

Article

Robust Model Predictive Control with a Dynamic Look-Ahead Re-Entry Strategy for Trajectory Tracking of Differential-Drive Robots

Diego Guffanti ^{1,*} , Moisés Filiberto Mora Murillo ^{2,3} , Santiago Bustamante Sanchez ⁴ ,
Javier Oswaldo Obregón Gutiérrez ^{5,6} , Marco Alejandro Hinojosa ⁴ , Alberto Brunete ⁷ , Miguel Hernando ⁷ 
and David Álvarez ⁷ 

¹ Centro de Investigación en Mecatrónica y Sistemas Interactivos—MIST, Universidad Indoamérica, Av. Machala y Sabanilla, Quito 170103, Ecuador

² Departamento de Mecánica y Ciencias Exactas, Instituto Superior Universitario Japón, Santo Domingo 230102, Ecuador; mmora@itsjapon.edu.ec or moisesmora@tsachila.edu.ec

³ Carrera de Electricidad, Instituto Superior Tecnológico Tsa'chila, Santo Domingo 230109, Ecuador

⁴ Carrera de Tecnología Superior en Mecánica Industrial, Instituto Superior Tecnológico Tsa'chila, Santo Domingo 230109, Ecuador; marcohinojosa@tsachila.edu.ec (M.A.H.)

⁵ Carrera de Electrónica, Instituto Superior Tecnológico Tsa'chila, Santo Domingo 230109, Ecuador; javierobregon@tsachila.edu.ec

⁶ Universidad Laica Eloy Alfaro de Manabí sede Santo Domingo, Santo Domingo 230108, Ecuador

⁷ Centre for Automation and Robotics (CAR UPM-CSIC), Universidad Politécnica de Madrid, 28012 Madrid, Spain; alberto.brunete@upm.es (A.B.); miguel.hernando@upm.es (M.H.); david.asanchez@upm.es (D.Á.)

* Correspondence: diego.guffanti@upm.es

Abstract

Accurate trajectory tracking remains a central challenge in differential-drive mobile robots (DDMRs), particularly when operating under real-world conditions. Model Predictive Control (MPC) provides a powerful framework for this task, but its performance degrades when the robot deviates significantly from the nominal path. To address this limitation, robust recovery mechanisms are required to ensure stable and precise tracking. This work presents an experimental validation of an MPC controller applied to a four-wheel DDMR, whose odometry is corrected by a SLAM algorithm running in ROS 2. The MPC is formulated as a quadratic program with state and input constraints on linear (v) and angular (ω) velocities, using a prediction horizon of $N_p = 15$ future states, adjusted to the computational resources of the onboard computer. A novel dynamic look-ahead re-entry strategy is proposed, which activates when the robot exits a predefined lateral error band ($\delta = 0.05$ m) and interpolates a smooth reconnection trajectory based on a forward look-ahead point, ensuring gradual convergence and avoiding abrupt re-entry actions. Accuracy was evaluated through lateral and heading errors measured via geometric projection onto the nominal path, ensuring fair comparison. From these errors, RMSE, MAE, P_{95} , and in-band percentage were computed as quantitative metrics. The framework was tested on real hardware at 50 Hz through 5 nominal experiments and 3 perturbed experiments. Perturbations consisted of externally imposed velocity commands at specific points along the path, while configuration parameters were systematically varied across trials, including the weight R , smoothing distance L_{smooth} , and activation of the re-entry strategy. In nominal conditions, the best configuration (ID 2) achieved a lateral RMSE of 0.05 m, a heading RMSE of 0.06 rad, and maintained 68.8% of the trajectory within the validation band. Under perturbations, the proposed strategy substantially improved robustness. For instance, in experiment ID 6 the robot sustained a lateral RMSE of 0.12 m and preserved 51.4% in-band, outperforming MPC without re-entry, which suffered from larger deviations



Academic Editor: Andrey V. Savkin

Received: 15 October 2025

Revised: 2 December 2025

Accepted: 15 December 2025

Published: 13 January 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article

distributed under the terms and

conditions of the [Creative Commons](https://creativecommons.org/licenses/by/4.0/)

[Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

and slower recoveries. The results confirm that integrating MPC with the proposed re-entry strategy enhances both accuracy and robustness in DDMR trajectory tracking. By combining predictive control with a spatially grounded recovery mechanism, the approach ensures consistent performance in challenging scenarios, underscoring its relevance for reliable mobile robot navigation in uncertain environments.

Keywords: Model Predictive Control (MPC); differential-drive robot; trajectory tracking; re-entry strategy; robust navigation

1. Introduction

Trajectory tracking for differential drive mobile robots (DDMRs) remains a central problem in robotics, especially in applications that require high precision, robustness to disturbances, and real-world deployment. Model Predictive Control (MPC) has become a popular method for this task because it allows explicit consideration of constraints, prediction of future system behavior, and optimization over a finite horizon. Although MPC has been widely studied in simulation, its performance in real hardware under perturbations and large deviations is less frequently evaluated.

Recent works have addressed various aspects of MPC in mobile robot control. Xu et al. [1] propose a Dynamic Model Predictive Control (DMPC) for a three-wheeled independent drive and steering mobile robot (3WID3WIS), demonstrating improved lateral and heading accuracy compared to traditional MPC, especially when the curvature of the path changes. Ali, Shen and Hashim [2] introduce a safety-critical MPC with control barrier functions for differential-drive robots, focusing on obstacle avoidance and guaranteeing safety constraints alongside trajectory following. Abed et al. [3] present a nonlinear neural network fractional trajectory tracking controller for DDMRs, improving accuracy in the presence of model uncertainties. Pang et al. [4] develop a time-varying linear MPC for autonomous vehicles (which share similar challenges of lateral error and heading control), achieving stability and high precision in real road trajectories. Zhang et al. [5] survey learning-based MPC methods for path tracking control under uncertain disturbances, underlining that many methods struggle when the robot deviates significantly from the nominal path. The work by Bouzoualegh, Guechi, and Kelaiaia [6] examines the use of dynamic vs. kinematic models for MPC in DDMRs, showing that incorporating more realistic dynamics can yield better tracking but also increases susceptibility to disturbances when not properly managed.

A recurring issue in these works is that, although MPC can correct for small errors, when deviations are large (due to disturbances, external forces, slippage, or sensor errors), MPC alone may require long correction times or may lead to overshoots. This suggests the potential benefit of an auxiliary strategy that assists MPC in re-entry or realigning to the reference trajectory more aggressively or smoothly whenever the error passes a certain threshold. This re-entry strategy can reduce the deviation in the worst-case scenario, limit the time outside acceptable error bounds, and improve the robustness of the real world.

In this paper, we propose an MPC implementation in a differential drive robot augmented with a re-entry strategy that is triggered when the lateral error exceeds a fixed threshold (0.05 m). We present an extensive experimental study (8 real experiments) that compares the conditions with and without the re-entry strategy, both under normal navigation and under perturbations. Our main contributions can be summarized as follows:

- A sensor-based implementation of a Model Predictive Controller (MPC) for trajectory tracking in a four-wheel differential-drive mobile robot (DDMR), validated on real

hardware under ROS 2 using SLAM-based odometry (LiDAR/IMU/encoders) for continuous state estimation.

- A novel re-entry strategy that is activated based on real-time sensory monitoring of the lateral deviation of the robot from the path. When the deviation exceeds a predefined band ($\delta = 0.05$ m), the method generates a smooth, sensor-triggered reconnection trajectory.
- Experimental evidence showing that the proposed sensing-driven control strategy enhances robustness against perturbations, reducing both lateral and heading errors, and increasing the percentage of the trajectory tracked within the validation band compared to MPC alone.
- A real-time feasibility study of the sensing-control loop, including a computational cost evaluation of the MPC pipeline with and without the re-entry mechanism. This analysis confirms that the proposed sensor-driven controller satisfies the timing constraints of the 20 ms sampling period in all tested conditions.

The rest of this paper is structured as follows. Section 2 reviews the state of the art in MPC for mobile robots. Section 3 describes the experimental DDMR used in this study, the MPC formulation, and the re-entry strategy. Section 5 discusses the implications, limitations, and proposes future research. Finally, Section 6 concludes this study.

2. State-of-the-Art

Model Predictive Control (MPC) is a well-established approach for trajectory tracking in wheeled mobile robots due to its ability to optimize performance under state and input constraints. Early and baseline formulations for DDMRs demonstrated the feasibility of MPC with both kinematic and dynamic models, laying the foundation for more robust and safety-aware extensions [6]. Beyond these foundations, a large body of work has explored MPC variants tailored to mobile platforms, including formulation choices (linear vs. nonlinear MPC), horizon design, and real-time implementation.

Safety-aware MPC has been advanced by incorporating Control Barrier Functions (CBFs) to enforce collision avoidance and state-space safety sets while preserving receding-horizon optimality. For DDMRs and other ground vehicles, linear or nonlinear MPC combined with barrier or feasibility constraints has been shown to maintain constraint satisfaction and stability in dynamic environments [2,7,8]. These formulations primarily target safety, feasibility, and constraint handling, but in general rely on nominal tracking when deviations occur, rather than explicitly modifying the reference to shorten recovery time.

Another active direction is robust MPC for mobile robots, where uncertainty and disturbances are handled via tube-based or dual-loop designs. Robust tube MPC, disturbance observers, and slip-aware controllers have been used in ground vehicles, skid-steer or manipulator platforms to attenuate disturbances and improve worst-case tracking guarantees [9–11]. Stochastic MPC has also been applied to wheeled robots to probabilistically satisfy constraints under additive disturbances [12]. Although these methods improve resilience, they still treat large deviations as disturbances around a fixed path, without a specific reincorporation mechanism designed to reshape the reference when the robot drifts beyond an application-relevant band.

There is also a growing emphasis on learning-aided or model-enhanced MPC to improve prediction fidelity and tracking performance by combining data-driven models (e.g., Gaussian processes) with predictive control to mitigate modeling mismatch under uncertain conditions [5,13]. Similar ideas are explored for non-holonomic and skid-steer robots, where NMPC, observers, or slip compensation are used to maintain precision under wheel-ground interactions [14]. These approaches typically improve the average

tracking accuracy, but practical recovery from significant deviations (e.g., slippage, pushes, or sensing failures) still depends on the ability of the MPC to quickly converge again.

In addition, recent research explores hybrid frameworks where MPC is combined with dynamic trajectory re-planning or online path updating, enabling the robot to modify the planned path in real time under changing environments or moving obstacles. Examples include MPC-based re-planning under uncertainty for manipulators [15], MPC combined with neurodynamic optimization for dynamic obstacle avoidance in car-like robots [16], and MPC-based path planning in multi-robot environments [17]. These works underline that MPC-based dynamic replanning remains an active line of research for mobile platforms operating in dynamic or uncertain conditions, supporting the motivation for lightweight replanning or re-entry mechanisms like the one presented in this study.

Time-varying or spatial MPC formulations have also been proposed to handle curvature changes and non-holonomic constraints on mobile platforms [14]. These contributions improve nominal tracking and robustness. However, most lack an evaluation protocol centered on band occupancy (time in a narrow lateral error band).

In summary, the previous state of the art demonstrates (i) safety-aware MPC with CBFs, (ii) robust/tube or stochastic MPC for bounded uncertainties, and (iii) learning-augmented MPC for better prediction. What remains under-reported is an experimentally validated practical mechanism to assist MPC during significant deviations by temporarily reshaping the reference trajectory. Our work addresses this gap through a smooth re-entry strategy activated by a threshold and evaluated using application-oriented metrics such as time within a narrow lateral band, number of band exits, and number of re-entries, validated in real-robot experiments under both nominal operation and deliberate perturbations.

3. Methods

3.1. Robot Platform

The experimental platform is a four-wheel differential-drive robot assembled on a Recon Chassis Kit (ServoCity, Winfield, KS, USA). Each wheel is actuated by a DC motor of type 5203 series Yellow Jacket Planetary Gear Motor (goBILDA, Winfield, IL, USA), with a 19.2:1 reduction ratio and a nominal speed of 312 RPM. Each motor integrates a quadrature encoder that provides a resolution of 537.7 pulses per revolution (PPR) in the output shaft. The wheels are filled with foam, ensuring stable operation on rigid outdoor surfaces. The actuation system is driven by two dual motor drivers (RoboClaw 2x15A), one per side of the differential configuration. These drivers are commanded by an MCU (Nucleo STM32F207ZG) that receives velocity commands and translates them into pulse-per-second signals for the motors. The robot is also equipped with a WitMotion WT901C-TTL 9-axis IMU (accelerometer, gyroscope, magnetometer, and inclinometer) and a Slamtec A2 2D LiDAR with a maximum range of 16 m, used for SLAM and localization.

The SLAM module is responsible for fusing wheel encoder odometry, IMU data, and 2D LiDAR scans to build the map and estimate the robot pose for trajectory tracking. We use the `slam_toolbox` package from ROS2 in online asynchronous mapping mode, which provides the corrected odometry through the `/odom_filtered` topic. The raw odometry is integrated in the `/base_footprint` to `/odom` transform, while LiDAR scan matching refines the `/map` to `/odom` transform, producing the SLAM-based pose feedback used by the MPC.

The reference trajectory used in all experiments consists of 1859 spatial poses (x, y, θ) recorded from the SLAM-based odometry. Its total length is approximately 20 m and consecutive points are sampled with an average spacing of 1.1 cm (maximum around 1.5 cm). The path includes straight segments and curved turns, and is used as the nominal trajectory $\mathcal{P}$ throughout the study.

The system is powered by a Tattu G-tech 5200 mAh 4S 35C 14.8V LiPo battery. The onboard computer is an Intel NUC BNUC11TNKV70002 11 Pro, equipped with an Intel Core i7-1185G7 processor (4 cores, 8 threads, base frequency 3.0 GHz, turbo up to 4.8 GHz, 12 MB cache), 16 GB RAM, and a 500 GB SSD, powered by a Tattu G-tech 5200 mAh 4S 35C 14.8V LiPo battery. This configuration provides sufficient computational power to reliably execute the control loop at 50 Hz (sampling time $T_s = 20$ ms), while also handling the perception, state estimation, and communication tasks required by the robot.

The control architecture is implemented in ROS 2 Foxy Fitzroy on Ubuntu 20.04 (Focal). The slam_toolbox package was used to perform mapping and localization, combining odometry from wheel encoders, IMU, and LiDAR scans. This fusion provided an accurate estimate of the odometry suitable for feedback to the MPC implemented in this study. Figure 1 shows the hardware layout of the experimental robot, highlighting the perception module and the embedded control electronics.

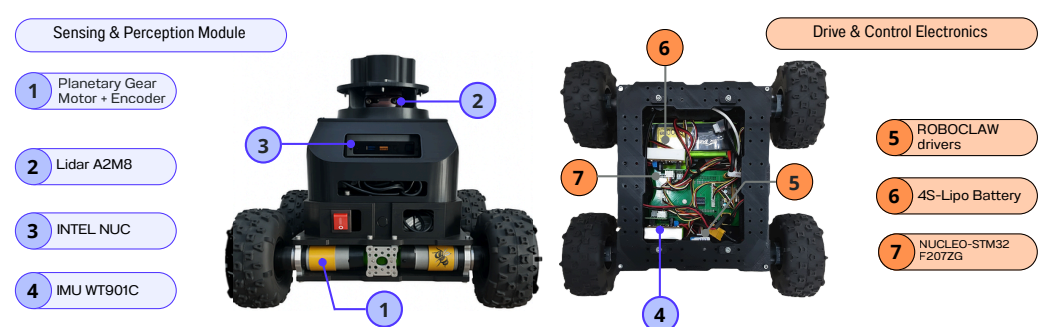


Figure 1. Hardware architecture of the mobile robot, showing the two main subsystems: (left) the Sensing and Perception Module including the LiDAR, planetary geared motors with encoders, IMU and on-board computer; (right) the Drive and Control Electronics including the motor drivers, embedded controller and power supply.

The experiments were carried out on a flat concrete surface outside, surrounded by buildings that improved the accuracy of the location of the SLAM system. The physical platform is capable of reaching linear velocities of up to approximately 1.7 m/s and angular velocities close to 2.0 rad/s. However, operating near these limits can place excessive stress on the motors and reduce control robustness. For this reason, this study was configured with conservative bounds of $v = \pm 1.5$ m/s and $\omega = \pm 1.75$ rad/s. These limits were selected as a trade-off between exploiting the full capability of the robot and ensuring safe and reliable operation during experiments.

The experimental setup is illustrated in Figure 2. The robot follows a predefined reference trajectory on a tiled floor, while three perturbation points (P1–P3) are marked along the path. These points are later used to inject controlled disturbances into the system, allowing us to evaluate the robustness of the control strategy under non-ideal conditions.

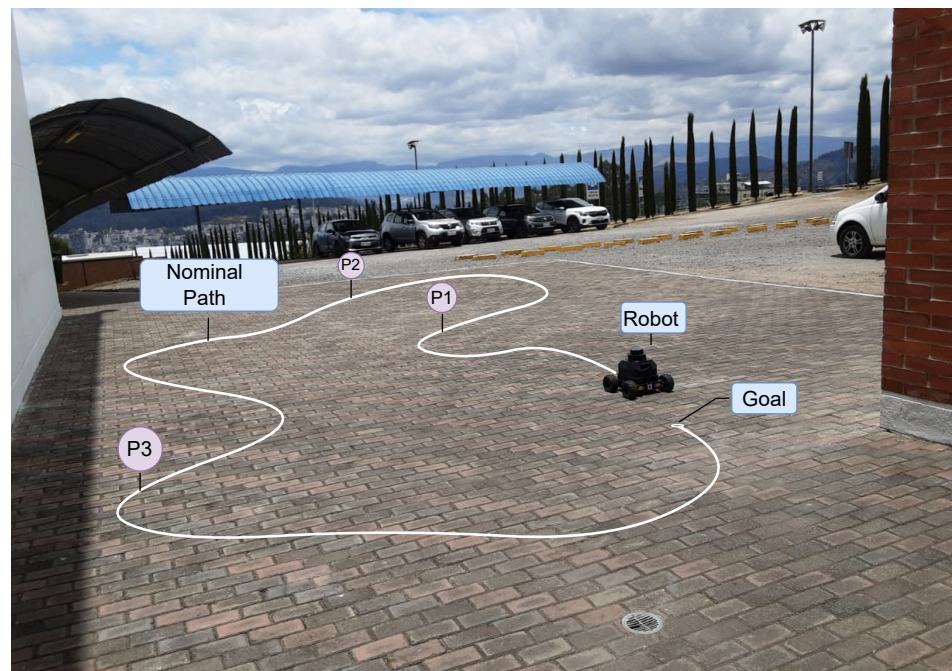


Figure 2. Experimental robot platform and nominal path with labeled perturbation points (P1–P3) used for subsequent evaluation.

3.2. MPC Formulation

The trajectory tracking problem is formulated as a discrete-time Model Predictive Control (MPC) optimization. The robot state vector is

$$\mathbf{x}_k = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix}^\top, \quad (1)$$

where (x_k, y_k) are the Cartesian coordinates and θ_k is the heading angle at time step k . The control inputs are the commanded linear and angular velocities

$$\mathbf{u}_k = \begin{bmatrix} v_k & \omega_k \end{bmatrix}^\top. \quad (2)$$

Let the robot state at time k be $\mathbf{x}_k = [x_k, y_k, \theta_k]^\top$, whose positional component $\mathbf{p}_k = [x_k, y_k]^\top$ is used for geometric computations on the reference path $\mathcal{P}$. The nominal path is defined as a finite sequence of M spatial poses,

$$\mathcal{P} = \{(x_i, y_i, \theta_i)\}_{i=1}^M, \quad (3)$$

where the positional component $\mathbf{p}_i = [x_i, y_i]^\top$ denotes the Cartesian coordinates and θ_i the heading of the i -th spatial point.

At each control cycle, the algorithm computes the lateral tracking error $e_y(k)$ between the robot position and each point of $\mathcal{P}$ as

$$e_y(k) = (\mathbf{p}_k - \mathbf{p}_i)^\top \begin{bmatrix} -\sin \theta_i \\ \cos \theta_i \end{bmatrix}, \quad (4)$$

and determines the closest point on the path as

$$\mathbf{p}_c = \arg \min_{\mathbf{p}_i \in \mathcal{P}} |e_y(k)|. \quad (5)$$

Consequently, the pose $\mathbf{p}_c = (x_c, y_c, \theta_c)$ defines the local tangent direction $\mathbf{t}_c = [\cos \theta_c, \sin \theta_c]^\top$ and serves as the spatial anchor for the prediction horizon. The scalar $e_y(k)$ quantifies the signed perpendicular deviation of the robot from the nominal path and is also used as the activation criterion for the re-entry mechanism (Section 3.3).

The MPC then extracts from $\mathcal{P}$ a local reference horizon $\mathcal{H}_k$ of length N_p , composed of the sequence of poses ahead of $\mathbf{p}_c$:

$$\mathcal{H}_k = \{\mathbf{x}_{c+1}^{ref}, \mathbf{x}_{c+2}^{ref}, \dots, \mathbf{x}_{c+N_p}^{ref}\}, \quad (6)$$

where each $\mathbf{x}_{c+i}^{ref} = [x_{c+i}, y_{c+i}, \theta_{c+i}]^\top, i = 1, \dots, N_p$, represents a spatial reference point along the path. Each element in this horizon corresponds to a discrete prediction step associated with the controller sampling period T_s , so that the predicted state $\mathbf{x}_{k+i}$ represents the expected robot pose at time $t_k + iT_s$. This spatial re-alignment ensures that the reference horizon always begins from the point on $\mathcal{P}$ closest to the robot, providing consistent geometric tracking even when the path was recorded at variable speeds. If $|e_y(k)| > \delta$, the nominal horizon $\mathcal{H}_k$ is replaced by a re-entry horizon as described in Section 3.3.

The nonlinear kinematic model of the differential-drive robot is

$$x_{k+1} = x_k + T_s v_k \cos(\theta_k), \quad (7)$$

$$y_{k+1} = y_k + T_s v_k \sin(\theta_k), \quad (8)$$

$$\theta_{k+1} = \theta_k + T_s \omega_k, \quad (9)$$

where T_s is the sampling time. For prediction within MPC, the model is linearized at each step around the operating point $(\hat{x}_k, \hat{y}_k, \hat{\theta}_k, \hat{v}_k, \hat{\omega}_k)$, yielding

$$\mathbf{x}_{k+1} \approx A_k \mathbf{x}_k + B_k \mathbf{u}_k, \quad (10)$$

where $A_k \in \mathbb{R}^{3 \times 3}$ and $B_k \in \mathbb{R}^{3 \times 2}$ are Jacobians.

Given a prediction horizon of length N_p , the stacked predicted state and control vectors are defined as

$$\mathbf{X} = [\mathbf{x}_{k+1}^\top \ \cdots \ \mathbf{x}_{k+N_p}^\top]^\top, \quad \mathbf{U} = [\mathbf{u}_k^\top \ \mathbf{u}_{k+1}^\top \ \cdots \ \mathbf{u}_{k+N_p-1}^\top]^\top. \quad (11)$$

Here, $\mathbf{X} \in \mathbb{R}^{3N_p}$ contains the predicted robot states, while $\mathbf{U} \in \mathbb{R}^{2N_p}$ represents the sequence of control inputs to be optimized by the MPC. Since no reference control sequence is provided by the nominal path, the control inputs are computed online as decision variables of the optimization problem.

The cost function penalizes both the deviation from the spatially aligned reference trajectory and the magnitude of the control effort, according to

$$J = \sum_{i=1}^{N_p} \left(\|\mathbf{x}_{k+i} - \mathbf{x}_{c+i}^{ref}\|_Q^2 + \|\mathbf{u}_{k+i-1}\|_R^2 \right), \quad (12)$$

where $Q \in \mathbb{R}^{3 \times 3} \succeq 0$ and $R \in \mathbb{R}^{2 \times 2} \succ 0$ are diagonal weighting matrices for the state and input, respectively. The matrix Q shapes the relative importance of position and orientation errors in the tracking term, while R penalizes excessive control effort and indirectly influences the smoothness of the commanded velocities. The vectors $\mathbf{x}_{c+i}^{ref}$ are the N_p reference poses contained in $\mathcal{H}_k$, and thus define the local tracking targets used in the optimization. This cost can be written in quadratic form as

$$J = \frac{1}{2} \mathbf{U}^\top H \mathbf{U} + f^\top \mathbf{U}, \quad (13)$$

where

$$H = \Gamma^\top Q_b \Gamma + R_b, \quad f = \Gamma^\top Q_b (\Phi \mathbf{x}_k - \mathbf{x}^{ref}), \quad (14)$$

with Φ and Γ the prediction matrices. The matrix $\Phi \in \mathbb{R}^{3N_p \times 3}$ propagates the current state through the prediction horizon, while $\Gamma \in \mathbb{R}^{3N_p \times 2N_p}$ is the control-to-state matrix (Toeplitz form) that maps the control input sequence to the predicted state trajectory:

$$\mathbf{X} = \Phi \mathbf{x}_k + \Gamma \mathbf{U}. \quad (15)$$

The block-diagonal matrices $Q_b = \text{blkdiag}(Q, \dots, Q)$ and $R_b = \text{blkdiag}(R, \dots, R)$ encode the stage weights along the horizon.

The MPC optimization problem at each sampling instant is therefore

$$\min_{\mathbf{U} \in \mathbb{R}^{2N_p}} J \quad (16)$$

$$\text{s.t. } \mathbf{x}_{k+i+1} = A_k \mathbf{x}_{k+i} + B_k \mathbf{u}_{k+i}, \quad (17)$$

$$v_{\min} \leq v_{k+i} \leq v_{\max}, \quad (18)$$

$$\omega_{\min} \leq \omega_{k+i} \leq \omega_{\max}, \quad (19)$$

$$\forall i = 0, \dots, N_p - 1. \quad (20)$$

Although the optimization is carried out over the control sequence $\mathbf{U}$, the cost function $J(\mathbf{U})$ penalizes the deviation between the predicted and reference trajectories, thus driving the robot state toward the desired path while regulating the control effort. The linear velocity was constrained to $v_{\min} = -1.5$ m/s and $v_{\max} = +1.5$ m/s, and the angular velocity to $\omega_{\min} = -1.75$ rad/s and $\omega_{\max} = +1.75$ rad/s, reflecting the physical capabilities of the drive motors. Although no additional constraints on the rate of change of v or ω were imposed, the weighting on control effort smooths the commanded inputs, preventing abrupt variations within the feasible bounds.

At each step, only the first optimal input $\mathbf{u}_k^* = [v_k^*, \omega_k^*]^\top$ is applied to the robot (receding horizon principle). The complete online implementation is summarized in Algorithm 1, which directly reflects the Python 3.8 node executed in the robot; it reads odometry, constructs the prediction horizon, builds the quadratic program, solves the optimization, and publishes the command. Optional disturbances can be enabled to test robustness. Algorithm 1 calls Algorithm 2 to compute the reference horizon $\mathcal{H}_k$ (nominal or re-entry) at each iteration.

Algorithm 1 Online MPC Control Loop with Optional Disturbances and Re-Entry Strategy

Require: $T_s, N_p, Q \succeq 0, R \succ 0$, bounds $u_{\min}, u_{\max}$, band δ , reference path $\mathcal{P}$, optional disturbances $\mathcal{D}$

```

1: while goal not reached do
2:   Read  $\mathbf{x}_k = [x_k, y_k, \theta_k]^\top$ 
3:   Compute lateral errors  $e_y(k)$  for all  $\mathbf{p}_i \in \mathcal{P}$ 
4:   Find closest point  $\mathbf{p}_c = (x_c, y_c, \theta_c) = \arg \min_{\mathbf{p}_i \in \mathcal{P}} |e_y(k)|$ 
5:    $\mathcal{H}_k \leftarrow \text{COMPUTEREFERENCEHORIZON}(\mathbf{x}_k, \mathbf{p}_c, \mathcal{P}, N_p, \delta)$ 
6:   Build  $\mathbf{X}^{ref}$  by stacking  $\mathbf{x}_{c+i}^{ref}$  from  $\mathcal{H}_k$ 
7:   Linearize model at  $(\hat{x}_k, \hat{y}_k, \hat{\theta}_k, \hat{v}_k, \hat{\omega}_k)$  to obtain  $A_k, B_k$ 
8:   Construct prediction matrices  $(\Phi, \Gamma)$  from  $(A_k, B_k)$  and  $N_p$ 
9:   Form  $Q_b = \text{blkdiag}(Q, \dots, Q), R_b = \text{blkdiag}(R, \dots, R)$ 
10:  Compute  $H = \Gamma^\top Q_b \Gamma + R_b, f = \Gamma^\top Q_b (\Phi \mathbf{x}_k - \mathbf{X}^{ref})$ 
11:  Solve  $\min_{\mathbf{U} \in \mathbb{R}^{2N_p}} \frac{1}{2} \mathbf{U}^\top H \mathbf{U} + f^\top \mathbf{U}$  s.t.  $u_{\min} \leq \mathbf{U} \leq u_{\max}$ 
12:  Extract and apply  $\mathbf{u}_k^* = [v_k^*, \omega_k^*]^\top$ 
13:  if a disturbance  $(p_j, v_j, \omega_j, \tau_j) \in \mathcal{D}$  is due then
14:    Apply  $(v_j, \omega_j)$  for  $\tau_j$  seconds; mark as applied; continue
15:  Update metrics ( $e_y$ , band exits, # re-entries)
```

▷ see Algorithm 2

3.3. Re-Entry Strategy

Although MPC can correct small deviations, large disturbances or wheel slip can drive the robot outside the admissible lateral band $\delta = 0.05$ m. In such cases, tracking the nominal path directly would require aggressive maneuvers, leading to instability or loss of robustness. To mitigate this, a *re-entry strategy* is introduced to temporarily replace the nominal reference horizon with a smooth interpolated sequence that guides the robot back into the band.

In each control cycle, the closest point $\mathbf{p}_c = (x_c, y_c, \theta_c) \in \mathcal{P}$ is already calculated as part of the horizon construction described in Section 3.2 and detailed in Algorithm 1. When the signed lateral deviation satisfies $|e_y(k)| > \delta$, the re-entry mechanism is activated.

A smoothing point $\mathbf{p}_s = (x_s, y_s, \theta_s) \in \mathcal{P}$ is selected ahead of $\mathbf{p}_c$ at a prescribed look-ahead distance L_{smooth} such that the Euclidean distance between them satisfies

$$\|\mathbf{p}_s - \mathbf{p}_c\| \geq L_{\text{smooth}}. \quad (21)$$

The re-entry seed is then built as

$$\mathcal{S} = \{\mathbf{p}_r, \mathbf{p}_s, \mathbf{x}_{s+1}^{\text{ref}}, \dots, \mathbf{x}_{s+N_p-2}^{\text{ref}}\}, \quad (22)$$

where $\mathbf{p}_r = [x_k, y_k, \theta_r]^\top$ represents the robot pose with an adjusted heading θ_r oriented toward $\mathbf{p}_s$, defined as

$$\theta_r = \tan^{-1}\left(\frac{y_s - y_k}{x_s - x_k}\right), \quad (23)$$

This alignment ensures that the first reference orientation in the temporary horizon is geometrically consistent with the direction of re-entry, rather than maintaining the robot's current heading θ_k .

Next, the seed sequence $\mathcal{S}$ is interpolated to obtain continuous functions $x(s)$, $y(s)$, and $\theta(s)$ parameterized by arc-length s . The cumulative distance along $\mathcal{S}$ is computed as

$$s_j = s_{j-1} + \|[x_j, y_j]^\top - [x_{j-1}, y_{j-1}]^\top\|, \quad s_0 = 0, \quad (24)$$

and resampled uniformly at arc-length increments Δs :

$$s_\ell = \ell \cdot \Delta s, \quad \ell = 0, 1, \dots, \lfloor s_{\max} / \Delta s \rfloor. \quad (25)$$

Evaluating the interpolants at $\{s_\ell\}$ yields

$$\tilde{\mathcal{P}} = \{(x(s_\ell), y(s_\ell), \theta(s_\ell))\}, \quad (26)$$

from which the first N_p samples are taken as the re-entry horizon $\mathcal{H}_k$. If fewer than N_p samples are available (e.g., near the end of the path), the horizon is padded with copies of the last nominal pose to maintain constant length.

This re-entry mechanism is depicted in Figure 3. Starting from the robot pose $\mathbf{p}_r$, the algorithm selects a smoothing point $\mathbf{p}_s$ ahead of the nearest nominal point $\mathbf{p}_c$ at a prescribed look-ahead distance L_{smooth} . A new interpolated trajectory $\tilde{\mathcal{P}}$ is then generated by combining $\mathbf{p}_r$, $\mathbf{p}_s$, and the following $N_p - 2$ nominal points. This interpolated sequence defines the temporary horizon for the MPC, ensuring smooth convergence back to the reference path $\mathcal{P}$.

When $|e_y| \leq \delta$, the horizon $\mathcal{H}_k$ coincides with the next N_p nominal points $\{\mathbf{x}_{c+1}^{\text{ref}}, \dots, \mathbf{x}_{c+N_p}^{\text{ref}}\}$. Otherwise, $\mathcal{H}_k$ is replaced by the resampled re-entry horizon obtained from $\tilde{\mathcal{P}}$. In both cases, $\mathcal{H}_k$ is stacked into $\mathbf{X}^{\text{ref}}$ for the quadratic program. The complete

MPC without terminal stabilizing ingredients, which directly supports our formulation. Furthermore, the general theoretical foundations of stability for MPC schemes based solely on the stage cost are rigorously established in the nonlinear MPC framework of Grüne and Pannek [19], reinforcing the validity of the control strategy adopted in this work. Note also that the re-entry trajectory is only used as a bounded, smooth and feasible external reference signal and does not modify the MPC formulation; therefore, the stability guarantees of the baseline controller remain preserved.

3.5. Experimental Setup and Evaluation

To validate the proposed MPC controller and the re-entry strategy, eight experiments were carried out on the real differential-drive robot. For clarity, experiments are renumbered consecutively from 1 to 8 in this paper. All runs used a fixed lateral band of $\delta = 0.05$ m and a prediction horizon of $N_p = 15$, corresponding to a prediction interval of 0.3 s given the controller sampling period $T_s = 0.02$ s. This horizon length was selected as a compromise between computational feasibility and sufficient anticipation of the robot dynamics, since larger horizons significantly increased the solver time on the onboard computer.

The diagonal weighting matrix $Q = [25.0, 25.0, 20.0]$ was kept fixed across all experiments so that the relative importance of position (x, y) and heading (θ) errors remained constant. These values were empirically chosen to give slightly higher priority to position accuracy than to orientation correction, promoting smooth yet precise trajectory tracking. Fixing Q allows us to evaluate the effect of different configurations of R on the trade-off between tracking accuracy and control smoothness without altering the tracking priority itself. Smaller R values lead to faster and more aggressive control actions, whereas larger R values promote smoother but slower responses. Additionally, both linear and cubic interpolation schemes were tested to assess the influence of trajectory smoothing on the re-entry behavior. Tables 1 and 2 summarize the complete configurations. The column L_{smooth} indicates the look-ahead distance used to place the smoothing point when re-entry is active (Section 3.3).

Table 1. Experiments without perturbations (IDs 1–5).

ID	Re-Entry	R	L_{smooth} [m]	Interp.	Justification
1	Active	[1.0, 2.0]	0.30	Linear	Baseline with re-entry. Tests linear interpolation vs. cubic.
2	Active	[1.0, 2.0]	0.25	Cubic	Re-entry with cubic interpolation. Shorter look-ahead to speed re-entry.
3	Active	[0.5, 1.0]	0.30	Cubic	Smaller R to allow stronger control action. Evaluates aggressiveness of corrections.
4	Active	[2.0, 4.0]	0.30	Cubic	Larger R to emphasize smoother control commands. Explores accuracy vs. smoothness trade-off.
5	Inactive	[1.0, 2.0]	–	–	Baseline without re-entry. Used as comparison against active strategy.

The second part introduces deliberate perturbations to probe robustness. These perturbations were triggered once the robot progressed beyond predefined indices of the nominal path $\mathcal{P} = \{(x_i, y_i, \theta_i)\}_{i=1}^M$, specifically at $i \in \{500, 1000, 1500\}$. Each perturbation overrides the MPC output for a fixed duration with the specified velocity command:

- P1: $(v, \omega) = (0.5 \text{ m/s}, -0.75 \text{ rad/s})$ for 3.0 s, forward motion with a strong right-turn.
 P2: $(v, \omega) = (0.5 \text{ m/s}, -0.5 \text{ rad/s})$ for 2.0 s, forward motion with a moderate right turn.
 P3: $(v, \omega) = (0.2 \text{ m/s}, 1.0 \text{ rad/s})$ for 2.0 s, forward motion with a left turn.

These cover sudden heading changes, brief stops/slowdowns, and lateral deviations.

Table 2. Experiments with perturbations (IDs 6–8).

ID	Re-Entry	R	L_{smooth} [m]	Interp.	Justification
6	Active	[1.0, 2.0]	0.25	Cubic	Re-entry with cubic interpolation. Shorter look-ahead for faster recovery.
7	Active	[0.5, 1.0]	0.30	Cubic	Re-entry with reduced R values. Tests robustness with stronger control effort.
8	Inactive	[1.0, 2.0]	–	–	Baseline without re-entry. Used as reference under perturbations.

As evaluation metrics, the Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and the 95th percentile of the absolute lateral deviation (P_{95}) were computed from the lateral error at the robot position $\mathbf{p}_k = (x_k, y_k)$ as follows:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{k=1}^N e_y(k)^2}, \quad (27)$$

$$\text{MAE} = \frac{1}{N} \sum_{k=1}^N |e_y(k)|, \quad (28)$$

$$P_{95} = 95\text{th percentile of } |e_y(k)|, \quad (29)$$

where N is the number of samples and T_s is the sampling period. The portion of the trajectory in which the lateral error remains within the validation band $\delta = 0.05$ m is also quantified. This metric is computed as

$$\text{Pct}_{\text{InBand}} = 100 \cdot \frac{1}{N} \sum_{k=1}^N [|e_y(k)| \leq \delta], \quad (30)$$

The resulting value expresses the fraction of the trajectory contained within the $\pm\delta$ band, normalized to a percentage scale.

In addition to the lateral error, a heading error $e_\theta(k)$ is calculated by comparing the measured orientation of the robot θ_k with the orientation θ_c of the tangent at the projection point $\mathbf{p}_c$. This ensures that the heading deviation is always evaluated consistently with respect to the closest segment of the nominal path. Quantitative metrics such as RMSE, MAE and the 95th percentile are then obtained for $e_\theta(k)$ using the same formulas as above.

Beyond tracking accuracy, the smoothness and dynamic quality of control signals were also assessed. Specifically, the *RateRMS* and *JerkRMS* metrics quantify the root mean square of the rate of change and the second derivative (jerk) of the commanded linear and angular velocities, respectively, reflecting how abruptly the controller varies its commands. Finally, the Integrated Smoothness Number (ISN) provides a dimensionless indicator of overall control smoothness, with higher values ($\text{ISN} \approx 1$) indicating smoother and more stable action.

4. Results

This section presents the quantitative and qualitative evaluation of the proposed MPC controller under two sets of conditions: (i) nominal operation without external perturbations (IDs 1–5) and (ii) perturbed operation with three injected disturbances (IDs 6–8). The results are reported both in tabular form, highlighting performance metrics such as lateral and angular errors, and graphically, showing the robot trajectories, error profiles, and control signals.

4.1. Experiments Under Nominal Conditions

Table 3 summarizes the quantitative results for experiments under nominal conditions (IDs 1–5), which are consistent with the qualitative observations in Figure 4. All controllers achieved stable trajectory tracking, although with different levels of accuracy and control smoothness. The best overall performance was obtained in ID 2 (active re-entry, $R = [1.0, 2.0]$, $L_{\text{smooth}} = 0.25$ m), which yielded the lowest lateral RMSE (0.05 m), smallest P95 error (0.10 m), and the highest in-band operation (68.8%). In terms of control behavior, ID 2 also achieved balanced smoothness, with moderate rate and jerk values ($\text{RateRMS}_v = 1.05$ m/s², $\text{RateRMS}_\omega = 8.21$ rad/s²; $\text{JerkRMS}_v = 68.44$ m/s³, $\text{JerkRMS}_\omega = 661.33$ rad/s³) and a high smoothness index ($\text{ISN} = 0.95$). ID 1 showed a comparable response (lateral RMSE 0.06 m, $\text{ISN} = 0.96$), remaining smooth and well-centered along the nominal trajectory. By contrast, ID 3 (reduced weights, $R = [0.5, 1.0]$) exhibited a more aggressive and oscillatory response, with increased lateral RMSE (0.15 m), higher $\text{RateRMS}_\omega = 16.88$ rad/s² and $\text{JerkRMS}_\omega = 1310.98$ rad/s³, and only 25.2% of the run within the ± 5 cm band. ID 4 achieved intermediate performance (lateral RMSE 0.10 m, $\text{RateRMS}_\omega = 8.12$ rad/s²), while the baseline case without re-entry (ID 5) maintained smooth velocity profiles ($\text{ISN} = 0.97$) but showed lower robustness and slightly larger tracking deviations.

Table 3. Quantitative results of experiments under nominal conditions (IDs 1–5).

Section	Metric	ID 1 (A,[1,2], 0.30, L)	ID 2 (A,[1,2], 0.25, C)	ID 3 (A,[0.5,1.0], 0.30, C)	ID 4 (A,[2,4], 0.30, C)	ID 5 (I,[1,2], 0.30, C)
Trajectory	RMSE e_y [m]	0.06	0.05	0.15	0.10	0.07
	MAE e_y [m]	0.04	0.04	0.12	0.07	0.06
	P95 e_y [m]	0.12	0.10	0.27	0.18	0.15
	RMSE e_θ [rad]	0.13	0.06	0.34	0.09	0.06
	MAE e_θ [rad]	0.10	0.04	0.27	0.07	0.05
	P95 e_θ [rad]	0.28	0.11	0.64	0.18	0.13
	Pct InBand [%]	63.3	68.8	25.2	48.5	52.6
Controller	RateRMS $\dot{v}$ [m/s ²]	1.90	1.05	4.24	0.87	1.44
	RateRMS $\dot{\omega}$ [rad/s ²]	3.70	8.21	16.88	8.12	1.43
	JerkRMS $\ddot{v}$ [m/s ³]	135.29	68.44	272.10	59.83	101.20
	JerkRMS $\ddot{\omega}$ [rad/s ³]	225.50	661.33	1310.98	655.11	88.94
	ISN [-]	0.96	0.95	0.92	0.94	0.97

The velocity subplots in Figure 4 provide further insight. In ID 3, the smaller values of R produced stronger corrective actions, which appear as high-amplitude oscillations in both v and ω . In contrast, ID 4 (larger R) generated slower commands, but this did not reduce the overall error as the robot required more time to re-enter the nominal path compared to ID 2. This behavior illustrates the trade-off between control aggressiveness and re-entry speed.

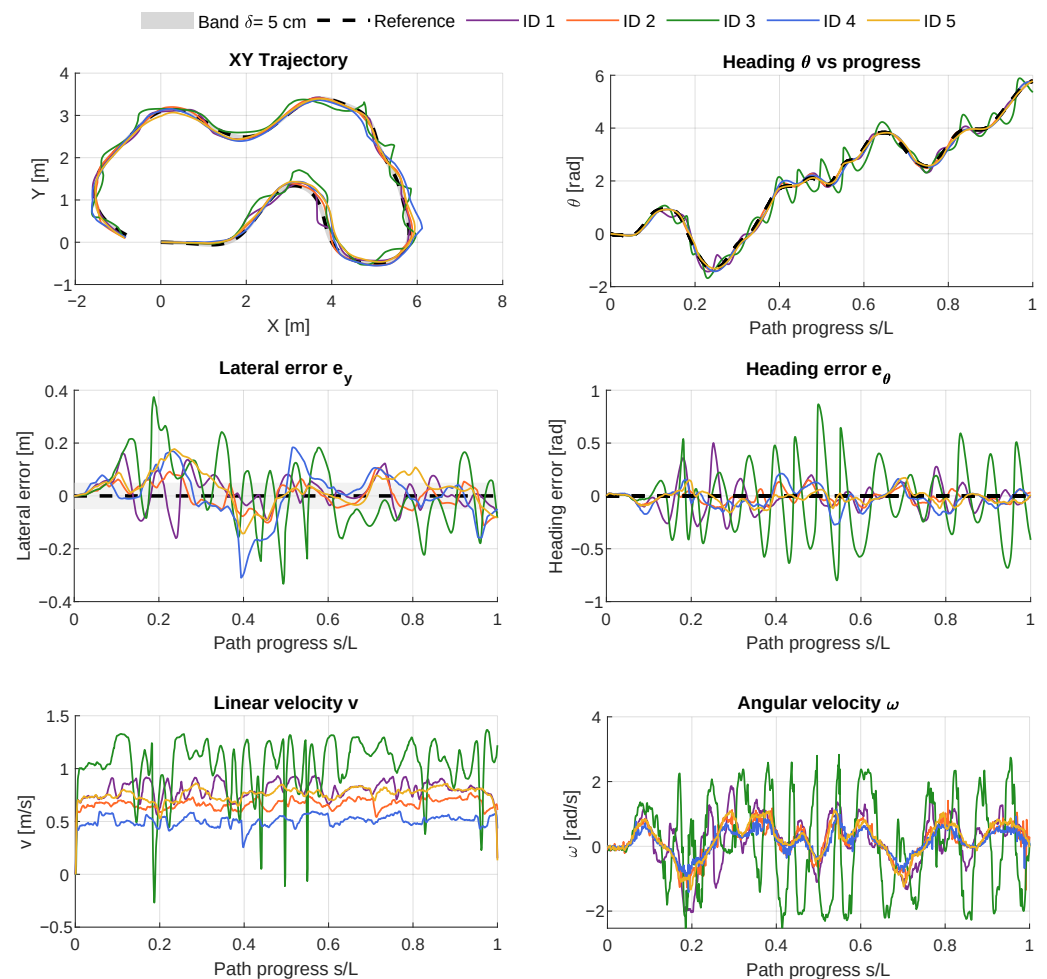


Figure 4. Experiments under nominal conditions (IDs 1–5) in Table 1. Comparison of the nominal path against tracked trajectories and error profiles. The x-axis represents the normalized path progress s/L , with s the arc length traveled along the reference trajectory and L its total length.

4.2. Experiments Under Perturbations

Table 4 reports the results for IDs 6–8, where external disturbances were injected into the velocity commands.

Table 4. Quantitative results of experiments under perturbations (IDs 6–8).

Section	Metric	ID 6 (A,[1,2], 0.25, C)	ID 7 (A,[0.5,1.0], 0.30, C)	ID 8 (I,[1,2], 0.30, C)
Trajectory	RMSE e_y [m]	0.12	0.19	0.19
	MAE e_y [m]	0.08	0.15	0.13
	P95 e_y [m]	0.25	0.37	0.37
	RMSE e_θ [rad]	0.45	0.46	0.47
	MAE e_θ [rad]	0.24	0.34	0.22
	P95 e_θ [rad]	1.16	0.85	1.28
	Pct InBand [%]	51.4	27.0	29.4
Controller	RateRMS $\ddot{v}$ [m/s ²]	2.19	4.09	3.50
	RateRMS $\dot{\omega}$ [rad/s ²]	10.34	14.20	5.46
	JerkRMS $\ddot{v}$ [m/s ³]	146.08	257.53	236.41
	JerkRMS $\dot{\omega}$ [rad/s ³]	773.85	1058.09	375.24
	ISN [-]	0.95	0.94	0.97

As expected, the disturbances significantly affected the tracking accuracy of all controllers. ID 6, with the same configuration as the best nominal case (ID 2), achieved the

most robust response under perturbations, with a lower lateral RMSE (0.12 m) and a higher percentage of in-band operation (51.4%) compared to IDs 7 and 8. In terms of controller behavior, ID 6 maintained moderate smoothness under disturbances ($\text{RateRMS}_v = 2.19 \text{ m/s}^2$, $\text{RateRMS}_\omega = 10.34 \text{ rad/s}^2$; $\text{JerkRMS}_v = 146.08 \text{ m/s}^3$, $\text{JerkRMS}_\omega = 773.85 \text{ rad/s}^3$) and preserved a high smoothness index ($\text{ISN} = 0.95$). By contrast, ID 7, configured with smaller R values, produced more aggressive but less stable control actions (higher rate and jerk values, $\text{ISN} = 0.94$), while ID 8 (inactive re-entry) resulted in smoother commands ($\text{ISN} = 0.97$) but prolonged deviations from the reference path and a lower in-band percentage (29.4%).

Figure 5 illustrates these effects. The shaded bands (P1–P3) indicate the intervals of injected perturbations. At P1 (strong right turn while moving forward), the robot is deflected to the right; ID 6 shows a rapid return to the band, whereas IDs 7 and 8 drift further before re-aligning. At P2 (moderate right turn), the induced deviation is smaller and the recovery of ID 6 is again faster than IDs 7–8. Finally, at P3 (left turn), the largest heading excursion is observed; ID 7 reacts more aggressively with visible oscillations before stabilizing, while ID 6 maintains a smoother recovery. These behaviors are evident in both the lateral error and velocity subplots, where the shaded bands delineate disturbance and re-entry phases. It should be noted that all subplots expressed with respect to the nominal path progress (s/L)—namely lateral error, heading θ , heading error e_θ , and the velocity profiles—are affected by this parameterization. Consequently, the shaded region at P3 appears narrower because the perturbation commands induce limited forward motion along the path, which reduces the apparent duration of the disturbance when plotted in normalized progress coordinates.

4.3. Sensitivity Analysis

The performance of the controller was evaluated under different parameter configurations to assess its sensitivity and to obtain practical tuning guidelines. In Table 3, the input weight R , the smoothing distance L_{smooth} , the interpolation mode, and the activation of the re-entry strategy were considered. Reducing L_{smooth} from 0.30 m to 0.25 m decreases the lateral RMSE by approximately 15% and increases the in-band operation from 63.3% to 68.8%, indicating that L_{smooth} effectively trades off re-entry speed against smoothness.

The parameter R has a strong influence on stability. Lower penalization ($R = [0.5, 1.0]$) increases the lateral RMSE from 0.05 m to 0.15 m (200% increase) and reduces the in-band percentage from 68.8% to 25.2% (63% reduction), while higher R produces smoother and less oscillatory control but slower re-entry. Under perturbations (Table 4), this trend is maintained, the RMSE increases from 0.12 m to 0.19 m (58% increase) when using a smaller R . These observations provide a practical tuning rule for R , in which larger values improve smoothness and robustness, while smaller values lead to faster but more aggressive convergence.

The type of interpolation also plays a role. Cubic interpolation yields the lowest nominal RMSE (0.05 m), compared to linear interpolation (0.06 m), suggesting a small but measurable improvement in convergence and smoothness. The activation of the re-entry mechanism also exhibits a clear sensitivity effect. Disabling the strategy (ID 5) increases the lateral RMSE to 0.07 m and reduces the in-band operation to 52.6%, while enabling it (ID 2) improves these values to 0.05 m and 68.8%. Under perturbations, enabling the re-entry reduces the RMSE from 0.19 m to 0.12 m (37% reduction) and increases the in-band percentage from 29.4% to 51.4%.

Overall, the tested configurations demonstrate predictable trends in the controller response, which can be used as practical guidelines for parameter tuning in real deployments.

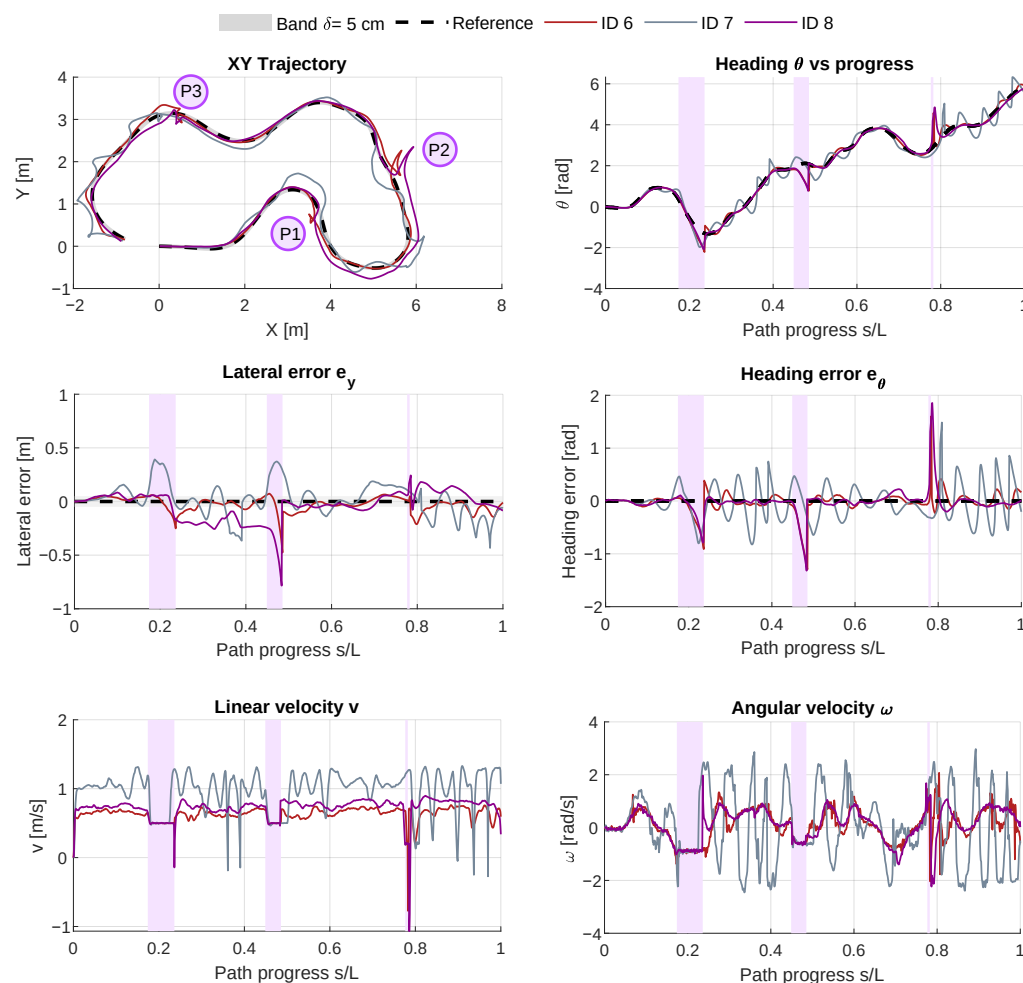


Figure 5. Experiments under perturbed conditions (IDs 6–8) in Table 2. Shaded regions correspond to perturbations P1–P3. The x -axis represents the normalized path progress s/L , with s the arc length traveled along the reference trajectory and L its total length.

4.4. Computational Requirements and Real-Time Feasibility

A real-time feasibility analysis was conducted using the best-performing configuration (ID2: $R = [1.0, 2.0]$, $L_{\text{smooth}} = 0.25$ m, cubic interpolation), both with and without the re-entry strategy. The controller was implemented in Python 3.8 using CVXPY as the modeling framework and OSQP as the numerical QP solver [20,21], and executed on an Intel NUC equipped with an Intel Core i7 at 3.0 GHz and 16 GB RAM.

Table 5 reports the computation times of the complete MPC loop. The average total computation time (Avg. Total) corresponds to the mean duration of one control iteration, including QP formulation, solver execution, reference update and input/output operations. The maximum observed computation time (Max. Total) reflects the worst-case duration measured during execution. The average QP solve time (Avg. QP Solve) represents the effort associated exclusively with solving the quadratic program.

Table 5. Computation time statistics for the MPC loop.

Mode	Avg. Total [ms]	Max. Total [ms]	Avg. QP Solve [ms]
MPC without re-entry	17.2–18.0	27.9	11.8–12.1
MPC with re-entry	19.8–21.6	27.4	12.2–12.5

Since the sampling period is 20 ms, the computation time per iteration must remain below this threshold to guarantee real-time execution. The MPC loop satisfies this re-

quirement on average in both configurations, and the QP solve time remains significantly below the sampling period (approximately 12 ms), which confirms that the computational bottleneck is not the MPC optimization problem itself.

Enabling the re-entry strategy increases the average computation time by approximately 2–3 ms, which represents an overhead of 11–15% with respect to the baseline. The maximum observed computation time remains almost unchanged (27.4 ms vs. 27.9 ms). This small increment is mainly due to reference update and feasibility checks, and does not modify the complexity of the MPC problem, since the QP solve time remains essentially the same in both cases (11.8–12.5 ms).

These results demonstrate that the proposed re-entry strategy introduces only a minor computational overhead while providing improved tracking and robustness. Furthermore, the implementation is based on Python 3.8 and ROS2 Foxy Fitzroy, which inherently introduce software overhead. A compiled implementation (for example, C++ with OSQP or an embedded QP solver) would further reduce execution time and provide additional margin. Overall, the proposed controller is compatible with real-time execution under the given 20 ms sampling period, even when the re-entry strategy is active.

5. Discussion

The experimental results demonstrate the effectiveness of the proposed contributions: the real-time implementation of MPC for trajectory tracking in a DDMR under ROS 2, the novel dynamic look-ahead re-entry strategy for band exits, and the robustness gains confirmed by systematic perturbation trials. Under nominal conditions, the best performance was obtained in configuration ID 2, where the re-entry strategy was active with $R = [1.0, 2.0]$, smoothing distance $L_{\text{smooth}} = 0.25$ m, and cubic interpolation. This setting yielded the lowest lateral error (RMSE 0.05 m, $P_{95} = 0.10$ m) and the highest in-band ratio (68.8%). When perturbations were introduced, the same configuration (ID 6) achieved the most stable recovery, maintaining RMSE of 0.12 m and more than 51% of the trajectory within the band, while weaker penalization (ID 7) or the absence of re-entry (ID 8) led to substantial error growth and longer excursions. These observations confirm that the re-entry trajectory smooths deviations without saturating the control inputs, fulfilling the second contribution of this work.

The literature also highlights the importance of carefully tuning the horizon and weightings in MPC to prevent oscillations or excessive delays. Xu et al. [1] reported improved heading and lateral accuracy in a 3WID3WIS robot by using dynamic MPC, especially in curved paths, which is consistent with our finding that interpolation and smoothing distance strongly affect the correction dynamics. Similarly, Bouzoualegh et al. [6] compared kinematic and dynamic MPC for DDMRs, showing that more realistic models can improve accuracy but increase sensitivity to disturbances; in our case, the re-entry strategy mitigated this sensitivity by reshaping the reference rather than relying solely on the internal model. Abed et al. [3] and Pang et al. [4] introduced fractional and time-varying MPC approaches that enhance stability and robustness, while Ali et al. [2] embedded safety guarantees through barrier functions. By contrast, our approach addresses robustness at the trajectory level; when the robot leaves the nominal path, a smooth local reference is reconstructed, guiding the system back with moderated curvature. This complements but does not replace robust/tube MPC formulations [10,22], which tighten constraints around the nominal trajectory, or learning-based MPC [5,13], which rely on data to compensate for uncertainties.

Recent work in dynamic trajectory re-planning further reinforces the importance of adapting the reference rather than depending exclusively on the predictive model. Lei et al. [15] and She et al. [17] demonstrated that online replanning improves performance

in uncertain environments by modifying the trajectory when disturbances or dynamic changes appear. Likewise, Zhang et al. [16] showed that dynamic obstacle avoidance benefits from modifying the local path rather than only tightening constraints. These studies support the motivation behind the re-entry strategy proposed here; providing a lightweight and smooth adjustment of the reference allows rapid recovery from deviations without increasing the MPC complexity or altering the model.

In contrast, the hierarchical framework by Wang et al. [23] integrates MPC with local planners (DWA) and adapts horizon and reference points in real time, achieving improvements in tracking error and reference path following under complex navigation tasks. The re-entry strategy directly reduces the duration and amplitude of error peaks, as evidenced in P1–P3, fulfilling the third contribution of this paper.

If a quantitative comparison is made with related works, it should be considered that the reported performance metrics depend strongly on each robot's mechanical design, sensor configuration, and testing environment. Therefore, quantitative comparisons are provided only to offer a general reference framework rather than a direct benchmark. For example, recent studies on skid-steer robots using NMPC achieved lateral RMSEs around 0.10–0.15 m in outdoor conditions [22], while we reached 0.05 m under nominal conditions and 0.12 m under perturbations. Gao et al. [9] reported robust tube MPC with cross-track errors between 0.08 and 0.12 m, values comparable to our perturbed cases but obtained without an explicit re-entry mechanism.

Other quantitative benchmarks from recent work further contextualize the improvements observed here. Wang et al. [13] showed that an NMPC enhanced with Gaussian processes reduced lateral RMSE by approximately 15.1% compared to a linear MPC with simple feedback under nominal conditions. This level of improvement is consistent with our configuration ID 2 (lateral RMSE 0.05 m), which exhibited similarly reduced tracking errors relative to less regularized configurations. Likewise, Chen et al. [10] demonstrated that Tube-RMPC decreased maximum trajectory deviation by 9.17% in S-curve maneuvers and reduced MSE by up to 29.38% in double lane change scenarios. Finally, Ravankar et al. [24] highlighted that arc-length parameterization with smoothing interpolation enhances tracking stability and reduces oscillations in mobile robot controllers—consistent with our observation that cubic interpolation with moderate smoothing distances improves re-entry robustness. Overall, although these results were obtained under different experimental setups, the comparative magnitudes suggest that the proposed re-entry strategy and cubic smoothing achieve tracking performance that is quantitatively competitive and qualitatively robust under both nominal and perturbed conditions.

The best-performing configuration in all eight experiments was ID 2/6, where the MPC controller achieved an effective trade-off between control smoothness and tracking accuracy. Its balanced input penalization, cubic interpolation, and moderate smoothing distance minimized both steady-state and transient errors, as reflected in low rate and jerk magnitudes together with a high smoothness index ($ISN \approx 0.95$). In particular, the re-entry strategy mitigated the accumulation of large deviations, shortened recovery times, and promoted smoother control profiles than the baseline MPC without re-entry.

However, some limitations remain and they point to concrete avenues for future research. First, the evaluation was restricted to a single robot platform and a flat outdoor environment; performance may vary on uneven terrain, at higher speeds, or with different sensing conditions. Second, formal safety mechanisms such as barrier functions [2] or invariant tubes [10] were not considered in this work, but they could be integrated to enhance safety guarantees. Third, the re-entry mechanism could be extended toward full dynamic replanning or adaptive tuning of L_{smooth} based on the measured deviation, or combined with learning-based corrections, as suggested in [13,25].

These directions outline the most promising next steps to improve the generalizability, safety guarantees, and applicability of the proposed method.

6. Conclusions

In this study, an MPC controller with a novel dynamic look-ahead re-entry strategy was implemented and validated on a four-wheel differential-drive robot under ROS 2. The proposed approach was experimentally tested along a nominal path under both nominal conditions and external perturbations.

Quantitative results demonstrated that, in nominal runs, the best-performing configuration (ID 2) achieved a lateral RMSE of 0.05 m, a heading RMSE of 0.056 rad, and an in-band ratio of 68.8%, outperforming other tested settings. Under perturbations, the re-entry strategy showed clear benefits; for instance, experiment ID 6 maintained a lateral RMSE of 0.12 m and a heading RMSE of 0.45 rad, while still keeping the robot within the validation band for 51.4% of the path, evidencing improved robustness compared to MPC without re-entry. Overall, these results confirm that the proposed re-entry mechanism with cubic smoothing interpolation enhances trajectory tracking stability and robustness, particularly in the presence of disturbances.

Future work will focus on adaptive tuning of the smoothing horizon L_{smooth} based on the measured deviation, as well as on combining the re-entry strategy with learning-based corrections to further improve performance in complex environments. This study highlights the potential of the proposed method as a baseline for future control architectures in autonomous mobile robots.

Beyond these contributions, the results of this work naturally lead to several avenues for future research. A first direction is the extension of the re-entry mechanism toward full dynamic trajectory re-planning, where the reference could be reshaped online in response to obstacles or environmental changes. A second direction is the integration of the method with safety-aware MPC formulations (such as control barrier functions or tube-based MPC), which would provide formal guarantees under constraints and uncertainties. A third opportunity lies in the adaptive or learning-based adjustment of the smoothing parameters and interpolation mode, so that the controller can automatically adapt to terrain, speed or disturbance variations. Finally, further investigation in embedded implementations or multi-robot settings would enable a broader range of practical deployments. These directions highlight the potential of the re-entry mechanism as a foundation upon which more comprehensive MPC-based navigation architectures can be developed.

Author Contributions: Conceptualization, D.G., M.F.M.M., M.A.H., S.B.S., J.O.O.G.; Formal analysis, D.G., M.F.M.M., M.A.H., S.B.S., J.O.O.G., A.B., M.H., D.Á.; Methodology, D.G., M.F.M.M., M.A.H., S.B.S., J.O.O.G.; Writing—original draft, D.G., M.F.M.M., A.B., M.H., D.Á. All authors have read and agreed to the published version of the manuscript.

Funding: This publication is part of the R&D&I project PID2023-150967OB-I00605 (Multirobot System for Gait Analysis, ROBOMESH), funded by MICIU/AEI/10.13039/501100011033 and the FEDER/EU.

Data Availability Statement: Dataset available on request from the authors.

Acknowledgments: The authors would like to express their gratitude to the R&D&I project PID2023-150967OB-I00605 (Multirobot System for Gait Analysis, ROBOMESH), funded by MICIU/AEI/10.13039/501100011033 and the FEDER/EU.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Xu, C.; Zhou, X.; Chen, R.; Li, B.; He, W.; Li, Y.; Ye, F. Trajectory Tracking for 3-Wheeled Independent Drive and Steering Mobile Robot Based on Dynamic Model Predictive Control. *Appl. Sci.* **2025**, *15*, 485. [\[CrossRef\]](#)
2. Ali, A.M.; Shen, C.; Hashim, H.A. A Linear MPC with Control Barrier Functions for Differential Drive Robots. *IET Control Theory Appl.* **2024**, *18*, 2693–2704. [\[CrossRef\]](#)
3. Abed, A.M.; Rashid, Z.N.; Abedi, F.; Zeebaree, S.R.M.; Sahib, M.A.; Jawad, A.J.M.; Ibraheem, G.A.R.; Maher, R.A.; Abdulkareem, A.I.; Ibraheem, I.K.; et al. Trajectory Tracking of Differential Drive Mobile Robots using Nonlinear Neural Network Fractional Controller. *Meas. Control* **2022**, *55*, 209–226. [\[CrossRef\]](#)
4. Pang, H.; Liu, N.; Hu, C.; Xu, Z. A Practical Trajectory Tracking Control of Autonomous Vehicles using LTV-MPC. *Proc. Inst. Mech. Eng. Part D J. Automob. Eng.* **2022**, *236*, 709–723. [\[CrossRef\]](#)
5. Zhang, K.; Wang, J.; Xin, X.; Li, X.; Sun, C.; Huang, J.; Kong, W. A Survey on Learning-Based Model Predictive Control: Toward Path Tracking Control of Mobile Platforms. *Appl. Sci.* **2022**, *12*, 1995. [\[CrossRef\]](#)
6. Bouzoualegh, S.; Guechi, E.H.; Kelaiaia, R. Model Predictive Control of a Differential-Drive Mobile Robot based on the Dynamic Model. *Acta Univ. Sapientiae Electr. Mech. Eng.* **2018**, *10*, 20–41. [\[CrossRef\]](#)
7. Zhou, X.; Liu, R.; Zhou, J.; Zhang, Z.; Wang, C.; Zhao, W. Robust Tube-MPC Trajectory Tracking Control for Four-Wheel Independent Steering Vehicles on Intermittent Snowy and Icy Roads. *Chin. J. Mech. Eng.* **2025**, *38*, 132. [\[CrossRef\]](#)
8. Ait Dahmad, H.; Ayad, H.; García Cerezo, A.; Mousannif, H. Adaptive Model Predictive Control for 4WD-4WS Mobile Robot: A Multivariate Gaussian Mixture Model–Ant Colony Optimization for Robust Trajectory Tracking and Obstacle Avoidance. *Sensors* **2025**, *25*, 3805. [\[CrossRef\]](#) [\[PubMed\]](#)
9. Gao, X.; Su, X.; An, A.; Zhang, H. Robust Model Predictive Tracking Control for the Wheeled Mobile Robot. *IET Cyber-Syst. Robot.* **2023**, *5*, 298–309. [\[CrossRef\]](#)
10. Chen, Y.; Sun, Y.; Li, J.; He, J.; He, C. Tube-based Robust Model Predictive Control for Autonomous Vehicle Control. *Appl. Sci.* **2025**, *15*, 6471. [\[CrossRef\]](#)
11. Aro, K.; Guevara, L.; Torres-Torriti, M.; Torres, F.; Prado, A. Robust Nonlinear Model Predictive Control for the Trajectory Tracking of Skid-Steer Mobile Manipulators with Wheel-Ground Interactions. *Robotics* **2024**, *13*, 171. [\[CrossRef\]](#)
12. Zheng, W.; Zhu, B. Stochastic Time-Varying Model Predictive Control for Trajectory Tracking of a Wheeled Mobile Robot. *Front. Energy Res.* **2021**, *9*, 767597. [\[CrossRef\]](#)
13. Wang, J.; Fader, M.T.H.; Marshall, J.A. Learning-based model predictive control for improved mobile robot path following using Gaussian processes and feedback linearization. *J. Field Robot.* **2023**, *40*, 1014–1033. [\[CrossRef\]](#)
14. Dorbetkhany, Z.; Murbabulatov, A.; Rubagotti, M.; Shintemirov, A. Spatial-Based Model Predictive Path Following Control for Skid Steering Mobile Robots. In Proceedings of the 2022 18th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA), Taipei, Taiwan, 28–30 November 2022; pp. 1–6. [\[CrossRef\]](#)
15. Lei, M.; Lu, L.; Laurenzi, A.; Rossini, L.; Romiti, E.; Malzahn, J.; Tsagarakis, N.G. An MPC-Based Framework for Dynamic Trajectory Re-Planning in Uncertain Environments. In Proceedings of the 2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids), Ginowan, Japan, 28–30 November 2022; pp. 594–601. [\[CrossRef\]](#)
16. Zhang, Z.; Yang, G.H. Dynamic obstacle avoidance for car-like mobile robots based on neurodynamic optimization with control barrier functions. *Neurocomputing* **2025**, *654*, 131252. [\[CrossRef\]](#)
17. She, Y.; Song, C.; Sun, Z.; Li, B. Optimized Model Predictive Control-Based Path Planning for Multiple Wheeled Mobile Robots in Uncertain Environments. *Drones* **2025**, *9*, 39. [\[CrossRef\]](#)
18. Worthmann, K.; Mehrez, M.W.; Zanon, M.; Mann, G.K.; Gosine, R.G.; Diehl, M. Regulation of Differential Drive Robots using Continuous Time MPC without Stabilizing Constraints or Costs. *IFAC-PapersOnLine* **2015**, *48*, 129–135. [\[CrossRef\]](#)
19. Grüne, L.; Pannek, J. *Nonlinear Model Predictive Control: Theory and Algorithms*, 2nd ed.; Communications and Control Engineering; Springer: Cham, Switzerland, 2017; p. 456. [\[CrossRef\]](#)
20. Stellato, B.; Banjac, G.; Goulart, P.; Bemporad, A.; Boyd, S. OSQP: An Operator Splitting Solver for Quadratic Programs. *Math. Program. Comput.* **2020**, *12*, 637–672. [\[CrossRef\]](#)
21. Diamond, S.; Boyd, S. CVXPY: A Python-embedded modeling language for convex optimization. *J. Mach. Learn. Res.* **2016**, *17*, 2909–2913.
22. Wang, J.; Li, T.; Zhang, Y. Trajectory Tracking Control of a Skid-Steer Mobile Robot Using Nonlinear MPC. *Appl. Sci.* **2023**, *14*, 122. [\[CrossRef\]](#)
23. Wang, Z.; Wang, S.; Xie, Y.; Xiong, T.; Wang, C. Autonomous Navigation of Mobile Robots: A Hierarchical Planning-Control Framework with Integrated DWA and MPC. *Sensors* **2025**, *25*, 2014. [\[CrossRef\]](#)

24. Ravankar, A.; Ravankar, A.A.; Kobayashi, Y.; Hoshino, Y.; Peng, C. Path smoothing techniques in robot navigation: State-of-the-art, current and future challenges. *Sensors* **2018**, *18*, 3170. [[CrossRef](#)] [[PubMed](#)]
25. Eschmann, H.; Ebel, D.; Althoff, M. Exploration-exploitation-based trajectory tracking of mobile robots using Gaussian processes and model predictive control. *Robotica* **2023**, *41*, 3040–3058. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.