Robust Path Tracking for Vehicles via Continuous-Time Residual Learning: An ICODE-MPPI Approach

Shugen Song[1], Wenjie Mei[2,†], and Chengyan Zhao[3]

***Abstract*—Model Predictive Path Integral (MPPI) control is a powerful sampling-based strategy for nonlinear autonomous systems. However, its performance is often bottlenecked by the fidelity of nominal dynamics. We propose ICODE-MPPI, a robust framework that leverages Input Concomitant Neural Ordinary Differential Equations (ICODEs) to learn and compensate for unmodeled residual dynamics. Unlike discrete-time learners, ICODEs maintain physical consistency and temporal continuity during the MPPI prediction horizon. Numerical simulations on complex trajectories demonstrate that ICODE-MPPI achieves up to a 69% reduction in cross-tracking error under persistent disturbances compared to standard MPPI control. Furthermore, our analysis confirms that ICODE-MPPI significantly suppresses control chattering, yielding smoother steering commands and improved robust performance.**

***Index Terms*—MPPI, ICODE, Residual Learning, Path Tracking, Control Smoothness.**

I. Introduction

Precise path tracking and proactive obstacle avoidance for autonomous ground vehicles (AGVs) are fundamental to intelligent transportation systems, particularly in unstructured or disturbed environments [1]. Model Predictive Control (MPC) has long been the paradigm for such tasks [2]; however, its gradient-based nature often struggles with non-convex cost landscapes. Recently, Model Predictive Path Integral (MPPI) control, a derivative-free and sampling-based variant, has gained prominence for its ability to handle complex constraints and nonlinear dynamics [3]. Its universality has been demonstrated in agile UAV maneuvers [4], USV path-following [5], and quadrotor pursuit [6].

Despite its robustness, the efficacy of MPPI is fundamentally sensitive to model-plant mismatch. Since the control law is derived from an expectation over sampled trajectories, inaccuracies in the forward dynamics model lead to biased rollouts, resulting in suboptimal control sequences, persistent tracking offsets, or high-frequency chattering [7], [8]. These challenges have necessitated a change toward data-driven control (DDC) to capture complex uncertainties more effectively [9].

Early data-driven extensions utilized multi-layer perceptrons (MLPs) [10] or Gaussian processes (GPs) [11]. However, MLPs often lack physical consistency in discrete-time transitions, while GPs suffer from prohibitive computational overhead in real-time optimization. Neural ordinary differential equations (Neural ODEs) [12] address these by parameterizing the state derivative, providing a continuous-time representation. For safety-critical control, architectures like ControlSynth ODEs (CSODEs) [13] and input concomitant neural ODEs (ICODEs) [14] further introduce convergence-guaranteed stability, demonstrating improved fidelity in modeling, for example, mechanical and electromechanical systems [15].

While many efforts have explored Transformers [16] or Reinforcement Learning [17] to enhance MPPI, integrating continuous-time, stability-guaranteed methods with its stochastic sampling architecture remains an open challenge. Inspired by residual learning [2] and robust sampling-based control [18], we propose the ICODE-MPPI framework. By employing ICODE as a continuous-time residual structure, our approach preserves the control-affine structure of the system while ensuring that high-frequency samples remain physically consistent and robust to disturbances.

The main contributions of this work are:

- We develop a residual dynamics framework based on ICODEs and iterative data aggregation. This hybrid strategy captures continuous-time uncertainties while maintaining high physical consistency.
- We integrate the trained ICODE model into the MPPI prediction horizon, enabling proactive compensation for environmental disturbances and significantly mitigating the steady-state tracking drift inherent in nominal models.
- We provide a comprehensive benchmarking of ICODE-MPPI against baselines, evaluating state-variable RMSE and control rate probability densities to demonstrate improved tracking precision and control smoothness.

The remainder of this paper is organized as follows. Section II presents the problem formulation and preliminaries. Section III details the proposed approach. Section IV discusses simulation and results, and Section V concludes the paper.

*This work is supported by the National Natural Science Foundation of China (NSFC) under grant 62403125, the Natural Science Foundation of Jiangsu Province under Grant BK20241283, and the Fundamental Research Funds for the Central Universities under grants 2242024k30037 and 2242024k30038.

[1]Shugen Song is School of Automation, Southeast University, Nanjing 210096, China 13955012259@163.com

[2]Wenjie Mei was with the School of Automation and the Key Laboratory of MCCSE of the Ministry of Education, Southeast University, Nanjing 210096, China, during the development of this work mei.wenjie@nju.edu.cn

[3]Chengyan Zhao is with Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology, 680-4 Kawazu, Iizuka-shi, Fukuoka, Japan zhao.chengyan961@mail.kyutech.jp

[†]Corresponding author.

II. Preliminaries and Problem Formulation

In this section, we formally present the stochastic optimal control problem and detail the vehicle kinematics and MPPI algorithm used for trajectory planning.

A. Problem Formulation

Consider a general discrete-time nonlinear dynamical system derived from a continuous-time process. The system evolution is governed by the following equation:

$$\mathbf{x}_{t+1} = \mathbf{F}(\mathbf{x}_t, \mathbf{v}_t) \tag{1}$$

where $\mathbf{x}_t \in \mathbb{R}^{n_x}$ denotes the system state vector, and $\mathbf{v}_t \in \mathbb{R}^{n_u}$ represents the noisy control input. The control input is modeled as the nominal control $\mathbf{u}_t$ perturbed by Gaussian noise:

$$\mathbf{v}_t = \mathbf{u}_t + \boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \Sigma) \tag{2}$$

where Σ is the covariance matrix representing the exploration variance.

The objective is to find an optimal control sequence $U^* = \{\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{H-1}\}$ that minimizes the expected cost over a finite horizon H. Formally, this optimization problem is formulated as follows:

$$\begin{aligned} \min_{U} \quad & J(U) = \mathbb{E}_{\mathbb{Q}}\left[\phi(\mathbf{x}_H) + \sum_{t=0}^{H-1}\left(s(\mathbf{x}_t) + \frac{1}{2}\mathbf{u}_t^\top R \mathbf{u}_t\right)\right] \\ \text{s.t.} \quad & \mathbf{x}_{t+1} = \mathbf{F}(\mathbf{x}_t, \mathbf{v}_t) \\ & \mathbf{v}_t = \mathbf{u}_t + \boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \Sigma) \\ & h(\mathbf{x}_t, \mathbf{u}_t) \leq 0 \\ & \mathbf{x}_0 = \mathbf{x}_{ini} \end{aligned} \tag{3}$$

where $\phi(\mathbf{x}_H)$ is the terminal cost, and the running cost is decomposed into a state-dependent cost $s(\mathbf{x}_t)$ and a quadratic control cost weighted by matrix R. The inequality $h(\mathbf{x}_t, \mathbf{u}_t) \leq 0$ represents physical constraints.

B. Vehicle Dynamics Modeling

For simplicity and to reduce computational complexity, this work adopts a bicycle model to represent the motion model of the vehicle [1].The state vector is defined as $\mathbf{x} = [x, y, \theta, v, \delta]^\top$, where (x, y) is the global position, θ is the heading angle, v is the longitudinal velocity, and δ is the steering angle. The control input vector is $\mathbf{u} = [a, \omega]^\top$, consisting of the longitudinal acceleration a and the steering rate ω.

The continuous-time dynamics are described by

$$\frac{d}{dt}\begin{pmatrix} x \\ y \\ \theta \\ v \\ \delta \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \frac{v}{l}\tan(\delta) \\ a \\ \omega \end{pmatrix} \tag{4}$$

where l is the wheelbase of the vehicle. This model captures the non-holonomic constraints of the vehicle without explicitly modeling the sideslip angle, suitable for trajectory planning at moderate speeds.

The vehicle motion model described by (4) can be rewritten in the general form of $\mathbf{x}_{t+1} = \mathbf{F}(\mathbf{x}_t, \mathbf{v}_t)$ by discretization, serving as the vehicle nominal model.

C. Model Predictive Path Integral Algorithm

MPPI solves the stochastic optimal control problem (3) by sampling trajectories, which has shown superior performance in handling complex nonlinear dynamics [3]. The algorithm consists of the following steps:

1) Trajectory Sampling: We sample K rollout trajectories by perturbing the nominal control sequence $\mathbf{u}_t$. The perturbed control input for the k-th sample at time step t is calculated by

$$\mathbf{v}_t^k = \mathbf{u}_t + \boldsymbol{\epsilon}_t^k, \quad \boldsymbol{\epsilon}_t^k \sim \mathcal{N}(0, \Sigma) \tag{5}$$

The system is forward simulated using the dynamics model to obtain state trajectories $\tau_k = \{\mathbf{x}_0^k, \ldots, \mathbf{x}_H^k\}$.

2) Cost Evaluation: For each trajectory τ_k, the total cost $S(\tau_k)$ is evaluated based on the running cost $q(\mathbf{x}_t, \mathbf{u}_t)$ and the terminal cost $\phi(\mathbf{x}_H)$:

$$S(\tau_k) = \phi(\mathbf{x}_H^k) + \sum_{t=0}^{H-1} q(\mathbf{x}_t^k, \mathbf{u}_t) \tag{6}$$

Note that the control cost is implicitly handled in the weighting step through the change of measurement method in MPPI control.

3) Weight Calculation: The importance weight w_k for the k-th trajectory is computed via the softmax function:

$$w_k = \frac{1}{\eta}\exp\left(-\frac{1}{\lambda}\left(S(\tau_k) - \rho\right)\right) \tag{7}$$

where λ is the temperature parameter, and $\rho = \min_k S(\tau_k)$ is a constant for numerical stability. The normalization factor η is given by:

$$\eta = \sum_{j=1}^{K}\exp\left(-\frac{1}{\lambda}\left(S(\tau_j) - \rho\right)\right) \tag{8}$$

4) Control Update: Finally, the resulting control sequence is updated by the probability-weighted average of the sampled noise:

$$\mathbf{u}_t^* = \mathbf{u}_t + \sum_{k=1}^{K} w_k \boldsymbol{\epsilon}_t^k \tag{9}$$

Furthermore, the obtained control sequence is then smoothed using a Savitzky-Golay filter. This formulation enables MPPI to approximate the optimal control by picking up low-cost trajectories while still maintaining exploration through the stochastic sampling process.

III. Proposed Approach

In this section, we present the proposed framework for robust trajectory tracking. We first detail the architecture of the Input Concomitant Neural Ordinary Differential Equations (ICODE) used to model the residual dynamics. Then, we describe the integration of ICODE with the MPPI controller, illustrated by a system flowchart.Finally, we discuss the data collection strategy and the training procedure.

A. ICODE Architecture for Residual Modeling

Standard Neural ODEs [12] typically model the derivative of the state as an autonomous function $\dot{\mathbf{x}}(t) = f_\theta(\mathbf{x}(t))$, treating external inputs as unknown parameters. However, for robotic systems, the state evolution is explicitly driven by control inputs. To accurately capture this interaction, we employ the ICODE architecture proposed in [14].

Unlike purely black-box models that concatenate state and control action into a single input vector, ICODE imposes a control-affine prior on the learned dynamics. This inductive bias ensures that the residual term aligns with the fundamental structure of robotic systems, where control inputs enter the state derivatives linearly, thereby improving sample efficiency.

We present the learned ICODE dynamics $\mathbf{f}_{res}(\mathbf{x}, \mathbf{u}; \theta)$ as:

$$\dot{\mathbf{x}}_{res}(t) = \mathbf{f}_{res}(\mathbf{x}(t), \mathbf{u}(t); \theta) = \mathbf{f}_\theta(\mathbf{x}(t)) + \sum_{j=1}^{m} \mathbf{g}_{\theta,j}(\mathbf{x}(t)) u_j(t) \tag{10}$$

where $\mathbf{u}(t) = [u_1, \ldots, u_m]^\top$ is the control input vector.

- $\mathbf{f}_\theta(\mathbf{x})\colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ represents the state-dependent autonomous drift (e.g., unmodeled friction, aerodynamic drag).
- $\mathbf{g}_{\theta,j}(\mathbf{x})\colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ represents the control-dependent gain fields, capturing how the j-th control input influences the state derivative (e.g., actuator effectiveness).

This affine decomposition, consistent with Eq. (2) in [14], provides a strong inductive bias. It ensures that the learned model respects the superposition principle of control inputs locally, significantly improving sample efficiency and generalization compared to fully connected networks. The total predicted dynamics used for MPPI rollouts combine the vehicle nominal model and the learned ICODE residual:

$$\dot{\mathbf{x}}_{pred}(t) = \mathbf{f}_{nom}(\mathbf{x}(t), \mathbf{u}(t)) + \mathbf{f}_{res}(\mathbf{x}(t), \mathbf{u}(t); \theta) \tag{11}$$

B. ICODE-MPPI Integration Algorithm

The proposed framework, as illustrated in Fig. 1,integrates the learned ICODE model into the MPPI prediction horizon to bridge the gap between idealized modeling and real-world disturbances. Specifically, the system dynamics are reformulated as a composite structure in (11): the nominal model provides the fundamental physical prior as defined in (4),while the ICODE module is explicitly trained to capture the complex, state-dependent residual dynamics that are otherwise neglected or difficult to model accurately.

At each control step, the MPPI controller leverages this augmented model within its forward prediction module. By simulating thousands of potential trajectories using the integrated Nominal-ICODE dynamics, the controller can anticipate environmental disturbances and unmodeled forces in real-time. Compared to a baseline controller relying solely on the nominal model, this integration allows the ICODE-MPPI scheme to compensate for residual errors proactively, ensuring robust path tracking in highly disturbed environments.

The ICODE architecture inherits the contraction-guaranteed property established in [14], ensuring convergence and numerical stability of the forward prediction. Nevertheless, a formal closed-loop stability proof for the overall ICODE-MPPI system remains an important direction for future work.

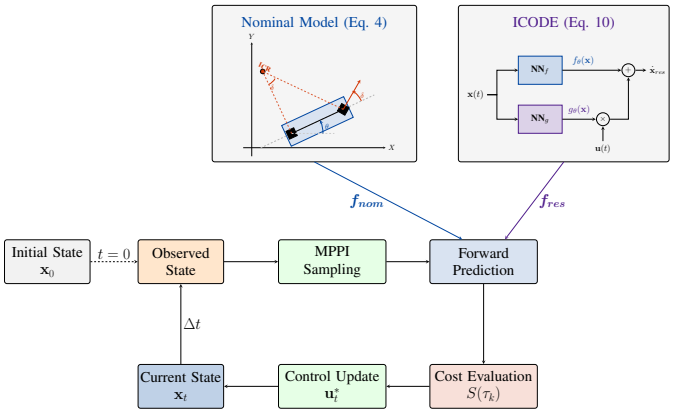


Fig. 1: Flowchart of the proposed ICODE-MPPI framework.

C. Data Collection and Iterative Training

To ensure the learned model accurately represents the vehicle's dynamics during high-performance maneuvers, we implement an iterative data aggregation framework [5]. This approach expands the dataset $\mathcal{D}$ by combining initial exploration with task-specific trajectories, addressing the distribution shift between random sampling and controlled path following.

1) Data Sources and Aggregation: The training dataset is constructed from two distinct data streams:

- **Random Exploration ($\mathcal{D}_{rand}$):** The vehicle is initially excited with random control inputs to capture the broad physical response and stability boundaries across the operational envelope.
- **Task-Specific Data ($\mathcal{D}_{task}$):** As training progresses, the MPPI controller utilizes the current model f_θ to execute the path-following task. The resulting state-action transitions are collected as on-policy data.

As illustrated in Fig. 2, these streams are aggregated such that $\mathcal{D} = \mathcal{D}_{rand} \cup \mathcal{D}_{task}$. This hybrid dataset ensures the model achieves high fidelity both in general dynamics and in the high-probability state regions near the reference trajectory.

2) Training Procedure: The neural network is optimized iteratively. In each cycle, a mini-batch $\mathcal{B}$ is sampled from the aggregated buffer $\mathcal{D}$ to minimize the multi-step prediction error. By integrating the nominal dynamics $\mathbf{f}_{nom}$ and the residual network $\mathbf{f}_{res}$ via an RK4 solver, the loss function is defined as:

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \| \mathbf{x}_{t+1}^{(i)} - \Phi_{RK4}(\mathbf{x}_t^{(i)}, \mathbf{u}_t^{(i)}; \mathbf{f}_{nom}, \mathbf{f}_{res}) \|^2 \tag{12}$$

The updated model f_θ is then redeployed into the MPPI planner for the next round of data collection, forming a self-correcting feedback loop that continuously reduces modeling uncertainty.

The dataset was aggregated over a single iteration cycle. The data was normalized with a mean of zero and a variance of one, and split into an 80% training set and a 20% test set.

Training was conducted using the Adam optimizer for up to 500 epochs, with early stopping applied based on the test set loss (with a patience threshold of 30 epochs).

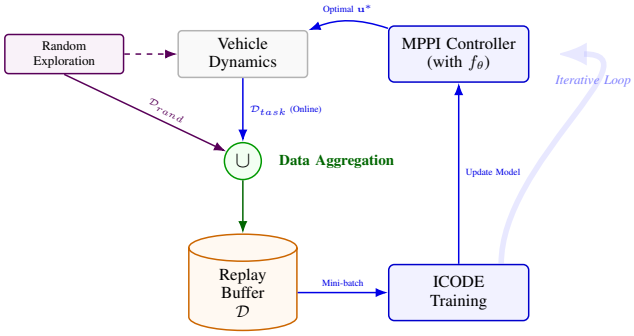


Fig. 2: The iterative learning framework with data aggregation.

TABLE I: System Configurations and Hyper-parameters

Category	Parameter	Value
Vehicle Kinematics	Wheelbase (L)	2.5 m
	Max Acceleration (a_{max})	2.0 m/s^2
	Max Steering Angle (δ_{max})	1.571 rad
	Sampling Time (Δt)	0.05 s
MPPI Controller	Prediction Horizon (H)	20 steps
	Number of Samples (K)	3000
	Temperature Parameter (λ)	0.05
	Control Noise Covariance (Σ)	diag(0.1, 0.5)
ICODE Network	Hidden dimension	[256, 256]
	Number of Layers	3
	Activation Function	Softplus
	Learning Rate	5×10^{-4}
	Optimizer	Adam

IV. Simulations and Results

To evaluate the efficacy and robustness of the proposed ICODE-MPPI framework, we conduct numerical simulation experiments on a trajectory tracking task under unmodeled environmental disturbances. We conduct simulations on three diverse reference paths: **Ellipse**, **Sine-wave**, and **Figure-8**. These trajectories present increasing levels of difficulty in terms of curvature change and control agility. This section details the simulation environment, the disturbance modeling, and the performance metrics used for comparison.

A. Experimental Setup

1) Dynamic Disturbance Environment: The environment is characterized by persistent external disturbances $\boldsymbol{\delta}(t)$. Unlike the ideal nominal model, the actual state evolution follows $\dot{\mathbf{x}} = \mathbf{f}_{nom} + \boldsymbol{\delta}(t)$. As implemented in our simulation, the disturbance is a composite sinusoidal signal:

$$\boldsymbol{\delta}(t) = [A_x \sin(\omega_x t), A_y \cos(\omega_y t), A_\theta \sin(\omega_\theta t), 0, 0]^\top \quad (13)$$

This setup manifests as time-varying drifting in the X, Y coordinates and the heading angle θ, creating a significant model mismatch for any controller relying solely on the nominal model.

2) Hyper-parameters: The key parameters extracted from our implementation are summarized in Table I. These ensure a fair comparison between the baseline MPPI and our ICODE-MPPI. Integrating the ICODE residual into MPPI rollouts increases the forward-prediction time due to the additional neural-network evaluation. Despite this acceptable computational overhead, ICODE-MPPI achieves substantially higher tracking accuracy, as demonstrated in Table II.

B. Trajectory Tracking and Qualitative Analysis

The spatial tracking performance across the three distinct trajectories is qualitatively evaluated in Fig. 3.

As shown in Fig. 3, the unmodeled time-varying disturbance $\boldsymbol{\delta}(t)$ induces a persistent steady-state offset in the nominal MPPI across all paths. However, the severity varies: in the Ellipse, the drift is relatively uniform, pushing the vehicle outward. In contrast, the Figure-8 and Sine-wave paths feature continuous curvature inversions. At these inflection points, the nominal controller struggles to quickly reverse its steering bias, resulting in severe overshooting.

ICODE-MPPI effectively suppresses this drift. By utilizing the ICODE to continuously approximate the residual dynamics $\boldsymbol{\delta}(t)$, the controller achieves a pre-steering effect. It anticipates the lateral push of the disturbance and compensates proactively, maintaining tight adherence to the reference even during sharp, transient maneuvers.

C. Quantitative Analysis and Statistical Robustness

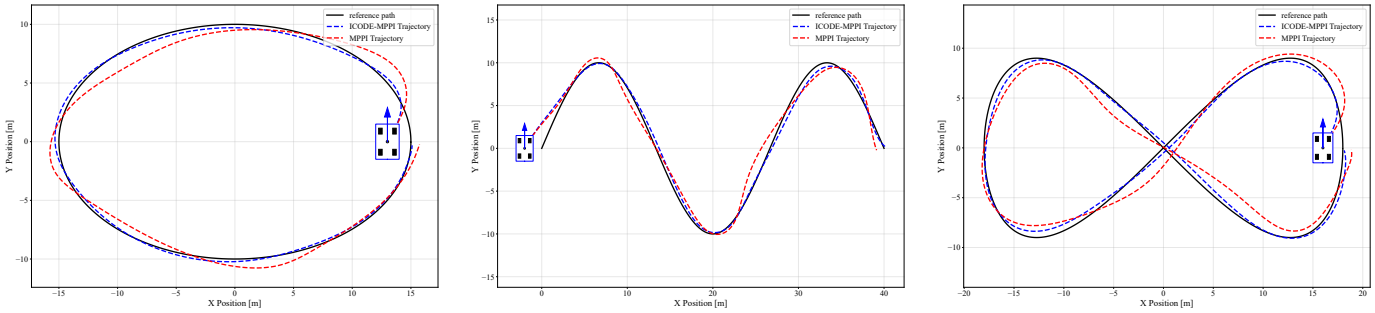
Table II quantifies the tracking precision using Root Mean Square Error (RMSE).

TABLE II: Tracking RMSE Comparison

Trajectory	Method	X Error (m)	Y Error (m)	Yaw Error (rad)
Ellipse	MPPI	0.589	0.557	0.231
	ICODE-MPPI	**0.448**	**0.171**	**0.244**
Sine-wave	MPPI	0.681	0.314	0.214
	ICODE-MPPI	**0.587**	**0.213**	**0.266**
Figure-8	MPPI	0.655	0.589	0.259
	ICODE-MPPI	**0.442**	**0.216**	**0.267**

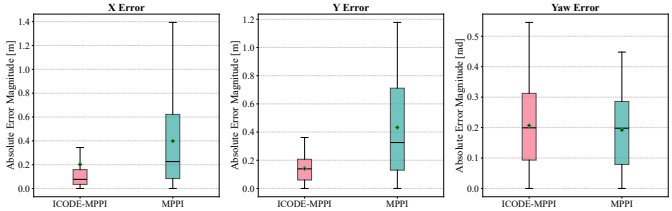
The data reveals that ICODE-MPPI achieves a significant reduction in positional errors, with the Y-direction error dropping by up to 69% in the Ellipse case. However, an interesting observation is that the yaw RMSE for ICODE-MPPI is slightly higher than the nominal model. This indicates a control trade-off: to maintain strict positional adherence (X, Y) against strong lateral disturbances, the controller performs more aggressive heading adjustments. In path-tracking tasks, prioritizing spatial safety over perfect heading alignment is a deliberate and necessary strategy for robust navigation.

To further analyze the error distribution, Fig. 4 presents the absolute error boxplots. In these plots, the **center line** denotes the median, the **green point** represents the mean, and the **whiskers** indicate the range of non-outlier maximum/minimum errors.

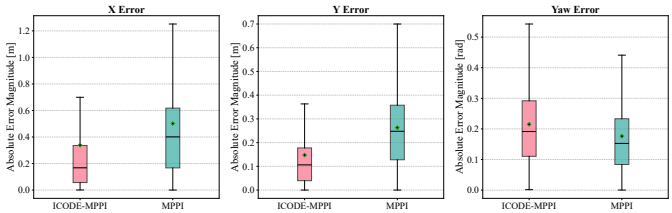


(a) Ellipse Trajectory (b) Sine-wave Trajectory (c) Figure-8 Trajectory

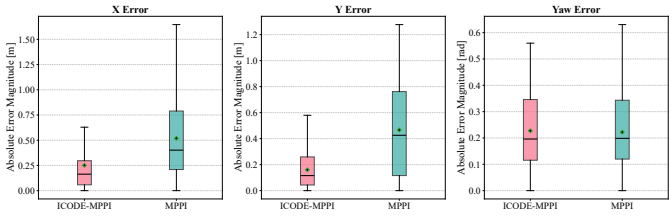
Fig. 3: Trajectory tracking results. The nominal MPPI shows significant steady-state drift under disturbances, particularly at the curvature inflection points, while ICODE-MPPI closely adheres to the reference.



(a) Error Distribution: Ellipse



(b) Error Distribution: Sine-wave



(c) Error Distribution: Figure-8

Fig. 4: Boxplots of state-variable tracking errors (X, Y, θ) across different trajectories.

The boxplots reveal stark differences in how the errors are distributed. Across all three trajectories, the X and Y positional boxes for ICODE-MPPI are drastically shorter and lower, proving its high resistance to lateral and longitudinal drift. However, the Yaw behavior is more nuanced. Notably, in the Ellipse and Sine-wave trajectory, the Yaw whiskers for ICODE-MPPI are longer than those of the nominal MPPI. This aligns with our RMSE findings: to maintain strict X and Y coordinates against periodic crosswinds, the ICODE framework occasionally permits marginally larger transient yaw deviations at the inflection points. It sacrifices a small degree of transient heading stability to guarantee absolute positional safety.

D. Control Smoothness and Stability

For real-world deployment, the control effort must be feasible and smooth. Fig. 5 illustrates the probability density of the control jerk (rate of change for acceleration a and steering δ). The key metrics are the peak value at zero (representing steady-state periods) and the distribution width (representing chattering intensity).Higher peaks indicate enhanced stability, while narrower distributions signify reduced chattering.

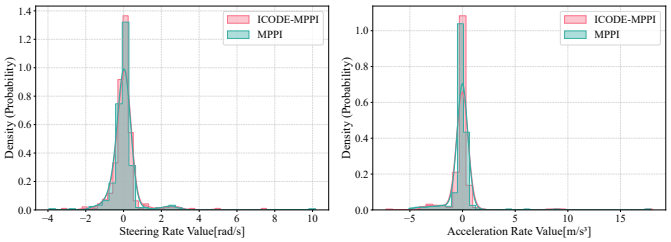
In all scenarios, the nominal MPPI exhibits a flat, wide distribution for steering rate, indicating severe control chattering.It constantly reacts to the disturbance with sudden, large-magnitude steering inputs.The higher peak at zero and the narrower distribution of ICODE-MPPI effectively demonstrate its enhanced stability and reduced chattering compared to the nominal MPPI.

Under peak disturbances, nominal MPPI frequently commands steering rates close to the saturation bound ($\delta_{\max} = 1.571$ rad) in a reactive manner. In contrast, ICODE-MPPI generates smoother, anticipatory corrections that remain farther from the actuator limits, preserving a larger safety margin.

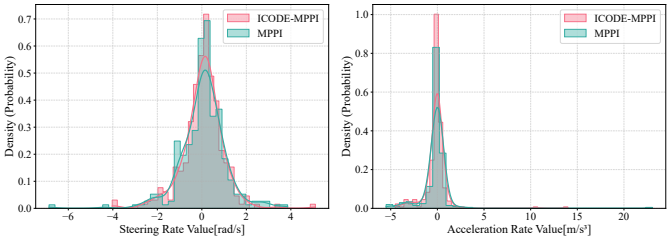
However, in certain trajectories (e.g., Ellipse), the steering rate is vastly improved while the acceleration distribution becomes slightly wider. This stems from lateral-longitudinal coupling. By maintaining aggressive and precise steering corrections to perfectly track the complex curves under heavy disturbances, the vehicle inevitably experiences higher lateral tire forces, causing slight losses in forward momentum. Consequently, the longitudinal acceleration controller must actuate more frequently to maintain reference speed. Trading modest longitudinal adjustments for substantial lateral safety and steering smoothness remains highly desirable.

V. Conclusion

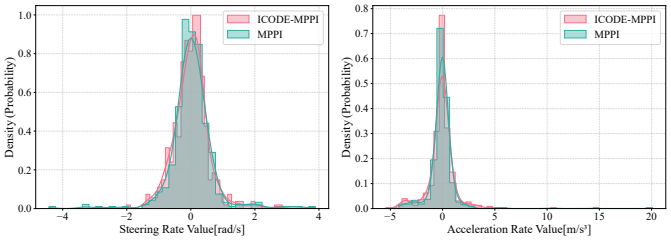
This paper presented ICODE-MPPI, a robust path-tracking framework that addresses the model mismatch challenge by embedding Input Concomitant Neural ODEs as residual learners. By explicitly modeling the residual terms of the system's



(a) Control Profile: Ellipse



(b) Control Profile: Sine-wave



(c) Control Profile: Figure-8

Fig. 5: Probability density of control input rates. ICODE-MPPI drastically smooths the steering commands, with acceptable trade-offs in longitudinal acceleration distribution.

state-space equations, our method achieves improved accuracy in tracking positional states (X, Y) under high-disturbance conditions. The simulation results across multiple complex trajectories confirm that ICODE-MPPI significantly reduces state-variable drift, while maintaining a strategic trade-off with transient heading alignment. Moreover, the analysis of control effort highlights that while steering chattering is drastically suppressed, the longitudinal-lateral coupling requires slightly more active acceleration adjustments to maintain reference velocities.

Future work will focus on: (i) validation under unseen stochastic disturbances, parametric variations, and actuator delays; (ii) theoretical closed-loop stability analysis leveraging the ICODE contraction property; (iii) empirical comparison against, for example, residual MPC, discrete-time MLP, Neural ODE and its variants baselines; (iv) dynamic obstacle avoidance and real-world deployment.

REFERENCES

[1] M. Testouri, G. Elghazaly, and R. Frank, "Towards a safe real-time motion planning framework for autonomous driving systems: An MPPI approach," *arXiv preprint arXiv:2308.01654*, 2023.

[2] H. Zhang, J. Ge, J. Su, K. Gu, F. Wang, W.-H. Chen, and S. Li, "Model predictive control with residual learning and real-time disturbance rejection: Design and experimentation," *Control Engineering Practice*, vol. 165, p. 106587, 2025.

[3] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1433–1440.

[4] M. Minařík, R. Pěnička, V. Vonásek, and M. Saska, "Model predictive path integral control for agile unmanned aerial vehicles," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 13 144–13 151.

[5] E. Liu, D. Wang, Z. Peng, L. Liu, and N. Gu, "Data-driven path following of unmanned surface vehicles based on model-based reinforcement learning and model predictive path integral control," in *2022 37th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. IEEE, 2022, pp. 1045–1049.

[6] E.-T. Jeong, S.-D. Lee, K.-M. Na, and C.-H. Lee, "Mppi control-based adaptive pursuit guidance for path-following control of quadrotors in the presence of wind disturbances," in *International Conference on Robot Intelligence Technology and Applications*. Springer, 2022, pp. 37–48.

[7] S. Ross and J. A. Bagnell, "Agnostic system identification for model-based reinforcement learning," in *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

[8] T. Kim, G. Park, K. Kwak, J. Bae, and W. Lee, "Smooth model predictive path integral control without smoothing," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 406–10 413, 2022.

[9] Z.-S. Hou and Z. Wang, "From model-based control to data-driven control: Survey, classification and perspective," *Information Sciences*, vol. 235, pp. 3–35, 2013.

[10] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7559–7566.

[11] 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, 2019.

[12] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," *Advances in neural information processing systems*, vol. 31, 2018.

[13] W. Mei, D. Zheng, and S. Li, "Controlsynth neural ODEs: Modeling dynamical systems with guaranteed convergence," *Advances in Neural Information Processing Systems*, vol. 37, pp. 99 232–99 261, 2024.

[14] Z. Li, W. Mei, K. Yu, Y. Bai, and S. Li, "ICODE: Modeling dynamical systems with extrinsic input information," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 14 358–14 370, 2025.

[15] W. Mei, X. Wang, Y. Lu, K. Yu, and S. Li, "Learning and current prediction of PMSM drive via differential neural networks," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 72, no. 3, pp. 489–493, 2025.

[16] S. Zinage, V. Zinage, and E. Bakolas, "Transformer-based model predictive path integral control," *arXiv preprint arXiv:2412.17118*, 2024.

[17] Y. Qu, H. Chu, S. Gao, J. Guan, H. Yan, L. Xiao, S. E. Li, and J. Duan, "RL-driven MPPI: Accelerating online control laws calculation with offline policy," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 2, pp. 3605–3616, 2023.

[18] M. S. Gandhi, B. Vlahov, J. Gibson, G. Williams, and E. A. Theodorou, "Robust model predictive path integral control: Analysis and performance guarantees," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1423–1430, 2021.