Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 5573467, 13 pages
https://doi.org/10.1155/2021/5573467

Research Article

Design of a Model Predictive Trajectory Tracking Controller for Mobile Robot Based on the Event-Triggering Mechanism

Ning He,[1] **Lipeng Qi**,[1] **Ruoxia Li**,[2] **and Yuesheng Liu**[1]

[1]*School of Mechanical and Electrical Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China*
[2]*School of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China*

Correspondence should be addressed to Ruoxia Li; rli@xauat.edu.cn

Received 21 January 2021; Revised 7 March 2021; Accepted 20 March 2021; Published 12 April 2021

Academic Editor: Defeng He

A novel model predictive control- (MPC-) based trajectory tracking controller for mobile robot is proposed using the event-triggering mechanism, and the aim is to solve the problem that the MPC optimization problem requires a large amount of online computation and communication resources. This method includes two different event-triggering strategies, namely, the event-triggering based on threshold curve and the event-triggering based on threshold band. The selection of the triggering threshold is achieved by applying the statistical method to the historical data of the trajectory tracking of the mobile robot under the classic MPC method. Simulation and experimental tests illustrate that the proposed approach is able to significantly reduce the computation and communication burdens without affecting the control performance. Furthermore, the experimental results show that compared with the classic MPC-based tracking method, the proposed two event-triggering strategies can reduce 28.1% and 75.7% of the computation load and 0.886 s and 2.385 s communication time.

1. Introduction

With the rapid development of computer engineering, electronic engineering, network communication engineering, and other disciplines, mobile robot technology has made great progress and has been widely used in scientific exploration, national defense and military, risk relief, life services, and other fields. Therefore, the trajectory tracking method of mobile robot using a conventional control strategy is difficult to achieve the desired control performance, so it is urgent to study the trajectory tracking strategy of a robot based on advanced control algorithms.

In recent years, many research achievements have been made on the trajectory tracking control of mobile robots. The mainstream tracking control algorithms at this stage included sliding mode control, model predictive control (MPC), and optimal control [1–4]. Due to the advantages in handling system constraints and balancing multiple control objectives, MPC has been applied in more and more practical mobile robots, such as wheeled robots [5–9] and underwater autonomous navigation robots [10, 11]. In [8], the authors designed a MPC-based trajectory tracking control method, which could ensure that the unmanned vehicle track the reference trajectory quickly and stably; the distance error and heading error are in a reasonable range, and the real-time performance meet the requirements. [9] proposed a real-time optimization scheme to reduce the control horizon and the control update frequency for MPC-based robot path tracking control, which could balance the real-time requirements and control accuracy. An MPC-based path tracking controller for the constrained under-driven autonomous underwater vehicle (AUV) system is designed in [10], and it is pointed out that a long prediction horizon could be chosen to ensure the final convergence of the control system. In addition, [11] proposed a new MPC method for the trajectory tracking of underwater robots, which used the linearized error model to formulate the MPC optimization problem such that the speed jump problem could be avoided without violating the control constraints.

As MPC controllers solve a constrained optimization problem at each sampling instant, the online computation burden is quite large and needs to be further optimized. In [12], the authors proposed a MPC strategy based on the event-

triggering mechanism, in which the optimization is only triggered when the difference between the predicted and the actual trajectories exceeded a certain threshold, which effectively reduced the online computation. [13] proposed an event-based MPC method and further proved the closed-loop stability property by constructing terminal constraint and terminal cost function. In [14], the authors proposed a novel aperiodic adaptive event-triggered communication mechanism for master-slave synchronization problem with aperiodic sampled data, which can effectively reduce the transmission load. For more details regarding communication links in tracking control, the reader is referred to [15] and references therein. [16] proposed an event-based MPC approach for linear discrete systems subject to external perturbation, in which the optimization problem included a time-vary tightened state constraint. It should be emphasized that compared with the standard MPC applied in practical robots, the above event-triggered MPC algorithms including additional items such as terminal and tightened constraints, which cannot be directly adopted, and therefore, the event-triggering mechanism should be further studied for MPC-based tracking control of robot systems.

Aiming at solving the abovementioned problems, this study proposes a more practical event-triggered MPC-based trajectory tracking method for mobile robot subject to external disturbances. Two novel event-triggering strategies, i.e., event-triggering based on the threshold curve and event-triggering based on the threshold band are developed. Furthermore, the selection of the threshold curve and threshold band is achieved via applying the statistical approach to the recorded historical trajectory data of the robot system with the classic MPC method. Finally, simulation and experimental tests show that the developed event-triggered MPC method can significantly reduce the computation and communication resources occupied by the MPC controller.

It is worth mentioning that the main motivation of this manuscript is that in order to facilitate the theoretical analysis, the majority of the existing event-triggered MPC methods have extra assumptions about the robot system and require additional items (e.g., terminal cost) in the MPC setting, which makes the corresponding MPC controller too computational expensive for many real robot systems. Since robot systems with relatively low hardware configuration are an important portion of industrial robots, it is of great importance to develop the event-triggered MPC method suitable for those robot systems. Furthermore, the main contribution of the proposed method can be illustrated in two aspects. On the one hand, compared with the event-triggered MPC with terminal cost/constraint, the proposed method employed the standard robot kinematic model with no extra assumption and has no additional terms in the MPC cost function, which can be directly applied to real robot systems. On the other hand, compared with event-triggered MPC with standard MPC framework, the proposed technique incorporated the external disturbances into the development of the triggering condition and adopted statistical methods to construct two different kinds of triggering strategies for performance improvement (Section 5.2).

2. Description of the Robot System

2.1. Kinematic Model of the Robot System. First, the kinematic model of the mobile robot is constructed. It is assumed that the robot system conforms to nonholonomic constraints, and the robot body has no lateral sliding. Considering that the velocity of the robot is generally low and the lateral acceleration such as centrifugal acceleration has little influence during steering [17], the kinematic model for the mobile robot can be established as

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} = \begin{bmatrix} \cos\varphi & 0 \\ \sin\varphi & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{1}$$

where the state is denoted as $[\, x \;\; y \;\; \varphi \,]^T$, consisting of the position of the mobile robot $[\, x \;\; y \,]$, as well as its orientation φ. The input signal is represented as $[\, v \;\; \omega \,]^T$, including linear velocity and angular velocity of the mobile robot.

The above kinematic model can then be expressed in a more general state space form as

$$\begin{aligned} &\dot{\xi} = f(\boldsymbol{\xi}, \mathbf{u}), \\ &\boldsymbol{\xi} = (x, y, \varphi)^T, \mathbf{u} = (v, \omega)^T, \end{aligned} \tag{2}$$

where $\boldsymbol{\xi}$ is the state, and $\mathbf{u}$ is the input; $f(\cdot)$ represents the corresponding mapping relationship. Since the trajectory status of the reference system is known, the relationship of the state vector and control vector of the reference system is expressed as

$$\begin{aligned} &\dot{\xi}_r = f(\boldsymbol{\xi}_r, \mathbf{u}_r), \\ &\boldsymbol{\xi}_r = (x_r, y_r, \varphi_r)^T, \mathbf{u}_r = (v_r, \omega_r)^T. \end{aligned} \tag{3}$$

Expand equation (2) by Taylor series at any reference point (ξ_r, u_r) and retain only the first-order terms and ignore the higher-order terms. The following equation can be obtained:

$$\dot{\xi} = f(\boldsymbol{\xi}_r, \mathbf{u}_r) + \frac{\partial f}{\partial \boldsymbol{\xi}}\Big|_{\substack{\boldsymbol{\xi} = \boldsymbol{\xi}_r \\ \mathbf{u} = \mathbf{u}_r}} (\boldsymbol{\xi} - \boldsymbol{\xi}_r) + \frac{\partial f}{\partial \mathbf{u}}\Big|_{\substack{\boldsymbol{\xi} = \boldsymbol{\xi}_r \\ \mathbf{u} = \mathbf{u}_r}} (\mathbf{u} - \mathbf{u}_r). \tag{4}$$

By making a difference between equations (3) and (4), the linearized error model is obtained as

$$\dot{\tilde{\xi}} = \begin{bmatrix} \dot{x} - \dot{x}_r \\ \dot{y} - \dot{y}_r \\ \dot{\varphi} - \dot{\varphi}_r \end{bmatrix} = \begin{bmatrix} 0 & 0 & -v_r \sin\varphi_r \\ 0 & 0 & v_r \cos\varphi_r \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x - x_r \\ y - y_r \\ \varphi - \varphi_r \end{bmatrix} + \begin{bmatrix} \sin\varphi_r & 0 \\ \cos\varphi_r & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v - v_r \\ \omega - \omega_r \end{bmatrix}. \tag{5}$$

The linearized error model of the mobile robot is further discretized by Euler method as

$$\overline{\xi}(k+1|t) = \mathbf{A}_k \overline{\xi}(k) + \mathbf{B}_k \overline{u}(k), \tag{6}$$

where $\mathbf{A}_k = \begin{bmatrix} 1 & 0 & -Tv_r \sin\varphi_r \\ 0 & 1 & Tv_r \cos\varphi_r \\ 0 & 0 & 1 \end{bmatrix}$, $\mathbf{B}_k = \begin{bmatrix} T\sin\varphi_r & 0 \\ T\cos\varphi_r & 0 \\ 0 & T \end{bmatrix}$, and T is the sampling time.

2.2. Augmented State Space Model. To ease the development of the model predictive trajectory tracking controller, an augmented state space model is constructed. Given the discrete time linear model of the mobile robot in (6), an augmented state is set as

$$\boldsymbol{\mu}(k|t) = \begin{bmatrix} \overline{\xi}(k|t) \\ \overline{u}(k-1|t) \end{bmatrix}. \tag{7}$$

Thus, the desired discrete state space equation is obtained:

$$\begin{cases} \boldsymbol{\mu}(k+1|t) = \widetilde{A}_{k,t}\boldsymbol{\mu}(k|t) + \widetilde{B}_{k,t}\Delta\mathbf{U}(k|t), \\ \boldsymbol{\eta}(k|t) = \widetilde{C}_{k,t}\boldsymbol{\mu}(k|t), \end{cases} \tag{8}$$

where $\widetilde{A}_{k,t} = \begin{bmatrix} \mathbf{A}_k & \mathbf{B}_k; & 0_{m\times n} & \mathbf{I}_m \end{bmatrix}$ is the system matrix, $\widetilde{B}_{k,t} = \begin{bmatrix} \mathbf{B}_k; & \mathbf{I}_m \end{bmatrix}$ is the control matrix, and $\widetilde{C}_{k,t} = \begin{bmatrix} \mathbf{I}_n & 0 \end{bmatrix}$ is the output matrix, and m, n are the dimensions of the state and control vectors. To simplify the representation, the following expressions are employed:

$$\begin{aligned} \widetilde{A}_{k,t} &= \widetilde{A}_t, \quad k = 1, \ldots, t + N_h - 1, \\ \widetilde{B}_{k,t} &= \widetilde{B}_t, \quad k = 1, \ldots, t + N_h - 1, \\ \widetilde{C}_{k,t} &= \widetilde{C}_t, \quad k = 1, \ldots, t + N_h - 1, \end{aligned} \tag{9}$$

where N_h is the horizon (including prediction horizon and control horizon).

3. Design of a Model Predictive Trajectory Tracking Controller

The objective function of the predictive controller contains information such as system state error and the change of control, based on which the optimal control problem can be formulated to ensure the mobile robot to track the reference trajectory quickly and smoothly. In this work, the cost function is formulated as

$$\begin{aligned} J(\boldsymbol{\xi}(t), \Delta\mathbf{U}(t)) = &\sum_{i=1}^{N_p} \left\| \boldsymbol{\eta}(t+i|t) - \boldsymbol{\eta}_{\text{ref}}(t+i|t) \right\|_{\mathbf{Q}}^2 \\ &+ \sum_{i=1}^{N_c-1} \left\| \Delta\mathbf{u}(t+k) \right\|_{\mathbf{R}}^2 + \alpha\varepsilon^2. \end{aligned} \tag{10}$$

where $\mathbf{Q}$ and $\mathbf{R}$ denote the state-weighting matrix and input-weighting matrix, respectively, ε is the relaxing factor, and α is the associated weighting coefficient. In this objective function, the first element shows the capability of the system to track the set point, and the second element illustrates its ability to constrain the change of the control input. The cost function (10) determines the closed-loop performance of the MPC system. More specifically, the dynamic and steady-state performances as well as the energy consumption of the robot system can be tuned by the design parameters (such as prediction horizon N_p, control horizon N_c, and weighting matrices $\mathbf{Q}$ and $\mathbf{R}$) in (10). The prediction horizon N_p mainly affects the stability and rapidity of the system. When N_p is small, the rapidity of the system response is guaranteed, but the robustness of the system is relatively poor. Therefore, the choice of N_p mainly involves the trade-off between robust stability and rapidity of the system. The control horizon N_c mainly affects the dynamic characteristics of the system. The smaller the N_c is selected, the worse the tracking performance of the system will be. A larger N_c makes the control signal more flexible and improves the overall dynamic characteristics of the system, but it is also prone to robust stability problems. The weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ restrict each other. The rapid response performance of the system can be improved by reducing $\mathbf{R}$ (or increasing $\mathbf{Q}$), and the steady-state performance and the energy consumption can be improved via increasing $\mathbf{R}$ (or reducing $\mathbf{Q}$), since the change of the control signal can be effectively suppressed and the control cost can be reduced. The relaxing factor and its weighting coefficient can be tuned to guarantee that MPC admits an optimal input trajectory at each sampling instant, which indirectly guarantees the feasibility of the optimal problem.

The prediction of the system output $\mathbf{Y}(t)$ during the prediction horizon is obtained:

$$\mathbf{Y}(t) = \boldsymbol{\Psi}_t \boldsymbol{\mu}(k|t) + \boldsymbol{\Theta}_t \Delta\mathbf{U}(t), \tag{11}$$

where $\mathbf{Y}(t) = \begin{bmatrix} \eta(t+1|t) \\ \eta(t+2|t) \\ \vdots \\ \eta(t+N_c|t) \\ \vdots \\ \eta(t+N_p|t) \end{bmatrix}$, $\boldsymbol{\Psi}_t = \begin{bmatrix} \widetilde{C}_t\widetilde{A}_t \\ \widetilde{C}_t\widetilde{A}_t^2 \\ \vdots \\ \widetilde{C}_t\widetilde{A}_t^{N_c} \\ \vdots \\ \widetilde{C}_t\widetilde{A}_t^{N_p} \end{bmatrix}$, $\Delta\mathbf{U}(t) = \begin{bmatrix} \Delta\mathbf{u}(t|t) \\ \Delta\mathbf{u}(t+1|t) \\ \vdots \\ \Delta\mathbf{u}(t+N_c|t) \end{bmatrix}$, and $\Theta_t = \begin{bmatrix} \widetilde{C}_t\widetilde{B}_t & 0 & 0 & 0 \\ \widetilde{C}_t\widetilde{A}_t\widetilde{B}_t & \widetilde{C}_t\widetilde{B}_t & 0 & 0 \\ \cdots & \cdots & \ddots & \cdots \\ \widetilde{C}_t\widetilde{A}_t^{N_c-1}\widetilde{B}_t & \widetilde{C}_t\widetilde{A}_t^{N_c-2}\widetilde{B}_t & \cdots & \widetilde{C}_t\widetilde{B}_t \\ \widetilde{C}_t\widetilde{A}_t^{N_c}\widetilde{B}_t & \widetilde{C}_t\widetilde{A}_t^{N_c-1}\widetilde{B}_t & \cdots & \widetilde{C}_t\widetilde{A}_t\widetilde{B}_t \\ \vdots & \vdots & \ddots & \vdots \\ \widetilde{C}_t\widetilde{A}_t^{N_p-1}\widetilde{B}_t & \widetilde{C}_t\widetilde{A}_t^{N_p-2}\widetilde{B}_t & \cdots & \widetilde{C}_t\widetilde{A}_t^{N_p-N_c-1}\widetilde{B}_t \end{bmatrix}$.

By substituting equation (11) into the cost function in (10), the output deviation in the prediction horizon can be denoted as

$$\mathbf{E}(t) = \boldsymbol{\Psi}_t \overline{\xi}(t|t) - \mathbf{Y}_{\text{ref}}(t), \tag{12}$$

where $\mathbf{Y}_{\text{ref}} = [\eta_{\text{ref}}(t+1|t), \ldots, \eta_{\text{ref}}(t+N_p|t)]^T$. Through some standard matrix operations, the cost function can be adjusted as

$$J(\boldsymbol{\xi}(t), \mathbf{u}(t-1), \Delta\mathbf{U}(t)) = \left[\Delta\mathbf{U}(t)^T, \varepsilon\right]^T \mathbf{H}_t \left[\Delta\mathbf{U}(t)^T, \varepsilon\right] + \mathbf{G}_t \left[\Delta\mathbf{U}(t)^T, \varepsilon\right] + \mathbf{P}_t, \tag{13}$$

where $\mathbf{H}_t = \begin{bmatrix} \Theta_t^T \mathbf{Q} \Theta_t + \mathbf{R} & 0 \\ 0 & \alpha \end{bmatrix}$, $\mathbf{G}_t = \begin{bmatrix} 2\mathbf{E}_t^T \mathbf{Q} \Theta_t & 0 \end{bmatrix}$, and $\mathbf{P}_t = \mathbf{E}(t)^T \mathbf{Q} \mathbf{E}(t)$.

Then, it can be shown that the constrained optimization problem of the predictive controller at each step is equivalent to the quadratic programming (QP) problem as

$$\begin{aligned} &\min_{\Delta\mathbf{U}_t, \varepsilon} \left[\Delta\mathbf{U}(t)^T, \varepsilon\right]^T \mathbf{H}_t \left[\Delta\mathbf{U}(t)^T, \varepsilon\right] + \mathbf{G}_t \left[\Delta\mathbf{U}(t)^T, \varepsilon\right] \\ &\text{subject to} \quad \begin{aligned} &\Delta\mathbf{U}_{\min} \le \Delta\mathbf{U}(k) \le \Delta\mathbf{U}_{\max} \\ &\mathbf{U}_{\min} \le \mathbf{u}(t-1) + \sum_{i=1}^{k} \Delta\mathbf{U}(i) \le \mathbf{U}_{\max} \\ &\mathbf{Y}_{\min} - \varepsilon \le \boldsymbol{\Psi}_t \boldsymbol{\mu}(k|t) + \boldsymbol{\Theta}_t \Delta\mathbf{U}(t) \le \mathbf{Y}_{\max} - \varepsilon \\ &k = t, \ldots, t + N_c - 1, \varepsilon > 0, \end{aligned} \end{aligned} \tag{14}$$

where $\Delta\mathbf{U}_{\min}$ and $\Delta\mathbf{U}_{\max}$ are the system control incremental sequence constraints, $\mathbf{U}_{\min}$ and $\mathbf{U}_{\max}$ are the system control sequence constraints, and $\mathbf{Y}_{\min}$ and $\mathbf{Y}_{\max}$ are the system output sequence constraints.

By solving problem (14) at a given sampling instant, the incremental control sequence $\Delta\mathbf{U}_t^* = [\Delta\mathbf{u}_t^*, \Delta\mathbf{u}_{t+1}^*, \ldots, \Delta\mathbf{u}_{t+N_c-1}^*]^T$ can be obtained. The first item in the sequence is applied as the actual control signal increment, i.e., $\mathbf{u}(t) = \mathbf{u}(t-1) + \Delta\mathbf{u}_t^*$. Then, the newly obtained state is utilized to update the optimization problem for the next instant. The aforementioned procedure is recursively performed, until the control process is completed. Besides, the convergence speed of the proposed MPC control algorithm can be controlled by the tunable parameters in the MPC cost function, namely, prediction horizon N_p, control horizon N_c, and weighting matrices $\mathbf{Q}$ and $\mathbf{R}$. In general, when convergence speed of the system response needs to be improved, a larger N_c and a smaller N_p are usually selected, and $\mathbf{R}$ is reduced (or $\mathbf{Q}$ is increased). It should be noted that in the physical sense, $N_c <= N_p$ needs to be satisfied, and the functions of the weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ are mutually restricted.

4. Design of Event-Triggering Strategies

Compared with classic time-triggered MPC, event-triggered MPC has the characteristic of only performing actions at the time when a predetermined event occurs, such as the error exceeds a certain threshold, and thus, it can effectively reduce the frequency of system sampling and update, which could effectively reduce computation and communication load.

Due to the inevitable existence of various disturbances in actual mobile robot systems, such as model mismatch and parameter perturbation, the design of triggering conditions should further consider the external disturbances of the considered system. Denote the disturbance signal as $w \in W$ with W being a compact set. Then, on the basis of the nominal model in equation (1), a perturbed model of the robot system can be described as $\dot{\xi} = f(\xi, \mathbf{u}) + \rho$, where $\rho \triangleq \sup_{w \in W} \|w(t)\|$ denotes the upper bound of disturbance.

It is worth nothing that although the external disturbances can be handled via incorporating terminal cost function, terminal constraint, and tightening set to the MPC algorithm in (10); these additional items will significantly increase the amount of online computation and seriously affect the real-time property and performance of the controller. For the robot system with limited computation ability, such modified MPC optimization problem may not be solved in time to update the control signals, which will lead to system performance deterioration and even instability. To solve the abovementioned problems, this study proposes an event-triggered MPC-based trajectory tracking method for the robot systems, which can effectively deal with the influence of external disturbances on the system without increasing the amount of online computation.

4.1. Event-Triggering Strategy Based on Threshold Curve. A triggering strategy based on the threshold curve is proposed in this section. More specifically, at each sampling time, if any component of the position coordinate of robot exceeds the corresponding component of the threshold curve vector, the MPC update is executed, based on which the next triggering time can defined as

$$\begin{aligned} \overline{t}_{k+1} \triangleq \inf_{s > t_k} \{ &s | \boldsymbol{\xi}_1(s|t_k) - \boldsymbol{\sigma}_x(s|t_k) > 0 \text{ or } \boldsymbol{\xi}_2(s|t_k) \\ &- \boldsymbol{\sigma}_y(s|t_k) > 0 \text{ or } \boldsymbol{\xi}_3(s|t_k) - \boldsymbol{\sigma}_\varphi(s|t_k) > 0 \}, \\ &s \in [t_k, t_k + N_c], \end{aligned} \tag{15}$$

where t_k is the current triggering time, $\boldsymbol{\xi}_1$, $\boldsymbol{\xi}_2$, and $\boldsymbol{\xi}_3$ are the values of position components of the systems x, y, and φ. And $\boldsymbol{\sigma}_x$, $\boldsymbol{\sigma}_y$, and $\boldsymbol{\sigma}_\varphi$ denote the event-triggering threshold corresponding to the state components, respectively.

Note that the threshold curve parameters $\boldsymbol{\sigma}_x$, $\boldsymbol{\sigma}_y$, and $\boldsymbol{\sigma}_\varphi$ can be selected via taking the mean values of the position coordinates of the robot from N groups of historical data collected during the similar working condition under the classic MPC, and thus, the three thresholds at each sampling time can be obtained through statistical processing via

$$\begin{cases} \boldsymbol{\sigma}_x(1, \ldots, k) = \dfrac{\sum_{i=1}^{N} x_{i,k}}{N}, \\ \boldsymbol{\sigma}_y(1, \ldots, k) = \dfrac{\sum_{i=1}^{N} y_{i,k}}{N}, \\ \boldsymbol{\sigma}_\varphi(1, \ldots, k) = \dfrac{\sum_{i=1}^{N} \varphi_{i,k}}{N}, \end{cases} \tag{16}$$

where $x_{i,k}$, $y_{i,k}$, and $\varphi_{i,k}$ are the position and the orientation information of the mobile robot in i^{th} group historical data, k is the k^{th} sampling time; $\boldsymbol{\sigma}_x(k)$, $\boldsymbol{\sigma}_y(k)$, and $\boldsymbol{\sigma}_\varphi(k)$ are the obtained thresholds corresponding to the position and orientation of the robot at sampling time k, respectively.

4.2. Event-Triggering Strategy Based on Threshold Band. To further reduce the frequency of solving the optimization problem and save computation resources, the second event-triggering strategy, which is based on threshold band, is proposed in this section. More specifically, at some sampling time, when the position coordinate of any state component of the mobile robot exceeds the upper or lower bound of the threshold band, it is considered to be the case where the triggering condition is satisfied. To formulate the event-triggering strategy, the triggering instant $\overline{t}_{k+1}$ is defined as

$$\begin{aligned}\overline{t}_{k+1} \triangleq \inf_{s>t_k}\{&s|\boldsymbol{\xi}_1(s|t_k) - \boldsymbol{\sigma}_{x\_u}(s|t_k) > 0\\ &\cdot \text{or}\, \boldsymbol{\xi}_1(s|t_k) - \boldsymbol{\sigma}_{x\_d}(s|t_k) < 0\\ &\cdot \text{or}\, \boldsymbol{\xi}_2(s|t_k) - \boldsymbol{\sigma}_{y\_u}(s|t_k) > 0\\ &\cdot \text{or}\, \boldsymbol{\xi}_2(s|t_k) - \boldsymbol{\sigma}_{y\_d}(s|t_k) < 0\\ &\cdot \text{or}\, \boldsymbol{\xi}_3(s|t_k) - \boldsymbol{\sigma}_{\varphi\_u}(s|t_k) > 0\\ &\cdot \text{or}\, \boldsymbol{\xi}_3(s|t_k) - \boldsymbol{\sigma}_{\varphi\_d}(s|t_k) < 0\},\quad s \in [t_k, t_k + N_c],\end{aligned} \tag{17}$$

where $\boldsymbol{\sigma}_{x\_u}$, $\boldsymbol{\sigma}_{x\_d}$, $\boldsymbol{\sigma}_{y\_u}$, $\boldsymbol{\sigma}_{y\_d}$, $\boldsymbol{\sigma}_{\varphi\_u}$, and $\boldsymbol{\sigma}_{\varphi\_d}$ denote the upper and lower threshold bounds of the position and orientation.

The selection method of the threshold band is that under the condition that the control effect does not decline; the threshold band can be formed by using the relationship between the threshold curve and the maximum disturbance. In actual working conditions, due to the existence of uncertain factors such as measurement noise, load changes, and external disturbances of the system, it is difficult to obtain accurate values of the parameters of the mobile robot model. Based on this, the model mismatch and parameter perturbation of the system are usually treated as a total disturbance.

According to a large number of historical data, the disturbance upper bound of the usual working scene can be measured, and a threshold band can be obtained by subtracting the disturbance upper bound from the threshold curve. Using the threshold band as the triggering condition can improve the robustness of the system, and the threshold band can be obtained as

$$\begin{cases}\boldsymbol{\sigma}_{x\_u}(1,\ldots,k) = \boldsymbol{\sigma}_x(1,\ldots,k) + \rho,\\ \boldsymbol{\sigma}_{x\_d}(1,\ldots,k) = \boldsymbol{\sigma}_x(1,\ldots,k) - \rho,\\ \boldsymbol{\sigma}_{y\_u}(1,\ldots,k) = \boldsymbol{\sigma}_y(1,\ldots,k) + \rho,\\ \boldsymbol{\sigma}_{y\_d}(1,\ldots,k) = \boldsymbol{\sigma}_y(1,\ldots,k) - \rho,\\ \boldsymbol{\sigma}_{\varphi\_u}(1,\ldots,k) = \boldsymbol{\sigma}_\varphi(1,\ldots,k) + \rho,\\ \boldsymbol{\sigma}_{\varphi\_d}(1,\ldots,k) = \boldsymbol{\sigma}_\varphi(1,\ldots,k) - \rho,\end{cases} \tag{18}$$

where $\boldsymbol{\sigma}_{x\_u}(k)$, $\boldsymbol{\sigma}_{y\_u}(k)$, $\boldsymbol{\sigma}_{\varphi\_u}(k)$, $\boldsymbol{\sigma}_{x\_d}(k)$, $\boldsymbol{\sigma}_{y\_d}(k)$, and $\boldsymbol{\sigma}_{\varphi\_d}(k)$ are the upper and lower threshold limits corresponding to the position and orientation of the robot at sampling time k.

Remark 1. The method proposed in this study relies on the robot state information mainly in two aspects, namely, calculating the triggering threshold and determining the triggering instant; the former needs to process a large amount of historical complete state information, and the latter needs to know the current complete state information of the robot. However, in certain scenarios (such as linear trajectory tracking), the complete state information can be reduced to only state information containing the horizontal and vertical coordinates. In addition, we can also reduce the limit of requiring the complete state information by changing the triggering condition, e.g., developing a new condition with only the horizontal and vertical coordinates, but it may affect the control performance to some extent and needs further investigation.

4.3. Design of the Event-Triggered MPC Algorithm. Based on the event-triggering mechanism proposed in Sections 4.1 and 4.2, the event-triggered MPC algorithm is provided. Note that $N_p = N_c$ is utilized based on the practical experience. In the developed algorithm, the online optimization is executed only when the triggering condition is satisfied. To further lessen the times to solve the optimization problem, the last element in the control sequence is continuously used after the control horizon expires. Thus, the control law is constructed as

$$\Delta\mathbf{U}^*_{t_k} = \begin{cases}\left[\Delta\mathbf{u}^*_{t_k}, \Delta\mathbf{u}^*_{t_k+1}, \ldots, \Delta\mathbf{u}^*_{t_k+N_c-1}\right]^T, & t_k \le t \le t_k + N_c - 1,\\ \left[\Delta\mathbf{u}^*_{t_k+N_c-1}, \ldots, \Delta\mathbf{u}^*_{t_k+N_c-1}\right]^T, & t > t_k + N_c - 1.\end{cases} \tag{19}$$

The pseudocode of the proposed event-triggered MPC algorithm is given in Algorithm 1.

Remark 2. Note that the computational complexity is minor while applying the proposed event-triggering mechanism. The reason is that only a simple-triggering condition is checked at each sampling instant, and the associated computation is quite simple in the machine level implementation, as operands in the operation register of the instruction can be used directly. More specifically, the (classic) MPC controller needs to solve a nonlinear optimization problem at each sampling instant, and if the associated computational complexity is O(MIN_ITER) with MIN_ITER being the minimum number of iterations to find a solution to the nonlinear optimization, the computational complexity for the developed triggering mechanism is only O(1), and therefore, the complexity of the triggering mechanism is minor and would not add much computational burden to the robot system. Besides, introducing the event-triggering mechanism to the MPC can effectively reduce the number of the sampling and updating instants, and thus, the entire computation resources will actually be significantly saved compared with its time-triggered counterpart.

Interpretations of Algorithm 1:

(1) The initialization parameters of the algorithm include prediction horizon N_p, initial state $\xi_0 = [\, x_0 \;\; y_0 \;\; \varphi_0 \,]^T$, triggering condition level $\boldsymbol{\sigma}$ (including threshold curve and threshold band), and weighting matrices $\mathbf{Q}$ and $\mathbf{R}$

Input: Prediction horizon N_p; triggering level σ; weighting matrices $\mathbf{Q}$ and $\mathbf{R}$
initial state $\xi_0 = [\, x_0 \;\; y_0 \;\; \varphi_0 \,]^T$; indices $k = 0$, $i = 0$.
Calculate $\Delta \mathbf{U}^*_{t_k} = [\Delta \mathbf{u}^*_{t_k}, \Delta \mathbf{u}^*_{t_k+1}, \ldots, \Delta \mathbf{u}^*_{t_k+N_c-1}]^T$ by solving the problem in (14)
Apply control input $\mathbf{u}(i|t_k) = \Delta \mathbf{U}^*_{t_k}(i)$
while the triggering condition is not satisfied **do**
    Apply control input $\mathbf{u}(i+1|t_k) = \mathbf{u}(i|t_k) + \Delta \mathbf{U}^*_{t_k}(i+1)$
    $i = i + 1$
end while
$k = k + 1$, $i = 0$
Calculate $\Delta \mathbf{U}^*_{t_k} = [\Delta \mathbf{u}^*_{t_k}, \Delta \mathbf{u}^*_{t_k+1}, \ldots, \Delta \mathbf{u}^*_{t_k+N_c-1}]^T$ by solving the problem in (14)
return 3

Algorithm 1: Threshold curve/band-based event-triggered MPC approach.

(2) When $t_k = 0$, the optimization problem in (14) is solved to obtain the control sequence $\Delta \mathbf{U}^*_{t_k} = [\Delta \mathbf{u}^*_{t_k}, \Delta \mathbf{u}^*_{t_k+1}, \ldots, \Delta \mathbf{u}^*_{t_k+N_c-1}]^T$, which uses the first element to form the control input $\mathbf{u}(t_k) = \mathbf{u}(t_k - 1) + \Delta \mathbf{U}^*_{t_k}(1)$, where $\mathbf{u}(t_k - 1) = 0$, and $\mathbf{u}(t_k)$ is applied to the system to obtain the new state vector $\xi(i+1|t_k)$

(3) The triggering conditions are checked via substituting $\xi(i+1|t_k)$ into equations (15) and (17) to see whether the predetermined thresholds are met

(4) When the triggering conditions are not valid, the elements in the input sequence are continuously applied without computing the optimization; otherwise, the optimization problem (14) is solved for the new control sequence, and the new input is utilized for getting the new state vector; then, the algorithm is back to step (3).

In order to quantify the computation and communication resources saved by using the event-triggering mechanism, S_1 and S_2 are defined as

$$S_1 = 1 - \left(\frac{T_e}{T_t}\right), \tag{20}$$

$$S_2 = (T_t - T_e)d, \tag{21}$$

where T_e and T_t are the event-triggering times and time-triggering times, and d is the measured mean network delay in a standard experimental environment. Therefore, equations (20) and (21) can quantificationally calculate the saved computation and communication resources.

It is worth nothing that a large amount of historical data is normally collected when the robot is controlled by the classical MPC before using the above strategy. Therefore, in view of different working conditions, the control strategy can also be used after statistical analysis of the previous historical data. Therefore, the control strategies proposed in this study also have a strong adaptability.

5. Simulation and Experimental Verifications

To verify the effectiveness of the proposed threshold curve/band-based MPC method, the simulation test and experimental verification are carried out, respectively.

5.1. Simulation Test. The simulation is performed on the MATLAB platform. Given the differential motion model of the mobile robot in equation (1), the objective of the mobile robot is to follow a given trajectory $y = 10$ from the initial position [0, 0, 0]. Simulation set-up sampling time $T = 0.05$ s, simulation steps $N_s = 250$, prediction and control horizons $N_p = N_c = 5$, weighting matrices $\mathbf{Q} = [1\ 0\ 0;\ 0\ 1\ 0;\ 0\ 0\ 0.5]$, $\mathbf{R} = [0.1\ 0;\ 0\ 0.1]$, weighting coefficient $\alpha = 10$, relaxing factor $\varepsilon = 0.025$, the upper bound of the disturbance $\rho = 0.05$, input incremental constraint $-0.05 \le \Delta \mathbf{u} \le 0.05$, input constraint $-0.2 \le \mathbf{u} \le 0.2$, and $N = 10$ is chosen to set the thresholds. In order to obtain the threshold curve and threshold band, 10 groups of logged coordinate data for trajectory tracking in previous tests with the classic MPC under a similar simulation environment are utilized. The threshold curve and band are then obtained by using the threshold selection method (equations (16) and (18)) proposed in Section 4, and the obtained values are shown in Figure 1.

Then, the control effect is compared with the standard time-triggering strategy, and the results are shown in Figure 2, in which TT denotes the time-triggered method, while ET1 and ET2 denote the event-triggering strategies using the threshold curve and threshold band methods, respectively.

Figure 2 illustrates that the proposed two event-triggering strategies have similar tracking performance compared with the standard time-triggered method. In fact, we can see that the event-triggering strategies both have faster response speed, indicating the advantage of the event-triggered controller with some practical and meaningful condition. In addition, the proposed method has been compared with an existing event-triggered MPC approach developed in literature [18], and the results show that event-triggering strategy proposed in this study can achieve better tracking performance and response speed (as illustrated in Figure 2).

Besides, Table 1 provides the effect of changing the design parameters on the closed-loop performance of the robot system for the proposed method. From Table 1, it is observed that weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ are mutually restricted, and by reducing $\mathbf{R}$ (or increasing $\mathbf{Q}$), the rapid response performance of the system can be improved; by increasing $\mathbf{R}$ (or reducing $\mathbf{Q}$), the control cost can be reduced and the steady-state performance of the system can be improved. As for the prediction horizon N_p and control horizon N_c, due to physical limitations, $N_c <= N_p$ must be satisfied, and in order to maintain the degree

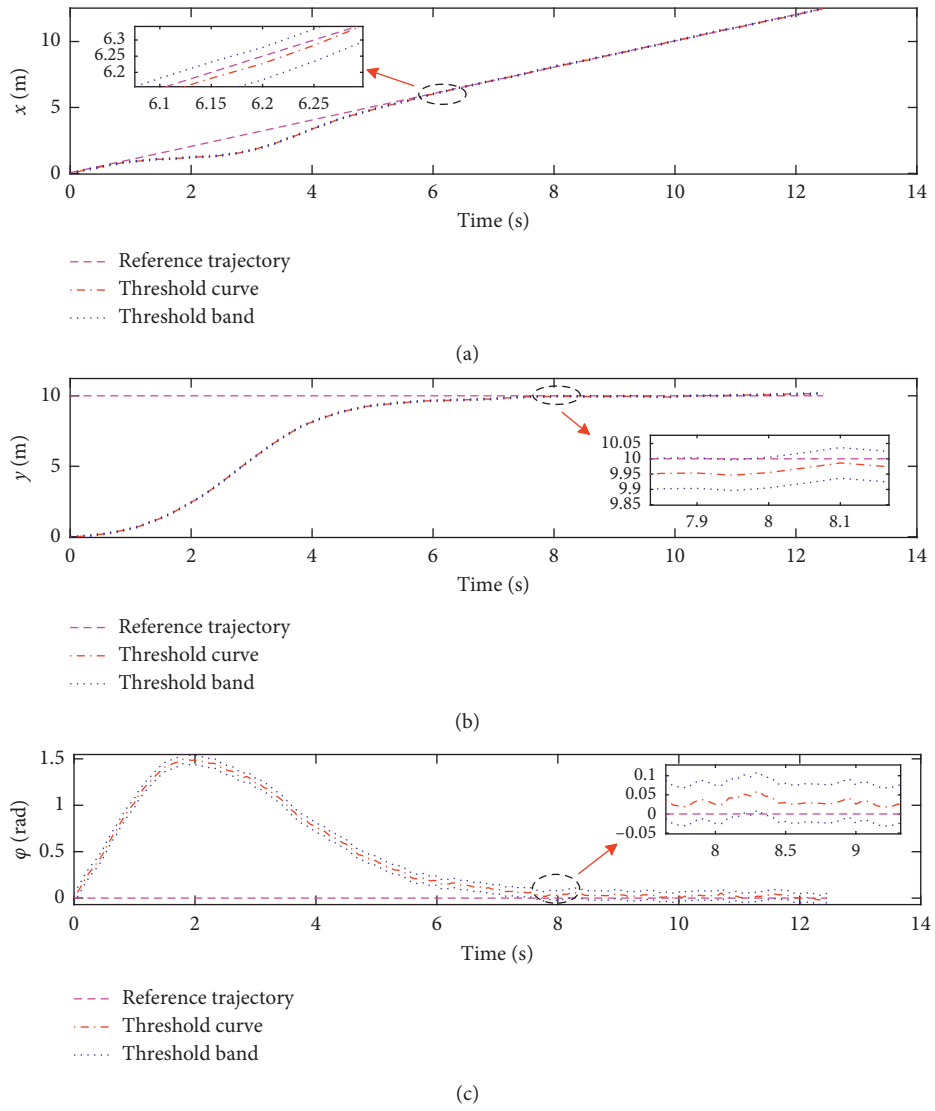

Figure 1: The obtained threshold curve and threshold band.

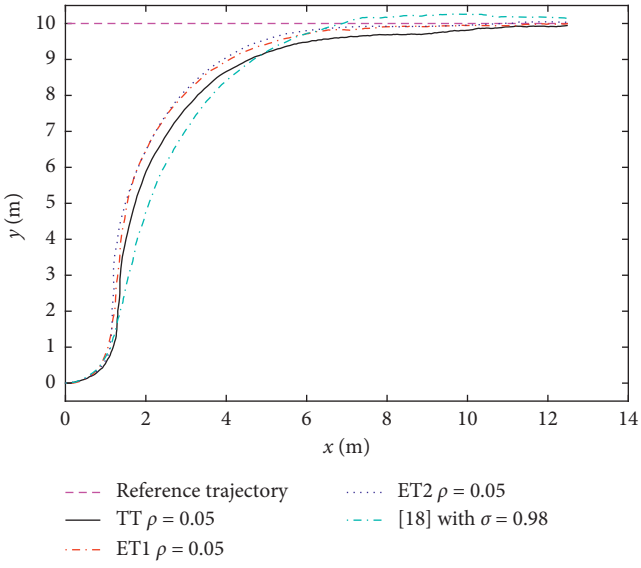

Figure 2: Trajectory tracking results.

of freedom for the MPC optimization, $N_c = N_p$ is normally considered in real applications. Given such a setting, it is observed that a larger prediction horizon N_p/control horizon N_c can provide a faster response speed but a certain compromise in steady-state performance will also be resulted.

Figure 3 shows the tracking error of state components x, y, and φ from the corresponding reference. Specifically, Figure 3 shows that the event-triggering strategies have faster response speeds than their time-triggering counterparts, and the threshold band method also has a faster response speed than the threshold curve method. Note that due to the existing of the external disturbance, the state components are changing slightly around the origin, rather than remain constants.

Furthermore, the triggering instants are shown in Figure 4, in which the triggered time steps of ET1 and ET2 are marked with red and blue circles with value 1. Note that the time-triggered MPC needs to update at every single sampling time (250 times in total), and the curve-based event-triggering strategy needs 184 times, and the band-based strategy only

Table 1: Performance comparison of different design parameters for the proposed method.

Prediction horizon (N_p)	Control horizon (N_c)	State-weighting matrix (Q)	Control-weighting matrix (R)	Max. steady-state error		
				$x(m)$	$y(m)$	φ(rad)
5	5	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$	$\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$	0.033	0.054	0.011
5	5	$\begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.25 \end{bmatrix}$	$\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$	0.028	0.038	0.013
8	8	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$	$\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$	0.042	0.053	0.037

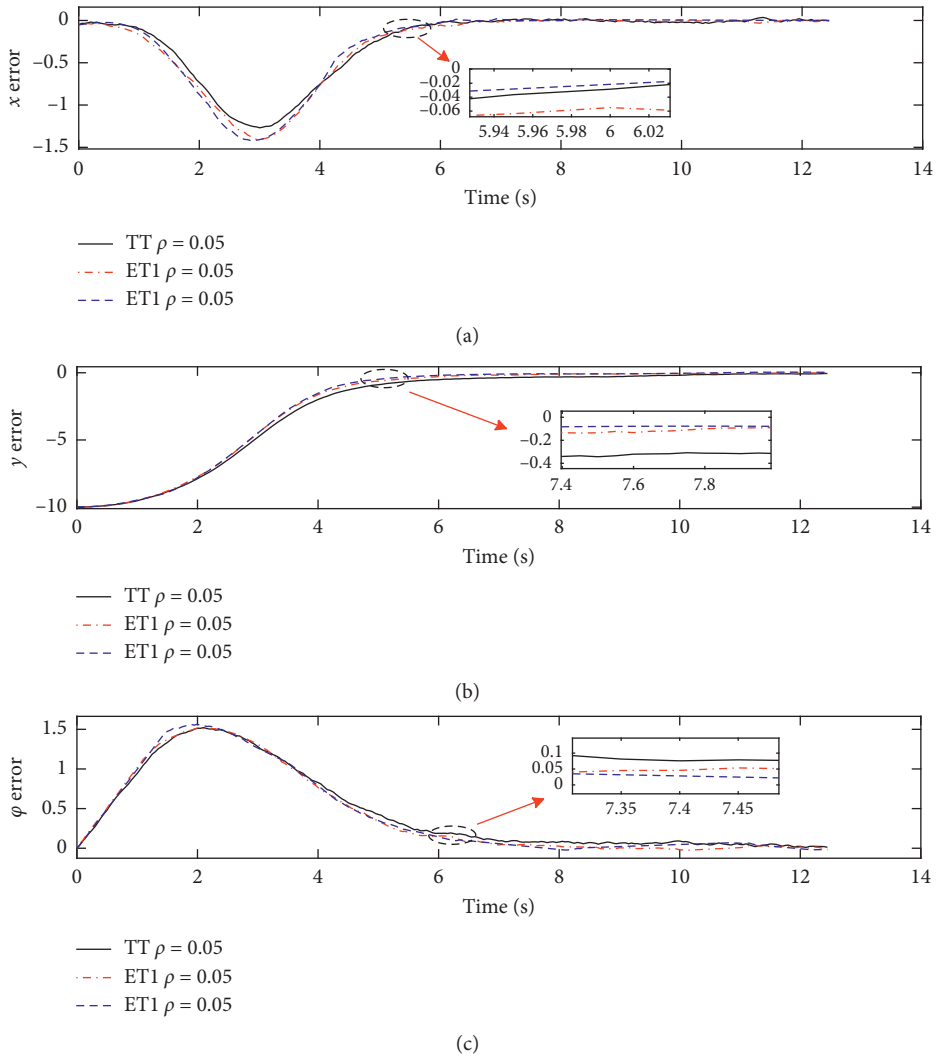

Figure 3: Trajectory tracking results of state components.

needs 65 times. Hence, the proposed strategies can significantly reduce the number of MPC update times to achieve trajectory tracking. According to equation (20), 26.4% and 74% computation resources are saved, respectively.

5.2. Experimental Verification. Then, the experiment test is carried out. Figure 5 shows the utilized crawler mobile robot and the experimental environment. The robot body is driven by two DC servo motors equipped with rotary encoders. The

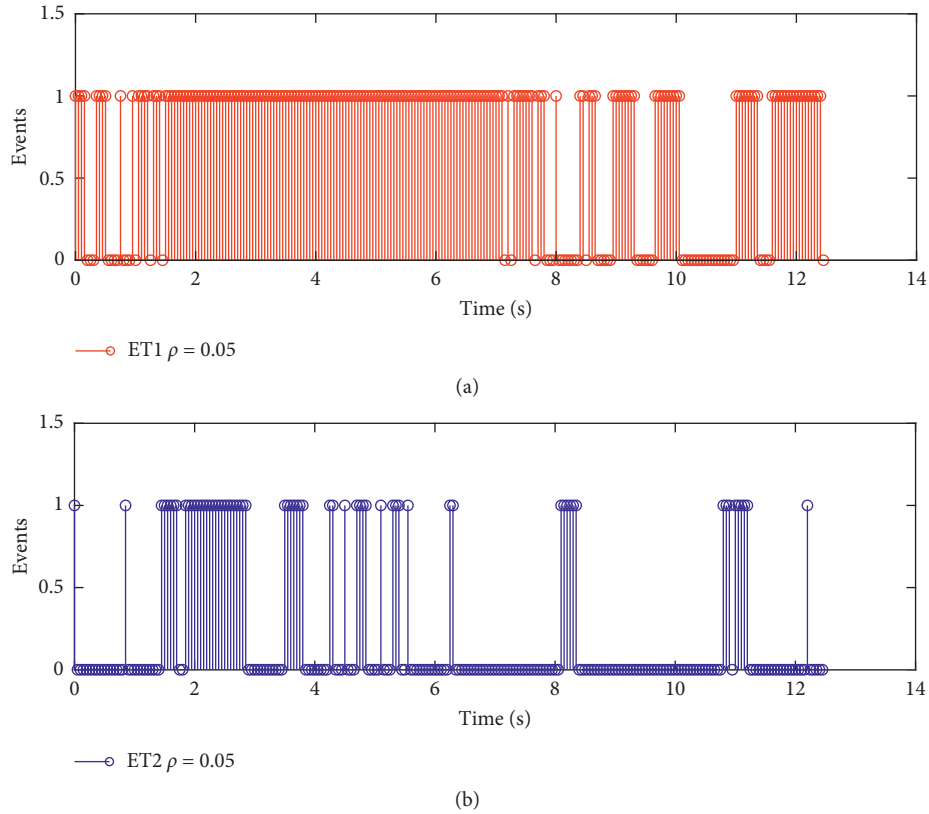

Figure 4: Triggering instants of the proposed methods.

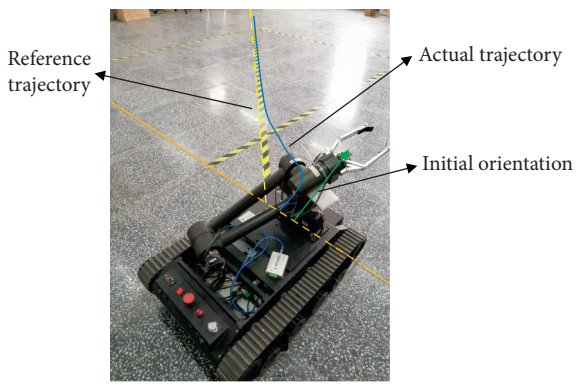

Figure 5: Mobile robot and the experimental environment.

reference trajectory is a diagonal line within an 8 m × 8 m rectangle.

The operating system on the robot is the Indigo version of the robot operating system (ROS). In order to establish communication with the same version of the ROS laptop, the 54 M bandwidth 802.11 g wireless communication protocol is used to realize remote wireless connection and control the movement of the robot. After network testing, the average network communication delay is about 17.04 ms, as shown in Figure 6.

Due to the error between the expected velocity command transmitted by the MPC node to the motion controller and the actual velocity measured by the encoder, it is necessary to calibrate the velocity of the mobile robot before the experiment. Let the robot travel a distance of 3 m at velocities of 0.1, 0.2, and 0.3 m/s, and each velocity was measured three times, and the average velocity is recorded, based on which the error can be calculated, and the detail data are shown in Table 2.

The initial position is [0, 0, 0], the target position is [8, 8, $\pi/4$], the prediction and control horizons are $N_p = N_c = 20$, the controller sampling frequency is 20 Hz, the input constraint is $-0.3 \leq \mathbf{u} \leq 0.3$, the disturbance limit is $\rho = 0.05$, the group number for constructing the triggering condition is $N = 10$, and other parameters are the same as the simulation. The upper bound of disturbance is selected as the maximum steady-state error in the experiment.

The real-time position and speed information of the mobile robot is obtained through wireless communication, and the ROS visualization (Rviz) tool and stageros tool in the host computer are used to record the real-time trajectory, as shown in Figure 7.

In addition, coordinate values recorded by a mobile robot in the global coordinate system are imported into MATLAB. According to equations (16) and (18), the threshold curve and threshold band can be calculated, as shown in Figure 8. Through the recorded data, we can get the tracking error of each state component under time-triggered and event-triggered methods, as shown in Figure 9.

It is observed that compared with the time-triggered method, the maximum error of each state component is

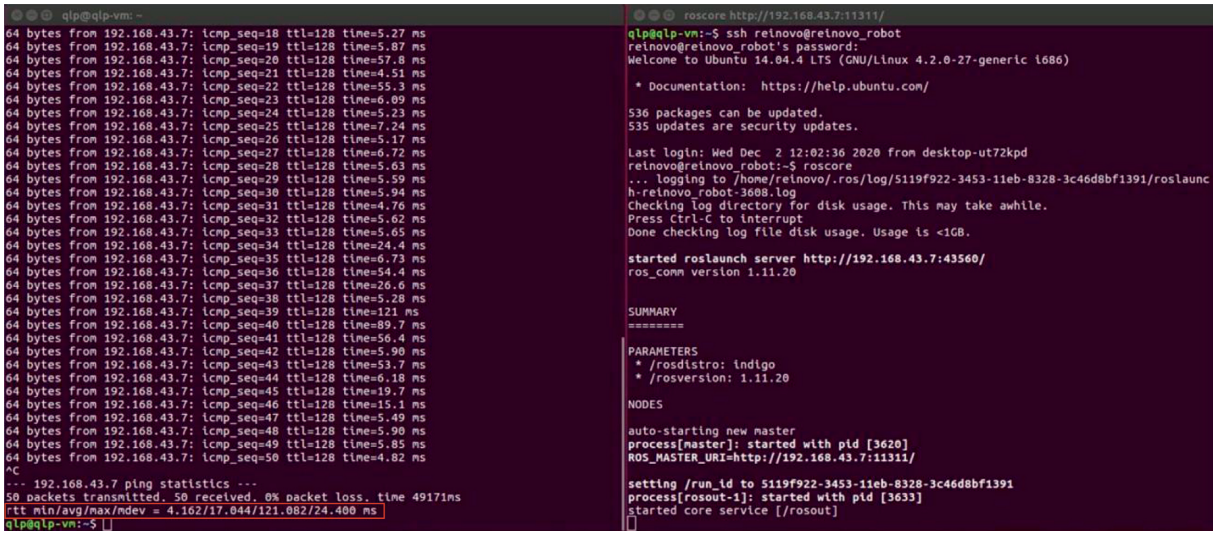

Figure 6: Network environment test.

Table 2: Velocity calibration of mobile robot.

Desired velocity (m/s)	Before calibration		After calibration	
	v (m/s)	Error (%)	v (m/s)	Error (%)
0.1	0.0808	19.2	0.0947	5.3
0.2	0.1573	21.4	0.1917	4.2
0.3	0.2397	20.1	0.2869	4.4

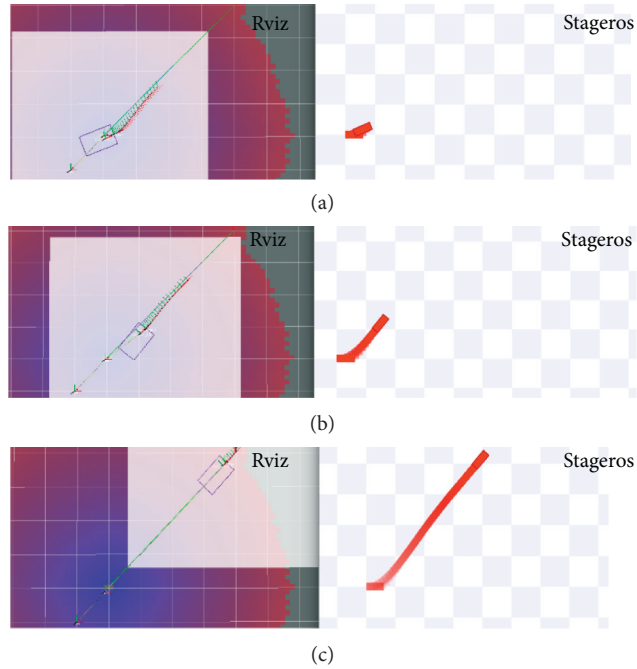

Figure 7: The real-time trajectory of mobile robot in host computer.

0.0268, 0.0141, and 0.0157 m for the curve-based event-triggering strategy, and the maximum error is 0.0258, 0.0333, and 0.0275 m for the band-based strategy, which illustrates that the developed event-triggering mechanism has a similar control effect as the standard method. Besides, from the 3rd subplot of Figure 9, the robot mainly completes the orientation adjustment process at 1.9 s and then gradually reduces the horizontal and vertical coordinate errors.

Figure 10 illustrates the corresponding triggering instants. Note that the time-triggered method needs to

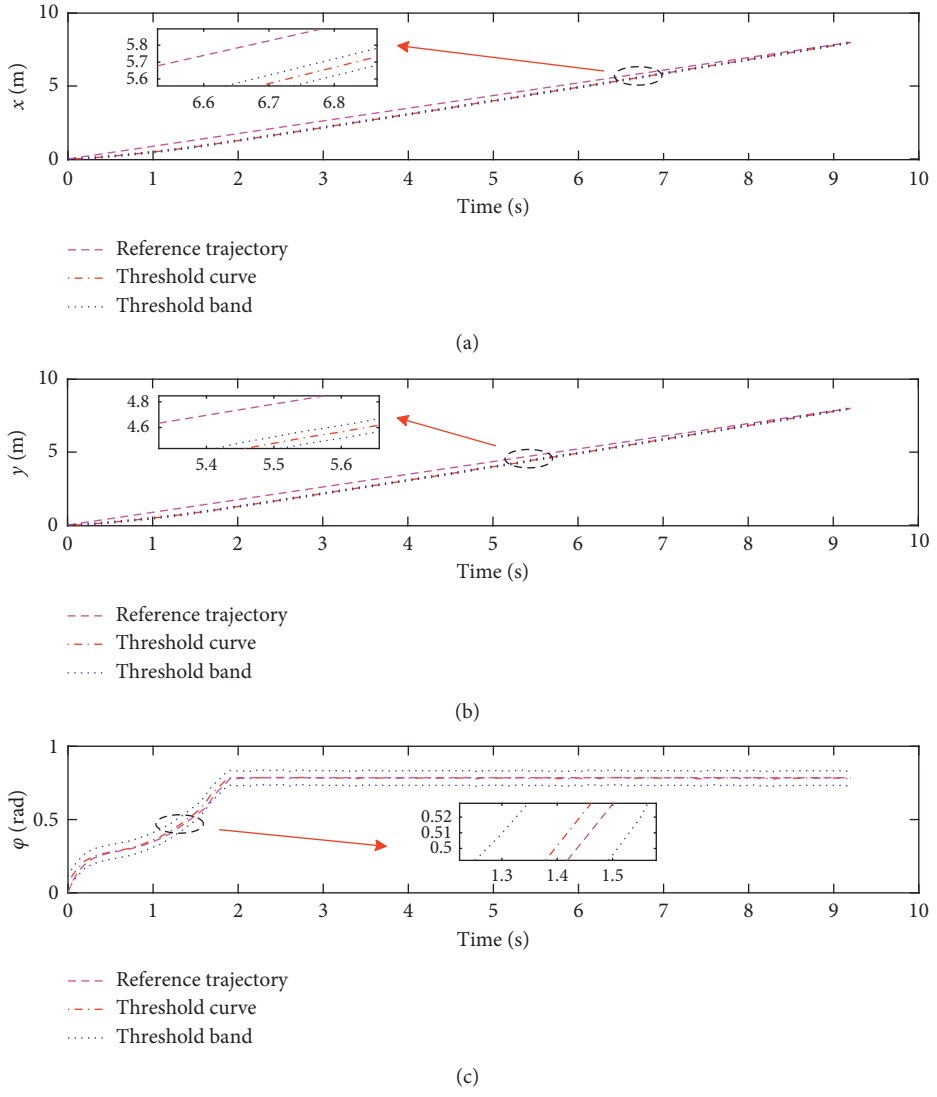

Figure 8: Threshold curve and threshold band in experiment.

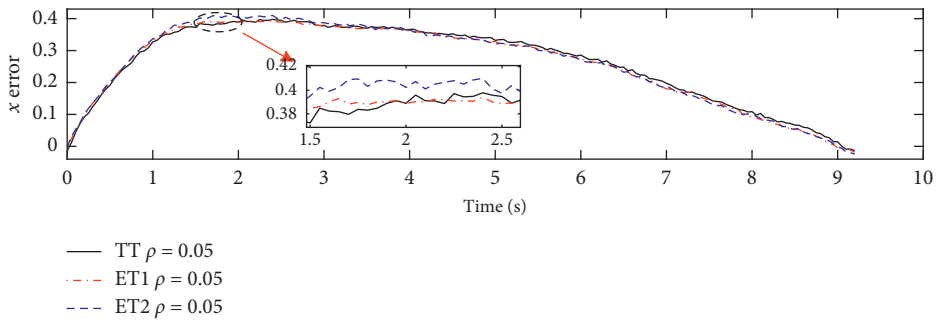

Figure 9: Continued.

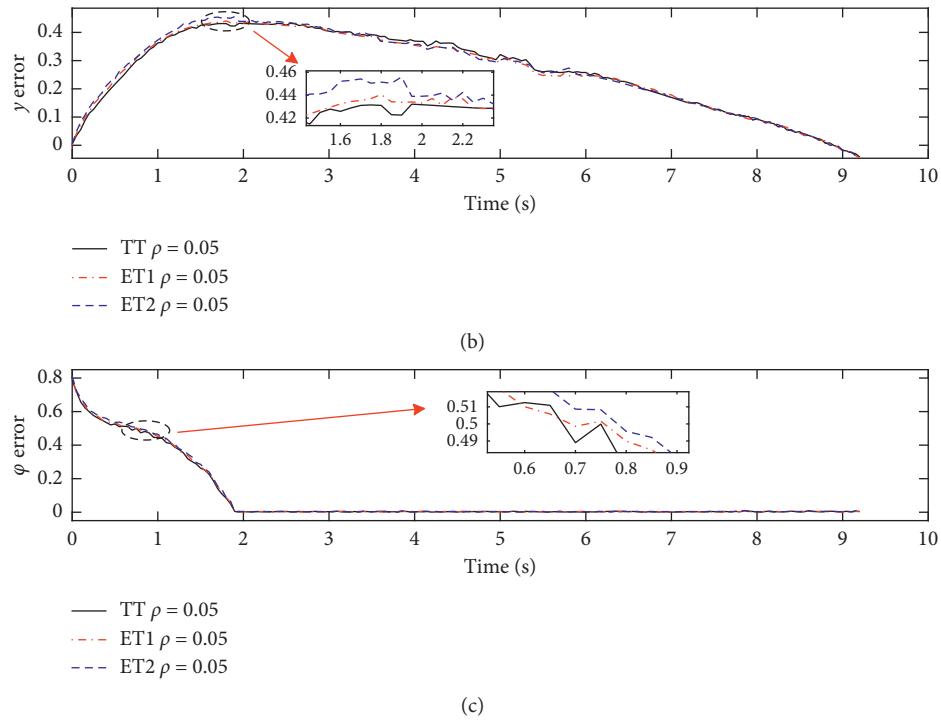

Figure 9: Trajectory tracking results of state components in experiment.

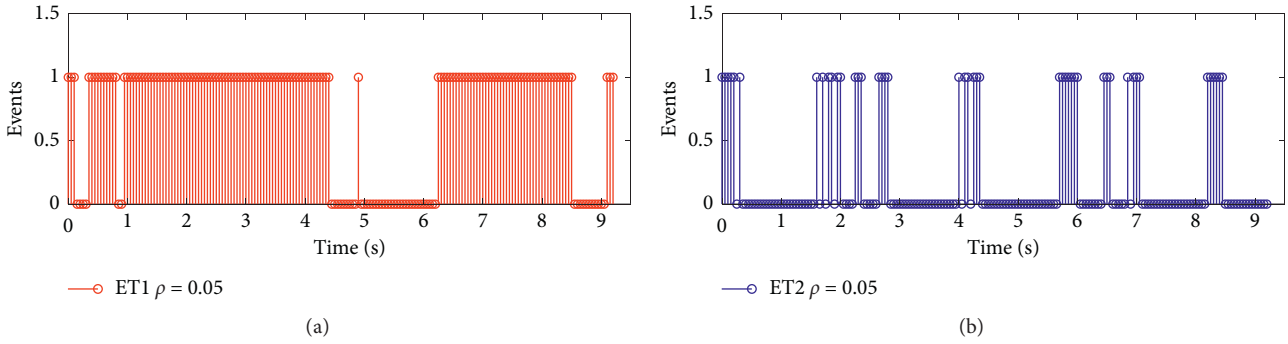

Figure 10: Event-triggering instants in experiment.

update the controller for 185 times in total with sampling time being 0.05 s, and event-triggering strategies only need 133 times and 45 times, respectively. Therefore, compared to the time-triggered counterpart, event-triggering strategies are able to significantly lessen the number of updates for the optimization problem. Furthermore, according to equation (20), it can be obtained that ET1 and ET2 can save 28.1% and 75.7% of computation resources, respectively. From equation (21) and communication delay from Figure 6, the communication time is reduced by 0.886 s and 2.385 s, respectively.

6. Conclusion

This study proposes a novel MPC-based trajectory tracking control method based on the event-triggering mechanism for mobile robot. The method includes two event-triggering strategies, the triggering strategy based on the threshold curve and the threshold band, respectively. The construction of the threshold curve and the threshold band is to employ the statistical method to process the recorded historical trajectory data. The simulation result and experimental verification show that compared with the classic time-triggered MPC, the proposed method can significantly reduce the update times of the optimization problem such that the computation and communication burdens can be effectively reduced without affecting the control performance of the system. In the future, the method to optimally select the design parameters in the event-triggering conditions and the way to handle more complex working conditions, e.g., other types of model uncertainties, time delays, and cyber attacks, are considered as potential research directions, and the target is to enhance the practicability of the event-triggered control technology.

Data Availability

The main contribution of the work is the proposed two new event-triggering strategies, and thus, the data are not the main supporting material.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Science Natural Foundation of China (61903291) and Special Scientific Research Project of Shaanxi Provincial Department of Education (Z20190269). The authors would like to thank the editor and anonymous reviewers for their helpful suggestions that have improved the quality of the study.

References

[1] J. Chen, Z. Shuai, H. Zhang, and W. Zhao, "Path following control of autonomous four-wheel-independent-drive electric vehicles via second-order sliding mode and nonlinear disturbance observer techniques," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 3, pp. 2460–2469, 2021.

[2] Z. Xu, L. He, N. He, and L. Qi, "Self-triggered model predictive control for perturbed underwater robot systems," *Mathematical Problems in Engineering*, vol. 2021, Article ID 4324389, 9 pages, 2021.

[3] M. Fakoor, S. Nikpay, and A. Kalhor, "On the ability of sliding mode and LQR controllers optimized with PSO in attitude control of a flexible 4-DOF satellite with time-varying payload," *Advances in Space Research*, vol. 67, no. 1, pp. 334–349, 2021.

[4] D. He, T. Qiu, and L. Lu, "Input-to-state stability of contractive EMPC of non-linear systems with bounded disturbances," *Control Theory & Applications, IET*, vol. 13, no. 5, pp. 651–658, 2019.

[5] I. Jmel, H. Dimassi, S. Hadj Said, and F. M'Sahli, "Adaptive observer-based output feedback control for two-wheeled self-balancing robot," *Mathematical Problems in Engineering*, vol. 2020, Article ID 5162172, 16 pages, 2020.

[6] H. Peng, W. Wang, Q. An, C. Xiang, and L. Li, "Path tracking and direct yaw moment coordinated control based on robust MPC with the finite time horizon for autonomous independent-drive vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 6053–6066, 2020.

[7] C. Shen, Y. Shi, and B. Buckham, "Path-following control of an AUV: a multiobjective model predictive control approach," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 3, pp. 1334–1342, 2019.

[8] P. Li, X. Jiang, and Y. Wei, "Predictive control method of autonomous vehicle based on tracking-error model," *Transactions of the Chinese Society for Agricultural Machinery*, vol. 48, no. 10, pp. 351–357, 2017.

[9] G. Bai, L. Liu, and Y. Meng, "Real-time path tracking of mobile robot based on nonlinear model predictive control," *Transactions of the Chinese Society for Agricultural Machinery*, vol. 51, no. 9, pp. 47–52, 2020.

[10] C. Liu, J. Gao, and D. Xu, "A model predictive path following control method for underactuated autonomous underwater vehicles," *Mechanical Science and Technology for Aerospace Engineering*, vol. 36, no. 11, pp. 1653–1657, 2017.

[11] M. Mei, D. Zhu, and W. Gan, "The tracking control of unmanned underwater vehicles based on model predictive control," *Control Engineering of China*, vol. 26, no. 10, pp. 1917–1924, 2019.

[12] H. Zhang, H. Zhang, and Z. Wang, "Event-triggered predictive path following control for unmanned autonomous vehicle," *Control and Decision*, vol. 34, no. 11, pp. 2421–2427, 2019.

[13] C. Liu, J. Gao, H. Li, and D. Xu, "Aperiodic robust model predictive control for constrained continuous-time nonlinear systems: an event-triggered approach," *IEEE Transactions on Cybernetics*, vol. 48, no. 5, pp. 1397–1405, 2018.

[14] Y. Wang, H. R. Karimi, and H. Yan, "An adaptive event-triggered synchronization approach for chaotic lur'e systems subject to aperiodic sampled data," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 3, pp. 442–446, 2019.

[15] Y. Wang, H. R. Karimi, H.-K. Lam, and H. Yan, "Fuzzy output tracking control and filtering for nonlinear discrete-time descriptor systems under unreliable communication links," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2369–2379, 2020.

[16] L. Lu, Y. Niu, and Y. Zou, "Design of robust model predictive controller based on event trigger strategies," *Journal of East China University of Science and Technology*, vol. 41, no. 4, pp. 515–522, 2015.

[17] K. Hashimoto, S. Adachi, and D. V. Dimarogonas, "Self-triggered model predictive control for nonlinear input-affine dynamical systems via adaptive control samples selection," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 177–189, 2017.

[18] S. Karim and H. Mohsen, "A decentralized event-based model predictive controller design method for large-scale systems," *Automatic Control and Information Sciences*, vol. 2, no. 1, pp. 26–31, 2014.