layer is quantified as:

$$\Delta J_{\%} = 100 \times \frac{J_{\mathrm{MPC}} - J_{\mathrm{MPC+RL}}}{J_{\mathrm{MPC}}},$$

where J_{MPC} and $J_{\mathrm{MPC+RL}}$ denote the cost values obtained with the baseline MPC and the RL-enhanced controller, respectively.

Safety is evaluated by verifying constraint satisfaction throughout the simulations, including obstacle avoidance, inter-agent collision avoidance, and compliance with nominal MPC control bounds, accounting for the admissible violation margins introduced by the RL correction.

The temporal evolution of the cost function and of the deployed RL-enhanced feedback controls are used to characterize transient and steady-state behavior and to assess the evolution of the learned policy particularly in terms of control effort.

6 Simulation results

6.1 Parallel trajectory tracking with obstacle avoidance (noise-free)

6.1.1 Pure MPC

The spatial trajectories obtained with the baseline MPC are shown in Fig. 2.

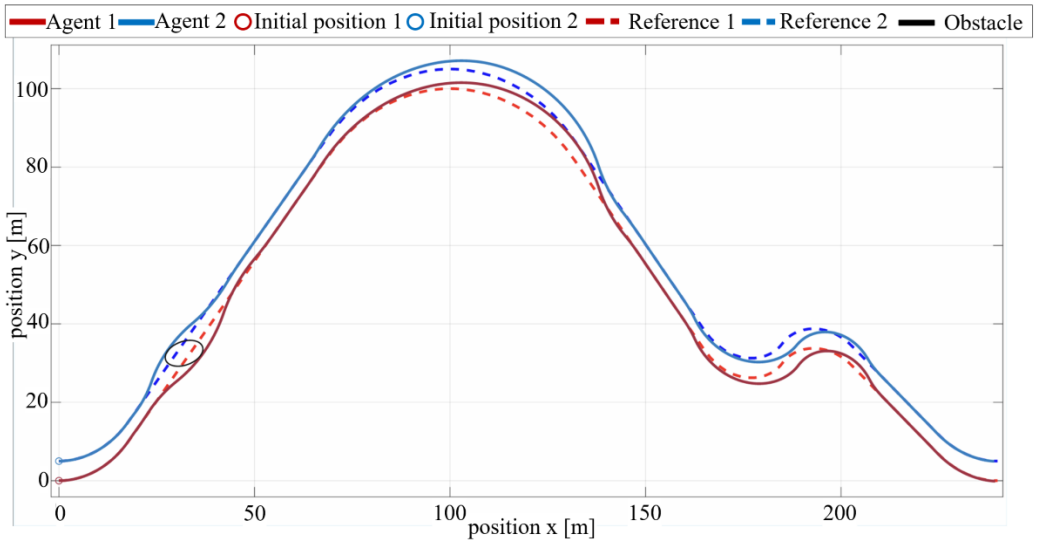


Figure 2: Spatial trajectories [m] of Agent 1 and Agent 2 with pure MPC.

Both agents exhibit increasing tracking errors when navigating the curvilinear segments. These errors originate from linearization inaccuracies and become amplified by control saturation, which limits the controller's ability to compensate for deviations in speed and heading.

As a result, positional errors accumulate along each curve and are only partially recovered during subsequent rectilinear segments. This behavior is reflected in the cost function evolution reported in Fig. 3.

Overall, the results highlight a key limitation of the linear MPC formulation: once tracking errors exceed a certain threshold, actuator constraints prevent rapid recovery, leading to degraded performance in highly nonlinear regions.

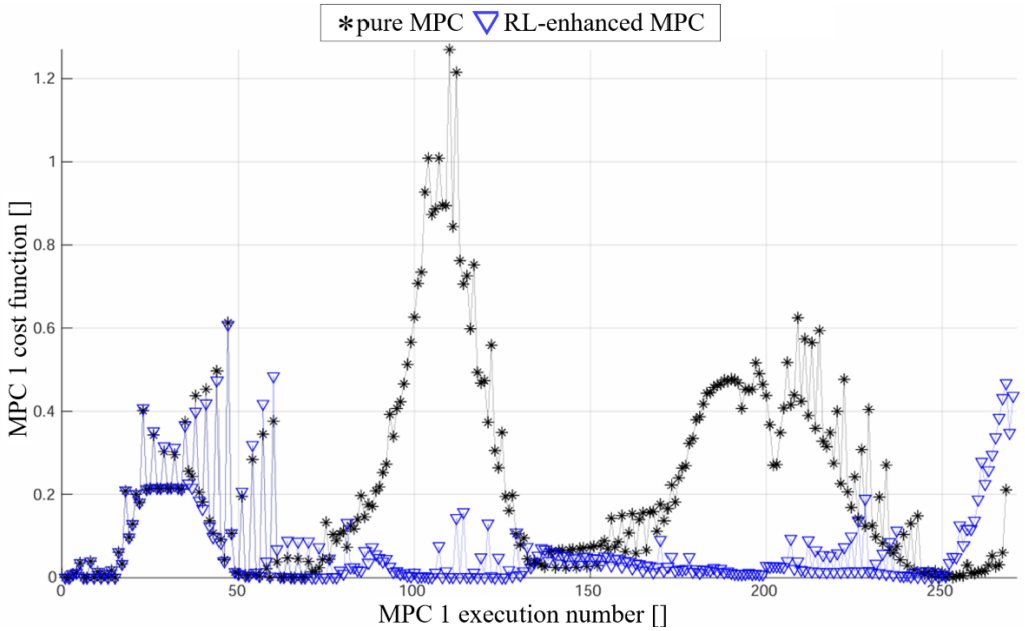


Figure 3: Comparison of the MPC internal performance index for Agent 1 with pure MPC and MPC+RL at each MPC execution.

6.1.2 RL-enhanced MPC

The trajectories obtained with the proposed MPC+RL architecture are shown in Fig. 4. Figures 2 and 4 are reported with equal axis scaling for direct comparison.

A substantial improvement in tracking performance is observed across all curvilinear segments. The RL correction effectively compensates for linearization errors, preventing the accumulation of deviations observed in the baseline MPC case.

The evolution of the control inputs (Figs. 5 and 6) shows that the RL layer introduces temporary violations of the nominal MPC control bounds, particularly during the first curve. These violations remain bounded within margins (Δ_1, Δ_2), whose approximate maximum values are reported in Table 1. The violations decrease as learning progresses, indicating improved policy stability.

After the initial transient, the controller achieves smoother control profiles and enhanced tracking accuracy, with negligible constraint violations in later mission phases.

The comparison of the cost functions in Fig. 3 confirms the effectiveness of the proposed approach. During the most demanding segments, the RL-enhanced controller achieves cost reductions exceeding 90% of the baseline MPC.

These results demonstrate that the RL module enables adaptive compensation of model inaccuracies and improves recovery capability while maintaining bounded deviations from the nominal MPC constraints.

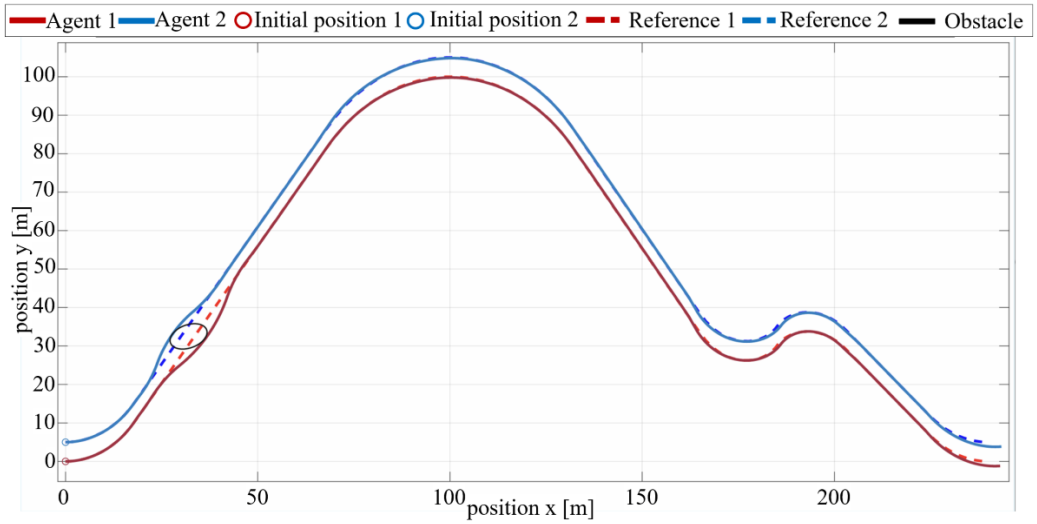


Figure 4: Spatial trajectories [m] of Agent 1 and Agent 2 with RL-enhanced MPC.

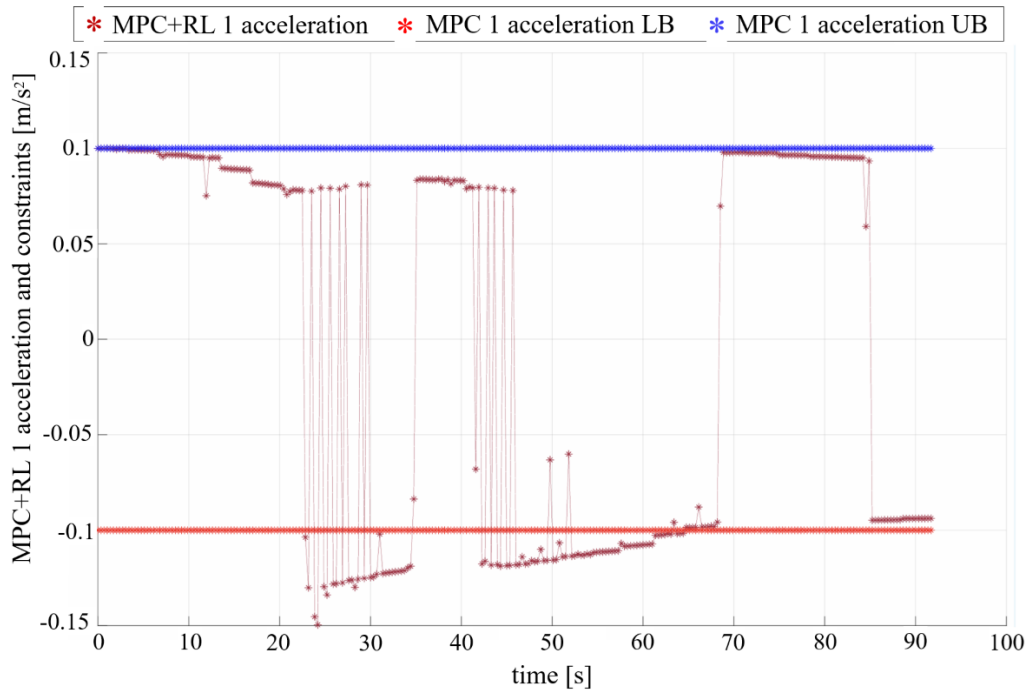


Figure 5: MPC+RL 1 acceleration $[\mathrm{m/s^2}]$ vs. time [s] relative to the MPC-based limits.

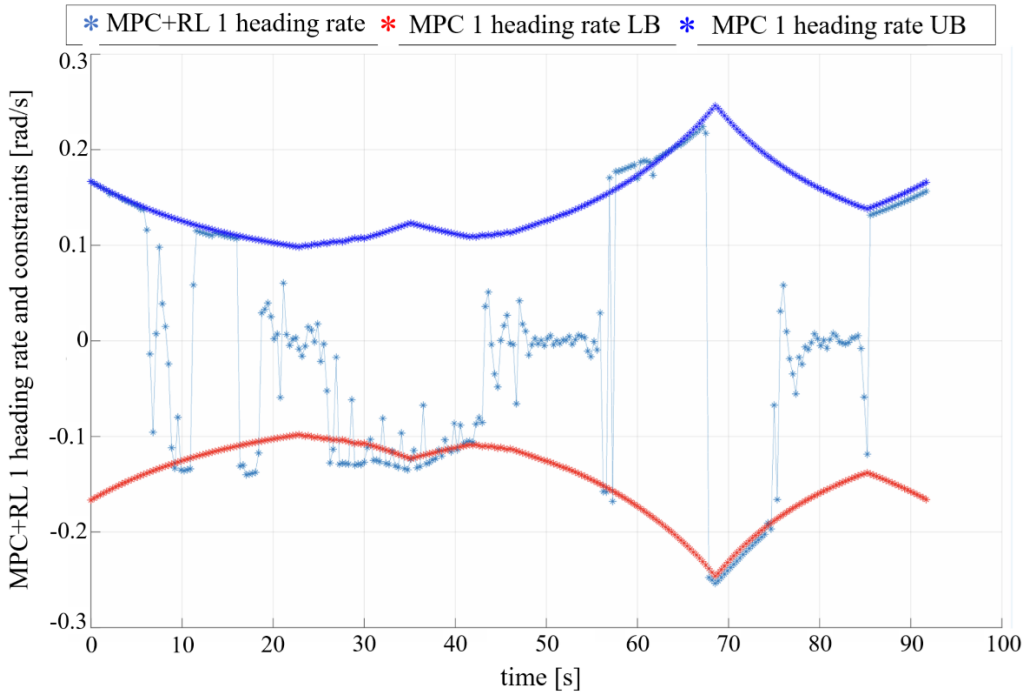


Figure 6: MPC+RL 1 heading rate [rad/s] vs. time [s] relative to the MPC-based limits.

6.2 Crossing trajectories with obstacle avoidance (noise-free)

This scenario evaluates the capability of the controllers to simultaneously handle inter-agent collision avoidance and obstacle avoidance under constrained conditions. Two symmetric obstacles are placed along the reference trajectories immediately downstream of the first crossing point.

6.2.1 Controllers comparison on safety performance

The comparison in Fig. 7 highlights a critical limitation of the baseline MPC controller regarding safety compliance. While inter-agent collision avoidance is maintained, Agent 1 fails to satisfy the obstacle avoidance constraint and violates the forbidden region downstream of the first crossing.

In contrast, the RL-enhanced MPC successfully handles both inter-agent conflict and obstacle avoidance. The inter-agent distance profile shown in Fig. 8 shows a slight violation of the inter-agent separation constraint at the first conflict event. However, the minimum distance is assumed to include a conservative safety margin and the observed violation has a significantly smaller order of magnitude with respect to the boundary value. Overall, the distance profile shows that the inter-agent separation remains close to the prescribed safety threshold throughout the mission, with only a small transient deviation during the first conflict event.

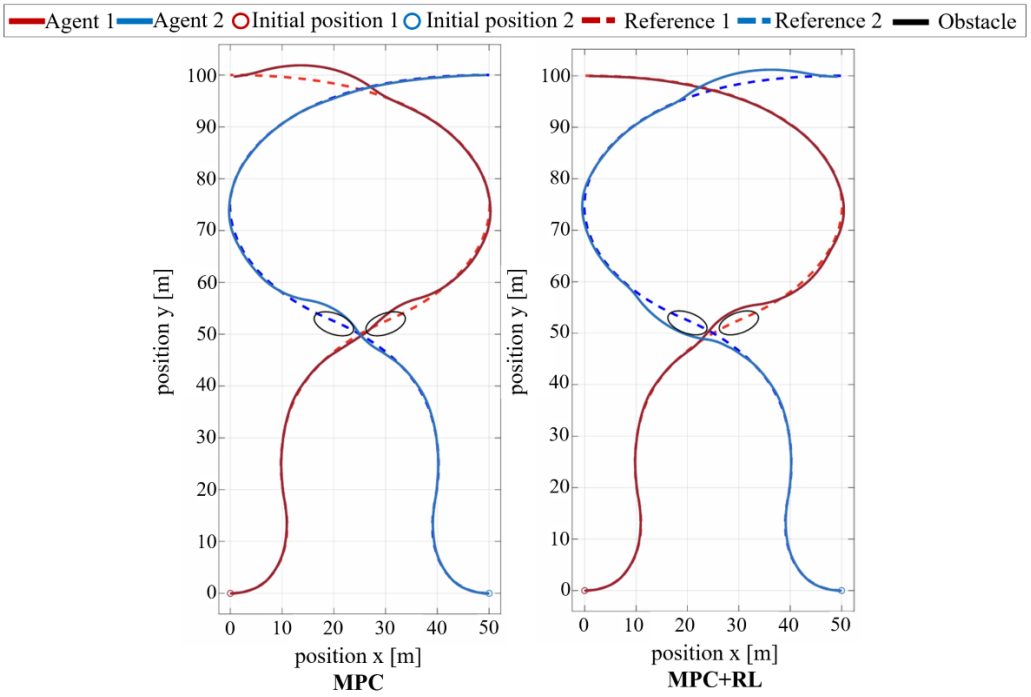


Figure 7: Spatial trajectories [m] of Agent 1 and Agent 2 with pure MPC and MPC+RL.

6.2.2 RL-induced control effort and constraint violations

For clarity, only the control histories of Agent 2 are reported, as it exhibits the largest constraint violations and represents the most critical case. Agent 1 shows qualitatively similar but less pronounced behavior.

The improved safety performance achieved by MPC+RL comes at the cost of significant violations of the nominal control limits. The control histories in Figs. 9 and 10 show that the largest deviations occur during the first conflict negotiation ($t \approx 25.6\,\mathrm{s}$).

In particular, the acceleration command exceeds the nominal upper bound by more than 100%, indicating that the RL correction actively exploits the relaxed admissible action space to satisfy simultaneous collision and obstacle avoidance constraints. This result provides an indication on the feasibility of the mission based on the actual system capabilities.

6.2.3 Tracking performance

Since Agent 1 equipped with pure MPC violates safety constraints, a direct performance comparison is meaningful only for Agent 2. The cost evolution shown in Fig. 11 indicates that MPC+RL maintains improved tracking performance even under safety-critical conditions where the baseline MPC fails to satisfy feasibility constraints.

These results highlight that, in the tested scenario, the RL layer enables the controller to resolve safety-critical situations that would otherwise lead to infeasibility under the nominal MPC formulation.

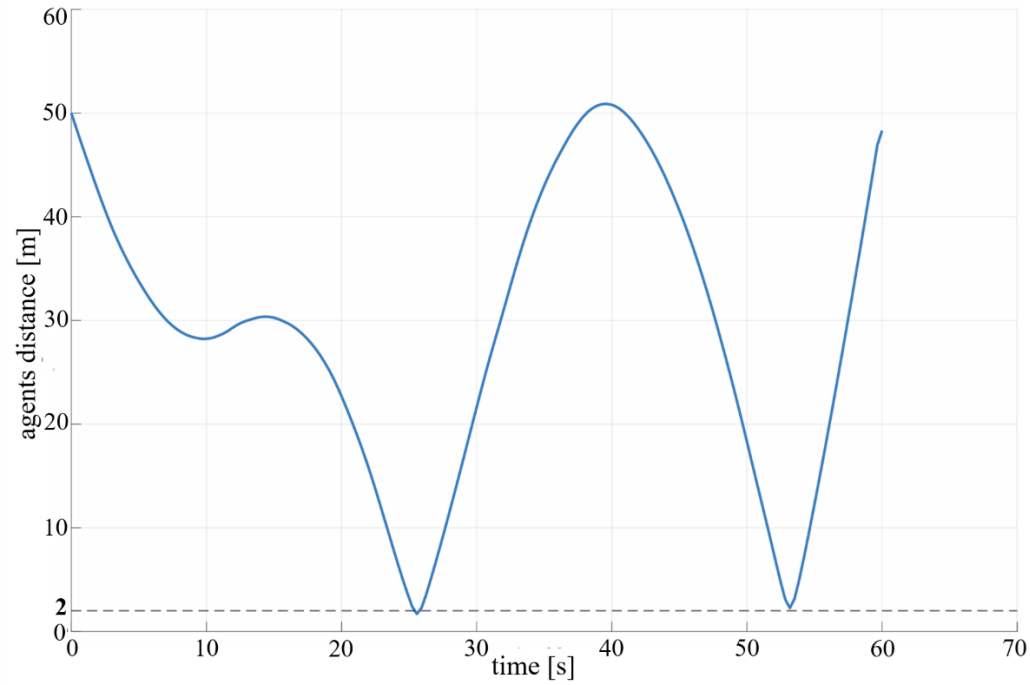


Figure 8: Distance [m] between agents with MPC+RL vs. time [s].

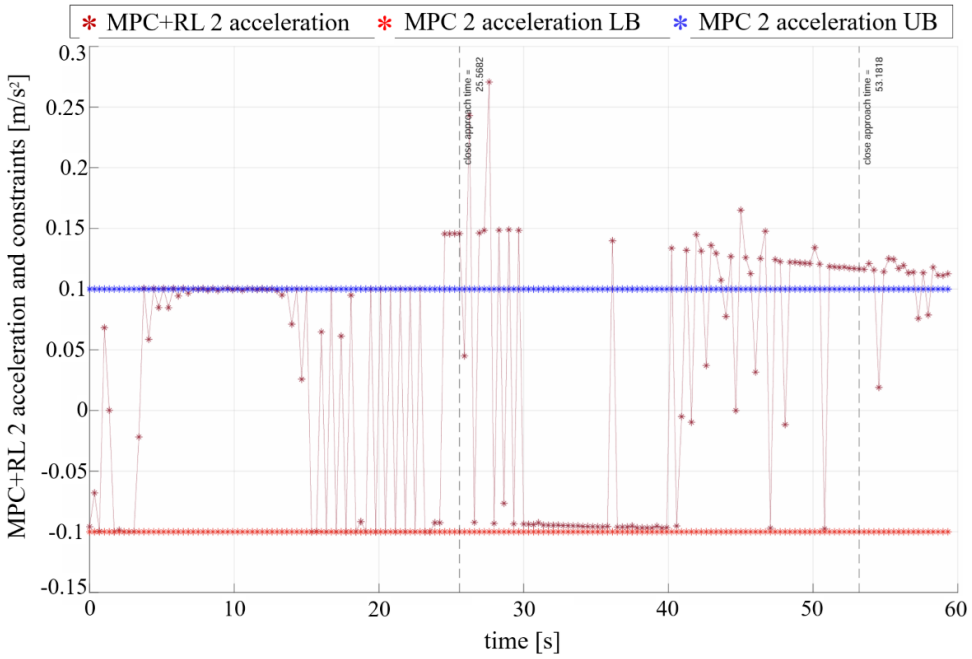


Figure 9: MPC+RL 2 acceleration $\left[\mathrm{m/s^2}\right]$ vs. time [s] relative to the MPC-based limits.

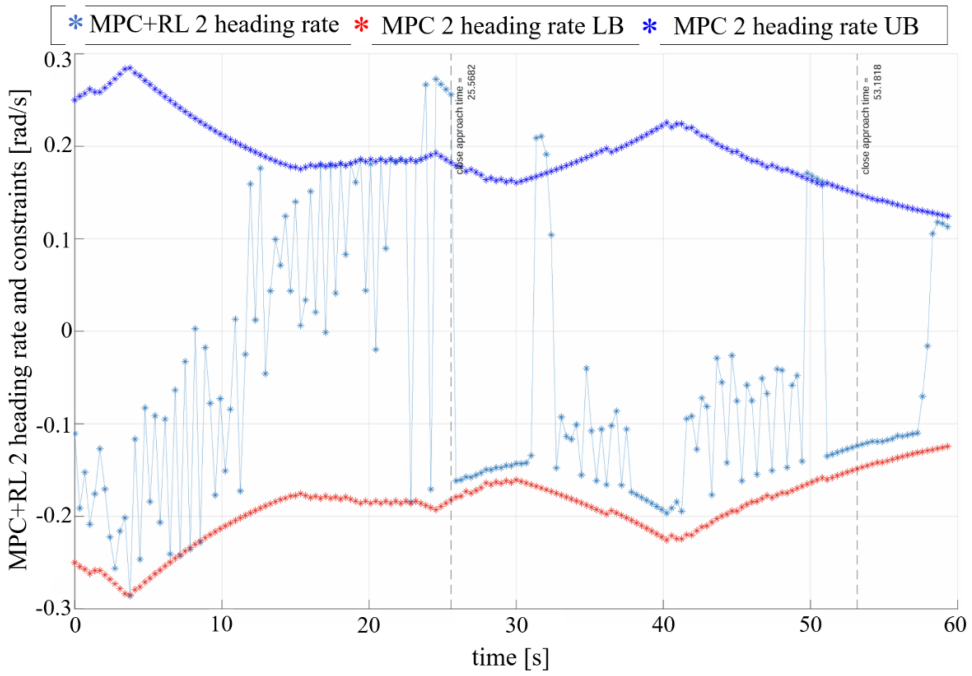


Figure 10: MPC+RL 2 heading rate [rad/s] vs. time [s] relative to the MPC-based limits.

6.3 Discussion

The presented results highlight the potential of combining linear MPC with an online model-free RL correction for multi-agent formation control in constrained environments.

In nominal conditions, the RL-enhanced controller consistently improves tracking performance with respect to the baseline MPC, particularly during negotiation of curvilinear segments where linearization errors are more pronounced. The observed cost reductions confirm that the learned feedback effectively compensates for model inaccuracies without requiring prior training.

More importantly, in safety-critical scenarios involving simultaneous obstacle avoidance and inter-agent conflict, the RL layer enables behaviors that are not achievable by the nominal MPC alone. In particular, the RL-enhanced controller successfully resolves situations that would otherwise lead to infeasibility or constraint violation.

This improvement, however, comes at the cost of temporary violations of the nominal MPC control bounds. The results show that the RL correction exploits the extended admissible action space to maintain feasibility under coupled constraints. While such violations remain bounded and occur primarily during critical phases, they highlight a fundamental trade-off between strict constraint enforcement and achievable performance in highly constrained environments.

From a learning perspective, the evolution of the control signals indicates that the policy progressively stabilizes after an initial transient phase. The magnitude of the control corrections and the associated violations decreases over time, indicating a tendency toward smoother and more efficient feedback strategies.

At the same time, the experiments reveal a strong dependency of the RL performance on scenario-specific tuning parameters. This sensitivity, together with the absence of formal sta-

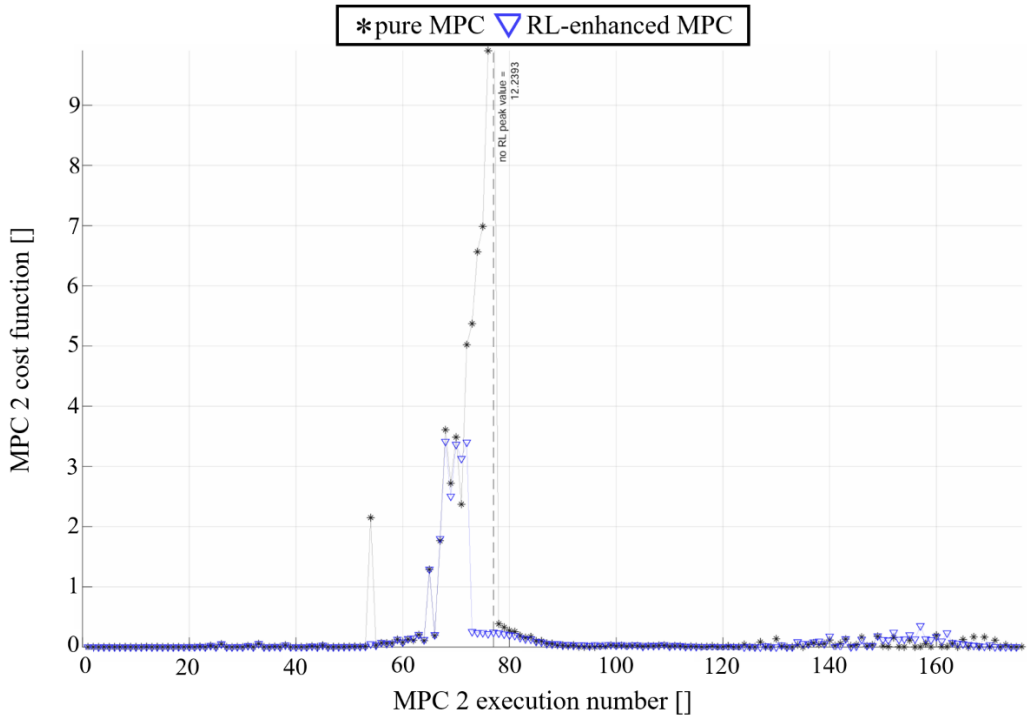


Figure 11: Comparison of the MPC internal performance index for Agent 2 with pure MPC and MPC+RL at each MPC execution.

bility guarantees, represents a limitation for real-world deployment and calls for further investigation on adaptive tuning and safety certification mechanisms. For deployment in real mission scenarios, a supervisory gateway architecture equipped with a certified fallback controller could ensure reliable safety-critical operation [3].

Overall, the proposed architecture demonstrates that a lightweight, model-free RL layer can effectively complement MPC by providing online adaptation capabilities, improving both tracking performance and robustness in complex multi-agent scenarios.

7 Conclusions

This work presents a decentralized guidance framework for multi-agent formation maneuvering based on linear MPC enhanced with an online model-free RL correction. The proposed approach enables each agent to improve its tracking performance by compensating for internal model inaccuracies of the MPC through a learned feedback correction. The RL component operates fully online, without prior training, and remains guided by the MPC baseline, preserving the overall safety-oriented structure of the controller.

Simulation results demonstrate that the RL-enhanced MPC significantly improves tracking accuracy in nominal conditions and enables successful resolution of safety-critical scenarios involving simultaneous obstacle avoidance and inter-agent collision constraints, where the baseline MPC alone may fail. The results of the considered case studies suggest that the proposed

RL-enhanced controller not only improves mission performance, but also successfully handles safety-critical situations involving coupled obstacle and inter-agent constraints that lead to infeasibility or constraint violations for the baseline MPC.

These improvements come at the cost of temporary violations of the nominal control limits. Moreover, careful tuning of the learning parameters is required to balance performance, stability, and constraint compliance. Despite these limitations, the proposed approach provides a promising direction for enhancing MPC-based control with adaptive, data-driven components while retaining a computationally lightweight control structure.

Future work will focus on improving the robustness and generality of the learning process, including adaptive tuning strategies, constraint-aware policy design, and validation in more realistic and experimental settings.

Acknowledgments

This paper is based on a Master's thesis completed in collaboration with Airbus Defence and Space GmbH in Manching, Germany. The authors would like to thank Airbus Defence and Space GmbH for providing the resources and support that made this research possible.

References

[1] A. Bahari Kordabad, D. Reinhardt, A. S. Anand, and S. Gros. Reinforcement learning for model predictive control: fundamentals and current challenges. *IFAC-PapersOnLine*, 56(2):5773–5780, 2023.

[2] M. Estrada, S. Li, and X. Cai. Feedback linearization of car dynamics for racing via reinforcement learning. *arXiv preprint arXiv:2110.10441*, 2021.

[3] D. M. Gierszewski, P. Piprek, and F. Holzapfel. Onboard trajectory optimization for radius-to-fix turns. In *Proceedings of the 34th Congress of the International Council of the Aeronautical Sciences (ICAS)*, Florence, Italy, 2024.

[4] S. Gros and M. Zanon. Towards safe reinforcement learning using nonlinear model predictive control and policy gradients: deterministic case. *arXiv preprint arXiv:1906.04034*, 2019.

[5] S. Gros and M. Zanon. Data-driven economic nonlinear model predictive control using reinforcement learning. *IEEE Transactions on Automatic Control*, 65(2):636–648, 2020.

[6] M. Mammarella, E. Capello, H. Park, G. Guglieri, and M. Romano. Tube-based robust model predictive control for spacecraft proximity operations in the presence of persistent disturbance. *Aerospace Science and Technology*, 77:585–594, 2018.

[7] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[8] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 2nd edition, 2017.

[9] R. Schaback. A practical guide to radial basis functions, 2007. Lecture notes, University of Göttingen.

[10] R. Stern. Multi-agent path finding: an overview. In *Artificial Intelligence*, volume 11684 of *Lecture Notes in Computer Science*, pages 96–115. Springer, 2019.

[11] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

[12] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106(1):25–57, 2006.

[13] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.