

Kinematics-Informed Data-Enabled Predictive Control for Mobile Robots

Binlin Zhang¹ and Erkan Kayacan¹

Abstract—Data-enabled predictive control (DeePC) enables model-free optimal control using input-output data; however, its performance degrades in nonlinear systems, particularly for trajectory tracking. In this paper, we propose a kinematics-informed DeePC (KiDeePC) framework for mobile robots that integrates known kinematic models with data-driven representations of unknown dynamics. The proposed approach enforces behavioral constraints only on the dynamic variables while propagating kinematic states analytically. This reduces the reliance on the data-driven component and improves feasibility. Simulation results on an unmanned ground vehicle demonstrate improved tracking accuracy, faster convergence, and smoother control inputs compared to conventional DeePC.

I. INTRODUCTION

Model predictive control (MPC) is a well-established, model-based approach for solving constrained finite-horizon optimal control problems [1]. Its ability to handle state, input, and output constraints while optimizing over a prediction horizon has made it widely used in robotics and autonomous systems. However, the performance of MPC heavily relies on the accuracy of the system model [2]. In practice, many systems exhibit nonlinear and complex behavior. As autonomous systems operate in uncertain and unstructured environments, deriving their dynamic model from first-principles or identifying their system parameters are challenging and labor-intensive [3].

To address this limitation, data-driven approaches have gained increasing attention in recent years [2]. These methods aim to learn system behavior directly from data without explicit modeling or system identification [4]. Notably, machine learning-based approaches have shown strong potential in handling complex and nonlinear systems. However, they often lack guarantees for enforcing safety constraints [5], and typically rely on extensive offline training [6], which can limit their applicability in autonomous systems.

As an alternative, data-enabled predictive control (DeePC) has been proposed. This method leverages behavioral systems theory [7] to represent system trajectories directly from input-output data while satisfying safety constraints. DeePC has been shown to be equivalent to model predictive control (MPC) for deterministic linear time-invariant (LTI) systems [8]. Despite its success for LTI systems, extending DeePC to nonlinear systems remains challenging. In particular, the linear representation in the Hankel matrix may become inaccurate when the system nonlinearities are significant. Several approaches have been proposed to enhance the robustness of

DeePC, including the incorporation of regularization terms and slack variables [9]–[12]. However, these methods focus primarily on point-to-point tracking tasks, rather than following a time-varying trajectory.

In contrast, trajectory tracking is a fundamental requirement in mobile robotics. In [10], nonlinear systems are considered as linear time-varying systems; therefore, the Hankel matrix is updated along the trajectory to enable tracking of time-varying references. Nevertheless, such approaches may suffer from infeasibility due to strict equality constraints, especially for high-dimensional systems.

A key observation in autonomous systems is that their kinematic models are readily available, as they depend only on geometric relationships. On the other hand, dynamic models are significantly more complex and vary across platforms due to actuation, mass distribution, and environmental interactions. This motivates the development of a kinematics-informed data-enabled predictive control (KiDeePC) framework. Rather than treating the system as fully known or unknown, the proposed approach incorporates available kinematic information into the DeePC algorithm, while learning the system dynamics directly from input-output data.

In this paper, we propose the KiDeePC framework for trajectory tracking of mobile robots. The main idea is to decompose the system behavior into kinematic and dynamic components, where the dynamics are captured from input-output data, while the kinematics are propagated using known analytical models. This structure improves tracking performance and alleviates feasibility issues by reducing the burden on the data-driven component.

The remainder of this paper is organized as follows. Section II presents the proposed KiDeePC method. Section III provides simulation results and comparisons. Section IV concludes the paper.

II. METHODOLOGY

A. Modeling of Mobile Robots

In general, the motion of a mobile robot can be described by a nonlinear control system of the form

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k),$$

where $\mathbf{x}_k \in \mathbb{R}^n$ and $\mathbf{u}_k \in \mathbb{R}^p$ denote the system state and control input at time step k , respectively, and f is a continuous function.

For mobile robots, the system behavior can be naturally decomposed into kinematic and dynamic components. Accordingly, we define the state vector as

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{\text{kin}} \\ \mathbf{x}_{\text{dyn}} \end{bmatrix},$$

¹School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore, Singapore binlin.zhang@ntu.edu.sg, erkan.kayacan@ntu.edu.sg

where the kinematic state \mathbf{x}_{kin} represents configuration variables of the system (e.g., position and orientation), while the dynamic state \mathbf{x}_{dyn} consists of velocity-related variables.

The kinematic model is generally consistent across the same class of mobile robots and can be readily derived from geometric information, which is given by

$$\mathbf{x}_{\text{kin},k+1} = f_{\text{kin}}(\mathbf{x}_{\text{kin},k}, \mathbf{x}_{\text{dyn},k}). \quad (1)$$

In contrast, the dynamics typically varies from one robot to another, and can be expressed as:

$$\mathbf{x}_{\text{dyn},k+1} = f_{\text{dyn}}(\mathbf{x}_{\text{dyn},k}, \mathbf{u}_k). \quad (2)$$

B. Kinematics-Informed DeePC Algorithm

Given an initial data length $T_{\text{ini}} \in \mathbb{Z}_{>0}$ and a prediction horizon $N \in \mathbb{Z}_{>0}$, the total window length is defined as $L = T_{\text{ini}} + N$. Let T denote the total number of collected data samples used to construct the Hankel matrices.

For a given input sequence $\{\mathbf{u}_k\}_{k=0}^{T-1}$, the Hankel matrix of windows L is defined as

$$\mathbf{H}_L(\mathbf{u}) = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \cdots & \mathbf{u}_{T-L} \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_{T-L+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{u}_{L-1} & \mathbf{u}_L & \cdots & \mathbf{u}_{T-1} \end{bmatrix}.$$

Similarly, $\mathbf{H}_L(\mathbf{x}_{\text{dyn}})$ is defined in the same way using the state sequence $\{\mathbf{x}_{\text{dyn},k}\}_{k=0}^{T-1}$.

With data-driven system representations and kinematic propagation, the trajectory tracking problem is formulated as the following finite-horizon optimization problem:

$$\min_{\substack{\alpha(t), \sigma(t) \\ \bar{\mathbf{u}}(t), \bar{\mathbf{x}}(t)}} \sum_{k=1}^N \|\bar{\mathbf{x}}_{\text{kin},k}(t) - \mathbf{x}_{\text{kin},k}^{\text{ref}}(t)\|_{\mathbf{Q}}^2 + \sum_{k=0}^{N-1} \|\bar{\mathbf{u}}_k(t)\|_{\mathbf{R}}^2 \quad (3a)$$

$$+ \lambda_{\alpha} \|\alpha(t)\|_2^2 + \lambda_{\sigma} \|\sigma(t)\|_2^2$$

$$\text{s.t. } \mathbf{x}_{\text{dyn},[-T,-1]}(t) = \hat{\mathbf{x}}_{\text{dyn},[-T,-1]}(t), \quad \mathbf{x}_0(t) = \hat{\mathbf{x}}_0(t), \quad (3b)$$

$$\begin{bmatrix} \bar{\mathbf{u}}(t) \\ \bar{\mathbf{x}}_{\text{dyn}}(t) + \sigma(t) \end{bmatrix} = \begin{bmatrix} \mathbf{H}_L(\mathbf{u}_{[t-T,t-1]}) \\ \mathbf{H}_L(\mathbf{x}_{\text{dyn},[t-T,t-1]}) \end{bmatrix} \alpha(t), \quad (3c)$$

$$\mathbf{x}_{\text{kin},k+1}(t) = f_{\text{kin}}(\mathbf{x}_{\text{kin},k}(t), \mathbf{x}_{\text{dyn},k}(t)), \quad (3d)$$

$$\begin{bmatrix} \bar{\mathbf{u}}_{[-T_{\text{ini}},-1]}(t) \\ \bar{\mathbf{x}}_{\text{dyn},[-T_{\text{ini}},-1]}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{u}_{[t-T_{\text{ini}},t-1]} \\ \mathbf{x}_{\text{dyn},[t-T_{\text{ini}},t-1]} \end{bmatrix}, \quad (3e)$$

$$\sum_{i=0}^{T-L} \alpha_i(t) = 1, \quad (3f)$$

$$\mathbf{u}_k(t) \in [\mathbf{u}_{\min}, \mathbf{u}_{\max}], \quad \mathbf{x}_k(t) \in [\mathbf{x}_{\min}, \mathbf{x}_{\max}], \quad (3g)$$

$$k = 0, 1, \dots, N-1, \quad (3h)$$

where $\bar{\mathbf{x}}$ and $\bar{\mathbf{u}}$ denote predicted states and control inputs. α is the decision variable that parameterizes the behavioral representation of the system, and $\lambda_{\alpha} > 0$ denotes its associated regularization weight. The slack variable σ is introduced to enhance robustness against measurement noise and model mismatch, penalized by $\lambda_{\sigma} > 0$. The matrices

$\mathbf{Q}, \mathbf{R} \succ 0$ are positive definite weighting matrices that penalize the tracking error and control effort, respectively. When the system states are not fully observable, constraint (3b) incorporates an observer to estimate the states. Based on the estimated states, the DeePC constraint (3c) obtains optimal control inputs, while the kinematic constraint (3d) propagates the kinematic states \mathbf{x}_{kin} using the dynamic states \mathbf{x}_{dyn} . Moreover, constraint (3e) enforces consistency between the predicted trajectories over the past horizon and the measured data. Furthermore, constraint (3f) introduces an affine consistency condition on α to eliminate scaling ambiguity caused by linearization. In addition, both the inputs and states are required to satisfy the constraints in (3g). Finally, the optimization problem in (3) is implemented in a standard receding horizon manner, as outlined in Algorithm 1.

Algorithm 1 Kinematics-Informed DeePC

Offline: Select the past data length T_{ini} (no smaller than the upper bound on the system order), prediction horizon N , weighting matrices $\mathbf{Q}, \mathbf{R} \succ 0$, regularization parameters $\lambda_{\alpha}, \lambda_{\sigma} > 0$, and input/state constraints $\mathbf{u}_{\min}, \mathbf{u}_{\max}, \mathbf{x}_{\min}, \mathbf{x}_{\max}$. Collect a persistently exciting dataset $\{\mathbf{u}_k, \mathbf{x}_{\text{dyn},k}\}_{k=0}^{T-1}$ and construct the initial Hankel matrices.

Online:

1. Acquire the most recent T input-state measurements $\{\mathbf{u}_k, \mathbf{x}_{\text{dyn},k}\}_{k=t-T}^{t-1}$ using a sliding window and update the Hankel matrices.
 2. Solve the optimization problem (3) to obtain the optimal control sequence $\{\mathbf{u}_k^*\}_{k=0}^{N-1}$.
 3. Apply the first control input \mathbf{u}_0^* .
-

III. SIMULATION

In this section, we apply both DeePC and KiDeePC schemes to the discrete-time nonlinear system obtained via Euler discretization with sampling time $T_s = 0.05s$. We consider an unmanned ground vehicle (UGV) system. The state vector is defined as follows:

$$\mathbf{x}_{\text{kin}} = [p_x \quad p_y \quad \theta]^T, \\ \mathbf{x}_{\text{dyn}} = [v \quad \omega]^T,$$

where p_x and p_y denote the planar position of the vehicle, θ is the heading angle, and v and ω represent the linear and angular velocities, respectively. The kinematic model for UGVs is usually described by

$$p_{x,k+1} = p_{x,k} + T_s v_k \cos \theta_k, \\ p_{y,k+1} = p_{y,k} + T_s v_k \sin \theta_k, \\ \theta_{k+1} = \theta_k + T_s \omega_k.$$

The control input is defined as $\mathbf{u} = [u_v, u_{\omega}]^T$, where u_v and u_{ω} correspond to control inputs related to throttle and steering, respectively. The dynamic models of UGVs can vary significantly across platforms; therefore, we assume the

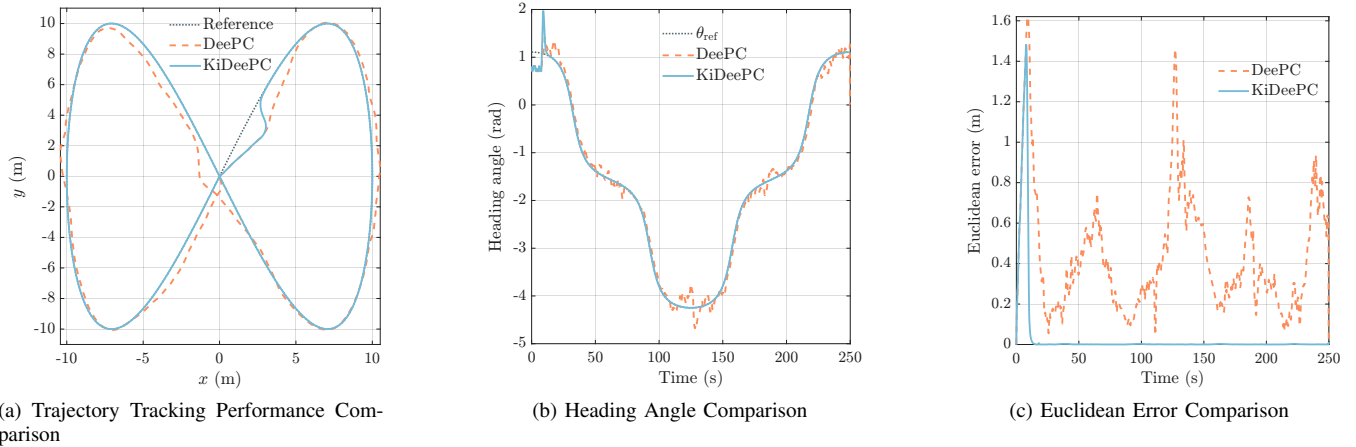


Fig. 1. Tracking Performance Comparison

following model:

$$v_{k+1} = v_k + T_s \frac{u_{1,k} - v_k}{\tau_v},$$

$$\omega_{k+1} = \omega_k + T_s \frac{u_{2,k} - \omega_k}{\tau_\omega},$$

where $\tau_v = 0.3$, $\tau_\omega = 0.2$.

We implement both the conventional DeePC algorithm in [10] and the proposed KiDeePC, as described in Algorithm 1, in MATLAB using the nonlinear optimization framework CasADi [13]. In the following simulation study, all states are assumed to be fully measurable and noise-free. The control objective is to track an eight-figure shape reference trajectory while satisfying the input and state constraints: $u_v, u_\omega \in [-1, 1]$, $v \in [0, 1]$, and $\omega \in [-1, 1]$. The parameters are chosen as $T_{\text{ini}} = 5$, $N = 30$, $L = T_{\text{ini}} + N = 35$, $T = 150$, and the total number of simulation time steps is $Num = 5000$. The weighting matrices and regularization parameters are set to $\mathbf{Q} = \text{diag}(50, 50, 10)$, $\mathbf{R} = \mathbf{I}$, $\lambda_\alpha = 10^{-6}$, and $\lambda_\sigma = 10^6$ for both methods. A multi-sine input signal is used for the offline data collection to ensure that the initial Hankel matrices are persistently exciting of order L for both methods. The tracking performance is evaluated using the Euclidean distance error.

Fig. 1 compares the trajectory tracking performance of the proposed KiDeePC and the conventional DeePC. It can be observed that the proposed method achieves more accurate and responsive tracking of the reference trajectory, with the Euclidean error quickly decreasing and remaining close to zero throughout the motion. In contrast, the conventional DeePC exhibits slower response and larger tracking errors in the considered high-dimensional nonlinear setting. Moreover, the tracking error continues to oscillate along the trajectory, indicating degraded tracking accuracy.

Fig. 2 presents the corresponding control inputs. The control signals generated by the conventional DeePC exhibit significant chattering, which is detrimental to real-time robotic systems. By contrast, KiDeePC produces smoother control inputs, thereby enhancing its practical applicability in real-time control.

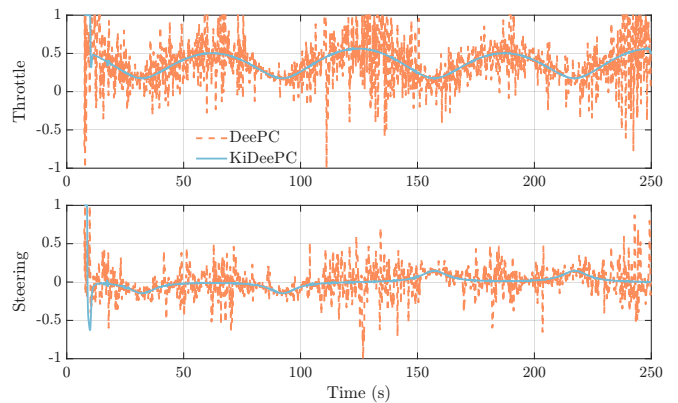


Fig. 2. Control Signal Comparison

In terms of quantitative performance, DeePC achieves a mean Euclidean error of 0.435 m, whereas KiDeePC attains a significantly lower error of 0.036 m. This demonstrates that KiDeePC substantially improves tracking accuracy compared to DeePC. Furthermore, DeePC encounters a considerable number of solver failures (404 out of 5000 instances), while KiDeePC completes all runs successfully without any failures, underscoring its superior robustness and numerical reliability. These findings suggest that the incorporation of kinematics effectively alleviates model mismatch in nonlinear settings, thereby enhancing both accuracy and stability.

IV. CONCLUSIONS

In this paper, we propose a kinematics-informed DeePC framework specifically designed for mobile robots to enable effective trajectory tracking. Furthermore, we validate the proposed approach through simulations and compare its performance with that of conventional DeePC. The results demonstrate that our method achieves higher tracking accuracy, faster convergence, and smoother control inputs. Future work will focus on extending the approach to real-time robotic systems through both simulations and experimental validation.

REFERENCES

- [1] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model predictive control: theory, computation, and design*. Nob Hill Publishing, 2020.
- [2] 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.
- [3] S. Yu, M. Köhler, Y. Huang, H. Chen, and F. Allgöwer, “Model predictive control for autonomous ground vehicles: a review,” *Autonomous Intelligent Systems*, vol. 1, 12 2021.
- [4] I. Markovsky and F. Dörfler, “Behavioral systems theory in data-driven analysis, signal processing, and control,” *Annual Reviews in Control*, vol. 52, 11 2021.
- [5] Z. wu, P. Christofides, W. Wu, Y. Wang, F. Abdullah, A. Alnajdi, and Y. Kadakia, “A tutorial review of machine learning-based model predictive control methods,” *Reviews in Chemical Engineering*, vol. 41, pp. 359–400, 12 2024.
- [6] F. Dörfler, J. Coulson, and I. Markovsky, “Bridging direct and indirect data-driven control formulations via regularizations and relaxations,” *IEEE Transactions on Automatic Control*, vol. 68, no. 2, pp. 883–897, 2022.
- [7] I. Markovsky, J. C. Willems, S. Van Huffel, and B. De Moor, *Exact and approximate modeling of linear systems: A behavioral approach*. SIAM, 2006.
- [8] J. Coulson, J. Lygeros, and F. Dörfler, “Data-enabled predictive control: In the shallows of the deepc,” in *2019 18th European control conference (ECC)*. IEEE, 2019, pp. 307–312.
- [9] E. Elokda, J. Coulson, P. N. Beuchat, J. Lygeros, and F. Dörfler, “Data-enabled predictive control for quadcopters,” *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8916–8936, 2021.
- [10] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, “Data-driven model predictive control: closed-loop guarantees and experimental results,” *at-Automatisierungstechnik*, vol. 69, no. 7, pp. 608–618, 2021.
- [11] —, “Linear tracking mpc for nonlinear systems—part ii: The data-driven case,” *IEEE Transactions on Automatic Control*, vol. 67, no. 9, pp. 4406–4421, 2022.
- [12] K. Worthmann, R. Strässer, M. Schaller, J. Berberich, and F. Allgöwer, “Data-driven mpc with terminal conditions in the koopman framework,” in *2024 IEEE 63rd Conference on Decision and Control (CDC)*. IEEE, 2024, pp. 146–151.
- [13] 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, 2019.