TELKOMNIKA Telecommunication Computing Electronics and Control
Vol. 24, No. 5, October 2026, pp. 1768~1778
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v24i5.27870

Hybrid neural network-augmented model predictive control for differential-drive robot tracking

Taoufik Belkebir[1], Hicham Belkebir[2], Anass Mansouri[2]
[1]LRSI, Faculty of Sciences and Technology (FST), University Sidi Mohammed Ben Abdellah, Fez, Morocco
[2]LRSI, National School of Applied Sciences (ENSA), Sidi Mohamed Ben Abdellah University, Fez, Morocco

Article Info

Article history:

Received Feb 16, 2026
Revised Apr 5, 2026
Accepted May 25, 2026

Keywords:

Differential-drive robot
Direct collocation
Multiple shooting
Neural network
Nonlinear model predictive control
Trajectory tracking
Warm-starting

ABSTRACT

Nonlinear model predictive control (NMPC) is widely adopted for mobile robot trajectory tracking, yet the interaction between transcription methods and learning-based enhancements remains unstudied for differential-drive platforms. This paper presents a unified CasADi/Interior Point OPTimizer (IPOPT) evaluation of fourth-order Runge-Kutta (RK4) multiple shooting (MS) and Hermite-Simpson (HS) direct collocation on a TurtleBot3 Burger across ten scenarios spanning five trajectory families, model mismatch, and sensor noise. A π-ambiguity in the standard $sin^2(\Delta\theta)$ heading cost is identified and corrected, reducing figure-eight tracking error by approximately 95%. RK4 MS achieves superior baseline accuracy in 9 of 10 experiments ($p < 0.01$). Three hybrid neural approaches are evaluated: warm-starting reduces solve time by 16% with no accuracy loss; adaptive transcription selection outperforms the baseline in 3 of 10 experiments; and neural approximate control achieves sub-millisecond inference but degrades on unseen trajectories (51%-64% fallback rate). A generalization hierarchy emerges: solver-in-the-loop methods generalize without measurable degradation, whereas solver-replacement methods require trajectory-specific training. These findings suggest that solver involvement is a key factor influencing generalization in learning-augmented predictive control. All results are statistically validated across 500 simulations.

Corresponding Author:

Taoufik Belkebir
LRSI, Faculty of Sciences and Technology (FST), University Sidi Mohammed Ben Abdellah
Fez 30000, Morocco
Email: belkebir.taoufik@usmba.ac.ma

1. INTRODUCTION

Model predictive control (MPC) is widely recognized as a leading framework for trajectory tracking in mobile robotics, owing to its systematic treatment of constraints and ability to incorporate nonlinear dynamics over a receding prediction horizon [1]–[4]. For differential-drive platforms such as the TurtleBot3 Burger widely used in research and education the nonholonomic kinematic constraints and limited actuation bandwidth ($v_{\max} = 0.22$ m/s) demand careful controller design [5], [6]. The transcription method used to convert the continuous-time optimal control problem (OCP) into a finite-dimensional nonlinear program (NLP) significantly impacts both solution accuracy and computational feasibility [1], [7], [8]. This paper addresses three underexplored aspects of nonlinear model predictive control (NMPC) for differential-drive robots: the choice of transcription method, the integration of learning-based enhancements, and an often-overlooked heading cost ambiguity.

While transcription methods (multiple shooting (MS), direct collocation, pseudospectral) have been extensively benchmarked for autonomous vehicles [3], [9]–[11], aerospace trajectory optimization [7], [8],

and autonomous mobile platforms [5], [12], their systematic comparison for differential-drive mobile robots with tight actuation limits has not yet been systematically studied. Existing implementations rely predominantly on fixed-step discretization without rigorous evaluation of alternatives [3], [5]. Furthermore, learning-augmented MPC including neural warm-starting [13], neural approximate control with safety guarantees [14]–[17], and adaptive algorithm selection [18] has been explored primarily for high-dimensional systems. Unlike the quadrotor platforms studied by Salzmann *et al*. [19] and Torrente *et al*. [20], or the racing and manipulation setups of [21]–[24], differential-drive robots operate under tighter velocity and actuation constraints, where even small transcription differences can affect real-time feasibility. Moreover, the heading error formulation an often-overlooked design choice has received insufficient attention despite its role in nonholonomic tracking stability: the widely-used $\sin^2(\Delta\theta)$ cost contains a π-ambiguity that can cause severe tracking failure on high-curvature trajectories.

Tang *et al*. [3] proposed geometric MPC for wheeled robots with curvature-aware cost design, but did not compare transcription methods or integrate learning-based augmentation. Nascimento *et al*. [5] developed modified NMPC for nonholonomic mobile robots, though only Euler discretization was evaluated. Rosenfelder *et al*. [25] employed Koopman-based predictive control for nonholonomic systems, without examining heading cost formulation. On the transcription side, Kelly [8] surveyed direct collocation methods and Verschueren *et al*. [7] developed the acados framework, yet neither addressed differential-drive-specific trade-offs. For learning-augmented MPC, Hertneck *et al*. [14] established approximate MPC with safety guarantees for general nonlinear systems; Karg and Lucia [15] demonstrated efficient deep-learning approximation of MPC laws; and Salzmann *et al*. [19] achieved real-time neural MPC for quadrotors. However, none of these works compared multiple hybrid strategies on the same platform or evaluated their generalization to unseen trajectories a gap this paper addresses. Additionally, Rosolia and Borrelli [13] developed learning MPC for iterative tasks, Cauligi *et al*. [26] applied supervised learning for mixed-integer MPC, Drgoňa *et al*. [16] proposed differentiable predictive control, and Hewing *et al*. [2] surveyed the field broadly, all targeting systems with higher-dimensional state spaces. This work makes four principal contributions:

- Heading cost correction: the standard $\sin^2(\Delta\theta)$ heading cost is shown to have two global minima ($\Delta\theta = 0$ and $\Delta\theta = \pi$). A corrected $(1 - \cos\Delta\theta) + \sin^2\Delta\theta$ formulation is proposed and proven to have a unique global minimum (Theorem 1), reducing figure-eight tracking error by approximately 95%. Notably, this ambiguity is difficult to detect in practice as it manifests only on high-curvature trajectories and has not been previously identified in the differential-drive NMPC literature.
- Systematic baseline evaluation: a unified CasADi/Interior Point OPTimizer (IPOPT) comparison of RK4 MS versus Hermite-Simpson direct collocation across ten scenarios, establishing RK4 MS superiority in 9 of 10 experiments ($p < 0.01$).
- Three hybrid neural approaches: (i) neural network (NN) warm-starting with 16% solve-time reduction; (ii) adaptive method selection that outperforms the baseline in 3 of 10 experiments with bounded regret; and (iii) neural approximate model predictive control (NA-MPC) with sub-millisecond inference and safety-constrained fallback.
- Generalization hierarchy: a controlled evaluation revealing that solver-in-the-loop methods generalize without measurable degradation, while solver-replacement methods show statistically significant accuracy loss ($p < 0.01$). This result addresses a gap in the literature, where prior work [14], [15], [19] evaluated hybrid methods individually rather than comparatively across generalization regimes on a common platform.

The remainder of this paper is organized as follows. Section 2 presents the research method, including the robot model, NMPC formulation, heading cost correction, transcription methods, and hybrid neural approaches. Section 3 reports the experimental results and discussion, and section 4 concludes.

2. METHOD

Figure 1 presents the overall control architecture. The reference trajectory is fed to the NMPC controller, which formulates an OCP with the corrected heading cost (Theorem 1). The NLP is solved by CasADi/IPOPT and the resulting controls are applied to the TurtleBot3 in a receding-horizon loop. Three hybrid neural augmentations are evaluated: M1 warm-starting the NLP with a learned initial guess, M2 adaptive selection of the transcription method, and M3 direct policy approximation that replaces online optimization with neural inference (with safety fallback).

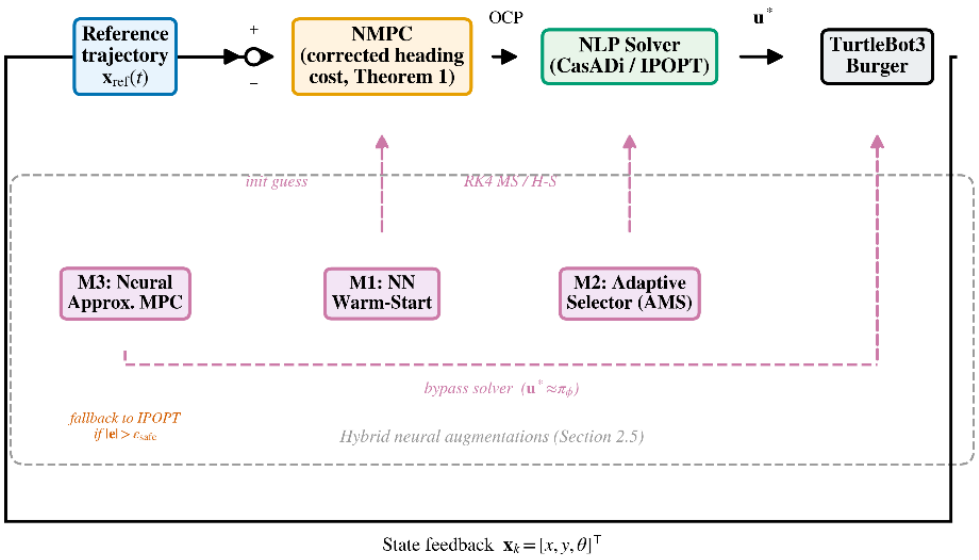

Figure 1. Control architecture

The NMPC pipeline (top row) solves a corrected-heading OCP at each control step. Three hybrid neural augmentations (dashed box) are evaluated: M1 provides a learned warm-start to accelerate convergence, M2 selects the transcription method adaptively, and M3 replaces online optimization with direct policy inference under safety-constrained fallback.

2.1. Differential-drive robot model

The TurtleBot3 Burger is modeled as a unicycle-type differential-drive robot [5], [6] with state $x = [x, y, \theta]^\top$ and control inputs $u = [v, \omega]^\top$. The continuous-time kinematics are:

$$\dot{x} = f(x,u) = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} \quad (1)$$

This model captures the essential nonholonomic constraint: the robot cannot move laterally, with all motion arising from the interplay of linear velocity v and angular velocity ω. The physical constraints are: $v \in [0,0.22]$ m/s (forward-only), $\omega \in [-2.84,2.84]$ rad/s, wheel separation $L = 0.160$ m, and wheel radius $r = 0.033$ m [6].

2.2. NMPC OCP

At each sampling time t_k, the following OCP is solved over the prediction horizon T [1], [7]:

$$\min_{x(\cdot),u(\cdot)} \int_0^T \ell\,(x,u,x_{ref})\,dt + \Phi(x(T)) \quad (2)$$

subject to dynamics (1), initial condition $x(0) = x_k$, and control bounds $u \in \mathcal{U}$. The objective balances tracking accuracy (stage cost ℓ) against terminal deviation (Φ), driving the robot to follow the reference trajectory while satisfying physical constraints. The stage cost with the corrected heading formulation (section 2.3) is:

$$\ell = w_p \parallel p - p_{ref} \parallel^2 + w_{\theta_1}(1 - cos\Delta\theta) + w_{\theta_2} sin^2\Delta\theta + w_v v^2 + w_\omega \omega^2 + w_{\Delta\omega}(\omega_{k+1} - \omega_k)^2 \quad (3)$$

where $\Delta\theta = \theta - \theta_{\text{des}}$ and $\theta_{\text{des}} = \text{atan2}(\dot{y}_{\text{ref}}, \dot{x}_{\text{ref}})$ is computed analytically from trajectory derivatives [3]. The first two terms penalize position and heading errors, while the remaining terms promote smooth, energy-efficient control inputs. The weights are: $w_p = 50$, $w_{\theta_1} = 10$, $w_{\theta_2} = 5$, $w_v = 0.1$, $w_\omega = 0.5$, $w_{\Delta\omega} = 5$, with terminal weight $w_T = 100$. The controller uses $N = 20$ intervals, $T = 2.0$ s horizon, and $\Delta t = 0.1$ s control period (10 Hz), solved via IPOPT through CasADi [27]. Key design choices include: (i) positive-only

velocity to prevent reverse-motion chattering [25]; (ii) tighter angular rate bounds ($\omega \in [-1.5,1.5]$ rad/s) for smoother trajectories; (iii) corrected heading cost (section 2.3); and (iv) angular rate smoothness penalty to prevent chattering.

2.3. Heading cost correction: resolving the π-ambiguity

The heading cost deserves dedicated treatment because, as shown below, a subtle structural flaw in the standard formulation can cause severe tracking failure that only manifests on high-curvature trajectories precisely the conditions under which differential-drive robots are most challenged. The heading cost $\ell_\theta^{\text{std}} = \sin^2(\Delta\theta)$ is commonly used in nonholonomic MPC formulations [3], [5] due to its smoothness and implicit angle wrapping. However, this formulation contains a structural ambiguity.

– Lemma 1 (π-periodicity)

The function $\sin^2(\Delta\theta)$ satisfies $\sin^2(\Delta\theta) = \sin^2(\Delta\theta + \pi)$, having two global minima per 2π-period: at $\Delta\theta = 0$ (correct heading) and $\Delta\theta = \pi$ (backward heading). On high-curvature trajectories (e.g., figure-eight crossover points), the NLP solver can converge to $\theta_{\text{des}} + \pi$, causing the robot to track the trajectory while pointing backward an unacceptable failure for forward-only differential-drive platforms. The following corrected heading cost is proposed:

$$\ell_\theta(\Delta\theta) = (1 - cos\Delta\theta) + sin^2\Delta\theta \tag{4}$$

– Theorem 1 (unique global minimum)

The heading cost (4) satisfies: (i) $\ell_\theta(\Delta\theta) \geq 0$ with equality if $\Delta\theta = 0$; (ii) ℓ_θ has a unique global minimum at $\Delta\theta = 0$; (iii) $\ell_\theta(\Delta\theta) = \frac{3}{2}\Delta\theta^2 + O(\Delta\theta^4)$ near the minimum.

Proof. Using $\sin^2\Delta\theta = 1 - \cos^2\Delta\theta$:

$$\ell_\theta = (1 - cos\Delta\theta)(2 + cos\Delta\theta) \tag{5}$$

Since $cos\Delta\theta \in [-1,1]$, we have $(1 - \cos\Delta\theta) \geq 0$ and $(2 + \cos\Delta\theta) \geq 1 > 0$. Their product is zero if $cos\Delta\theta = 1$, i.e., $\Delta\theta = 0$. The derivative $\ell_\theta' = \sin\Delta\theta(1 + 2\cos\Delta\theta)$ yields critical points at $\Delta\theta \in \{0, \pm\pi, \pm 2\pi/3\}$ in $[-\pi, \pi]$. Evaluating: $\ell_\theta(0) = 0$ (global minimum), $\ell_\theta(\pm\pi) = 2$ (local minimum), $\ell_\theta(\pm 2\pi/3) = 9/4$ (local maxima). The second derivative confirms $\ell_\theta''(\pm 2\pi/3) < 0$ (maxima) and $\ell_\theta''(0) = 3 > 0$ (minimum).

The corrected formulation combines the disambiguation of $(1 - cos\Delta\theta)$ with the local shaping of $\sin^2\Delta\theta$, yielding the steepest gradient (3 × the cosine-only variant) near the minimum while remaining smooth and 2π-periodic. The function ℓ_θ also satisfies Lyapunov candidate properties (positive definite, zero at equilibrium), making it suitable for NMPC stability analysis under standard terminal cost assumptions [1], [2].

2.4. Transcription methods

Two transcription methods with matching $O(h^4)$ local truncation error are compared: Hermite-Simpson direct collocation and RK4 MS [7], [8], [28]. Hermite-Simpson direct collocation uses cubic Hermite interpolation with Simpson's rule quadrature [8], [27]. For each interval $[t_k, t_{k+1}]$ with step $h = T/N$, the midpoint state is interpolated as $\mathrm{x}_m = \frac{1}{2}(\mathrm{x}_k + \mathrm{x}_{k+1}) + \frac{h}{8}(f_k - f_{k+1})$, and the Simpson quadrature constraint is $\mathrm{x}_{k+1} = \mathrm{x}_k + \frac{h}{6}(f_k + 4f_m + f_{k+1})$. The midpoint states are additional decision variables, yielding $(2N + 1)n_x + (N + 1)n_u = 163$ variables and $6N = 120$ constraints at $N = 20$. RK4 MS divides the horizon into intervals and uses explicit 4th-order Runge-Kutta integration within each [7], [28]. Continuity is enforced as NLP constraints (shooting gaps). This yields $(N + 1)n_x + Nn_u = 103$ variables and $3N = 60$ constraints a 37% smaller problem. The simpler NLP structure (fewer variables, sparser Jacobian, no implicit midpoint coupling) contributes to more robust convergence, particularly under model mismatch.

2.5. Hybrid NN methods

Learning-augmented MPC methods can be categorized by their degree of solver involvement. Solver-in-the-loop methods retain the full optimization solver and use the NN only to accelerate convergence (e.g., warm-starting). Solver-selection methods use a learned classifier to choose among solvers but still execute traditional optimization. Solver-replacement methods replace online optimization entirely with direct policy inference, using safety mechanisms to fall back to the solver when needed. Three representative approaches spanning this spectrum are evaluated, trained on 4,650 samples from two trajectory families (circle, figure-eight) under diverse conditions.

– Method 1: NN warm-start (NN-WS)

A fully connected network (8→128→128→64→103, rectified linear unit (ReLU) activations) predicts near-optimal NLP decision variables from the robot state and reference [13]. Input features are $[x, y, \theta, x_{\text{ref}}, y_{\text{ref}}, \dot{x}_{\text{ref}}, \dot{y}_{\text{ref}}, \kappa]$. The prediction initializes IPOPT via $opti.set\_initial()$, retaining full solver feasibility and optimality guarantees. Training uses Adam optimizer, 500 epochs (final mean squared error (MSE): 8.68×10^{-4}).

– Method 2: adaptive method selector (AMS)

A binary classifier (9→64→32→1, sigmoid output) predicts which transcription method achieves lower error at each control step [18]. An additional input feature σ_{noise} encodes the noise level. Training uses binary cross-entropy, 300 epochs (accuracy: 89.7%). Both solvers are dispatched normally only the selection is learned.

– Method 3: NA-MPC

A policy network (9→128→64→32→2, tanh output scaled to control bounds) directly predicts $[v^*, \omega^*]$, replacing online optimization with direct policy inference [14], [16]. A safety mechanism monitors tracking error: if $\| p - p_{ref} \| > \varepsilon_{safe} = 2$ cm, the controller falls back to full IPOPT (RK4 MS) [17]. Training uses MSE loss, 500 epochs (final MSE: 0.192).

3. RESULTS AND DISCUSSION

3.1. Experimental setup

All methods are evaluated on five trajectory families: (i) smooth circular ($r = 0.3$ m), (ii) smooth figure-eight ($A = 0.25$ m), (iii) lemniscate of Bernoulli ($a = 0.35$ m), (iv) S-curve (Fourier-based), and (v) multi-waypoint path (6 points, cubic spline). Trajectories span curvature profiles from $\kappa_{\max} = 3.33$ to $68.2\ \text{m}^{-1}$. Trajectories 1–2 are used for NN training; trajectories 3–5 are held out for generalization testing. Figure 2 illustrates the trajectory library.

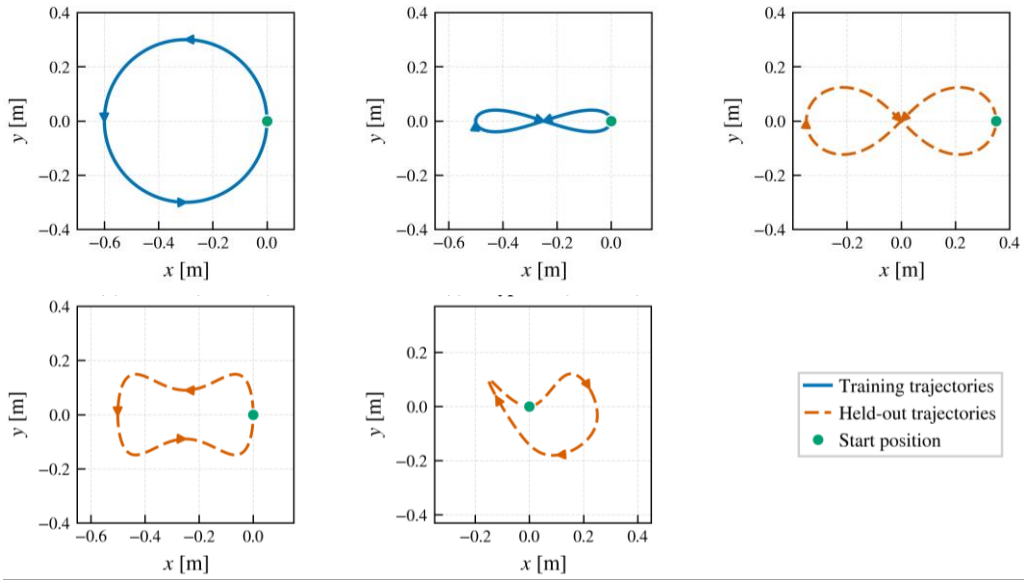

Figure 2. Five trajectory families used for evaluation

Circle and figure-eight (solid lines) serve as the NN training set; lemniscate, S-curve, and waypoints (dashed lines) are held out for generalization testing. Curvature ranges span $\kappa_{\max} = 3.33$ to $68.2\ \text{m}^{-1}$. Ten experiments evaluate robustness and generalization: nominal (Exp. 1), model mismatch $\pm 10\%$ (Exps. 2–3), Gaussian noise $\sigma \in \{0.5,1,2\}$ cm (Exps. 4–6), figure-eight (Exp. 7), and three held-out trajectories (Exps. 8–10). All experiments use $N = 20$, $T = 2.0$ s, $\Delta t = 0.1$ s, repeated with 10 random seeds (42–51), totaling 500 simulations (5 methods × 10 experiments × 10 seeds). Simulations run on Intel Xeon (4 CPU cores, 30 GB RAM). Statistical significance is assessed via the Wilcoxon signed-rank test with $\alpha = 0.01$.

3.1.1. Baseline results

Table 1 presents the baseline comparison between RK4 MS and Hermite-Simpson direct collocation. RK4 MS achieves lower or equal tracking error in 9 of 10 experiments (significantly better in 7, non-significant in 2). The only exception is the lemniscate (Exp. 8), where H-S achieves a statistically significant but marginal 0.001 cm advantage. Both methods maintain sub-centimeter nominal accuracy (0.57 cm vs. 0.71 cm). Under high noise ($\sigma \geq 1$ cm, Exps. 5–6), the difference becomes non-significant, indicating that measurement noise dominates transcription effects. Average solve times are comparable: 142 ms (RK4 MS) vs. 152 ms (H-S). Notably, the corrected heading cost (Theorem 1) reduces figure-eight error from 6.09 cm to 0.29 cm (95% reduction).

Table 1. Baseline comparison: RK4 MS vs. Hermite-Simpson direct collocation. Mean tracking error (cm) over 10 seeds. Significance: $\star\star\ p < 0.01$, ns = not significant

#	Experiment	RK4 MS	H-S	Δ (cm)	p	Winner
1	Nominal	0.567	0.714	0.147	0.002**	RK4 MS
2	Model −10%	1.006	1.163	0.157	0.002**	RK4 MS
3	Model +10%	0.227	0.357	0.130	0.002**	RK4 MS
4	Noise σ=0.5 cm	0.704	0.772	0.068	0.002**	RK4 MS
5	Noise σ=1 cm	0.937	0.927	−0.010	0.695	—
6	Noise σ=2 cm	1.238	1.292	0.054	0.375	—
7	Figure-eight	0.292	0.318	0.026	0.002**	RK4 MS
8	Lemniscate	0.462	0.461	−0.001	0.002**	H-S
9	S-curve	0.348	0.477	0.129	0.002**	RK4 MS
10	Waypoints	0.320	0.324	0.005	0.002**	RK4 MS

3.1.2. Hybrid method results

Table 2 presents the five-method comparison. Figure 3 visualizes the tracking error distributions across all ten experiments, illustrating that no single method dominates all scenarios. Key insight: NN-WS matches RK4 MS accuracy on all experiments (including held-out trajectories) while reducing solve time by 16%. AMS achieves the lowest error in noise and figure-eight scenarios, whereas NA-MPC shows substantial degradation on held-out trajectories (Exps. 8–10).

Table 2. Five-method comparison: mean tracking error (cm) over 10 seeds. Stars ($\star$) indicate the strict winner per experiment; boldface indicates the best result. NN-WS ties RK4 MS on all deterministic experiments

#	Experiment	RK4 MS	H-S	NN-WS	AMS	NA-MPC	Winner
1	Nominal	**0.567**	0.714	0.567	0.567	0.656	RK4=WS
2	Model -10%	**1.006**	1.163	1.006	1.006	1.617	RK4=WS
3	Model +10%	**0.227**	0.357	0.227	0.227	0.331	RK4=WS
4	Noise 0.5 cm	0.704	0.772	0.713	**0.703**$\star$	0.783	AMS
5	Noise 1 cm	0.937	0.927	0.922	**0.885**$\star$	0.913	AMS
6	Noise 2 cm	1.238	1.292	1.290	1.255	**1.195**$\star$	NA-MPC
7	Figure-eight	0.292	0.318	0.292	**0.289**$\star$	1.401	AMS
8	Lemniscate	0.462	**0.461**$\star$	0.462	0.466	2.064	H-S
9	S-Curve	**0.348**	0.477	0.348	0.390	2.059	RK4=WS
10	Waypoints	**0.320**	0.324	0.320	0.322	1.947	RK4=WS

No single method dominates all scenarios. Strict accuracy wins are distributed between AMS (3/10), NA-MPC (1/10), and H-S (1/10), while NN-WS ties RK4 MS on 5/10 experiments including held-out trajectories. NN warm-start preserves all IPOPT guarantees, as the NN modifies only the initial guess. It achieves identical accuracy to RK4 MS on all deterministic experiments (1–3, 7–10) including held-out trajectories while reducing average solve time from 142 ms to 121 ms (16% reduction) [13]. Adaptive method selector achieves 89.7% classification accuracy and outperforms the baseline in 3 of 10 scenarios, including the lowest error on figure-eight (0.289 cm). Its misclassification risk is bounded: for accuracy α and method gap $\delta_i = |e_{\mathrm{RK4}}^{(i)} - e_{\mathrm{H\text{-}S}}^{(i)}|$, the expected excess error satisfies $\mathbb{E}[e_{AMS}] \leq e^* + (1-\alpha)\delta_i$ [18]. With $\alpha = 0.897$ and $\delta_{\max} = 0.157$ cm, the worst-case excess is 0.016 cm negligible. Neural approximate MPC achieves a substantial latency reduction on training trajectories (142 ms → 0.03 ms, approximately ×4700) but performance diverges on held-out trajectories: tracking error increases to 1.95–2.06 cm (4.5–5.9× worse than RK4 MS), and safety fallback rates rise to 51–64%. The safety mechanism [14], [17] prevents catastrophic failure: even at 64.3% fallback (lemniscate), tracking error remains bounded at 2.06 cm.

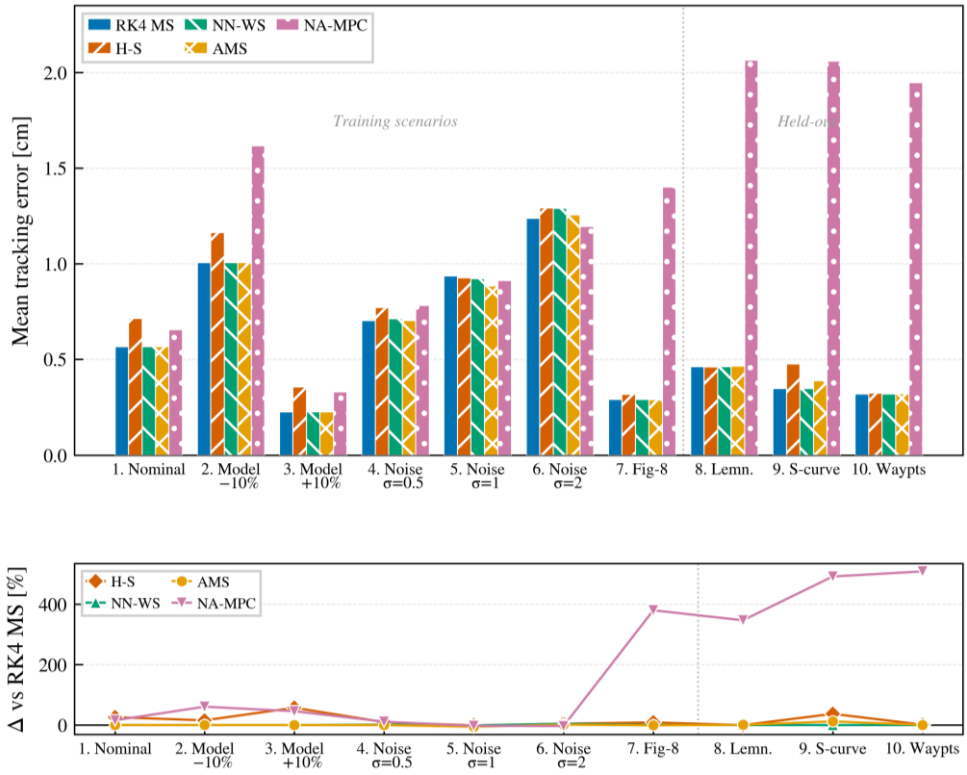

Figure 3. Five-method comparison across ten experiments

3.1.3. Generalization analysis

Table 3 presents the controlled evaluation of generalization to trajectories outside the training distribution. Three distinct generalization behaviors emerge, forming a hierarchy directly reflecting the degree of solver involvement:

- NN-WS generalizes without measurable degradation. Tracking error is identical to RK4 MS on all held-out trajectories. The solver-in-the-loop architecture ensures this: IPOPT converges to optimality regardless of initial guess quality [13].
- AMS generalizes well. Maximum degradation relative to RK4 MS is 12% (S-curve: 0.390 vs. 0.348 cm). The classifier correctly selects RK4 MS for most held-out conditions.
- NA-MPC shows statistically significant degradation ($p < 0.01$). Error increases from 0.66–1.40 cm (training) to 1.95–2.06 cm (held-out), with fallback rates rising from 0–16% to 51–64%. The safety mechanism correctly detects out-of-distribution behavior [14], [17].

Table 3. Generalization results: training vs. held-out trajectories. Mean tracking error (cm) and NA-MPC fallback rate (%)

Group	Trajectory	RK4 MS	NN-WS	AMS	NA-MPC	Fallback
Training	Circle	0.567	0.567	0.567	0.656	0.0%
Training	figure-eight	0.292	0.292	0.289	1.401	16.0%
Held-out	Lemniscate	0.462	0.462	0.466	2.064	64.3%
Held-out	S-Curve	0.348	0.348	0.390	2.059	50.8%
Held-out	Waypoints	0.320	0.320	0.322	1.947	62.7%

This hierarchy solver-in-the-loop $>$ solver-selection $>$ solver-replacement suggests a design principle: the degree of solver involvement in the control pipeline is a key factor influencing generalization capability for learning-augmented MPC, consistent with findings of Cauligi *et al.* [26] and Nubert *et al.* [23]. From a theoretical perspective, this hierarchy can be understood through the lens of approximation and generalization theory. Solver-in-the-loop methods (NN-WS) use the NN only to provide an initial guess; the solver then corrects any approximation error, effectively acting as a regularizer that bounds the output to the

feasible and optimal set. Solver-selection methods (AMS) reduce the problem to binary classification, which has lower sample complexity than full policy regression. Solver-replacement methods (NA-MPC) must approximate the entire input-to-control mapping, a higher-dimensional function that is inherently more sensitive to distribution shift [2], [15]. This analysis provides a principled basis for the observed hierarchy beyond the empirical evidence.

3.2. Discussion

Table 3 and Figure 4 provides an overview of the baseline and hybrid method results across all ten experimental scenarios, consolidating the findings from the preceding subsections. Heading cost impact: among the factors evaluated, heading cost formulation has the largest measured impact on complex trajectories. The figure-eight results (Table 1) confirm that the π-ambiguity in $\sin^2(\Delta\theta)$ causes severe practical failure on high-curvature trajectories, reducing tracking error by approximately 95% when corrected, as shown in Figure 4(a). This finding is applicable to other nonholonomic MPC formulations employing $\sin^2$-based heading costs [3], [5]. Practical guidelines: for general deployment, RK4 MS with NN warm-start offers the most favorable speed/accuracy trade-off, as summarized in Figure 4(b). AMS is preferred when operating conditions vary significantly and trajectory-dependent performance gaps emerge. NA-MPC suits compute-constrained platforms operating within the training distribution, though safety fallback to full IPOPT is mandatory, given the elevated fallback rates on held-out trajectories shown in Figure 4(c) [14], [16], [17]. The generalization hierarchy described in section 3.1.3, illustrated in Figure 4(d), provides a principled basis for this method selection.

Statistical rigor: all claims are supported by Wilcoxon signed-rank tests ($n = 10$ seeds). NN-WS shows no significant difference from RK4 MS in any experiment, confirming accuracy preservation. NA-MPC shows statistically significant differences ($p < 0.01$) in 8 of 10 experiments, confirming that its accuracy degradation is not a sampling artifact. Limitations and future work: the current evaluation is simulation-only; hardware deployment on a physical TurtleBot3 Burger with real-time ROS2 integration is planned as future work. NA-MPC accuracy degrades on held-out trajectories, requiring expanded training data or online adaptation for broader deployment. Future directions include online learning for adaptive NN refinement [2], data-driven dynamics via Koopman operators [25] or physics-informed NNs [29], integration with safety-constrained neural controllers [23], and comparison with acados [7] and distributed MPC architectures [30] for embedded real-time deployment. Key takeaway: NN-WS (solver-in-the-loop) consistently matches or improves upon RK4 MS, while NA-MPC (solver-replacement) shows marked degradation on held-out trajectories (Exps. 8–10)

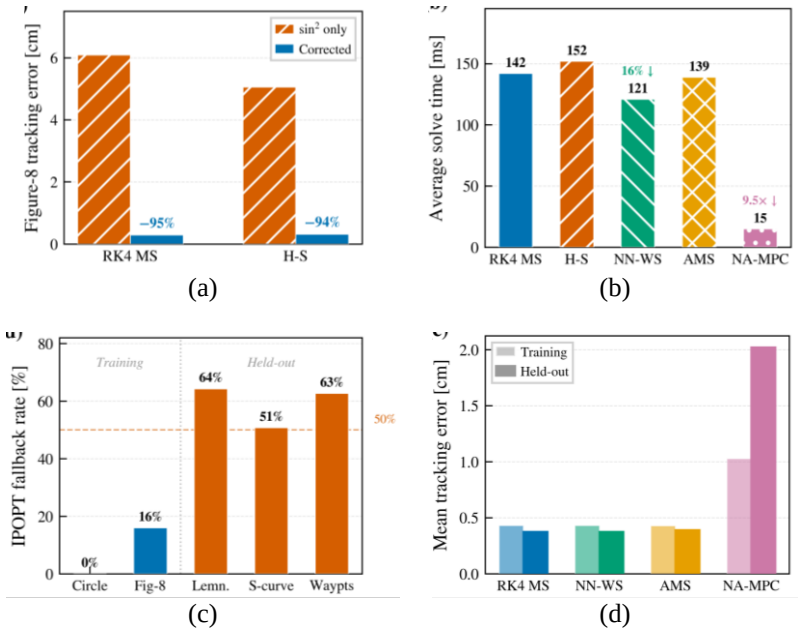

Figure 4. Summary of baseline and hybrid method results across all ten scenarios: (a) figure-eight tracking error before and after the corrected heading cost formulation, (b) average solve time across methods, (c) IPOPT fallback rate on training versus held-out trajectories, and (d) mean tracking error on training versus held-out trajectories by method

4. CONCLUSION

This paper presented a unified evaluation of NMPC transcription methods and three learning-augmented hybrid approaches for differential-drive trajectory tracking on a TurtleBot3 Burger. The main findings are: (i) the corrected heading cost (Theorem 1) eliminates a π-ambiguity in the standard $sin^2(\cdot)$ formulation, yielding approximately 95% error reduction on high-curvature trajectories; (ii) RK4 MS provides the most robust baseline across diverse conditions; (iii) NN warm-starting is the recommended augmentation, combining solver-guaranteed accuracy with 16% solve-time reduction; and (iv) the generalization hierarchy revealed by the results correlates directly with the degree of solver involvement. This hierarchy offers a practical design principle: when selecting learning-augmented MPC architectures, the degree of solver involvement should be matched to the expected variability of the deployment environment. These results point toward a broader opportunity for real-time NMPC deployment on resource-constrained embedded platforms, where combining classical optimization guarantees with NN efficiency can enable robust and computationally tractable control.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Taoufik Belkebir	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	
Hicham Belkebir	✓	✓				✓				✓		✓		
Anass Mansouri	✓					✓				✓		✓	✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The authors confirm that the data supporting the findings of this study are available within the article. The simulation code is available from the corresponding author upon reasonable request.

REFERENCES

[1] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, Jun. 2000, doi: 10.1016/S0005-1098(99)00214-9.

[2] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-Based Model Predictive Control: Toward Safe Learning in Control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 269–296, May 2020, doi: 10.1146/annurev-control-090419-075625.

[3] J. Tang *et al.*, "GMPC: Geometric Model Predictive Control for Wheeled Mobile Robot Trajectory Tracking," *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4822–4829, May 2024, doi: 10.1109/LRA.2024.3381088.

[4] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model predictive control: Theory, Computation, and Design*, 2nd ed. Santa Barbara, CA: Nob Hill Publishing, 2017.

[5] T. P. Nascimento, C. E. T. Dórea, and L. M. G. Gonçalves, "Nonlinear model predictive control for trajectory tracking of nonholonomic mobile robots: A modified approach," *International Journal of Advanced Robotic Systems*, vol. 15, no. 1, Jan. 2018, doi: 10.1177/1729881418760461.

[6] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot Operating System 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, May 2022, doi: 10.1126/scirobotics.abm6074.

[7] R. Verschueren *et al.*, "Acados—a Modular Open-Source Framework for Fast Embedded Optimal Control," *Mathematical Programming Computation*, vol. 14, no. 1, pp. 147–183, Mar. 2022, doi: 10.1007/s12532-021-00208-8.

[8] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation∗," *SIAM Review*, vol. 59, no. 4, pp.

849–904, Jan. 2017, doi: 10.1137/16M1062569.
[9] H. Belkebir and T. Belkebir, "Enhanced trajectory tracking for autonomous vehicles using a nonlinear model predictive controller," *Turkish Journal of Electromechanics & Energy*, vol. 9, no. 3, pp. 114–123, 2024.
[10] T. Belkebir, H. Belkebir, and A. Mansouri, "Real-Time Embedded NMPC for Autonomous Vehicle Path Tracking with Curvature-Aware Speed Adaptation and Sensitivity Analysis," *Automation*, vol. 7, no. 2, 2026, doi: 10.3390/automation7020044.
[11] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, Mar. 2016, doi: 10.1109/TIV.2016.2578706.
[12] L. Chen *et al.*, "Milestones in Autonomous Driving and Intelligent Vehicles: Survey of Surveys," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1046–1056, Feb. 2023, doi: 10.1109/TIV.2022.3223131.
[13] U. Rosolia and F. Borrelli, "Learning model predictive control for iterative tasks. A data-driven control framework," *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 1883–1896, Jul. 2018, doi: 10.1109/TAC.2017.2753460.
[14] M. Hertneck, J. Kohler, S. Trimpe, and F. Allgower, "Learning an Approximate Model Predictive Controller with Guarantees," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 543–548, Jul. 2018, doi: 10.1109/LCSYS.2018.2843682.
[15] B. Karg and S. Lucia, "Efficient Representation and Approximation of Model Predictive Control Laws via Deep Learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3866–3878, Sep. 2020, doi: 10.1109/TCYB.2020.2999556.
[16] J. Drgoňa, K. Kiš, A. Tuor, D. Vrabie, and M. Klaučo, "Differentiable predictive control: Deep learning alternative to explicit model predictive control for unknown nonlinear systems," *Journal of Process Control*, vol. 116, pp. 80–92, Aug. 2022, doi: 10.1016/j.jprocont.2022.06.001.
[17] K. P. Wabersich and M. N. Zeilinger, "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems," *Automatica*, vol. 129, p. 109597, Jul. 2021, doi: 10.1016/j.automatica.2021.109597.
[18] P. Kerschke and H. Trautmann, "Automated algorithm selection on continuous black-box problems by combining exploratory landscape analysis and machine learning," *Evolutionary Computation*, vol. 27, no. 1, pp. 99–127, 2018, doi: 10.1162/evco_a_00236.
[19] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-Time Neural MPC: Deep Learning Model Predictive Control for Quadrotors and Agile Robotic Platforms," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2397–2404, Apr. 2023, doi: 10.1109/LRA.2023.3246839.
[20] G. Torrente, E. Kaufmann, P. Fohn, and D. Scaramuzza, "Data-Driven MPC for Quadrotors," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3769–3776, Apr. 2021, doi: 10.1109/LRA.2021.3061307.
[21] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-Based Model Predictive Control for Autonomous Racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, Oct. 2019, doi: 10.1109/LRA.2019.2926677.
[22] M. Imad, O. Doukhi, D. J. Lee, J. chul Kim, and Y. J. Kim, "Deep Learning-Based NMPC for Local Motion Planning of Last-Mile Delivery Robot," *Sensors*, vol. 22, no. 21, p. 8101, Oct. 2022, doi: 10.3390/s22218101.
[23] J. Nubert, J. Köhler, V. Berenz, F. Allgöwer, and S. Trimpe, "Safe and Fast Tracking on a Robot Manipulator: Robust MPC and Neural Network Control," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3050–3057, Apr. 2020, doi: 10.1109/LRA.2020.2975727.
[24] E. Arcari *et al.*, "Bayesian Multi-Task Learning MPC for Robotic Mobile Manipulation," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3222–3229, Jun. 2023, doi: 10.1109/LRA.2023.3264758.
[25] M. Rosenfelder, L. Bold, H. Eschmann, P. Eberhard, K. Worthmann, and H. Ebel, "Data-driven predictive control of nonholonomic robots based on a bilinear Koopman realization: Data does not replace geometry," *Robotics and Autonomous Systems*, vol. 194, p. 105156, Dec. 2025, doi: 10.1016/j.robot.2025.105156.
[26] A. Cauligi, P. Culbertson, E. Schmerling, M. Schwager, B. Stellato, and M. Pavone, "CoCo: Online Mixed-Integer Control Via Supervised Learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1447–1454, Apr. 2022, doi: 10.1109/LRA.2021.3135931.
[27] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, Mar. 2019, doi: 10.1007/s12532-018-0139-4.
[28] R. Quirynen, S. Gros, B. Houska, and M. Diehl, "Lifted collocation integrators for direct optimal control in ACADO toolkit," *Mathematical Programming Computation*, vol. 9, no. 4, pp. 527–571, Dec. 2017, doi: 10.1007/s12532-017-0119-0.
[29] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, Feb. 2019, doi: 10.1016/j.jcp.2018.10.045.
[30] T. Schiz and H. Ebel, "Reducing the communication of distributed model predictive control: Autoencoders and formation control," *Control Engineering Practice*, vol. 165, p. 106560, Dec. 2025, doi: 10.1016/j.conengprac.2025.106560.

BIOGRAPHIES OF AUTHORS

Taoufik Belkebir received the Engineering degree in Electronics and Industrial Computing from the National School of Applied Sciences (ENSA), Oujda, Morocco, in 2012, and the Master's degree in Project Management in 2016. He is currently pursuing the Ph.D. degree with the LRSI Laboratory, Faculty of Sciences and Technology (FST), University Sidi Mohammed Ben Abdellah, Fez, Morocco. His research interests include optimal control, model predictive control, reinforcement learning, and autonomous mobile robotics. He can be contacted at email: belkebir.taoufik@usmba.ac.ma.

Hicham Belkebir received the advanced studies Diploma in Optical Information Processing in 2007 and the National Doctorate degree in Photonics from the Faculty of Sciences, Meknes, Morocco, in 2015. He is currently a Professor with the National School of Applied Sciences (ENSA), Fez. His research interests include optical modulators, planar photonic crystals, and optoelectronic systems. He can be contacted at email: hicham.belkebir@usmba.ac.ma.

Anass Mansouri received the Ph.D. degree in Microelectronics and Embedded Systems from the Faculty of Sciences and Technologies, Fez, Morocco, in 2009. He is currently a Professor with the National School of Applied Sciences (ENSA), Fez. His research interests include VLSI and embedded architecture design, video, image processing, and software/hardware design and optimization. He can be contacted at email: anass.mansouri@usmba.ac.ma.