Robotics and Autonomous Systems 194 (2025) 105152

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Trajectory tracking model predictive control for mobile robot based on deep Koopman operator modeling

Minan Tang [a],*, Yaqi Zhang [a], Shuyou Yu [b], Jinping Li [a], Kunxi Tang [a]

[a] School of Automation and Electrical Engineering, Lanzhou Jiaotong University, Lanzhou, Gansu, 730070, China
[b] School of Communication Engineering, Jilin University, Changchun, Jilin, 130022, China

ARTICLE INFO

MSC:
93-10

Keywords:
Mobile robot
Model predictive control
Self-triggered control
Deep koopman operator
Extended state observer

ABSTRACT

Trajectory tracking serves as a pivotal performance metric for mobile robot systems, and is crucial for improving the efficiency of robots. The intricate kinematic and dynamic properties of robot systems pose substantial challenges in achieving accurate modeling and effective control, which remain pressing issues within the current research domain. This study focuses on wheeled mobile robot, relying on the deep Koopman operator theory, combined with the extended state observer (ESO) and the adaptive predictive time domain self-triggered model predictive control (APST-MPC) method, to propose a data-driven solution for the trajectory tracking control issue of wheeled mobile robot under uncertain model parameters. Firstly, the dynamic model of the mobile robot is constructed by the deep Koopman operator method. Secondly, to counteract operational disturbances encountered by the robot, an ESO is designed for disturbance estimation and subsequent compensation within the controller. Thirdly, to reduce the computational load, APST-MPC is employed to enhance the trajectory tracking control of wheeled mobile robot. Ultimately, the efficacy of the proposed trajectory tracking controller is confirmed through simulation experiments. The simulation outcomes confirm the deep Koopman operator theory's efficacy in establishing a robot model with considerable accuracy, the tracking error of the robot is reduced by 46.03% and the total number of triggering times of the system is reduced by more than 59.8% by the APST-MPC controller based on ESO compared with the MPC controller.

1. Introduction

Significant advancements in computer and artificial intelligence technologies have fostered the robust development of mobile robot technology. Mobile robots have been extensively deployed across a multitude of domains [1], including critical areas such as industrial transportation [2], agricultural pest control [3], fire and rescue operations [4], healthcare [5], planetary exploration [6], and military reconnaissance [7]. This widespread adoption is attributable to their inherent advantages, which include lightweight construction, maneuverability, flexibility, and the capacity to endure substantial loads. Building upon the realization of autonomous mobility, further enhancing the operational efficiency of mobile robots has emerged as a critical challenge to address. Undoubtedly, accurate trajectory tracking control technology stands at the heart of resolving this issue.

During the operation of mobile robots, numerous complex factors come into play. In addition to common external disturbances, such as friction and gusts, which affect their motion, there are also internal factors to consider, including parameter uncertainty, modeling errors, and unmodeled dynamics [8]. These factors are interrelated and pose significant challenges to the accurate trajectory tracking of mobile robots. Therefore, effectively suppressing or compensating for the impact of these disturbing factors on the system is crucial for enhancing the accuracy of trajectory tracking control. This improvement is of significant importance for boosting the overall operational capability of mobile robots. It is not only related to the performance of mobile robots across various application domains but also constitutes a key determinant for their stable and efficient operation under intricate and fluctuating conditions.

In the domain of trajectory tracking control for wheeled mobile robots, the primary challenge lies in accurately following the preset reference trajectory in the shortest possible time [9]. During this process, it is essential to consider various factors comprehensively, encompassing factors such as temporal expenditure, accuracy, and system steadiness, to guarantee the high accuracy of trajectory tracking. To enhance the resilience of robotic trajectory tracking control, academics worldwide have invested significant effort in conducting in-depth research. A multitude of control strategies have been introduced

* Corresponding author.
E-mail addresses: tangminan@mail.lzjtu.cn (M. Tang), 12211534@stu.lzjtu.edu.cn (Y. Zhang), shuyou@jlu.edu.cn (S. Yu), 11220936@stu.lzitu.edu.cn (J. Li), 12221583@stu.lzjtu.edu.cn (K. Tang).

https://doi.org/10.1016/j.robot.2025.105152
Received 2 April 2025; Received in revised form 6 June 2025; Accepted 3 August 2025
Available online 11 August 2025

and applied in trajectory tracking control applications, including robust control [10,11], neural network control [12,13], sliding mode control [14,15], adaptive control [16,17] and so on. Model predictive control (MPC) has garnered extensive application and attention within the realm of robot trajectory tracking control in recent years. This is attributable to its notable advantages, which include straightforward modeling, efficient management of multivariate and constrained issues [18], timely compensation for uncertainties caused by model mismatches and external disturbances, and excellent dynamic performance. For instance, relevant research findings have been articulated in Refs. [19–21], thereby substantiating the practical efficacy and application potential of MPC in this domain.

To achieve high-performance trajectory tracking for mobile robots, the MPC method relies on accurate model information. However, obtaining highly accurate models is challenging due to the inherent strong nonlinearity and parameter uncertainty of robots. Common mechanism modeling methods depend on various physical variables [22,23], and in practice, some parameters cannot be measured and are difficult to estimate [24,25], which greatly limits the application range of the models. The data-driven modeling method provides a novel approach to addressing this issue, as it can directly extract and construct models of unknown systems from available data. The Koopman operator [26] is capable of mapping a nonlinear system into a linear infinite dimensional space, thereby achieving the global linearization of the nonlinear system [27]. Kim et al. [28] developed a vehicle model grounded in Koopman theory for capturing nonlinear dynamic behaviors of lane-keeping systems under various driving conditions. The extended dynamic mode decomposition (EDMD) method was employed to estimate the Koopman operator in a confined, finite-dimensional setting, and the stochastic MPC scheme grounded in the Koopman operator was designed to address the modeling error associated with the approximated Koopman operator within the EDMD approach. Li et al. [29] introduced a Koopman modeling approach based on input reinforcement learning to enhance the predictive capability of the Koopman model for multiple future steps by boosting the state and known input through two deep neural networks (DNNs) while training the Koopman model with nonlinear input in a high-dimensional state space.

The operating environment of mobile robots is frequently highly complex, with various interference factors such as friction and external disturbances, adversely impacting the normal operation of the robots, thereby affecting the accuracy and efficiency of task execution. Therefore, to ensure that mobile robots can operate stably and efficiently in complex environments and achieve their designated task objectives, it is imperative to implement effective measures to mitigate the interference of these disturbance factors on the system, thereby improving the robustness and reliability of the system. Hameed et al. [30] considered all torque disturbances and parameter uncertainties of mobile robot as generalized disturbances, which were observed and eliminated in real-time utilizing a nonlinear sliding mode extended state observer. Rodríguez-Arellano et al. [31] proposed a new observer-based H_∞ controller capable of resisting both matched and mismatched disturbances. This controller compensated for the disturbances by employing an observer to estimate disturbances and convert the closed-loop system into one subject to bounded disturbances uniformly. To tackle the challenge of increasing disturbance boundary changes caused by sudden transitions, Wu et al. [32] designed an adaptive disturbance observer and constructed a new switching law based on barrier functions to suppress residual disturbance estimation error of the adaptive disturbance observer under transient conditions.

In the application of MPC, the solution of optimization problems occurs at each sampling time [33], which demands substantial computational resources. However, in practice, solely the initial control variable in the computed control sequence is typically implemented in the actual system [34], resulting in a significant consumption of calculation work. To effectively address this issue, the self-triggered mechanism has emerged as necessary, with its core objective being the reduction of the frequency at which optimization problems are solved. This mechanism enables the system to obtain the latter triggering time without relying on continuous attention, thereby reducing unnecessary computations and enhancing the efficiency of resource utilization. Furthermore, the self-triggered mechanism also does not require additional hardware support, which not only reduces the operational costs of the system but also streamlines the maintenance process. Heshmati-alamdari et al. [35] adopted a vision-driven self-triggered control approach to obtain control inputs and the subsequent triggering times, thereby avoiding continuous measurements by the visual system and reducing processing time and energy consumption. Cao et al. [36] designed a self-triggered scheme grounded in the Lyapunov function to reduce the computational burden associated with MPC, wherein the controller transitions to the terminal controller once the tracking error enters the terminal region.

Table 1 summarizes the relevant research on robot trajectory tracking control in ascending order of publication year.

The combination of different modeling techniques and control methods for robots has been widely used for trajectory tracking control of mobile robots. From Table 1, it can be seen that most reference adopts mechanism modeling methods, and the research on data-driven modeling methods is still in the stage of continuous exploration. There is relatively little consideration for disturbances in existing reference. In recent years, the deep Koopman operator theory has emerged as a novel tool for nonlinear system modeling, providing a new perspective for solving robot modeling challenges and simplifying control design. However, how to combine the deep Koopman operator with existing control strategies to enhance the accuracy and robustness of trajectory tracking remains an open problem. Furthermore, the extended state observer (ESO) as an effective tool for disturbance observation and compensation, and the adaptive predictive time domain self-triggered model predictive control (APST-MPC) algorithm as a control strategy to reduce computational burden, their integrated application is expected to further enhance the trajectory tracking performance of mobile robot. In light of this, this study integrates deep Koopman theory, ESO, and APST-MPC approach to conduct an in-depth exploration of the trajectory tracking control problem for wheeled mobile robot. This method can not only handle model uncertainty and external disturbances, but also reduce computational requirements while ensuring control accuracy. The primary research contributions encompass the following three points:

(1) This study proposes a deep Koopman operator modeling framework that integrates DNN and EDMD, achieving a global linearization representation of the strong nonlinear dynamic system of mobile robot, reducing dependence on accurate model parameters, significantly improving the model's generalization ability to unmodeled dynamics and parameter uncertainties, and simplifying the design complexity of MPC controller.
(2) This study designs a unified disturbance processing strategy based on ESO, which models the internal parameter perturbations and external disturbances of mobile robot as "total disturbances", and uses linear ESO for real-time estimation and compensation. Under the premise of ensuring input-state stability, the triggering interval is expanded, the triggering frequency of the system is reduced, and the robustness of the controller is significantly enhanced.
(3) This study combines adaptive predictive time domain and self-triggered mechanisms to propose the APST-MPC algorithm to balance control accuracy and computational efficiency. By dynamically adjusting the predictive time domain length and non-periodic triggering conditions, the computational burden of the controller is significantly reduced, the utilization of computing resources is optimized, and the operating efficiency of the system is enhanced while maintaining the tracking accuracy.

Table 1
Reference review on robot trajectory tracking control.

Authors (year)	Model/Method	Disturbances estimation method	Control method/Optimize MPC calculation
Yang (2018) [19]	Kinematics model/Mechanism	No	Model predictive control/No
Shu (2018) [17]	Dynamics model/Mechanism	No	Adaptive control
Cao (2019) [36]	Kinematics model/Mechanism	No	Model predictive control/Yes
Zhao (2021) [12]	Kinematics model/Mechanism	No	Fuzzy+Neural network
Cen (2021) [14]	Kinematics model/Mechanism	No	Sliding mode control
Chen (2022) [15]	Dynamics model/Mechanism	No	Adaptive optimal control
Hameed (2023) [30]	Kinematics+dynamics model/Mechanism	Nonlinear sliding mode extended state observer	Active disturbance rejection control
Cenerini (2023) [34]	Kinematics model/Mechanism	No	Model predictive control/No
Cao (2023) [13]	Kinematics model/Data driven	New fractional exponential activation function	Zeroing neural network
Rodríguez-Arellano (2023) [31]	Kinematics model/Mechanism	Disturbance observer	H_∞ control
You (2024) [20]	Kinematics model/Mechanism	No	PID+Model predictive control /Yes
Kim (2025) [28]	Dynamics model/Data driven	No	Model predictive control/No

The subsequent sections of this paper will be structured as outlined below. Section 2 will construct a deep Koopman linear model for wheeled mobile robot. Section 3 will utilize an ESO to compensate for operational disturbances. In Section 4, an APST-MPC controller will be designed to make the robot track the reference trajectory accurately. Section 5 will conduct simulations and analyzes the accuracy of the deep Koopman linear model and the effectiveness of the trajectory tracking control. Section 6 will provide the conclusions.

2. Problem statement and system modeling

Due to the tire characteristics, the dynamics of the robot exhibit strong coupling and nonlinearity. This paper proposes a DNN approach grounded in the Koopman operator to get the dynamic model of the robot system. The weights of the DNN are optimized exclusively with pre-gathered datasets comprising state and control inputs, thereby attaining an accurate depiction of the robotic system's dynamics. The obtained model comprises a stationary nonlinear DNN that maps the state space into the lifted state space, coupled with the linear model that evolves within this lifted state space.

2.1. Koopman operator using the EDMD

Consider the mobile robot as a discrete-time nonlinear dynamic system characterized by external inputs

$$\boldsymbol{x}(k+1) = f\left(\boldsymbol{x}(k), \boldsymbol{u}(k)\right) \tag{1}$$

where $k \in \mathbb{Z}$ is the discrete time step, $\boldsymbol{x}(k) \in \mathcal{N} \subseteq \mathbb{R}^n$ is the state variables of the system, and $\boldsymbol{x}(k) = \begin{bmatrix} x(k) & y(k) & \theta(k) \end{bmatrix}^{\mathrm{T}}$. $\boldsymbol{u}(k) \in \mathcal{M} \subseteq \mathbb{R}^m$ are the control inputs of the system, and $\boldsymbol{u}(k) = \begin{bmatrix} v(k) & \delta(k) \end{bmatrix}^{\mathrm{T}}$. x and y are the current positions of the robot, θ is the heading angle, v is operating speed, and δ is the front wheel steering angle. $f(\boldsymbol{x}, \boldsymbol{u})$ denotes the system's nonlinear evolution of its state dynamics temporally. $\mathcal{N}$ and $\mathcal{M}$ are the state space, $\mathbb{R}$ is the real domain, and $\mathbb{R}^n$ represents the n-dimensional Euclidean space of real-valued vectors, n and m are the dimensions of the system states and inputs, respectively. Fig. 1 shows the model of a mobile robot.

Merge state variables $\boldsymbol{x}(k)$ and control inputs $\boldsymbol{u}(k)$ into an expanded state

$$\zeta = \begin{bmatrix} \boldsymbol{x}(k) \\ \boldsymbol{u}(k) \end{bmatrix} \tag{2}$$

where $\boldsymbol{u} := u_0^\infty$ signifies the set of all control inputs within the control space $\mathcal{M}$. The Koopman operator of the extended state ζ on (1) is written as

$$\boldsymbol{\varphi}\left(\boldsymbol{\zeta}_{k+1}\right) = \mathcal{K}\boldsymbol{\varphi}(\boldsymbol{\zeta}_k) \tag{3}$$

where $\mathcal{K}$ denotes the infinite-dimensional Koopman operator and $\boldsymbol{\varphi}(\boldsymbol{\zeta}_k) \in \mathbb{R}$ is an observable function in lifted space that belongs to an infinite-dimensional Hilbert space.

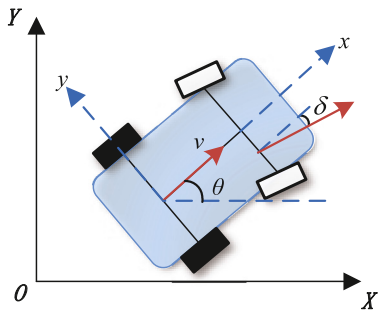


Fig. 1. Mobile robot model.

The state $\boldsymbol{\zeta}_k$ of the original nonlinear system is finite-dimensional, $\mathcal{K}$ exhibits infinite dimensional properties and linearity in the observable state space [37] adopting $\boldsymbol{\varphi}(\boldsymbol{\zeta}_k) = \begin{bmatrix} \boldsymbol{\varphi}_1(\boldsymbol{x}_k)^{\mathrm{T}} & \cdots & \boldsymbol{\varphi}_L(\boldsymbol{x}_k)^{\mathrm{T}} & \boldsymbol{u}_k^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{L+m}$ as a set of observable functions, where L denotes the count of observable functions of the $\boldsymbol{x}$. Due to the fact that the future control inputs $\boldsymbol{u}_{k+1}$ does not require prediction, only the first L rows of $\boldsymbol{\varphi}\left(\boldsymbol{\zeta}_{k+1}\right)$ are concerned. The approximate Koopman operator can be decomposed into $\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix}$, where $\boldsymbol{A} \in \mathbb{R}^{L\times L}$, $\boldsymbol{B} \in \mathbb{R}^{L\times m}$ are the linear constant matrices of the lifted model. Consequently, the Koopman model obtained in the observable space is

$$\boldsymbol{\varphi}\left(\boldsymbol{\zeta}_{k+1}\right) = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix} \begin{bmatrix} \boldsymbol{\varphi}\left(\boldsymbol{x}_k\right) \\ \boldsymbol{u}_k \end{bmatrix} \tag{4}$$

Assuming there is a dataset $\boldsymbol{D} = \{(\boldsymbol{x}(k), \boldsymbol{u}(k)), k = 1, \ldots, M\}$ that meets system (1), matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ can be determined by resolving the least-squares error (5) using the dataset $\boldsymbol{D}$. Define the matrix $\boldsymbol{C} \in \mathbb{R}^{n\times L}$, map $\boldsymbol{\varphi}(\boldsymbol{x})$ back to the $\boldsymbol{x}$, and solve the minimization problem of (6) to obtain the matrix $\boldsymbol{C}$ in the least squares sense.

$$\min_{\boldsymbol{A},\boldsymbol{B}} \sum_{k=1}^{M} \left\| \boldsymbol{\varphi}(\boldsymbol{x}_k^+) - \boldsymbol{A}\boldsymbol{\varphi}(\boldsymbol{x}_k) - \boldsymbol{B}\boldsymbol{u}_k \right\|_F^2 \tag{5}$$

$$\min_{\boldsymbol{C}} \sum_{k=1}^{M} \left\| \boldsymbol{x}_k - \boldsymbol{C}\boldsymbol{\varphi}(\boldsymbol{x}_k) \right\|_F^2 \tag{6}$$

where $\boldsymbol{x}_k^+$ is the evolution of $\boldsymbol{x}_k$ along with $\boldsymbol{u}_k$, and $\|\cdot\|$ is Euclidean norm. By solving (5) and (6), the matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are derived as follows.

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix} = \boldsymbol{V}\boldsymbol{G}^{\mathrm{T}}\left(\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}\right)^{\dagger} \tag{7}$$

$$\boldsymbol{C} = \boldsymbol{X}\boldsymbol{H}^{\dagger} \tag{8}$$

Among them $\dagger$ is the Moore–Penrose pseudoinverse. $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{\varphi}(\boldsymbol{x}_1) & \cdots & \boldsymbol{\varphi}(\boldsymbol{x}_M^+) \end{bmatrix}$, $\boldsymbol{G} = \begin{bmatrix} \boldsymbol{E} \\ \boldsymbol{U} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\varphi}(\boldsymbol{x}_1) & \cdots & \boldsymbol{\varphi}(\boldsymbol{x}_M) \\ \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_M \end{bmatrix}$, $\boldsymbol{H} = \begin{bmatrix} \boldsymbol{\varphi}(\boldsymbol{x}_1) & \cdots & \boldsymbol{\varphi}(\boldsymbol{x}_M) \end{bmatrix}$, $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_M \end{bmatrix}$.

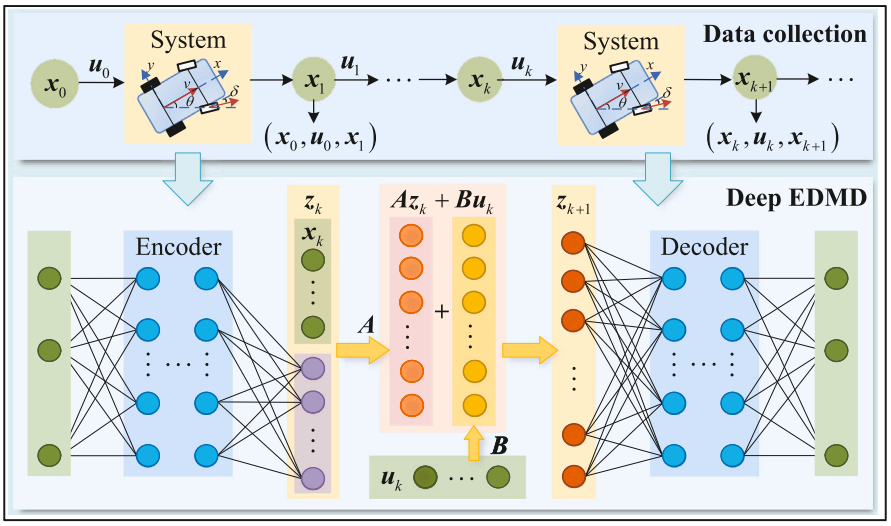


Fig. 2. Deep EDMD architecture.

2.2. Deep EDMD method modeling

The robot dynamics modeling algorithm based on the deep EDMD integrates a DNN into the EDMD approach to obtain the estimation of the finite-dimensional Koopman operator. This algorithm utilizes the DNN to automatically construct the observable subspace associated with the Koopman operator. The approximate dynamics generated by the deep EDMD can be formulated as follows

$$\begin{cases} \boldsymbol{z}_{k+1} = \mathcal{K}\boldsymbol{\Phi}_k \\ \hat{\boldsymbol{x}}_k = \tilde{\boldsymbol{\Phi}}_k\left(\boldsymbol{z}_k, \boldsymbol{\phi}_{\mathrm{d}}\right) \end{cases} \tag{9}$$

where $\mathcal{K} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix} \in \mathbb{R}^{L\times N}$, $N = L + m$. $\boldsymbol{z}_k$ is the lifted observable space obtained by the encoder with weight $\boldsymbol{\phi}_{\mathrm{e}}$. $\boldsymbol{\Phi}_k = \begin{bmatrix} \boldsymbol{z}_k^{\mathrm{T}} & \boldsymbol{u}_k^{\mathrm{T}} \end{bmatrix} \in \mathbb{R}^N$, where $\tilde{\boldsymbol{\Phi}}_k$ is the decoder, with a weight of $\boldsymbol{\phi}_{\mathrm{d}}$.

Fig. 2 presents the architecture of the deep EDMD approach. The deep EDMD approach is predicated on the encoder and decoder, serving as the observable functions and the Koopman model, respectively. Firstly, the original state variables $\boldsymbol{x}_k$ are mapped to the lifted observable space $\boldsymbol{z}_k$ through the encoder. Secondly, $\boldsymbol{z}_k$ and $\boldsymbol{u}_k$ are combined to form a lifted state, which is used to construct linear evolution in the lifted space. Again, the subsequent states $\boldsymbol{z}_{k+1}$ are calculated by the linear Koopman model. Finally, the state of the robot is restored from the lifted state space by a decoder composed of fully connected layers.

The output of DNN is defined by

$$\boldsymbol{y}_j^{(l)} = \sigma_j^{(l)}\left(\boldsymbol{W}_j^{(l)}\boldsymbol{y}_j^{(l-1)} + b_j^{(l)}\right) \tag{10}$$

where $j = \mathrm{e}, \mathrm{d}$ denotes the subscript of the encoder or decoder respectively. $\sigma^{(l)}$, $\boldsymbol{W}^{(l)} \in \mathbb{R}^{n_l\times n_{l-1}}$ and $b^{(l)} \in \mathbb{R}^{n_l}$ indicate the activation function, weight, and bias of the hidden layer l, respectively. $l = 1, \ldots, n_h$, n_h is the number of hidden layers of the encoder and decoder, and n_l is the number of hidden layer neurons.

Obtain the lifted state $\boldsymbol{z}_k$ through the encoder,

$$\boldsymbol{z}_k = \left[\boldsymbol{x}_k^{\mathrm{T}} \quad \left(\boldsymbol{y}_{\mathrm{e}}^{n_h}\right)^{\mathrm{T}}\right]^{\mathrm{T}} \tag{11}$$

where $\boldsymbol{y}_{\mathrm{e}}^{n_h}$ represents the output from the final layer of the encoder's NN, $\boldsymbol{y}_{\mathrm{e}}^{(0)} = \boldsymbol{x}_k$. $\boldsymbol{z}_{k+1} = \mathfrak{S}\boldsymbol{\Phi}_k$ is a linear time-invariant system, where $\mathfrak{S}$ represents the multi-step linear evolution of $\mathcal{K}$, extended to p-step ahead state as follows.

$$\boldsymbol{z}_{k+p} = \boldsymbol{A}^p\boldsymbol{z}_k + \sum_{i=1}^{p}\boldsymbol{A}^{i-1}\boldsymbol{B}\boldsymbol{u}_{k+p-i} \tag{12}$$

The reconstructed state $\hat{\boldsymbol{x}}_k$ is calculated by the decoder based on $\boldsymbol{z}_k$

$$\hat{\boldsymbol{x}}_k = \tilde{\boldsymbol{\Phi}}_k\left(\boldsymbol{z}_k, \boldsymbol{\phi}_{\mathrm{d}}\right) \tag{13}$$

Ensure the consistency of the Koopman linear system and the original nonlinear system evolution over time, and minimize the state prediction loss.

$$L_{\mathrm{p}} = \frac{1}{p}\sum_{i=1}^{p}\left\|\boldsymbol{x}_{k+i} - \tilde{\boldsymbol{\Phi}}_k\left(\mathfrak{S}^i\boldsymbol{\Phi}_k, \boldsymbol{\phi}_{\mathrm{d}}\right)\right\|_2^2 \tag{14}$$

The modeling accuracy is enhanced through the reduction of prediction discrepancies in the lifted observation space, thereby minimizing the loss within the lifted linear space.

$$L_{\mathrm{l}} = \frac{1}{p}\sum_{i=1}^{p}\left\|\boldsymbol{z}_{k+i} - \mathfrak{S}^i\boldsymbol{\Phi}_k\right\|_2^2 \tag{15}$$

Ensure that deep Koopman learns effective observation functions so that the state variables can be recovered from the lifted state through the decoder, minimizing the reconstruction error loss.

$$L_{\mathrm{r}} = \frac{1}{p}\sum_{i=1}^{p}\left\|\boldsymbol{x}_i - \tilde{\boldsymbol{\Phi}}_k\left(\boldsymbol{z}_i, \boldsymbol{\phi}_{\mathrm{d}}\right)\right\|_2^2 \tag{16}$$

To avoid overfitting, the L_2 regularization loss function is used, that is

$$L_2 = \left\|\boldsymbol{\phi}_{\mathrm{e}}\right\|_2^2 + \left\|\boldsymbol{\phi}_{\mathrm{d}}\right\|_2^2 \tag{17}$$

The estimation steps of the Koopman operator are optimized by introducing the aforementioned loss function, aiming to minimize the weighted loss function L.

$$L = \alpha L_{\mathrm{p}} + \beta L_{\mathrm{l}} + \gamma L_{\mathrm{r}} + \mu L_2 \tag{18}$$

where $\alpha, \beta, \gamma, \mu$ are the corresponding weights of different losses, indicating the importance of each loss.

Algorithm 1 delineates the specific process of the robot modeling method utilizing the deep Koopman operator.

After training by the deep Koopman operator method, get the mobile robot's approximated dynamic linear model

$$\begin{cases} \boldsymbol{z}_{k+1} = \boldsymbol{A}\boldsymbol{z}_k + \boldsymbol{B}\boldsymbol{u}_k \\ \hat{\boldsymbol{x}}_k = \tilde{\boldsymbol{\Phi}}_k\left(\boldsymbol{z}_k, \boldsymbol{\phi}_{\mathrm{d}}\right) \end{cases} \tag{19}$$

3. Design of extended state observer

The deep Koopman operator constructs a model of the robot's system utilizing offline The deep Koopman operator can construct high-dimensional models for nonlinear systems and predict robot states by analyzing the input and output data of the system. However, it is challenging to accurately measure friction and external disturbances in practical operating environments. Relying solely on offline data models

Algorithm 1 Deep EDMD modeling

Initialize $\boldsymbol{\phi}_e$, $\boldsymbol{\phi}_d$, $\boldsymbol{A}$, $\boldsymbol{B}$, time step p, $i = 0, \cdots, p$, training period E, loss weight $\alpha, \beta, \gamma, \mu$, batch size b_s, $j = 1, \cdots, b_s$, small scalar $\sigma > 0$;
while $E < E_{\max}$ or $|L| > \sigma$ **do**
  Reset the training set;
  while the training cycle is not terminated **do**
    Sample the robot's state and control input data sequences;
    Get the lifted state according to $\boldsymbol{z}_k = \left[\boldsymbol{x}_k^{\mathrm{T}} \quad \left(\boldsymbol{y}_e^{n_h}\right)^{\mathrm{T}}\right]^{\mathrm{T}}$ and the reconstruction state according to $\hat{\boldsymbol{x}}_k = \tilde{\boldsymbol{\Phi}}_k\left(\boldsymbol{z}_k, \boldsymbol{\phi}_d\right)$;
    Execute the p-step Koopman operator according to $\boldsymbol{z}_{k+p} = \boldsymbol{A}^p \boldsymbol{z}_k + \sum_{i=1}^{p} \boldsymbol{A}^{i-1} \boldsymbol{B} \boldsymbol{u}_{k+p-i}$;
    Solve the weighted loss L minimization problem update $\boldsymbol{\phi}_e$, $\boldsymbol{\phi}_d$, $\boldsymbol{A}$, $\boldsymbol{B}$;
  end while
  $E = E + 1$
end while

makes it challenging to accurately predict the robot's state in practical applications, potentially reducing control accuracy and even leading to system instability. To obtain a higher accurate mobile robot's model, it is necessary to employ an observer to estimate and compensate for disturbances in real-time. The ESO is an advanced method for disturbance estimation that converts internal disturbances within the system into an additional state variable [38]. By analyzing actual input–output data, the ESO can accurately estimate the system's state, thereby improving control accuracy.

The mobile robot system consists of a nominal system without disturbances and a disturbance system with disturbances. The control variable $\boldsymbol{u}(k)$ is represented as

$$\boldsymbol{u}(k) = \boldsymbol{u}_n(k) - \boldsymbol{u}_d(k) \tag{20}$$

where $\boldsymbol{u}_n(k)$ denotes the nominal system control variable and $\boldsymbol{u}_d(k)$ denotes the disturbance compensation control variable.

Expand the disturbances into a new state variable $\boldsymbol{z}_2$, denoted as $\boldsymbol{z}_2(k) = \boldsymbol{d}(k)$, and denoted as $\boldsymbol{z}_2(k+1) = \boldsymbol{\chi}(k)$. The expanded state equation of the mobile robot disturbance system is obtained from the robot training model (19) as

$$\begin{cases} \boldsymbol{z}_d(k+1) = \boldsymbol{C}\boldsymbol{A}\boldsymbol{z}_d(k) + \boldsymbol{C}\boldsymbol{B}\boldsymbol{u}_d(k) + \boldsymbol{y}_1(k) \\ \boldsymbol{z}_2(k+1) = \boldsymbol{\chi}(k) \end{cases} \tag{21}$$

According to the expanded new system (21), a linear ESO is established.

$$\begin{cases} \boldsymbol{e}_1(k) = \widehat{\boldsymbol{z}}_d(k) - \boldsymbol{z}_d(k) \\ \widehat{\boldsymbol{z}}_d(k+1) = \boldsymbol{C}\boldsymbol{A}\widehat{\boldsymbol{z}}_d(k) + \widehat{\boldsymbol{z}}_2(k) + \boldsymbol{C}\boldsymbol{B}\boldsymbol{u}_d(k) - \boldsymbol{\ell}_1 \boldsymbol{g}_1\left(\boldsymbol{e}_1(k)\right) \\ \widehat{\boldsymbol{z}}_2(k+1) = -\boldsymbol{\ell}_2 \boldsymbol{g}_2\left(\boldsymbol{e}_1(k)\right) \end{cases} \tag{22}$$

where $\widehat{\boldsymbol{z}}_d$ is the estimated value of the system state $\boldsymbol{z}_d$, $\widehat{\boldsymbol{z}}_2$ is the estimated value of the disturbance $\boldsymbol{z}_2$, $\boldsymbol{\ell}_1$ and $\boldsymbol{\ell}_2$ are the gain matrices of the observer, and $-\boldsymbol{\ell}_i \boldsymbol{g}_i\left(\boldsymbol{e}_1\right)$ is the nonlinear feedback form. When the nonlinear function $\boldsymbol{g}_i\left(\boldsymbol{e}_1\right)$ satisfies $\boldsymbol{e}_1 \boldsymbol{g}_i\left(\boldsymbol{e}_1\right) > 0, \forall \boldsymbol{e}_1 \neq 0$, it can make the system's state asymptotically converge to the true value, i.e., $\lim_{k\to\infty} \widehat{\boldsymbol{z}}_d(k) = \boldsymbol{z}_d(k)$. In particular, when $\boldsymbol{g}_i\left(\boldsymbol{e}_1\right) = \boldsymbol{e}_1$, the system can be simplified into the following ESO structure.

$$\begin{cases} \boldsymbol{e}_1(k) = \widehat{\boldsymbol{z}}_d(k) - \boldsymbol{z}_d(k) \\ \widehat{\boldsymbol{z}}_d(k+1) = \boldsymbol{C}\boldsymbol{A}\widehat{\boldsymbol{z}}_d(k) + \widehat{\boldsymbol{z}}_2(k) + \boldsymbol{C}\boldsymbol{B}\boldsymbol{u}_d(k) - \boldsymbol{\ell}_1 \boldsymbol{e}_1 \\ \widehat{\boldsymbol{z}}_2(k+1) = -\boldsymbol{\ell}_2 \boldsymbol{e}_1 \end{cases} \tag{23}$$

The corresponding observation error system is

$$\boldsymbol{e}_i(k+1) = \boldsymbol{C}\boldsymbol{A}_i \boldsymbol{e}_i(k) + \boldsymbol{\xi}_i(k) \tag{24}$$

included among these

$$\boldsymbol{e}_i(k) = \begin{bmatrix} \boldsymbol{e}_1(k) \\ \boldsymbol{e}_2(k) \end{bmatrix}, \boldsymbol{C}\boldsymbol{A}_i = \begin{bmatrix} \boldsymbol{C}\boldsymbol{A} - \boldsymbol{\ell}_1 & \boldsymbol{I} \\ -\boldsymbol{\ell}_2 & \boldsymbol{0} \end{bmatrix}, \boldsymbol{\xi}_i = \begin{bmatrix} \boldsymbol{0} \\ -\boldsymbol{\chi}(k) \end{bmatrix},$$
$$\boldsymbol{e}_1(k) = \widehat{\boldsymbol{z}}_d(k) - \boldsymbol{z}_d(k), \boldsymbol{e}_2(k) = \widehat{\boldsymbol{z}}_2(k) - \boldsymbol{z}_2(k)$$

Theorem 1 (*[39]*). *If the observation error system (24) has a continuous function $\boldsymbol{X} : \mathbb{R}^{2n} \to \mathbb{R}^+$, it satisfies (25) when there are $\boldsymbol{K}_\infty$-class functions $\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2$, and satisfies (26) when there are $\boldsymbol{K}_\infty$-class function $\boldsymbol{\alpha}_3$ and $\boldsymbol{K}$-class function $\boldsymbol{\gamma}$*

$$\boldsymbol{\alpha}_1\left(\left|\boldsymbol{e}_i\right|\right) \leq \boldsymbol{X}\left(\boldsymbol{e}_i(k)\right) \leq \boldsymbol{\alpha}_2\left(\left|\boldsymbol{e}_i\right|\right), \forall \boldsymbol{e}_i \in \mathbb{R}^{2n_i} \tag{25}$$

$$\boldsymbol{X}\left(\boldsymbol{e}_i(k+1)\right) - \boldsymbol{X}\left(\boldsymbol{e}_i(k)\right) \leq -\boldsymbol{\alpha}_3\left(\left|\boldsymbol{e}_i\right|\right) + \boldsymbol{\gamma}\left(\left|\boldsymbol{\xi}_i\right|\right), \forall \boldsymbol{e}_i \in \mathbb{R}^{2n_i}, \forall \boldsymbol{s} \in \mathbb{R}^{2n_i} \tag{26}$$

Then $\boldsymbol{X}_i$ is an input-state stable Lyapunov function (ISS-Lyapunov) and the system has input-to-state stability (ISS).

Proof. For the observation error system (24), given a positive definite matrix $\boldsymbol{D}_i \in \mathbb{R}^{2\times n_i}$, by selecting an appropriate observer gain $\boldsymbol{\ell}_i$, the eigenvalues of the state matrix $\boldsymbol{C}\boldsymbol{A}_i$ of (24) are all in the unit circle, and then by solving the equation

$$\left(\boldsymbol{C}\boldsymbol{A}_i\right)^{\mathrm{T}} \boldsymbol{S}_i \left(\boldsymbol{C}\boldsymbol{A}_i\right) - \boldsymbol{S}_i = -\boldsymbol{D}_i \tag{27}$$

The symmetric positive definite matrix $\boldsymbol{S}_i$ is obtained, and the ISS-Lyapunov function of (24) is obtained

$$\boldsymbol{X}_i\left(\boldsymbol{e}_i\right) = \boldsymbol{e}_i^{\mathrm{T}} \boldsymbol{S}_i \boldsymbol{e}_i \tag{28}$$

When $\boldsymbol{\alpha}_1(r) = \lambda_{\max}\left(\boldsymbol{S}_i\right) r^2$, $\boldsymbol{\alpha}_2(r) = \lambda_{\min}\left(\boldsymbol{S}_i\right) r^2$, $\boldsymbol{X}_i\left(\boldsymbol{e}_i\right)$ satisfies (25). When $\boldsymbol{\alpha}_3(r) = 0.5\lambda_{\min}\left(\boldsymbol{D}_i\right) r^2$, $\boldsymbol{\gamma}(r) = \left(\left[2\left|\left(\boldsymbol{C}\boldsymbol{A}_i\right)^{\mathrm{T}} \boldsymbol{S}_i\right|^2 / \lambda_{\min} \boldsymbol{D}_i\right] + \left|\boldsymbol{S}_i\right|^2\right) r^2$, $\boldsymbol{X}_i\left(\boldsymbol{e}_i\right)$ satisfies (26). Therefore, the function (28) is the ISS-Lyapunov function of the error system (24) and (24) has ISS, thus the observation error is bounded. □

Remark 1. The ISS system's output does not approach the origin but rather to a neighborhood of the origin. Meanwhile, the ISS of the system can be guaranteed by finding an ISS-Lyapunov function [40].

Remark 2. By integrating ESO and the deep Koopman operator, the system's ability to resist disturbance has been significantly improved. Compared with traditional methods that rely solely on model compensation, this method significantly improves anti-disturbance performance and does not rely on accurate system parameters. In addition, this strategy is not only applicable to specific system parameters or environments, but also has applicability under a wider range of conditions, thereby enhancing the model's generalization ability.

The $\hat{\boldsymbol{d}}(k)$ obtained by the ESO is utilized to calculate the $\boldsymbol{u}_d(k)$.

$$\boldsymbol{u}_d(k) = (\boldsymbol{C}\boldsymbol{B})^{-1} \hat{\boldsymbol{d}}(k) \tag{29}$$

4. Design of APST-MPC controller

This section proposes an APST-MPC design scheme based on the robot model constructed in Section 2, aiming to illustrate how to utilize the trained model to achieve more accurate trajectory tracking control of the robot while reducing the solution frequency.

Fig. 3 illustrates the control architecture of the ESO-APST-MPC. The linear model of the robot derived from deep Koopman operator training covers both the nominal model and the disturbance model. In this study, the design of an APST-MPC controller is specifically tailored for the nominal model, while an ESO is formulated to estimate the disturbances for the disturbance model. The model trained with the deep Koopman operator exhibits greater similarity to the actual system during the control process, even in the presence of disturbances. Under

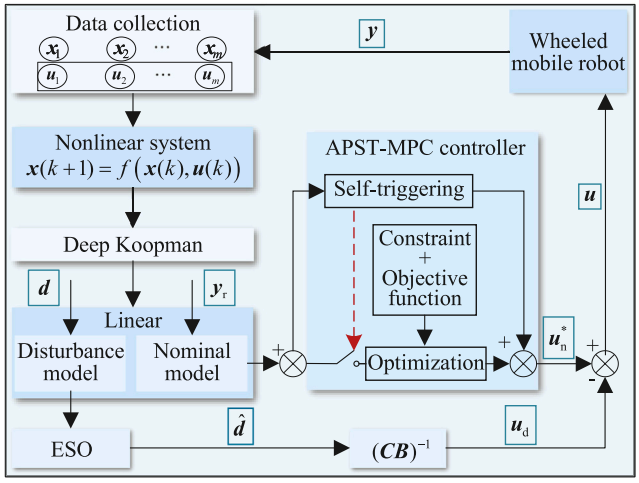


Fig. 3. ESO-APST-MPC control structure diagram.

the control of the APST-MPC controller, the mobile robot is capable of accurately tracking the reference trajectory with fewer triggers, thereby significantly reducing the consumption of computing resources and improving the overall performance and efficiency of the system while ensuring control accuracy.

4.1. Optimization problem

The nominal model of the robot without disturbances is

$$\begin{cases} \boldsymbol{z}_{\mathrm{n}}(k+1) = \boldsymbol{A}\boldsymbol{z}_{\mathrm{n}}(k) + \boldsymbol{B}\boldsymbol{u}_{\mathrm{n}}(k) \\ \boldsymbol{y}_{\mathrm{n}}(k) = \boldsymbol{C}\boldsymbol{z}_{\mathrm{n}}(k) \end{cases} \tag{30}$$

The output expression in the predictive time domain is

$$\boldsymbol{y}_{\mathrm{n}}(k+1) = \boldsymbol{F}\boldsymbol{z}_{\mathrm{n}}(k) + \boldsymbol{M}\boldsymbol{u}_{\mathrm{n}}(k) \tag{31}$$

Among them

$$\boldsymbol{y}_{\mathrm{n}}(k+1) = \begin{bmatrix} \boldsymbol{y}_{\mathrm{n}}(k+1|k) \\ \boldsymbol{y}_{\mathrm{n}}(k+2|k) \\ \vdots \\ \boldsymbol{y}_{\mathrm{n}}(k+N|k) \end{bmatrix}, \boldsymbol{u}_{\mathrm{n}}(k) = \begin{bmatrix} \boldsymbol{u}_{\mathrm{n}}(k|k) \\ \boldsymbol{u}_{\mathrm{n}}(k+1|k) \\ \vdots \\ \boldsymbol{u}_{\mathrm{n}}(k+N-1|k) \end{bmatrix}$$

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{CA} \\ \boldsymbol{CA}^2 \\ \vdots \\ \boldsymbol{CA}^N \end{bmatrix}, \boldsymbol{M} = \begin{bmatrix} \boldsymbol{CB} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{CAB} & \boldsymbol{CB} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{CA}^{N-1}\boldsymbol{B} & \boldsymbol{CA}^{N-2}\boldsymbol{B} & \cdots & \boldsymbol{CB} \end{bmatrix}$$

In order to guarantee the robot effectively and stably track reference trajectory, the objective function is designed

$$\begin{aligned} \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}(k), \boldsymbol{u}_{\mathrm{n}}(k), N\right) = & \sum_{i=0}^{N-1} \left(\left\| \boldsymbol{y}_{\mathrm{e}}(k+i|k) \right\|_{\boldsymbol{\tau}_1}^2 + \left\| \boldsymbol{u}_{\mathrm{e}}(k+i|k) \right\|_{\boldsymbol{\tau}_2}^2 \right) \\ & + \left\| \boldsymbol{y}_{\mathrm{e}}(k+N|k) \right\|_{\boldsymbol{\tau}_3}^2 \end{aligned} \tag{32}$$

where $\boldsymbol{y}_{\mathrm{e}}(k)$ is the error between the reference state $\boldsymbol{y}_{\mathrm{r}}(k)$ and the actual state, $\boldsymbol{u}_{\mathrm{e}}(k)$ is the error between the reference control input $\boldsymbol{u}_{\mathrm{r}}(k)$ and the actual control variable, and $\boldsymbol{\tau}_i, i = 1, 2, 3$ are the weight coefficients.

4.2. Adaptive predictive time domain self-triggered control

The predictive time domain determines the length of the MPC method to predict the behavior of the system. by. A longer predictive time domain ensures system stability but increases the computational load. Conversely, a shorter predictive time domain diminishes the system's anti-disturbance capability, but can simplify the optimization problem. In MPC, the predictive time domain is typically a constant that remains constant value, remaining consistent at each step of solving the optimization problem. However, as the tracking error is close to the terminal region, the shorter predictive time domain can already satisfy the terminal constraints. Based on this observation, this section designs an adaptive predictive time domain calculation method, which employs an adaptive method to reduce the predictive time domain, thereby reducing the dimensionality and computational complexity when solving optimization problem [41].

Traditional MPC operates on a time-triggered, periodic basis, applying solely the initial control variable to the control system. The event-triggered mechanism necessitates the measurement of the system state and the calculation of the tracking error at each step [42], and the determination of the triggering condition and the latter triggering time determined by this error. A self-triggered approach for activating the optimal problem solver is proposed in this section. The latter triggering time for the self-triggered mechanism is obtained at the prior triggering time, which requires fewer state information while ensuring the suboptimal convergence performance.

The following convergence performance can be guaranteed in the time-triggered MPC with a fixed predictive time domain.

$$\begin{aligned} \boldsymbol{G}\left(\boldsymbol{y}_{\mathrm{e}}(k|k), \boldsymbol{u}_{\mathrm{e}}(k|k)\right) \le & \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*(k), \boldsymbol{u}_{\mathrm{e}}^*(k), N\right) \\ & - \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*(k+1), \boldsymbol{u}_{\mathrm{e}}^*(k+1), N\right) \end{aligned} \tag{33}$$

Under the influence of the APST-MPC controller, factors such as disturbances inherent to the mobile robot system, the aperiodicity of control, and the adaptability of the prediction range will affect the robot's tracking effect. Different from the time-triggered MPC, $\{k_j\}$ is defined as the triggering time sequence, and the next time is updated as

$$k_{j+1} = k_j + I_k, k_0 = 0 \tag{34}$$

where k_j is the triggering time, the triggering interval time I_k is determined according to the current actual state variables. The range of I_k is $\varepsilon \le I_k \le N_{k_j}$, where ε is an adjustable parameter.

To ensure the control effect of the controller design (35).

$$\begin{aligned} & \frac{1}{\eta} \sum_{i=0}^{I_k-1} \boldsymbol{G}\left(\boldsymbol{y}_{\mathrm{e}}\left(k_j+i\middle|k_j\right), \boldsymbol{u}_{\mathrm{e}}\left(k_j+i\middle|k_j\right)\right) \\ & \le \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*\left(k_{j+1}\right), \boldsymbol{u}_{\mathrm{e}}^*\left(k_{j+1}\right), N_{k_{j+1}}\right) - \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*\left(k_j\right), \boldsymbol{u}_{\mathrm{e}}^*\left(k_j\right), N_{k_j}\right) \end{aligned} \tag{35}$$

where $\eta \ge 1$ denotes the level of performance loss. According to (33)

$$\begin{aligned} & \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*\left(k_{j+1}\right), \boldsymbol{u}_{\mathrm{e}}^*\left(k_{j+1}\right), N_{k_{j+1}}\right) - \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*\left(k_j\right), \boldsymbol{u}_{\mathrm{e}}^*\left(k_j\right), N_{k_j}\right) \\ & \le -\sum_{i=0}^{I_k-1} \boldsymbol{G}\left(\boldsymbol{y}_{\mathrm{e}}\left(k_j+i\middle|k_j\right), \boldsymbol{u}_{\mathrm{e}}\left(k_j+i\middle|k_j\right)\right) \end{aligned} \tag{36}$$

To ensure the suboptimal performance of the controller (35), the triggering interval time I_k satisfies the following condition.

$$I_k \le (1 - \frac{1}{\eta}) \times \sum_{i=0}^{I_k-1} \boldsymbol{G}\left(\boldsymbol{y}_{\mathrm{e}}\left(k_j+i\middle|k_j\right), \boldsymbol{u}_{\mathrm{n}}\left(k_j+i\middle|k_j\right)\right) \tag{37}$$

In practical application, the upper bound in (37) is taken as the value of triggering interval time I_k, $I_k \in \mathbb{N}_{[1,N_{k_j}]}$.

Define N_{k_j} as the predictive time domain at k_j, and obtain the adaptive update of the predictive time domain at the time k_{j+1} as [42]

$$N_{k_{j+1}} = N_{k_j} - S_k, N_0 = N_{\mathrm{p}} \tag{38}$$

where S_k represents the contraction of the predictive time domain, which is determined at time k_j. The constant N_p ensures the existence of a solution to the optimization problem at the initial time. To ensure the stability of the system, the predictive time domain is updated according to (38), and its contraction is

$$S_k = \min\left\{ I_k - 1, N_{k_j} - \hat{N}_{k_j} \right\} \tag{39}$$

where $\hat{N}_{k_j} = \inf\left\{i : \boldsymbol{y}_{\mathrm{e}}^*(k_j+i|k_j) \in \Omega_\Gamma\right\}$ is the shortest predictive time domain that ensures the iteration of the optimization problem is feasible.

The optimal state sequence $\boldsymbol{y}_{\mathrm{n}}^*(k_j)$ and the optimal control sequence $\boldsymbol{u}_{\mathrm{n}}^*(k_j)$ in N_{k_j} are derived by calculating the minimization problem at k_j.

$$\begin{aligned}\boldsymbol{u}_{\mathrm{n}}^*(k_j) = \arg\min_{\boldsymbol{u}_n(k_j)} \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}(k_j), \boldsymbol{u}_{\mathrm{n}}(k_j), N_{k_j}\right)\\ \text{s.t.} \quad \begin{array}{c}\boldsymbol{y}_{\mathrm{e}}(k_j|k_j) = \boldsymbol{y}_{\mathrm{e}}(k_j)\\ \boldsymbol{u}_{\mathrm{n}}(k_j+i|k_j) \in \mathbb{U}\\ \boldsymbol{y}_{\mathrm{e}}(k_j+i+1|k_j) = f\left(\boldsymbol{y}_{\mathrm{e}}(k_j+i|k_j), \boldsymbol{u}_{\mathrm{n}}(k_j+i|k_j)\right)\\ \boldsymbol{y}_{\mathrm{e}}(k_j+N_{k_j}|k_j) \in \Omega_\Gamma\end{array}\end{aligned} \tag{40}$$

where $i = 0, \ldots, N_{k_j}-1$, $\Omega_\Gamma = \left\{\boldsymbol{y}_{\mathrm{e}} : \|\boldsymbol{y}_{\mathrm{e}}\|_{\boldsymbol{P}}^2 \le \Gamma^2\right\}$, and $\Gamma > 0$ is terminal region. The terminal constraint serves as a pivotal mechanism for ensuring that the end state within the predictive time domain accurately enter the predetermined terminal set, thereby providing a strong guarantee for the system's stability. At each triggering time, the first constraint condition is employed to obtain the robot's actual state, which is utilized as the initial state for the controller. Subsequently, the robot's subsequent state is accurately predicted through combining the control input constraint and the system output (31). This rigorous control strategy enables the accurate adjustment of the controller parameters at each triggering time, ensuring that the robot's behavior in the predictive time domain meets the preset trajectory and performance requirements, thereby achieving accurate control of the robot's motion and improving the overall performance and reliability of the system.

The flow of the APST-MPC algorithm for mobile robot is summarized in Algorithm 2.

Algorithm 2 APST-MPC algorithm

Initialize system information, time index $i = 0$;
while Not reach the maximum simulation time of the system **do**
  Solve the optimization problem (40) at k_j to obtain the optimal control sequence $\boldsymbol{u}_{\mathrm{n}}^*(k_j)$ and the optimal state $\boldsymbol{y}_{\mathrm{n}}^*(k_j)$;
  if Not reach k_{j+1} **then**;
    Apply the first I_k control variables in the optimal control sequence $\boldsymbol{u}_{\mathrm{n}}^*(k_j)$ to the nominal system;
    $i = i+1$
    Measure the actual state $\boldsymbol{y}_{\mathrm{n}}(k_j+i)$ and go to step 5;
  else
    Determine the next triggering time k_{j+1} from $k_{j+1} = k_j + I_k$;
    Update the predictive time domain $N_{k_{j+1}}$ at the time k_{j+1} based on $N_{k_{j+1}} = N_{k_j} - S_k$;
  end if
  Update the triggering time $k_{j+1} \to k_j$ and return to step 1;
end while

The first control variable of $\boldsymbol{u}_{\mathrm{n}}^*(k_j)$ obtained through the optimization problem (40) is the nominal system control variable $\boldsymbol{u}_{\mathrm{n}}(k_j)$ at time k_j, and the actual control variables $\boldsymbol{u}(k)$ of the mobile robot are derived from (20) is

$$\boldsymbol{u}(k) = \boldsymbol{u}_{\mathrm{n}}^*(k_j) - (\boldsymbol{CB})^{-1}\hat{\boldsymbol{d}}(k) \tag{41}$$

4.3. Stability analysis

Theorem 2. *Suppose the mobile robot is governed by $\boldsymbol{u}_n^*(k_j+i|k_j)$ at time k_j+i, (34) determines the triggering time, and (38) determines the update of the predictive time domain. The robot system is ISS, which ensures that the tracking error enters Ω_Γ within a bounded time frame.*

Proof. Define the Lyapunov function $\boldsymbol{V}(k_j) = \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*(k_j), \boldsymbol{u}_{\mathrm{e}}^*(k_j), N_{k_j}\right)$, and the difference $\Delta\boldsymbol{V}_{j+1}$ between the time k_{j+1} and k_j is

$$\begin{aligned}\Delta\boldsymbol{V}_{j+1} &= \boldsymbol{V}(k_{j+1}) - \boldsymbol{V}(k_j)\\ &= \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*(k_{j+1}), \boldsymbol{u}_n^*(k_{j+1}), N_{k_{j+1}}\right)\\ &\quad - \boldsymbol{J}\left(\boldsymbol{y}_{\mathrm{e}}^*(k_j), \boldsymbol{u}_n^*(k_j), N_{k_j}\right)\end{aligned} \tag{42}$$

According to (33), it can be obtained

$$\Delta\boldsymbol{V}_{j+1} \le -\sum_{i=0}^{I_k-1} \boldsymbol{G}\left(\boldsymbol{y}_{\mathrm{e}}\left(k_j+i\middle|k_j\right), \boldsymbol{u}_{\mathrm{e}}\left(k_j+i\middle|k_j\right)\right) \tag{43}$$

Assumption The performance function $\boldsymbol{G}\left(\boldsymbol{y}_{\mathrm{e}}\left(k_j+i\middle|k_j\right), \boldsymbol{u}_{\mathrm{e}}\left(k_j+i\middle|k_j\right)\right)$ is locally Lipschitz continuous, with Lipschitz constant L_g. Supposing $\boldsymbol{G}(0,0) = 0$, there exist positive integers $a > 0$, $b \ge 1$ [43] . and can obtain

$$\boldsymbol{G}\left(\boldsymbol{y}_{\mathrm{e}}\left(k_j+i\middle|k_j\right), \boldsymbol{u}_{\mathrm{e}}\left(k_j+i\middle|k_j\right)\right) \ge a\left\|\boldsymbol{y}_{\mathrm{e}}(k+i|k), \boldsymbol{u}_{\mathrm{e}}(k+i|k)\right\|^b \tag{44}$$

From (44)

$$\Delta\boldsymbol{V}_{j+1} \le -a\left\|\boldsymbol{y}_{\mathrm{e}}(k+i|k), \boldsymbol{u}_{\mathrm{e}}(k+i|k)\right\|^b \tag{45}$$

From this, it can be concluded that $\Delta\boldsymbol{V}_{j+1} < 0$, therefore the system (30) is stable. According to the self-triggered condition (34), it is obtained that

$$\Delta\boldsymbol{V}_{j+1} \le -\frac{1}{\eta}\sum_{i=0}^{I_k-1} \boldsymbol{G}\left(\boldsymbol{y}_{\mathrm{e}}\left(k_j+i\middle|k_j\right), \boldsymbol{u}_{\mathrm{e}}\left(k_j+i\middle|k_j\right)\right) \tag{46}$$

If $\boldsymbol{y}_{\mathrm{e}} \notin \Omega_\Gamma$, (46) satisfies

$$\begin{aligned}\Delta\boldsymbol{V}_{j+1} &\le -\frac{\lambda_{\min}(Q)}{\eta\lambda_{\max}(R)}\Gamma^2\\ \Delta\boldsymbol{V}_{j} &\le -\frac{\lambda_{\min}(Q)}{\eta\lambda_{\max}(R)}\Gamma^2\\ &\vdots\\ \Delta\boldsymbol{V}_{1} &\le -\frac{\lambda_{\min}(Q)}{\eta\lambda_{\max}(R)}\Gamma^2\end{aligned} \tag{47}$$

By adding the inequalities in (47), we get

$$\boldsymbol{V}(k_{j+1}) \le \boldsymbol{V}(k_0) - k_{j+1}\frac{\lambda_{\min}(Q)}{\eta\lambda_{\max}(R)}\Gamma^2 \tag{48}$$

From (48), when $j \to \infty$, $\boldsymbol{V}(k_{j+1}) < 0$, and the Lyapunov function positive definite should be greater than 0. Therefore, the tracking error $\boldsymbol{y}_{\mathrm{e}}$ can enter Ω_Γ in a finite time. □

Remark 3. Adaptive predictive time domain and self-triggered strategies can adjust the complexity of optimization problems in real time according to the system dynamics, thereby achieving more efficient computational performance. This contrasts with traditional MPC methods that typically require constant and possibly excessive computing resources. Compared with the resource waste caused by the fixed predictive time domain in traditional MPC methods, this strategy reduces computational load while ensuring control performance by adjusting the prediction range and triggering conditions.

Remark 4. It is worth noting that although the ISS properties of ESO and APST-MPC are independently established (Theorems 1–2), the unified Lyapunov proof of the combined system is still challenging, primarily due to the coupling effect between the approximation error of the deep Koopman model and the dynamics of the disturbance observer. This paper verifies its empirical stability through systematic experiments, and future work will explore rigorous joint analysis from a theoretical perspective.

5. Experimental validation

In this study, simulation experiments are conducted utilizing the Matlab/Simulink platform, aiming to comprehensively validate the

Table 2
Hyperparameters in deep Koopman.

Hyperparameter	Value	Hyperparameter	Value
Learning rate	10^{-4}	b_s	128
L	13	p	43
α	1	β	1
γ	1	μ	1

effectiveness of the deep Koopman operator model and the ESO-APST-MPC controller. The accuracy of the deep Koopman operator in modeling nonlinear system is quantitatively evaluated by constructing a detailed simulation model to simulate the complex working conditions. Concurrently, the simulation experiments for the ESO-APST-MPC controller focus on its comprehensive performance, anti-disturbance capability, and computational efficiency in trajectory tracking.

5.1. Deep Koopman model validation

Considering that the deep Koopman operator represents a data-driven approach to modeling, the initial phase involves gathering the mobile robot's operational data. Consequently, this study utilizes a trajectory dataset of the mobile robot to train the deep Koopman model. Specifically, a corresponding dataset was constructed by simulating the dynamic equation of the mobile robot within the Matlab environment. In data collection, the sampling period is 0.02 s, and the initial state is set to zero. The dataset covers 20,000 state trajectories, with each trajectory containing 201 state vectors and 200 control vectors. To ensure the validity of the data and model's capacity for generalization, the dataset is partitioned into three segments: a training set, a validation set, and a testing set, allocated at ratios of 80%, 10%, and 10%, respectively. Furthermore, to enhance the diversity of training data, this study generates random sampling starting points within the range $[0, p]$, and samples the data sequence from various starting points before the commencement of each training cycle. This strategy facilitates the model's exposure to a broader spectrum of data distributions throughout the training process, thereby enhancing its adaptability to diverse operating conditions. In the deep Koopman, the encoder and decoder are five-layer structures, denoted as $[n \quad 64 \quad 128 \quad 64 \quad L-n]$ and $[L \quad 128 \quad 64 \quad 64 \quad n]$, respectively. Table 2 lists the hyperparameters used in the simulation.

During the process where the mobile robot follows a double-shift reference trajectory, the simulation compares the local linearization method, the Koopman model, and the deep Koopman model to highlight the effectiveness of the modeling method obtained in this study, and sets the same initial state and control input. Figs. 4 and 5 display the simulation comparison results for the robot's state dimension and control input under identical control conditions. Fig. 6 presents a comparison of errors between the state and control variables and their actual values when different modeling methods are employed. The true values depicted in the figure are sampled from the test dataset.

The process of deep Koopman operator modeling involves the application of deep neural networks, which results in a relatively lengthy training cycle. However, this data-driven approach can achieve higher modeling accuracy as it can learn more accurate dynamic representations from complex nonlinear systems. In contrast, the Koopman operator modeling method features a shorter training time due to its reliance on the least squares solutions. Nevertheless, this method typically necessitates the manual design of observable functions, which may limit the model's expressive capability, potentially at the cost of some modeling accuracy. When comparing the deep Koopman method proposed in this study with both the local linearization method and the traditional Koopman method, it is evident that the deep Koopman method possesses an advantage in terms of convergence speed. The local linearization method is only linearized at $\boldsymbol{x}_0$, which may lead to inadequate modeling accuracy across the global range. In the practical applications of the Koopman method, the design of observable function often depends on expert knowledge, which may impact the model's generalization capability and accuracy.

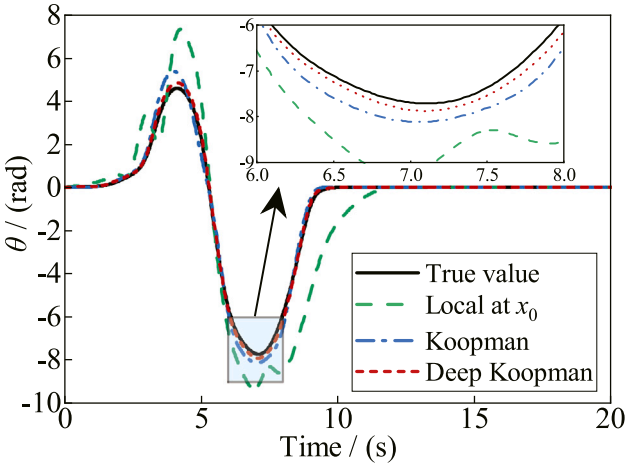


Fig. 4. Comparison of heading angle accuracy.

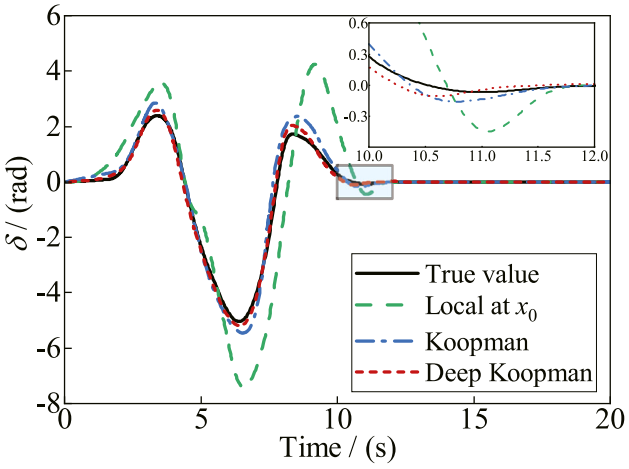


Fig. 5. Comparison of front wheel steering angle accuracy.

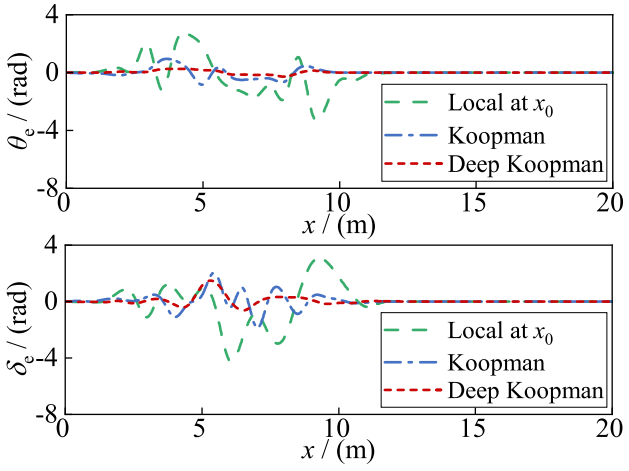


Fig. 6. Comparison of modeling error.

Table 3
RMSE values of state variable.

Modeling method	RMSE
Local linearization model	1.65
Koopman model	0.44
Deep Koopman model	0.12

The root mean square error (RMSE) reflects the model's accuracy by calculating the square root of the average sum of the squared differences between the actual and predicted values, serving as a crucial metric for evaluating the model's accuracy. RMSE is shown in (49).

$$\mathrm{RMSE} = \sqrt{\frac{1}{O}\sum_{k=1}^{O}\left(\theta - \hat{\theta}\right)^2} \tag{49}$$

where O is the size of the test set, θ represents the true value of the heading angle, and $\hat{\theta}$ represents the predicted value of the heading angle. To evaluate the predictive performance of the model and show the relative advantages of the deep Koopman model, quantitative analysis was carried out. RMSE is employed as a quantitative index to compare the accuracies of the local linearization model, the Koopman model, and the deep Koopman model. Table 3 lists the RMSE values corresponding to the state variables under various modeling methods in Fig. 4.

The tabulated numerical outcomes in Table 3 demonstrate that the RMSE of the heading angle modeled by the deep Koopman method proposed in this study is the lowest among all the compared methods. Both the Koopman model and the deep Koopman model exhibit good prediction accuracy. Specifically, the Koopman model shows an improvement of 73.33% over the local linearization model, while the deep Koopman model demonstrates an improvement of 92.73% compared to the local linearization model.

5.2. Controller validation

To demonstrate the effectiveness of ESO-APST-MPC controller, simulations were conducted to assess its disturbance estimation capability and trajectory tracking ability under various reference trajectories. Set the sampling period to 0.05 s, the initial predictive time domain to $N_0 = 25$, and the weight coefficients of the objective function to $\boldsymbol{\tau}_1 = \mathrm{diag}(0.6,0.6,1)$, $\boldsymbol{\tau}_2 = \mathrm{diag}(0.5,0.5)$, $\boldsymbol{\tau}_3 = \mathrm{diag}(0.6,0.6,0.6)$.

When the tracking simulations are conducted under both circular and double-shift conditions, the robot is set to operate from (0,0). To demonstrate the advantages of the APST-MPC controller designed in this study, the tracking effects of APST-MPC and standard MPC controllers are compared through simulation methods under various working conditions. The trajectory tracking results are depicted in Figs. 7 and 8, while the control inputs and states are presented in Figs. 9 and 10.

Fig. 7 emerges the performance comparison of the mobile robot tracking circular reference trajectory, demonstrating that the APST-MPC method more closely tracks the reference trajectory, demonstrating superior tracking accuracy. Fig. 9 further illustrates the control input and state errors during the circular trajectory tracking process, and the results indicate that the tracking errors gradually approaches zero, with the APST-MPC method exhibiting smoother control inputs and smaller tracking errors, which verifies its advantages in reducing control input fluctuations and enhancing tracking accuracy. Fig. 8 compares the performance of the mobile robot in double-shift trajectory tracking, showing that the APST-MPC method also achieves higher tracking accuracy, particularly in areas where the trajectory changes rapidly. Fig. 10 further provides the control input and states during the double-shift trajectory tracking process, showing that the APST-MPC method offers smoother control input and tracking errors that are consistently lower than those of the MPC method, further confirming its

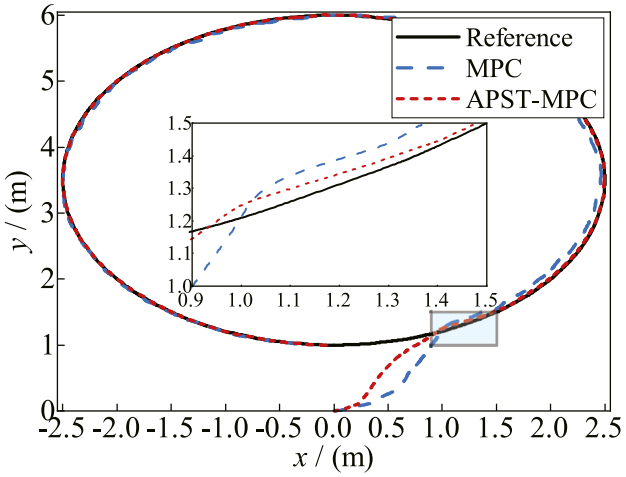


Fig. 7. Comparison of heading angle accuracy.

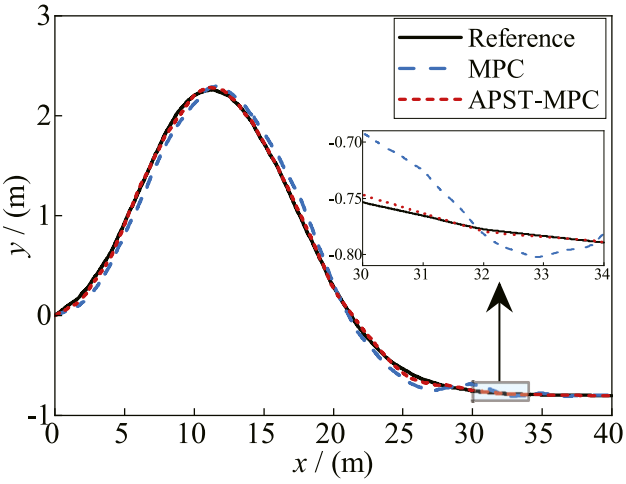


Fig. 8. Comparison of front wheel steering angle accuracy.

effectiveness in improving the trajectory tracking control performance. In summary, the simulation results indicate that the APST-MPC method exhibits enhanced tracking accuracy and better control performance in the trajectory tracking control of mobile robot, effectively enhancing the execution capability of mobile robot in complex trajectory tracking tasks.

The minimum triggering interval time ε affects the triggering frequency and tracking performance. A larger ε can reduce the update frequency of the predictive time domain and the solution frequency of the optimization problem, but at the expense of tracking performance. Conversely, a smaller ε can ensure better control performance, but at the cost of a high solution frequency for the optimization problem and a slower change of the predictive time domain. For this reason, this simulation sets the parameter ε to 0.3, 0.6 and 0.9 to observe the effects on the triggering interval time and the predictive time domain, respectively. Fig. 11 records the triggering times and predictive time domain for the MPC method and the APST-MPC method under different parameter settings.

Fig. 11 illustrates that as ε increases, the triggering interval time also increases and tends to sample uniformly as ε approaches 0. The distribution of triggering times is relatively sparse at the beginning and as the objective function decreases, the distribution of triggering times becomes dense. This occurs because when the objective function is relatively large, more performance loss can be tolerated. Conversely, as

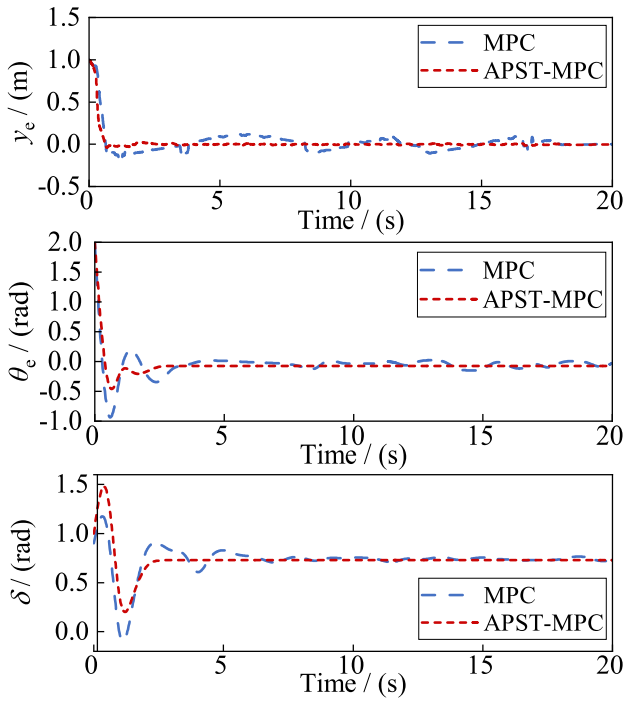


Fig. 9. Comparison of heading angle accuracy.

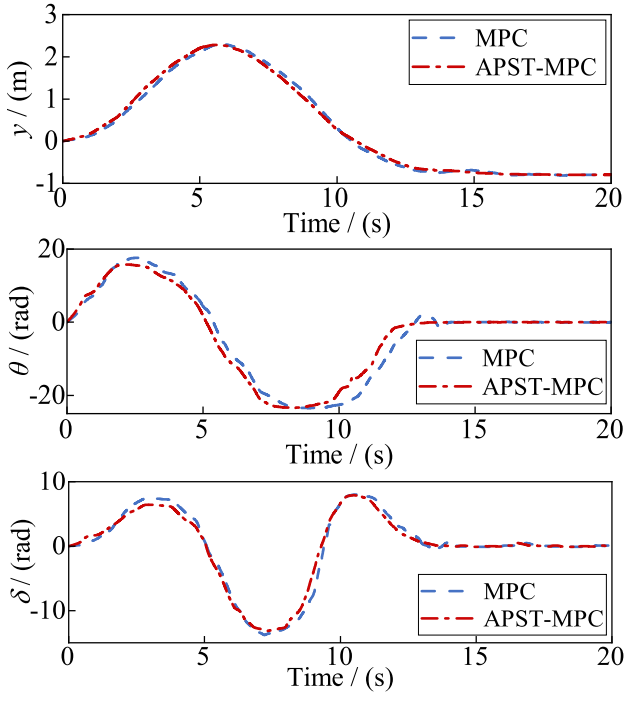


Fig. 10. Comparison of front wheel steering angle accuracy.

the objective function approaches zero, less performance loss is permissible, leading to an increase in triggering frequency. Since the sampling of the APST-MPC method is non-periodic and the sampling frequency is significantly reduced compared to periodic sampling, which reduces the number of times optimization problems need to be solved. Concurrently, the predictive time domain is shortened over time, and as the system state gradually approaches the terminal region, the dimension of the optimization problem is continuously reduced at each step,

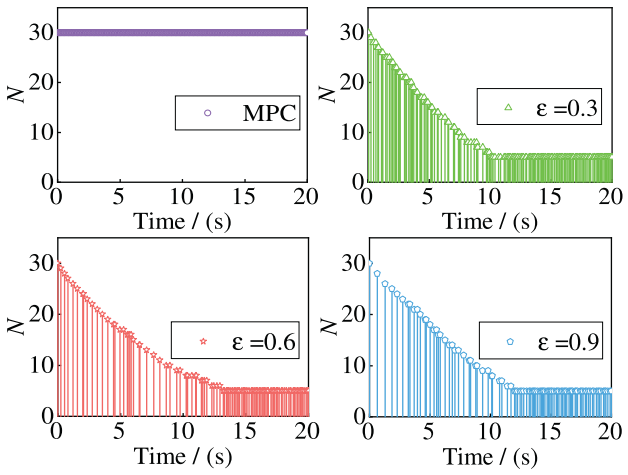


Fig. 11. Triggering time and predictive time domain.

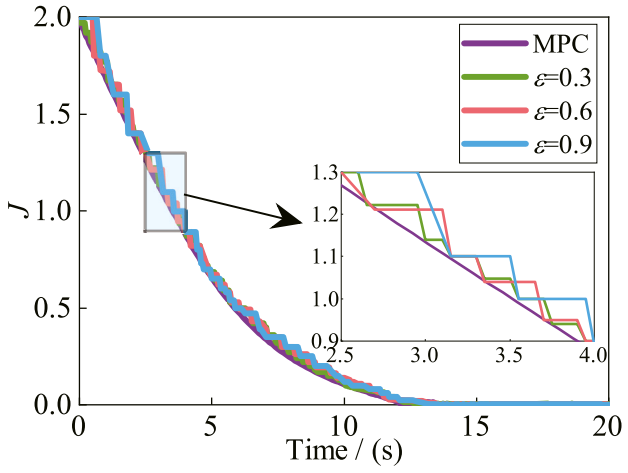


Fig. 12. Objective function.

Table 4
Total optimization times for different control methods.

Control method	Total optimization time/(ms)
MPC	31 986
APST-MPC $\varepsilon = 0.3$	2885
APST-MPC $\varepsilon = 0.6$	1808
APST-MPC $\varepsilon = 0.9$	1580

which not only alleviates the computational load across the entire time domain but also reduces the complexity during each iteration update.

Fig. 12 illustrates the upper bounds of the objective function under various parameter settings. It is evident from the figure that the APST-MPC method proposed in this paper satisfies the suboptimal performance, but reflects that increasing the value of ε within this method may reduce the control performance.

To assess the computational complexity of the optimization problem at each step, the optimization time for each step is determined through simulation. Fig. 13 presents the optimization times for the optimization problem under various control methods, while Table 4 tabulates the total optimization times for these different methods.

Fig. 13 demonstrates that the optimization time for each step of the APST-MPC method decreases over time relative to the MPC method, indicating a reduction in the computational complexity per step. The data presented in Table 4 underscore the effectiveness of APST-MPC concerning reducing computational complexity.

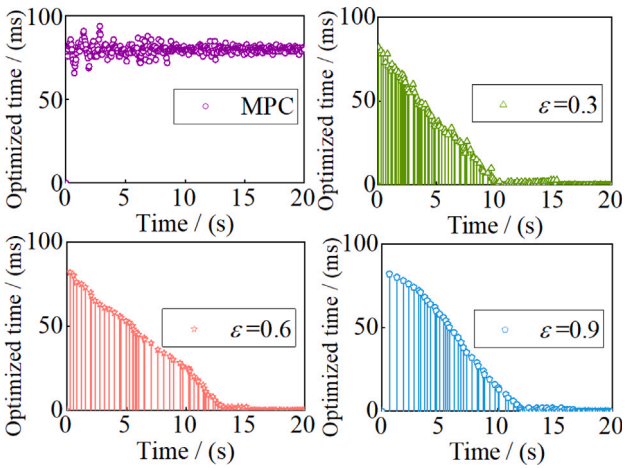


Fig. 13. Optimization times.

To verify the estimation accuracy of the ESO, random disturbances are introduced to the robot at $t = 0$ s, and the disturbance value changed abruptly at $t = 9$ s to simulate the attack behavior encountered by the mobile robots during trajectory tracking. Figs. 14 and 15 present the circular and double-shift tracking trajectories of the mobile robot system under the control of APST-MPC and ESO-APST-MPC, respectively. Fig. 16 displays the true disturbance values alongside the estimated obtained by the ESO.

Fig. 14 illustrates the tracking of a circular trajectory by the mobile robot under disturbance conditions, and the results indicate that the ESO-APST-MPC method can more closely follow the reference trajectory under these conditions, demonstrating superior tracking accuracy and robustness. Fig. 15 presents the tracking accuracy of the mobile robot on the double-shift trajectory under disturbance conditions, and the results indicate that the ESO-APST-MPC method also exhibits higher tracking accuracy in double-shift trajectory tracking, particularly in areas where the trajectory changes rapidly, effectively reducing tracking error and improving trajectory tracking accuracy. Figs. 14 and 15 reflect that the ESO-APST-MPC method can adjust the attitude faster to track the reference trajectory with smaller tracking error after the mobile robot is subjected to simulated attack under the two reference trajectories. Fig. 16 compares the true disturbance value with the estimated value obtained by the ESO. The results indicate that the ESO can effectively estimate disturbances in real-time, with a small error between the estimated and true values, thus verifying the effectiveness and accuracy of the ESO in disturbance observation. To summarize, when there are external disturbances, the ESO-APST-MPC approach can significantly enhance the mobile robot's trajectory tracking performance, demonstrating superior accuracy and robust disturbance rejection capabilities, while maintaining its own stability. This method provides an effective control strategy for high accuracy trajectory tracking of the mobile robot in complex environments.

5.3. Experimental analysis

To substantiate the actual control capability of the controller designed in this study, the robot platform depicted in Fig. 17 was selected for the experiments. The mobile robot platform is equipped with cameras, LiDAR, ultrasonic sensors, and Wi-Fi modules. In practical applications, mobile robots may need to perform tasks involving multiple directional changes, sometimes requiring sharp turns, which puts higher demands on the robot's dynamic response and control accuracy. Therefore, circular trajectory, double-shift trajectory, and square trajectory containing multiple directional changes and angles were selected for testing in the experiment. The controller parameter settings were

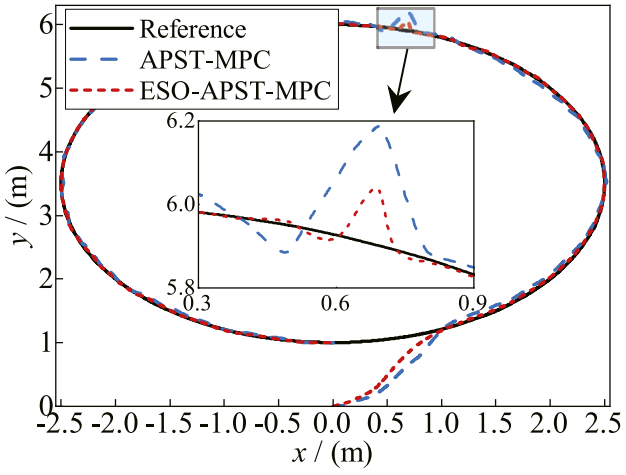


Fig. 14. Circular tracking trajectory with disturbances.

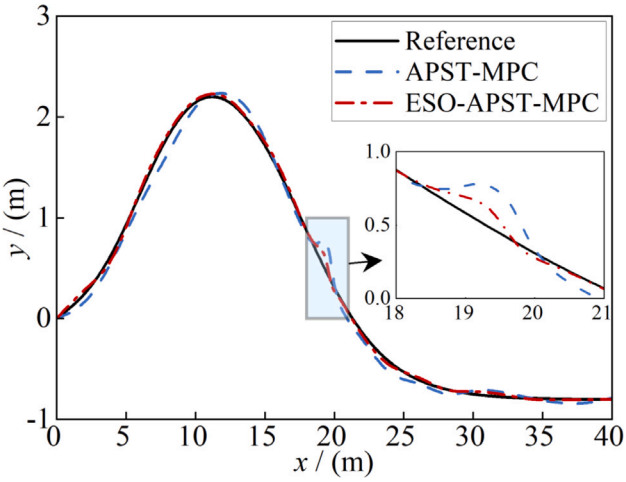


Fig. 15. Double-shift tracking trajectory with disturbances.

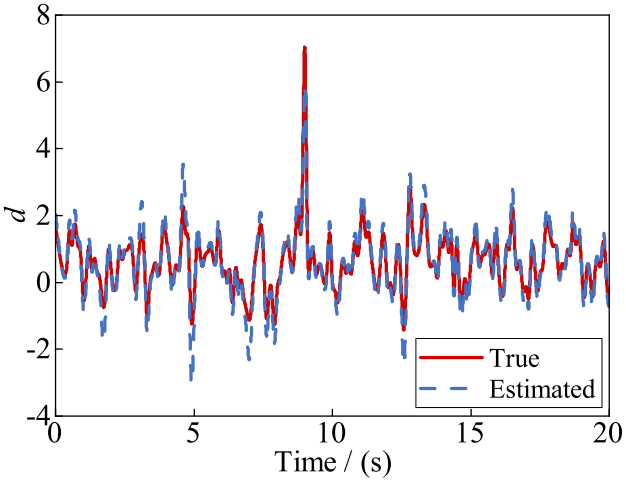


Fig. 16. The true value and the estimated value of the disturbances.

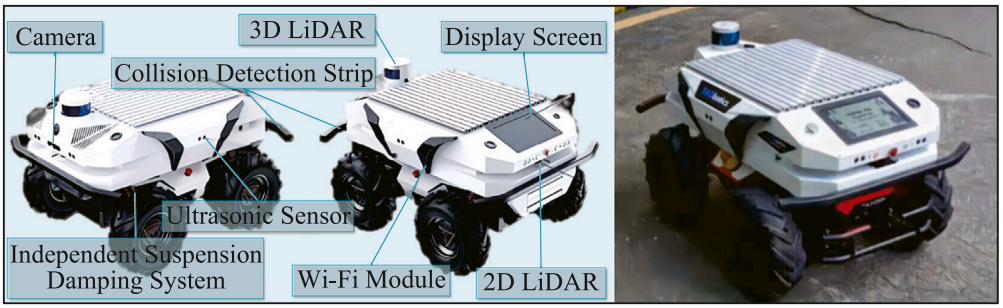
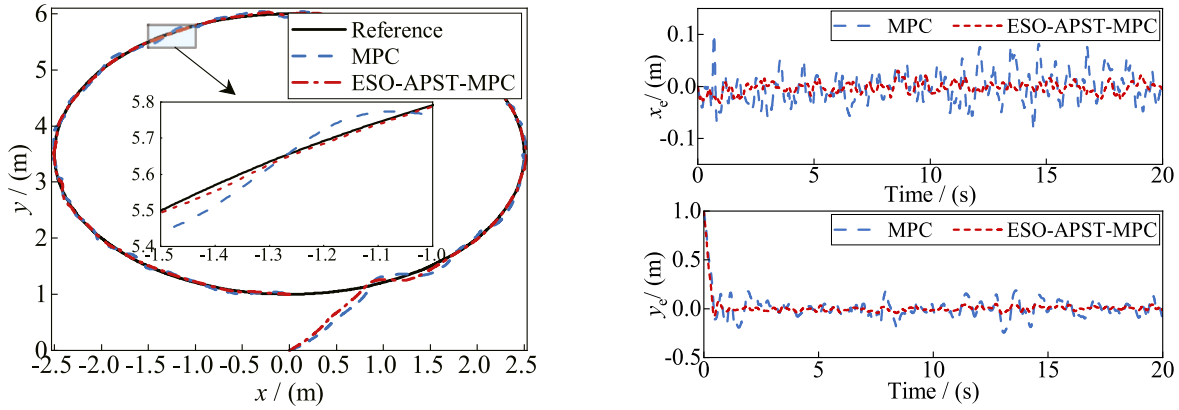
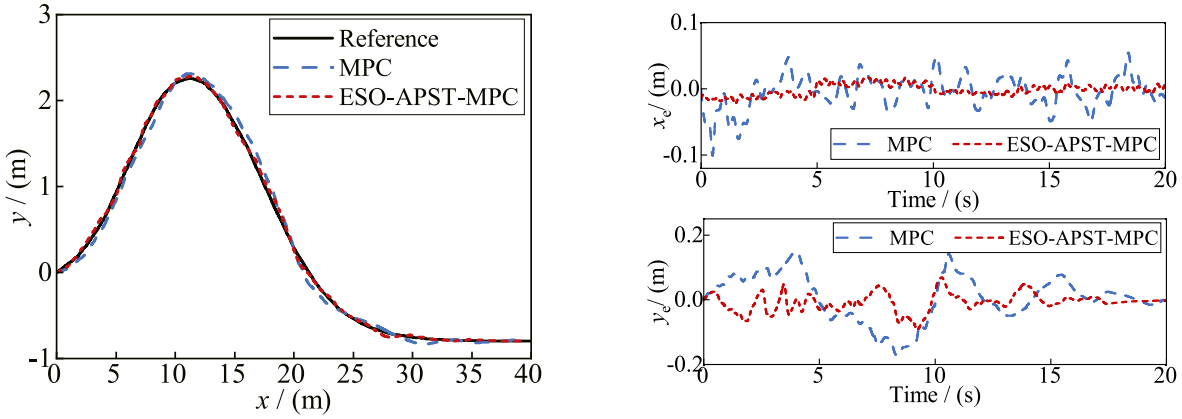


Fig. 17. Robot formatfrei.



(a) Trajectory (b) State errors

Fig. 18. Experimental circular trajectory.



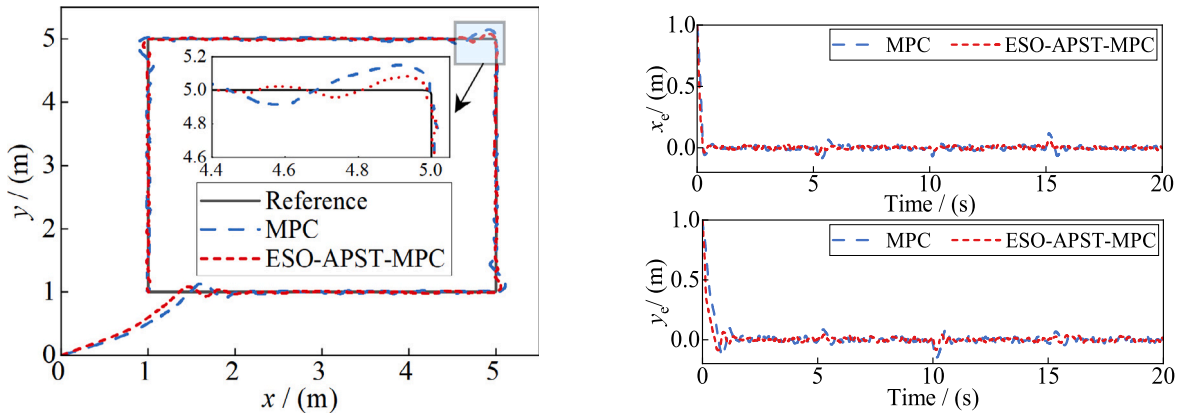
(a) Trajectory (b) State errors

Fig. 19. Experimental double-shift trajectory.

consistent with those used in the numerical simulation experiments. The mass of the wheeled mobile robot applied in the experiments is approximately 110 kg, with a maximum speed of 1.5 m/s and a minimum turning radius of 0 m.

The experimental results are presented in Figs. 18–20, which respectively depict the trajectory tracking effects and tracking errors of the mobile robot in the X-axis and Y-axis directions under various tracking trajectories. The maximum tracking errors of the robot after tracking the reference trajectory are provided in Table 5.

Figs. 18–20 show the tracking circular, double-shift, and square trajectory figures, as well as tracking error comparison figures. It can be observed from these figures and Table 5 that the controller proposed in this study exhibits smaller errors compared to the MPC controller. This is because ESO can effectively estimate disturbances affecting the robot, ensuring tracking accuracy while reducing the sampling frequency, shortening the predictive time domain, and decreasing the computational complexity of the optimization problem. This demonstrates that the control approach proposed in this study possesses superior anti-disturbance and trajectory tracking capabilities. Fig. 20 adopts square trajectory that is not covered by the training set. By comparing the tracking errors under trajectory outside the training data distribution,



(a) Trajectory (b) State errors

Fig. 20. Experimental square trajectory.

Table 5
Maximum tracking errors.

Control method	Circular trajectory		Double-shift trajectory		Square trajectory	
	$\|x_e\|$ maximum value	$\|y_e\|$ maximum value	$\|x_e\|$ maximum value	$\|y_e\|$ maximum value	$\|x_e\|$ maximum value	$\|y_e\|$ maximum value
MPC	0.1354	0.1991	0.1339	0.1775	0.1349	0.1508
ESO-APST-MPC	0.0333	0.0724	0.0251	0.0958	0.0602	0.0914

it shows that the combined system still has practical stability even if there is model mismatch.

6. Conclusion

In this study, the trajectory tracking control problem for wheeled mobile robot was addressed, an MPC approach employing deep Koopman operator modeling was put forth to facilitate accurate and swift tracking of the reference trajectory. During the modeling process, the data-driven deep Koopman operator theory was employed to globally describe the dynamic properties of the robot system, and a robot's high-dimensional linear model was constructed. In the process of disturbance estimation, ESO was applied to estimate and compensate for operational disturbances affecting the output of the controller, thus reducing the impact of disturbances on the effect of the robot's trajectory tracking control and enhancing the system's robustness. In the controller design process, the APST-MPC method was utilized to reduce the dimension and frequency of solving the optimization problem, thereby decreasing the computational burden of the controller while ensuring the robot's tracking accuracy. Finally, a Matlab/Simulink simulation model was constructed to explore in depth the advantages of the deep Koopman operator in high accuracy modeling and the pivotal role of the ESO-APST-MPC controller in achieving efficient trajectory tracking control. The experimental findings verified that the proposed approach met the expected goals concerning modeling accuracy and control performance. The accurate system model in high-dimensional linear space constructed by the deep Koopman operator enhanced the real-time performance of the APST-MPC controller, improved the prediction accuracy of the model, and strengthened the adaptive ability and robustness of the system. It provided an efficient and accurate solution for mobile robot's trajectory tracking control within intricate settings.

This study validated the engineering feasibility of the modular, data-driven system through experiments, but did not provide a rigorous stability proof for the combination. Future work will explore rigorous joint analysis in theory. In addition, combining data-driven modeling methods with advanced control strategies can further improve accuracy and efficiency. And extend the method to various types and scenarios of robots, and conduct physical platform testing to ensure practical feasibility and effectiveness. The goal is to make mobile robots more reliable and autonomous in complex environments, expanding their application scope.

CRediT authorship contribution statement

Minan Tang: Writing – original draft, Methodology, Resources, Funding acquisition, Writing – review & editing, Project administration, Conceptualization, Supervision, Investigation. **Yaqi Zhang:** Visualization, Investigation, Software, Data curation, Writing – original draft, Methodology, Validation, Formal analysis. **Shuyou Yu:** Validation, Methodology, Writing – review & editing, Supervision. **Jinping Li:** Supervision, Formal analysis, Validation, Investigation. **Kunxi Tang:** Resources, Visualization, Investigation, Software, Writing – review & editing, Methodology.

Funding

This work was supported by the National Natural Science Foundation of China [grant numbers 62363022, 61663021, 71763025, and 61861025]; the Natural Science Foundation of Gansu Province [grant number 23JRRA886]; and the Gansu Provincial Department of Education: Industrial Support Plan Project [grant number 2023CYZC-35].

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] M. Jabari, A. Botta, L. Tagliavini, C. Visconte, G. Quaglia, A Safe, high-precision reinforcement learning-based optimal control of surgical continuum robots: A monotone tube boundary approach with prescribed-time control capability, Robot. Auton. Syst. (2025) 104992, http://dx.doi.org/10.1016/j.robot.2025.104992.

[2] F. Espinosa, C. Santos, J. Sierra-García, Multi-AGV transport of a load: state of art ancdentralized proposa, Rev. Iberoam. Autom. Inform. Ind. 18 (1) (2021) 82–91, http://dx.doi.org/10.4995/riai.2020.12846.

[3] R. Berenstein, Y. Edan, Human-robot collaborative site-specific sprayer, J. Field Robot. 34 (2017) 1519–1530, http://dx.doi.org/10.1002/rob.21730.

[4] B. Madhevan, R. Sakkaravarthi, M. Singh, R. Diya, D. Jha, Modelling, simulation and mechatronics design of a wireless automatic fire fighting surveillance robot, Def. Sci. J. 67 (5) (2017) 572–580, http://dx.doi.org/10.14429/dsj.67.10237.

[5] S. Wan, Z. Gu, Q. Ni, Cognitive computing and wireless communications on the edge for healthcare service robots, Comput. Commun. 149 (2020) 99–106, http://dx.doi.org/10.1016/j.comcom.2019.10.012.

[6] G. Yang, Y. Liu, L. Guan, Design and simulation optimization of obstacle avoidance system for planetary exploration mobile robots, J. Phys. Conf. Ser. 1176 (2019) 032038, http://dx.doi.org/10.1088/1742-6596/1176/3/032038.

[7] M. Rafeeq, S. Toha, S. Ahmad, M. Razib, Locomotion strategies for amphibious robots-a review, IEEE Access 9 (2021) 26323–26342, http://dx.doi.org/10.1109/ACCESS.2021.3057406.

[8] W. Xue, B. Zhou, F. Chen, E. Ghaderpour, A. Mohammadzadeh, Design of a robust intelligent controller based neural network for trajectory tracking of high-speed wheeled robots, Phys. Scr. 99 (11) (2024) 115023, http://dx.doi.org/10.1088/1402-4896/ad8486.

[9] L. An, X. Huang, P. Yang, Z. Liu, Adaptive bézier curve-based path following control for autonomous driving robots, Robot. Auton. Syst. 189 (2025) 104969, http://dx.doi.org/10.1016/j.robot.2025.104969.

[10] Y. Hu, W. Zhou, Y. Liu, M. Zeng, W. Ding, S. Li, Efficient online planning and robust optimal control for nonholonomic mobile robot in unstructured environments, IEEE Trans. Emerg. Top. Comput. Intell. 8 (5) (2024) 3559–3575, http://dx.doi.org/10.1109/TETCI.2024.3424527.

[11] B. Chai, K. Zhang, M. Tan, J. Wang, An optimal robust trajectory tracking strategy for the wheeled mobile robot, Int. J. Control. Autom. Syst. 22 (2024) 1050–1065, http://dx.doi.org/10.1007/s12555-022-0902-1.

[12] L. Zhao, G. Wang, X. Fan, Y. Li, The analysis of trajectory control of non-holonomic mobile robots based on internet of things target image enhancement technology and backpropagation neural network, Front. Neurorobotics 15 (2021) 634340, http://dx.doi.org/10.3389/fnbot.2021.634340.

[13] Y. Cao, B. Liu, J. Pu, Robust control for a tracked mobile robot based on a finite-time convergence zeroing neural network, Front. Neurorobotics 17 (2023) 1242063, http://dx.doi.org/10.3389/fnbot.2023.1242063.

[14] H. Cen, B. Singh, Nonholonomic wheeled mobile robot trajectory tracking control based on improved sliding mode variable structure, Wirel. Commun. Mob. Comput. 2021 (2021) 9, http://dx.doi.org/10.1155/2021/2974839.

[15] Y. Chen, Y. Chen, Trajectory tracking design for a swarm of autonomous mobile robots: a nonlinear adaptive optimal approach, Math. 10 (20) (2022) 3901, http://dx.doi.org/10.3390/math10203901.

[16] Z. Shao, J. Zhang, Vision-based adaptive trajectory tracking control of wheeled mobile robot with unknown translational external parameters, IEEE/ASME Trans. Mechatronics 29 (1) (2024) 358–365, http://dx.doi.org/10.1109/TMECH.2023.3278027.

[17] P. Shu, M. Oya, J. Zhao, A new adaptive tracking control scheme of wheeled mobile robot without longitudinal velocity measurement, Internat. J. Robust Nonlinear Control 28 (5) (2018) 1789–1807, http://dx.doi.org/10.1002/rnc.3985.

[18] L. Bossi, C. Rottenbacher, G. Mimmi, L. Magni, Multivariable predictive control for vibrating structures: An application, Control Eng. Pract. 19 (10) (2011) 1087–1098, http://dx.doi.org/10.1016/j.conengprac.2011.05.003.

[19] H. Yang, M. Guo, Y. Xia, L. Cheng, Trajectory tracking for wheeled mobile robots via model predictive control with softening constraints, IET Control Theory Appl. 12 (2) (2018) 206–214, http://dx.doi.org/10.1049/iet-cta.2017.0395.

[20] Y. You, Z. Yang, H. Zhuo, Y. Sui, Research on ground mobile robot trajectory tracking control based on MPC and ANFIS, Control Eng. Pract. 152 (2024) 106040, http://dx.doi.org/10.1016/j.conengprac.2024.106040.

[21] J. Tang, S. Wu, B. Lan, Y. Dong, Y. Jin, G. Tian, W. Zhang, L. Shi, GMPC: geometric model predictive control for wheeled mobile robot trajectory tracking, IEEE Robot. Autom. Lett. 9 (5) (2024) 4822–4829, http://dx.doi.org/10.1109/LRA.2024.3381088.

[22] M. Tang, Y. Zhang, W. Wang, B. An, Y. Yan, Yaw stability control of unmanned emergency supplies transportation vehicle considering two-layer model predictive control, Actuators 13 (3) (2024) 103, http://dx.doi.org/10.3390/act13030103.

[23] M. Baglioni, A. Jamshidnejad, A novel MPC formulation for dynamic target tracking with increased area coverage for search-and-rescue robots, J. Intell. Robot. Syst. 110 (4) (2024) 140, http://dx.doi.org/10.1007/s10846-024-02167-3.

[24] B. Vicente, S. James, S. Anderson, Linear system identification versus physical modeling of lateral–longitudinal vehicle dynamics, IEEE Trans. Control Syst. Technol. 29 (3) (2021) 1380–1387, http://dx.doi.org/10.1109/tcst.2020.2994120.

[25] C. Ho, K. Ahn, Design of an adaptive fuzzy observer-based fault tolerant controller for pneumatic active suspension with displacement constraint, IEEE Access 9 (2021) 136346–136359, http://dx.doi.org/10.1109/ACCESS.2021.3115909.

[26] K. B.O., Hamiltonian systems and transformation in hilbert space, Proc. Natl. Acad. Sci. USA 17 (5) (1931) 315–318, http://dx.doi.org/10.1073/pnas.17.5.315.

[27] M. Haseli, J. Cortés, Learning koopman eigenfunctions and invariant subspaces from data: symmetric subspace decomposition, IEEE Trans. Autom. Control 67 (7) (2022) 3442–3457, http://dx.doi.org/10.1109/TAC.2021.3105318.

[28] J. Kim, Y. Quan, C. Chung, W. Choi, K-SMPC: Koopman operator-based stochastic model predictive control for enhanced lateral control of autonomous vehicles, IEEE Access 13 (2025) 13944–13958, http://dx.doi.org/10.1109/ACCESS.2025.3530984.

[29] Z. Li, M. Han, D. Vo, X. Yin, Machine learning-based input-augmented Koopman modeling and predictive control of nonlinear processes, Comput. Chem. Eng. 191 (2024) 108854, http://dx.doi.org/10.1016/j.compchemeng.2024.108854.

[30] I. Hameed, L. Abbud, J. Abdulsaheb, A. Azar, M. Mezher, A. Jawad, W. Abdul-Adheem, I. Ibraheem, N. Kamal, A new nonlinear dynamic speed controller for a differential drive mobile robot, Entropy 25 (3) (2023) 514, http://dx.doi.org/10.3390/e25030514.

[31] J. Rodríguez-Arellano, R. Miranda-Colorado, L. Aguilar, M. Negrete-Villanueva, Trajectory tracking nonlinear hoo controller for wheeled mobile robots with disturbances observer, ISA Trans. 142 (2023) 372–385, http://dx.doi.org/10.1016/j.isatra.2023.07.037.

[32] H. Wu, S. Wang, Y. Xie, H. Li, S. Zheng, L. Jiang, Adaptive abrupt disturbance rejection tracking control for wheeled mobile robots, IEEE Robot. Autom. Lett. 9 (9) (2024) 7787–7794, http://dx.doi.org/10.1109/LRA.2024.3432353.

[33] S. Le Cleac'h, T. Howell, S. Yang, C. Lee, J. Zhang, A. Bishop, M. Schwager, Z. Manchester, Fast contact-implicit model-predictive control, IEEE Trans. Robot. 40 (2024) 1617–1629, http://dx.doi.org/10.1109/TRO.2024.3351554.

[34] J. Cenerini, M. Mehrez, J. Han, S. Jeon, W. Melek, Model predictive path following control without terminal constraints for holonomic mobile robots, Control Eng. Pract. 132 (2023) 105406, http://dx.doi.org/10.1016/j.conengprac.2022.105406.

[35] S. Heshmati-alamdari, A. Eqtami, G. Karras, D. Dimarogonas, K. Kyriakopoulos, A self-triggered position based visual servoing model predictive control scheme for underwater robotic vehicles, Mach. 8 (2) (2020) 33, http://dx.doi.org/10.3390/machines8020033.

[36] Q. Cao, Z. Sun, Y. Xia, L. Dai, Self-triggered MPC for trajectory tracking of unicycle-type robots with external disturbance, J. Franklin Inst. 356 (11) (2019) 5593–5610, http://dx.doi.org/10.1016/j.jfranklin.2019.03.015.

[37] M. Williams, I. Kevrekidis, C. Rowley, A data-driven approximation of the Koopman operator: extending dynamic mode decomposition, J. Nonlinear Sci. 25 (6) (2015) 1307–1346, http://dx.doi.org/10.1007/s00332-015-9258-5.

[38] J. Han, From PID to active disturbance rejection control, IEEE Trans. Ind. Electron. 56 (3) (2009) 900–906, http://dx.doi.org/10.1109/TIE.2008.2011621.

[39] A. Liu, R. Zhang, L. Yu, W. Zhang, Extended state observer-based decentralized model predictive control, Control Decis. 31 (6) (2016) 1093–1098, http://dx.doi.org/10.13195/j.kzyjc.2015.0542.

[40] H. Huang, D. He, L. Yu, Robust nonlinear predictive control based on polytopic description systems, Acta Automat. Sinica 38 (12) (2012) 1906–1912, note=.

[41] Z. Sun, Y. Xia, L. Dai, P. Campoy, Tracking of unicycle robots using event-based MPC with adaptive prediction horizon, IEEE/ASME Trans. Mechatronics 25 (2) (2020) 739–749, http://dx.doi.org/10.1109/TMECH.2019.2962099.

[42] Y. Zhang, M. Tang, H. Zhang, B. An, Y. Yan, W. Wang, K. Tang, Emergency supplies transportation robot trajectory tracking control based on koopman and improved event-triggered model predictive control, Internat. J. Robust Nonlinear Control 34 (13) (2024) 9089–9111, http://dx.doi.org/10.1002/rnc.7449.

[43] J. Yoo, A. Molin, M. Jafarian, H. Esen, D. Dimarogonas, K. Johansson, Event-triggered model predictive control with machine learning for compensation of model uncertainties, in: 2017 IEEE 56th Annual Conference on Decision and Control, Melbourne, Australia, 2017, pp. 5463–5468, http://dx.doi.org/10.1109/CDC.2017.8264468.

Minan Tang received his Ph.D. degree in transportation information engineering and control from Lanzhou Jiaotong University in 2011, China. He gained his Master's degree in electronic information engineering from Lanzhou Jiaotong University in 2006, China. He was a postdoctoral researcher in intelligent transportation control of Lanzhou University of Technology, China, in 2012. He is currently a professor and doctoral supervisor of the school of automation and electrical engineering of Lanzhou Jiaotong University, China. His research interests are complex system modeling and control, intelligent transportation system, process control system,

clean energy generation system and grid connection and power supply quality control of power system. Professional e-mail address: tangminan@mail.lzjtu.cn.

Yaqi Zhang received her Master's degree in transportation from Lanzhou Jiaotong University, China, in 2024. She gained her Bachelor's degree from Shandong Jiaotong University, China, in 2021. She is currently pursuing the Ph.D. degree in traffic information engineering and control at Lanzhou Jiaotong University, China. Her research interests include robot control, intelligent control, and complex industrial process control. Professional e-mail address: 12211534@stu.lzjtu.edu.cn.

Shuyou Yu received his Ph.D. degree in control theory and control engineering from Jilin University in 2010, China. He gained his Master's degree in control theory and control engineering from Jilin University in 2005, China. In 2011, he was an associate researcher at the University of Stuttgart in Germany. He is currently a professor and doctoral supervisor of the school of communication engineering of Jinlin University, China. His research interests are reinforcement learning and predictive control, human–computer augmentation, and human–computer collaborative games. Professional e-mail address: shuyou@jlu.edu.cn.

Jinping Li received her Bachelor's degree in electrical engineering and automation from Harbin University of Science and Technology, China, in 2020. She is currently pursuing the Ph.D. degree in traffic information engineering and control at Lanzhou Jiaotong University, China. Her research interests are clean energy generation system, intelligent control, and energy-transportation integration. Professional e-mail address: 11220936@stu.lzjtu.edu.cn.

Kunxi Tang received his Bachelor's degree in the School of Automation and Electrical Engineering at Lanzhou Jiaotong University, China, in 2022. He is currently pursuing the M.S. degree in transportation from Lanzhou Jiaotong University. His research interests include mobile robot control. Professional e-mail address: 12221583@stu.lzjtu.edu.cn.