Event-Triggered MPC With Linear Inter-Event Control for AV Path Tracking

Zhaodong Zhou and Jun Chen, *Senior Member, IEEE*

***Abstract*—Model predictive control (MPC) is widely used for autonomous vehicle path tracking due to its ability to handle system constraints and optimize performance over a prediction horizon. However, frequent online optimization imposes high computational demands, making the real-time implementation of MPC challenging. Event-triggered MPC aims to solve this issue by updating control actions only when a predefined condition is met, but it executes precomputed control sequences in an open-loop fashion between events, potentially allowing errors to accumulate. This letter proposes an event-triggered MPC framework integrated with a linear inter-event control mechanism to address this limitation. The proposed inter-event controller applies a least-squares-based linear model to generate control inputs in real-time during inter-event periods, enabling continuous feedback corrections. Experimental results on a Quanser QCar2 platform demonstrate that the proposed approach improves tracking accuracy by 10% while significantly reducing the number of MPC optimizations compared to standard event-triggered MPC, offering an efficient solution for real-time path tracking problem.**

***Index Terms*—Motion control, autonomous vehicle navigation model predictive control, event-triggered control.**

I. Introduction

ACCURATE path tracking is one of the core challenges in autonomous vehicle (AV) control [1], [2], [3], [4], [5]. The objective of path tracking is to ensure that the vehicle can follow a reference trajectory with minimal deviation while maintaining stability and safety under various driving conditions. As autonomous driving systems become increasingly complex, path tracking controllers must handle nonlinear vehicle dynamics, dynamic constraints, and external disturbances in real-time [6], [7]. Traditional controllers, such as pure pursuit controller [8], Stanley controller [9], fuzzy controller [10], and PID controller [11], provide simple implementations but often struggle to balance precision and robustness, and need extensive amount of calibration especially in complex driving scenarios [12], [13]. To address these challenges, model predictive control (MPC) has recently been widely studied and applied in AV path tracking due to its ability to predict future system behavior and handle dynamic constraints, thus achieving robust control [14], [15], [16]. By solving a finite horizon optimal control problem in real-time, MPC computes control actions that minimize a cost function while satisfying actuator limits, vehicle dynamics, and safety considerations [17], [18]. However, this online optimization requires substantial computational resources, especially for real-time applications where frequent updates are necessary to cope with uncertainties and fast-varying driving scenarios [19].

To reduce the high computational burden of traditional time-triggered MPC, event-triggered MPC (eMPC) schemes have been introduced in recent years [20], [21]. In eMPC, the controller continuously monitors the system state and only solves the optimization problem when certain event conditions are met, such as exceeding a predefined error threshold [22], [23]. By skipping unnecessary updates when the system operates within acceptable bounds, eMPC can significantly reduce the number of optimization calls compared to time-triggered MPC, leading to improved computational efficiency [24]. However, most existing eMPC implementations employ a simple inter-event execution strategy: once an optimization is triggered and solved, the resulting control sequence is typically applied in an open-loop, sample-and-hold manner until the next event [25], [26]. Without additional feedback during these inter-event intervals, tracking errors can accumulate under model mismatch and disturbances [27], particularly when triggering is sparse. More broadly, event-triggered MPC offers a key design freedom not only in the event-trigger condition but also in how control inputs are executed between consecutive optimization instances. While open-loop inter-event execution is computationally friendly, the drift it induces can degrade performance and may need more frequent re-optimizations to correct, partially offsetting the intended computational savings. Inter-sample controllers can alleviate this issue, yet they are commonly realized as fixed-gain feedback [28] and therefore can not adapt to the current operating regime.

To address this issue, this letter proposes an enhanced event-triggered MPC framework with linear inter-event control. The key idea is to incorporate a lightweight inter-event controller that updates control inputs in real-time based on current vehicle state. Specifically, after each MPC optimization, a linear feedback gain between the resulted optimal control sequence and the optimal vehicle state sequence is learned using linear least squares regression [29]. During inter-event periods, this linear feedback gain is then used to generate control commands adaptively at each control loop, allowing real-time correction of deviations before they grow large enough to trigger a new event. Compared to conventional approach where the optimal control

Received 21 March 2026; accepted 6 May 2026. Date of publication 11 May 2026; date of current version 19 May 2026. This article was recommended for publication by Associate Editor Y. Chen and Editor L. Pallottino upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation under Award #2237317 and in part by the National Institute of Standards and Technology under Award 60NANB24D138. *(Corresponding author: Jun Chen.)*

The authors are with ECE Department, Oakland University, Rochester, MI 48309 USA (e-mail: zhaodongzhou@oakland.edu; junchen@oakland.edu).

Digital Object Identifier 10.1109/LRA.2026.3692329

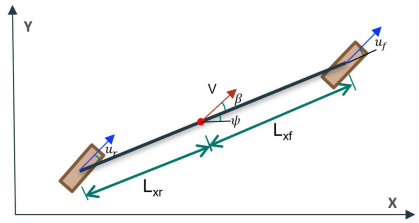

Fig. 1. Vehicle kinematic bicycle model.

sequence is executed in an open-loop fashion [6], [20], [30], this hybrid structure maintains the long-term optimality of MPC while enhancing responsiveness between events. In addition, the proposed inter-event control does not require solving any Riccati equation in real-time (note that the system is nonlinear), and hence is more computationally tractable than LQR [31]. We note that combining MPC with a linear feedback term has classical roots in robust MPC such as tube MPC [32]. However, the proposed framework is different from tube-based robust MPC formulation: it does not construct invariant tubes, tighten constraints, or assume an explicit bounded disturbance set. Instead, the learned linear feedback is used as an inter-event approximation of the MPC policy, with the primary goal of reducing the frequency of solving the nonlinear optimization while preserving tracking performance in real-time. The proposed method is validated experimentally using a Quanser Qcar2 ground vehicle platform [33]. By testing under multiple event thresholds and speeds, the performance of the inter-event controller is evaluated in terms of both tracking accuracy and computational load. Comparing to conventional event-triggered MPC, inter-event controller increase the tracking accuracy around 10% and reduce the event frequency around 50%. The results demonstrate that the proposed approach effectively reduces the number of MPC optimizations while maintaining and even improving tracking precision compared to conventional event-triggered MPC. This makes the proposed approach especially attractive for real-time AV applications where computational resources are limited and efficient control execution is critical.

The remainder of this letter is organized as follows. Section II describes the system model and the conventional MPC designs, and Section III presents the proposed event-triggered MPC with linear inter-event control. Section IV discusses the experimental setup and test results, while Section V concludes the paper.

II. AV Path Tracking

A. Vehicle Model

In this letter, the vehicle model used in MPC design is a kinematic bicycle model as illustrated in Fig. 1. We define the state vector at the vehicle's center of gravity (CG) as $x = \begin{bmatrix} p_x & p_y & \psi \end{bmatrix}^T$, where px and py represent the coordinate of positions of the vehicle, and ψ denotes the vehicle's heading angle. All state variables are expressed relative to the global frame. The time derivative of this state vector is given by $\dot{x} = \begin{bmatrix} \dot{p_x} & \dot{p_y} & \dot{\psi} \end{bmatrix}^T$. The kinematic bicycle model is used to describe the vehicle motion, which captures the dynamics for path tracking at certain speeds, with the forward Euler method the discretized time model is expressed as:

$$p_{x,n+1} = p_{x,n} + v_n \cos(\psi_n + \beta_n)dt \tag{1a}$$

$$p_{y,n+1} = p_{y,n} + v_n \sin(\psi_n + \beta_n)dt \tag{1b}$$

$$\psi_{n+1} = \psi_n + \frac{v_n \cos(\beta_n)}{L_{xf} + L_{xr}} \tan(u_{f,n})dt. \tag{1c}$$

where the subscript n represent the discrete time step, β represents the slip angle of the vehicle, v denotes the velocity of the vehicle's CG, L_{xf} and L_{xr} are the lengths from the CG to the front and rear axles, respectively, and u_f is correspond to front steering angles. The slip angle β is defined as:

$$\beta = \arctan\left(\frac{L_{xr}\tan(u_f)}{L_{xf} + L_{xr}}\right). \tag{2}$$

B. Time-Triggered MPC

For MPC-based path tracking control, at each time instant t, the controller operates in a receding horizon manner. First, the current system state $\hat{x}_t$ is measured. Then, based on the vehicle model, system constraints, and reference trajectory, the controller solves a finite horizon optimal control problem (OCP) to obtain the optimal predicted state sequence $\mathbf{X}(t) = \{x_{t+1}, x_{t+2}, \ldots, x_{t+N}\}$ and the corresponding optimal control input sequence $\mathbf{U}(t) = \{u_t, u_{t+1}, \ldots, u_{t+N-1}\}$, where N denotes the prediction horizon. In conventional periodic MPC (or time-triggered MPC), only the first control input u_t is applied to the vehicle, while the optimization process is repeated at the next time step with updated state information.

The OCP solved by MPC at each time step is formulated as follows:

$$\begin{aligned} \min_{x} \quad J = & \sum_{n=1}^{N} \left|\left| p_{x,t+n} - p_{x,t+n}^{ref} \right|\right|_{Q_p}^2 \\ & + \sum_{n=1}^{N} \left|\left| p_{y,t+n} - p_{y,t+n}^{ref} \right|\right|_{Q_p}^2 + \sum_{k=0}^{N-1} ||u_{t+n}||_{Q_u}^2 \\ & + \sum_{n=0}^{N-1} ||u_{t+n} - u_{t+n-1}||_{Q_d}^2 \end{aligned} \tag{3a}$$

$$\text{s.t.} \quad x_t = \hat{x}_t \tag{3b}$$

$$\text{System dynamics } (1), \quad 1 \le n \le N \tag{3c}$$

$$u_{min} \le u_{t+n} \le u_{max}, \quad 0 \le n \le N-1 \tag{3d}$$

$$du_{min} \le u_{t+n} - u_{t+n-1} \le du_{max}, \quad 0 \le n \le N-1. \tag{3e}$$

The four terms in the cost function (3a) respectively penalize the predicted position tracking errors along the longitudinal and lateral axes, the control effort, and the rate of change of control inputs to ensure smooth steering behavior. Q_p, Q_u, and Q_d are the corresponding weighting matrices. The constraints (3d) and (3e) impose physical limitations on the control input magnitude and its change rate, respectively.

C. Event-Triggered MPC

Conventional event-triggered MPC usually relies on a predefined event, and as long as the event occurs (i.e., is triggered), the OCP (with new feedback) will be re-solved to generate a new control sequence. This letter adopts a threshold-based event-trigger mechanism similar to that presented in [22], [23], [34], and further incorporates an anticipation lateral error to reduce overshoot when approaching a steep turn.

In particular, the event condition primarily considers the lateral deviation of the vehicle from the reference path. Specifically, the lateral offset $Y(t)$ is defined as the shortest distance from the current vehicle position $(p_{x,t}, p_{y,t})$ to the reference path segment defined by two adjacent waypoints $(p^{ref}_{x,1}, p^{ref}_{y,1})$ and $(p^{ref}_{x,2}, p^{ref}_{y,2})$, and is computed as:

$$Y = \frac{|(p^{ref}_{x,2} - p^{ref}_{x,1})(p^{ref}_{y,1} - p_{y,t}) - (p^{ref}_{x,1} - p_{x,t})(p^{ref}_{y,2} - p^{ref}_{y,1})|}{\sqrt{(p^{ref}_{x,2} - p^{ref}_{x,1})^2 + (p^{ref}_{y,2} - p^{ref}_{y,1})^2}}. \tag{4}$$

To make the event-trigger condition predictive rather than reactive, we predict the vehicle position after a look-ahead time T_L using the kinematic vehicle model (1) under the currently available control policy, yielding $(\bar{p}_{x,t+T_L}, \bar{p}_{y,t+T_L})$. Therefore, the corresponding predictive lateral error $\bar{Y}$ is then computed using the same geometric definition:

$$\bar{Y} = \frac{\left|(p^{ref}_{x,2} - p^{ref}_{x,1})(p^{ref}_{y,1} - \bar{p}_{y,t+T_L}) - (p^{ref}_{x,1} - \bar{p}_{x,t+T_L})(p^{ref}_{y,2} - p^{ref}_{y,1})\right|}{\sqrt{(p^{ref}_{x,2} - p^{ref}_{x,1})^2 + (p^{ref}_{y,2} - p^{ref}_{y,1})^2}}. \tag{5}$$

The event-trigger condition used in this study is then defined as:

$$e = \begin{cases} 1 & \text{if } Y > \sigma \text{ or } \bar{Y} > \sigma \text{ or } i > i_{max} \\ 0 & \text{Otherwise} \end{cases}, \tag{6}$$

where σ is the predefined lateral error threshold, and i denotes the number of consecutive steps since the last MPC optimization. The parameter i_{max} limits the maximum number of steps that can elapse between two MPC optimizations and should not exceed the prediction horizon p to ensure feasibility.

If $e = 1$, the OCP defined in (3) is re-solved with new state feedback to obtain a new optimal control sequence. If $e = 0$, the controller continues to apply the previously optimized control sequence by shifting to the next control input, operating at a fixed control frequency, as described in Algorithm 1. In this way, the event-triggered MPC reduces computation by solving the optimization only when necessary, while relying on open-loop execution of previous optimal control sequence between two events.

Remark 1: The look-ahead position $(\bar{p}_{x,t+T_L}, \bar{p}_{y,t+T_L})$ is obtained via a forward simulation of the kinematic vehicle model (1) over the T_L using a discrete time rollout. Specifically, let Δt denote the controller sampling time and $H = T_L/\Delta t$. Starting from the measured state $\bar{x}_0 = \hat{x}(t)$,

$$\bar{x}_{k+1} = f(\bar{x}_k, \bar{u}_k), \quad k = 0, \ldots, H-1, \tag{7}$$

Algorithm 1: Event-triggered MPC for AV Path Tracking.

1 **Initialization:** $i \leftarrow 0$; $\mathbf{U}_{t_1} \leftarrow 0$;
2 **Procedure** eMPC($\hat{x}_t$, i, U_{t_1});
3 $i \leftarrow i + 1$;
4 $e \leftarrow$ computing (6);
5 **if** $e = 1$ **then**
6 $\quad i \leftarrow 0$;
7 $\quad (\mathbf{X}_t, \mathbf{U}_t) \leftarrow$ Solving OCP (3);
8 $\quad u \leftarrow \mathbf{U}_t(1)$;
9 $\quad \mathbf{U}_{t_1} \leftarrow \mathbf{U}_t$;
10 **else**
11 $\quad u \leftarrow \mathbf{U}_{t_1}(i+1)$;
12 **end**
13 **return** u, i, $\mathbf{U}_{t_1}$;

where $f(\cdot)$ is the kinematic model (1) and $\hat{u}_k$ denotes the steering command used during prediction. In this work, $\hat{u}_k$ is generated by the currently available control policy without re-solving the OCP, e.g., using the learned local feedback mapping which is derived from the previous MPC solution (MPC with linear interval control, see Section III for details), or by taking the remaining inputs of the previously optimized control sequence with a zero-order-hold strategy when the sequence is exhausted (conventional event-triggered MPC). The look-ahead position is then obtained from $\bar{x}_H$ ($(\bar{p}_{x,t+T_L}, \bar{p}_{y,t+T_L}) = (\bar{x}_H(1), \bar{x}_H(2))$. In our implementation, $T_L = 1\tilde{}s$ and $\Delta t = 0.2\,\mathrm{s}$, resulting in a 5-step rollout.

III. Event-Triggered MPC With Linear Inter-Event Control

Since conventional event-triggered MPC, as described in Algorithm 1, relies on open-loop control in between two MPC events (Line 11), it can be prone to model mismatch and external disturbance. This section proposes a new mechanism to determine the inter-event control. The proposed linear inter-event control mechanism is based on a linear relationship between the steering control and features extracted from current vehicle position, represented as $u = KP$, where

$$K = \begin{bmatrix} k_0 & k_1 & k_2 & k_3 & k_4 & k_5 & k_6 \end{bmatrix} \tag{8}$$

$$P = \begin{bmatrix} 1 & p_x & p_y & \sin\psi & \cos\psi & p_x^2 & p_y^2 \end{bmatrix}^T. \tag{9}$$

The components of P correspond to a constant bias term, planar position terms (p_x, p_y), a smooth orientation representation $(\sin\psi, \cos\psi)$ to avoid angle wrap-around, and quadratic position terms (p_x^2, p_y^2) to capture mild local nonlinearities. The feedback gain K has 7 parameters to be determined in real-time. This small size makes the gain easy to fit online from the limited data and keeps the inter-event computation low. Since the gain is only used for short inter-event corrections, and MPC is re-solved when the error grows, a more complex model is not necessary. If needed, more features can be added to increase the dimension of P and K.

In other words, in between two MPC events, the control u is linear on the feature vector P with feedback gain K being

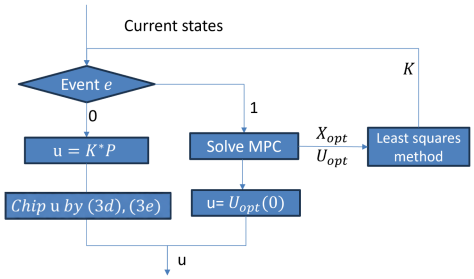

Fig. 2. Flow chart of the proposed event-triggered MPC with linear inter-event control.

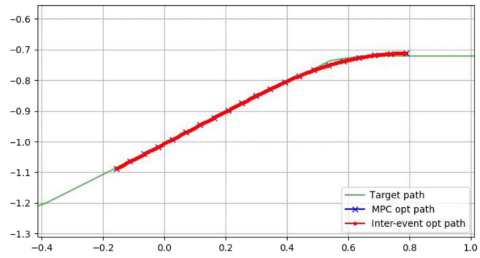

Fig. 3. Demonstration of the proposed inter-event control during lane change maneuver.

updated each time an MPC is triggered. More specifically, when the OCP (3) is solved, denote

$$\mathbf{P}_{\text{opt}} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ p_{x,0} & p_{x,1} & \cdots & p_{x,p-1} \\ p_{y,0} & p_{y,1} & \cdots & p_{y,p-1} \\ \sin(\psi_o) & \sin(\psi_1) & \cdots & \sin(\psi_{p-1}) \\ \cos(\psi_0) & \cos(\psi_1) & \cdots & \cos(\psi_{p-1}) \\ p_{x,0}^2 & p_{x,1}^2 & \cdots & p_{x,p-1}^2 \\ p_{y,0}^2 & p_{y,1}^2 & \cdots & p_{y,p-1}^2 \end{bmatrix}^T, \quad (10)$$

and the optimal input sequence over the prediction horizon as

$$\mathbf{U}_{\text{opt}} = \begin{bmatrix} u_0 & u_1 & \cdots & u_{p-1} \end{bmatrix}^T. \quad (11)$$

Then the feedback gain $K^* = \begin{bmatrix} k_0 & k_1 & \cdots & k_6 \end{bmatrix}$ can be found by least square as follow,

$$K^* = \arg\min_K \left\| \mathbf{P}_{\text{opt}} K^T - \mathbf{U}_{\text{opt}} \right\|^2, \quad (12)$$

where matrix $\mathbf{P}_{\text{opt}}$ as defined in (10) is computed based on the predicted optimal vehicle position over the prediction horizon as calculated by OCP (3). The least squares solution is computed via

$$K^* = \mathbf{P}_{\text{opt}}^{\dagger} \mathbf{U}_{\text{opt}}, \quad (13)$$

where $(\cdot)^{\dagger}$ denotes the Moore-Penrose pseudoinverse which is computed via SVD. This avoids explicitly inverting $\mathbf{P}_{\text{opt}}^{\top}\mathbf{P}_{\text{opt}}$ and improves numerical robustness under potential feature collinearity.

Remark 2: Although K^* is identified using the MPC predicted optimal segment $(\mathbf{X}_{\text{opt}}, \mathbf{U}_{\text{opt}})$ the inter-event control is applied in closed loop using the current state at every time step, i.e.,

$$u_t = K^* P(\hat{x}_t), \quad (14)$$

where $P(\hat{x}_t)$ is recomputed from the current state $\hat{x}_t$, and is defined as

$$P(\hat{x}_t) = \begin{bmatrix} 1 & \hat{p}_{x,t} & \hat{p}_{y,t} & \sin\hat{\psi}_t & \cos\hat{\psi}_t & \hat{p}_{x,t}^2 & \hat{p}_{y,t}^2 \end{bmatrix}^T. \quad (15)$$

Fig. 2 illustrates the flow chart of the proposed event-triggered MPC framework integrated with linear inter-event control. The control system operates in two modes, switching dynamically based on a predefined event-trigger condition: a full MPC optimization is executed when the event is triggered, while a lightweight linear feedback control is applied when the event is not triggered.

The proposed event-triggered MPC framework integrated with linear inter-event control is summarized in Algorithm 2. At each time step, the current system state is evaluated against the event-trigger condition in Line 2. If the condition is satisfied (e.g., deviation from the reference exceeds a specified threshold), the system solves an MPC optimization problem to obtain the new optimal state and control sequences $(\mathbf{X}_{\text{t}}, \mathbf{U}_{\text{t}})$ in Line 7. The first control input $u = \mathbf{U}_{\text{t}}(1)$ is then applied to the system in Line 8. Simultaneously, in Lines 9-11 the optimal trajectory and optimal control sequence are passed to the Least Squares Module (LSM), where updated feedback gain K^* is computed using the most recent data. These coefficients are used to define a control command in between two events.

When the event-trigger condition is not met, i.e., the system state remains within acceptable error bounds, the controller bypasses MPC computation and instead generates the control input using the current vehicle position and the previously estimated feedback gain (Lines 13-14). On Line 13, $P(\hat{x}_t)$ denotes a feature vector computed from the current estimated vehicle state $\hat{x}_t = [\hat{p}_{x,t}, \hat{p}_{y,t}, \hat{\psi}_t]^T$ at each control step (thus P is updated whenever $\hat{x}_t$ is updated). In addition, input constraints (3d) and (3e) are enforced by clipping the value in Lines 15-18.

This hybrid control structure offers a favorable balance between control performance and computation efficiency. By reducing the frequency of expensive MPC computations, the computational and communication overheads are lowered while adequate tracking performance can be maintained. As a result, it is particularly well-suited for real-time vehicle path tracking applications in embedded systems with limited resources.

Example 1: Fig. 3 illustrates an example of the proposed event-triggered MPC with linear inter-event control applied during the lane change scenario. At a certain time step, an MPC is triggered, and an optimal state sequence is obtained. The blue-cross indicate the predicted optimal trajectory points generated by the MPC over the prediction horizon. The red-dot line denotes the predicted vehicle trajectory if the linear feedback gain K^* from the optimal input and state trajectories is used. It can be seen that this trajectory closely follows the original MPC prediction. Even in high curvature segment, the linear control

Algorithm 2: Event-triggered MPC with linear inter-event control for AV path tracking.

Initialization: $i \leftarrow 0$; $K^* \leftarrow 0$;
Procedure eMPC/K($\hat{x}_t$, i, $\mathbf{U}_t$, u_{pre}, K^*);
$i \leftarrow i + 1$;
$e \leftarrow$ computing (6);
if $e = 1$ **then**
  $i \leftarrow 0$;
  $(\mathbf{X}_t, \mathbf{U}_t) \leftarrow$ Solving OCP (3);
  $u \leftarrow \mathbf{U}_t(1)$;
  $\mathbf{U}_{opt} \leftarrow \mathbf{U}_t$;
  $\mathbf{P}_{opt} \leftarrow$ computing (10);
  $K^* \leftarrow$ computing (12);
else
  $P \leftarrow P(\hat{x}_t)$;
  $u \leftarrow$ computing (14);
  $u \leftarrow \text{clip}(u, u_{min}, u_{max})$;
  $\delta u \leftarrow u - u_{pre}$;
  $\delta u \leftarrow \text{clip}(\delta u, du_{min}, du_{max})$;
  $u \leftarrow u_{pre} + \delta u$
end
$u_{pre} \leftarrow u$;
return u, i, u_{pre}, K^*;

can still keep the vehicle close to the optimal path without solving a new optimal control problem. This example demonstrates the feasibility and tracking accuracy of the proposed inter-event control, particularly in short prediction intervals and locally smooth regions of the path.

Remark 3: During inter-event periods, Algorithm 2 uses a lightweight state-feedback law learned from the most recent MPC solution, providing closed-loop correction instead of open-loop execution. To ensure closed-loop stability, we first trigger MPC once the tracking error or anticipation tracking error exceeds the threshold, and second enforce a maximum inter-event duration $i_{\max}$ so that MPC is solved at least once every $i_{\max}$ steps. These safeguards bound the deviation and limit operation without re-optimization, consistent with the bounded tracking errors observed in experiments.

Remark 4: In this letter, the linear form $u = KP$ is adopted mainly due to its fast online identification and low computation cost. Since only a limited data is available at each trigger instance, a simple linear model helps avoid overfitting and can be estimated via least squares as in (12). The gain K is intended as a local approximation of the MPC policy over short inter-event periods; the event-trigger condition (6) (together with the bound i_{max}) supervises this approximation by re-solving MPC when deviations grow. More general parametric nonlinear policies $u = f(P)$ are possible, but they typically require more data and higher computation for online training, this extension is then left for future work.

IV. Results and Discussions

While Example 1 demonstrate the proposed approach during offline analysis, this section presents experimental results to validate its efficacy.

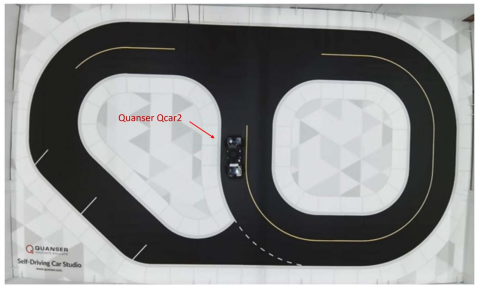

Fig. 4. Experiment environment with Quanser Qcar2 Self-Driving Studio.

TABLE I
MPC Parameters

N	6	Q_u	1	u_{min} (rad)	-0.97
dt (ms)	500	Q_d	1	du_{max} (rad)	0.15
Q_p	20	u_{max} (rad)	0.97	du_{min} (rad)	-0.15

A. Experiment Setup

To validate the proposed event-triggered MPC with linear inter-event control, experiments are conducted on a Quanser QCar2 1/10-scale autonomous vehicle platform, see Fig. 4. The QCar2 integrates an onboard NVIDIA Jestion AGS Orin computer and sensor suite for autonomous driving research. The vehicle states (x, y, ψ) are obtained from the onboard sensing and state estimation pipeline. Specifically, wheel odometry, IMU, and 2D LiDAR measurements are fused via an EKF to provide real-time pose and velocity feedback to the controller. Control commands are computed onboard in real time and transmitted to the drive and steering actuators through the QCar2 low-level interface. Throughout all tests, the longitudinal speed is kept at a constant value, and MPC is applied to lateral control. Experiments are performed in an indoor environment, where the reference trajectory is predefined and tracked repeatedly under different triggering thresholds.

Table I shows the MPC parameters used in all experiments. The prediction horizon is set to $N = 6$ with a sampling period of $dt = 500$ ms. The cost weights penalize position tracking error (Q_p), control effort (Q_u), and steering rate (Q_d). The steering angle and steering-rate constraints are enforced by $u \in [u_{min}, u_{max}]$ and $du \in [du_{min}, du_{max}]$, respectively, to ensure actuator feasibility and smooth control actions. Real vehicle experiments inherently include unmodeled disturbances and uncertainties (e.g., road–tire friction variations, small surface irregularities, actuator delays, and sensor noise), so the reported results reflect performance under realistic disturbances rather than an ideal disturbance-free setting.

Remark 5: Given the limited onboard computation budget, we set $N = 6$ and use a sparse discretization $dt = 0.5$ s, yielding a fixed prediction window of 3s with a reduced number of optimization variables of MPC. This prediction horizon is necessary to anticipate upcoming curvature changes and maintain stable tracking. In a standard event-triggered MPC baseline, using $dt = 0.5$s would imply that, with the absence of an event, the control input is updated only at 2 Hz if the controller follows the MPC sampling rate, which is inconsistent with the high-rate

TABLE II
TIME-TRIGGERED MPC TRACKING ERROR AND TRIGGERING STATISTICS OVER 10 LAPS (MEAN ± STD OVER 10 LAPS) ACROSS DIFFERENT SPEEDS

v (m/s)	RMSE (m)	Mean Error (m)	Event freq. (Hz)	# of Events pre lap
0.32	0.0144 ± 0.0019	0.0128 ± 0.0014	18.76 ± 0.72	1039.57 ± 23.6
0.26	0.0142 ± 0.0012	0.0112 ± 0.0009	19.36 ± 1.36	1351.18 ± 45.4
0.20	0.0151 ± 0.0011	0.0118 ± 0.0011	19.61 ± 0.96	1676.75 ± 62.4

TABLE III
TRACKING PERFORMANCE OVER 10 LAPS (MEAN ± STD) AND RELATIVE IMPROVEMENT OF EMPC/K OVER EMPC FOR EACH (σ, v) SETTING

σ	v (m/s)	RMSE (m)		Mean Eerror (m)		Improvement (%)	
		eMPC	eMPC/K	eMPC	eMPC/K	ΔRMSE	ΔMean
0.02	0.32	0.0154 ± 0.0009	0.0148 ± 0.0013	0.0126 ± 0.0008	0.0118 ± 0.0010	3.39	6.24
0.02	0.26	0.0135 ± 0.0008	0.0132 ± 0.0012	0.0110 ± 0.0008	0.0105 ± 0.0009	1.82	4.21
0.02	0.20	0.0154 ± 0.0034	0.0144 ± 0.0011	0.0114 ± 0.0022	0.0113 ± 0.0009	6.91	1.51
0.04	0.32	0.0203 ± 0.0019	0.0186 ± 0.0032	0.0155 ± 0.0016	0.0144 ± 0.0020	8.38	6.97
0.04	0.26	0.0172 ± 0.0013	0.0165 ± 0.0011	0.0145 ± 0.0012	0.0134 ± 0.0009	4.08	7.96
0.04	0.20	0.0189 ± 0.0024	0.0178 ± 0.0016	0.0151 ± 0.0023	0.0140 ± 0.0014	5.92	7.35
0.06	0.32	0.0290 ± 0.0039	0.0239 ± 0.0026	0.0236 ± 0.0034	0.0185 ± 0.0021	17.73	21.73
0.06	0.26	0.0250 ± 0.0023	0.0244 ± 0.0016	0.0219 ± 0.0017	0.0192 ± 0.0013	7.52	12.28
0.06	0.20	0.0268 ± 0.0011	0.0259 ± 0.0023	0.0237 ± 0.0011	0.0218 ± 0.0022	10.04	8.02

TABLE IV
COMPUTATION AND TRIGGERING STATISTICS OVER 10 LAPS (MEAN ± STD) FOR EACH (σ, v) SETTING

σ	v (m/s)	Event freq. (Hz)		# of Events per lap	
		eMPC	eMPC/K	eMPC	eMPC/K
0.02	0.32	10.003 ± 0.591	6.245 ± 1.154	552.9 ± 49.9	349.3 ± 66.6
0.02	0.26	7.914 ± 0.469	4.389 ± 0.716	546.1 ± 34.2	287 ± 95.2
0.02	0.20	8.653 ± 1.180	5.534 ± 0.755	610.2 ± 53.3	476.6 ± 65.0
0.04	0.32	3.563 ± 0.870	1.424 ± 0.482	210.1 ± 44.9	81.6 ± 18.2
0.04	0.26	2.740 ± 0.598	0.819 ± 0.155	186.7 ±40.6	56.7 ± 11.2
0.04	0.20	2.797 ± 0.931	1.268 ± 0.891	252.2 ± 69.4	121.0 ± 34.6
0.06	0.32	2.656 ± 1.126	0.954 ± 0.563	148.3 ± 63.6	54.1 ± 31.8
0.06	0.26	1.370 ± 0.419	0.885 ± 0.276	85.8 ± 25.6	60.6 ± 20.3
0.06	0.20	0.952 ± 0.161	0.707 ± 0.350	84.8 ± 14.7	66.2 ± 36.8

command streaming of the QCar2 low-level interface. To decouple command streaming from the MPC update rate and to enable a fair real-time comparison between conventional eMPC and the proposed eMPC with linear inter-event control (eMPC/K), we implement a time-indexed zero-order-hold (ZOH) for the eMPC baseline: when there is no event, the previously optimized input sequence is replayed according to the elapsed time since the last solve, while commands are streamed at the outer-loop rate. As a result, both eMPC and eMPC/k operate under the same high-rate interface, and the observed performance differences are attributable to the inter-event control strategy rather than implementation timing.

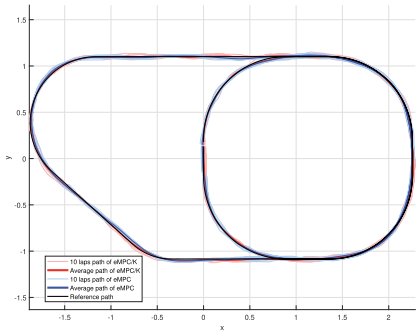

Fig. 5. Path tracking performance with event-trigger threshold $\sigma = 0.02$ m, and $v = 0.32$ m/s, the black curve is the reference path, and the blue and red lines show the vehicle trajectories under eMPC and eMPC/K, respectively.

B. Test Results

We use the baseline time-triggered MPC for comparison, and evaluate the event-triggered MPC (eMPC) and the proposed event-triggered MPC with linear inter-event control (eMPC/K) under three event-trigger thresholds, $\sigma \in 0.02, 0.04, 0.06$. Each configuration is tested at three different speeds $v \in \{0.2, 0.26, 0.32\}$ (m/s), and the QCar2 completes 10 laps for each setting. Tables II–IV report the numerical results for all settings, where each entry is computed over 10 laps and presented as mean ± standard deviation and the improvement of eMPC/K in percentage.

As a representative example, Fig. 5–7 present the path-tracking results at $v = 0.32$ m/s under three triggering thresholds $\sigma \in \{0.02, 0.04, 0.06\}$. In each figure, the black curve denotes the reference path, the light blue and light red curves show the individual lap trajectories for eMPC and eMPC/K, respectively, and the solid blue and solid red curves represent the corresponding average trajectories over 10 laps. Overall, both controllers reliably track the reference route and remain close to the reference across repeated laps. When the threshold is strict ($\sigma = 0.02$), the lap trajectories are tightly clustered and the average paths largely overlap with the reference path, indicating consistent tracking and good repeatability. As σ increases to 0.04 and 0.06, the trajectory envelope becomes wider, with larger

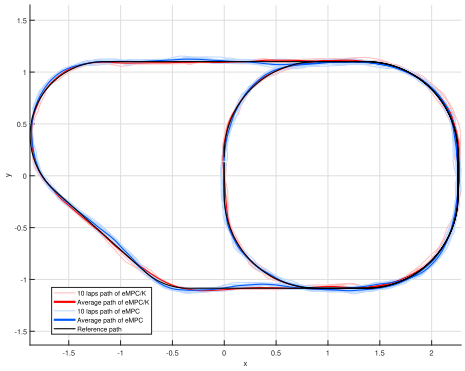

Fig. 6. Path tracking performance with event-trigger threshold $\sigma = 0.04$ m, and $v = 0.32$ m/s, the black curve is the reference path, and the blue and red lines show the vehicle trajectories under eMPC and eMPC/K, respectively.

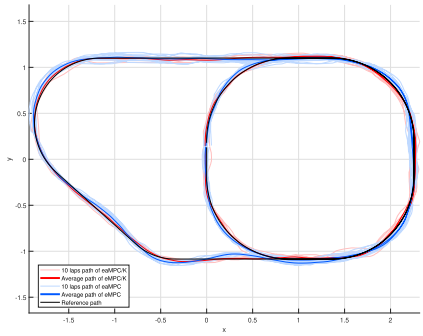

Fig. 7. Path tracking performance with event-trigger threshold $\sigma = 0.06$ m, and $v = 0.32$ m/s, the black curve is the reference path, and the blue and red lines show the vehicle trajectories under eMPC and eMPC/K, respectively.

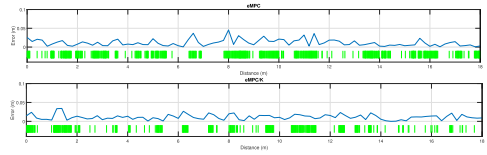

Fig. 8. Tracking error and MPC activation when $\sigma = 0.02$ under different frameworks. Vertical lines indicate MPC computation.

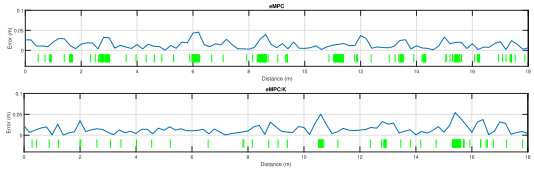

Fig. 9. Tracking error and MPC activation when $\sigma = 0.04$ under different frameworks. Vertical lines indicate MPC computation.

visible deviations occur mainly in high-curvature and transition regions, which is consistent with less frequent replanning under a relaxed triggering condition. Across all three thresholds, the average trajectory of eMPC/K remains comparable to the eMPC, and eMPC/K is slightly closer to the reference path, indicating that the linear inter-event control can limit the accumulation of deviation between MPC activations while preserving stable closed-loop behavior at the tested speed.

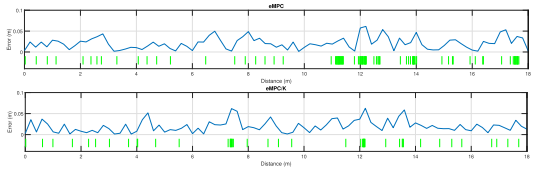

Fig. 10. Tracking error and MPC activation when $\sigma = 0.06$ under different frameworks. Vertical lines indicate MPC computation.

Figs. 8–10 show one randomly selected run at $v = 0.32$ m/s. In each figure, the top plot is eMPC and the bottom plot is eMPC/K. The blue line is the lateral tracking error along the traveled distance, and the green vertical lines mark when the controller actually solves an MPC problem. At $\sigma = 0.02$, the green lines are very dense for both methods, meaning MPC is solved frequently. In this case, both controllers keep the error relatively small and the two error curves look similar, since both are constantly re-planning. When the threshold increases to $\sigma = 0.04$, the green lines become much sparser. The error still stays bounded, but occasional peaks become more visible because the controller allows larger deviation before re-solving MPC. Compared with eMPC, eMPC/K reaches a similar error level with fewer green lines, showing that the linear inter-event control can increase the tracking performance between MPC events. At $\sigma = 0.06$, MPC is triggered only occasionally. Even then, eMPC/K still keeps the error in a comparable range while requiring fewer MPC solves, whereas eMPC shows more clustered MPC activations around segments where the error rises. Overall, these plots illustrate that eMPC/K can reduce MPC computations while keeping tracking error at a similar level, especially when the triggering threshold is relaxed.

Table II summarizes the tracking accuracy and MPC triggered frequency of time-triggered MPC, and shows time-triggered MPC has stable tracking across the tested speeds. As reported in Table II, the RMSE and the mean error remains in a narrow range indicating a steady accuracy at v$\in\{0.20,0.26,0.32\}$ m/s. As expected for periodic updates, the controller maintains a nearly constant update frequency of about 19–20 Hz for all speeds, leading to a large number of MPC updates per lap. This table therefore provides a computation-heavy baseline for comparison: time-triggered MPC achieves consistent tracking accuracy but requires dense optimizations throughout the entire run.

Table III summarizes the tracking accuracy in terms of RMSE and mean lateral error, reported as mean $\pm$ standard deviation over 10 laps for each (σ, v) configuration. Across the tested thresholds and speeds, eMPC/K achieves tracking performance comparable to that of eMPC, while often providing modest improvements. The benefit becomes increasingly evident as the triggering condition is relaxed. When averaged over the three tested speeds for each threshold, eMPC/K reduces the mean lateral error by 3.9% and the RMSE by 4.1% at $\sigma = 0.02$. At $\sigma = 0.04$, these average reduction increases to 7.4% for mean lateral error and 6.1% for RMSE. The largest gains are obtained at $\sigma = 0.06$, where the mean lateral error is reduced by 14.0% on average and the RMSE by 11.8%. The reported standard

deviations are generally small relative to the corresponding mean errors, indicating that the tracking accuracy is consistent across the 10 repeated laps for each (σ, v) setting.

Table IV summarized the MPC event frequency and the number of MPC events per lap as mean $\pm$ std over 10 laps. For a fixed (σ, v), eMPC/K consistently reduces event frequency compared to eMPC, indicating that the linear inter-event control significant reduces the frequency of the MPC computation.

V. Conclusion

In this letter, an event-triggered model predictive control (MPC) framework with integrated linear inter-event control is proposed for autonomous vehicle path tracking. By using a least-squares-based linear feedback gain to generate control actions during inter-event periods, the proposed method achieves real-time error correction between MPC updates, improving tracking accuracy while reducing the number of computationally expensive MPC optimizations. Experimental results demonstrate that, compared to conventional event-triggered MPC, the proposed approach maintains better tracking performance with fewer MPC activations, thus offering an efficient solution suitable for real-time embedded systems. Future work will include broader validation and analysis. We plan to evaluate the method for more diverse trajectories, and to conduct controlled disturbance tests. We will also add additional baselines such as periodic MPC with matched solver budgets and simple fixed-gain inter-event controllers. Finally, we will validate the approach on full-scale autonomous vehicle platforms under real-world operating conditions.

References

[1] M. Reda, A. Onsy, A. Y. Haikal, and A. Ghanbari, "Path planning algorithms in the autonomous driving system: A comprehensive review," *Robot. Auton. Syst.*, vol. 174, 2024, Art. no. 104630.

[2] D. C. Herraez, M. Zeller, D. Wang, J. Behley, M. Heidingsfeld, and C. Stachniss, "RaI-SLAM: Radar-inertial SLAM for autonomous vehicles," *IEEE Robot. Autom. Lett.*, vol. 10, no. 6, pp. 5257–5264, Jun. 2025.

[3] L. Guvenc, B. Aksun-Guvenc, S. Zhu, and S. Y. Gelbal, *Autonomous Road Vehicle Path Planning and Tracking Control*. Hoboken, NJ, USA: Wiley, 2021.

[4] Z. Xu et al., "DriveGPT4: Interpretable end-to-end autonomous driving via large language model," *IEEE Robot. Autom. Lett.*, vol. 9, no. 10, pp. 8186–8193, Oct. 2024.

[5] L. Paull et al., "Duckietown: An open, inexpensive and flexible platform for autonomy education and research," in *Proc. IEEE Int. Conf. Robot. Autom*, Singapore, May 2017, pp. 1497–1504.

[6] C. Rother, Z. Zhou, and J. Chen, "Development of a four-wheel steering scale vehicle for research and education on autonomous vehicle motion control," *IEEE Robot. Autom. Lett.*, vol. 8, no. 8, pp. 5015–5022, Aug. 2023.

[7] Q. Yao, Y. Tian, Q. Wang, and S. Wang, "Control strategies on path tracking for autonomous vehicle: State of the art and future challenges," *IEEE Access*, vol. 8, pp. 161211–161222, 2020.

[8] M. Samuel, M. Hussein, and M. B. Mohamad, "A review of some pure-pursuit based path tracking techniques for control of autonomous vehicle," *Int. J. Comput. Appl.*, vol. 135, no. 1, pp. 35–38, 2016.

[9] A. AbdElmoniem, A. Osama, M. Abdelaziz, and S. A. Maged, "A path-tracking algorithm using predictive Stanley lateral controller," *Int. J. Adv. Robot. Syst.*, vol. 17, no. 6, 2020, Art. no. 1729881420974852.

[10] G. Antonelli, S. Chiaverini, and G. Fusco, "A fuzzy-logic-based approach for mobile robot path tracking," *IEEE Trans. Fuzzy Syst.*, vol. 15, no. 2, pp. 211–221, Apr. 2007.

[11] J. E. Normey-Rico, I. Alcalá, J. Gómez-Ortega, and E. F. Camacho, "Mobile robot path tracking using a robust PID controller," *Control Eng. Pract.*, vol. 9, no. 11, pp. 1209–1214, 2001.

[12] D. Chu, H. Li, C. Zhao, and T. Zhou, "Trajectory tracking of autonomous vehicle based on model predictive control with PID feedback," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2239–2250, Feb. 2023.

[13] M. Elgouhary and A. S. El-Wakeel, "Learning to tune pure pursuit in autonomous racing: Joint lookahead and steering-gain control with PPO," *IEEE Robot. Autom. Lett.*, vol. 11, no. 4, pp. 5041–5048, Apr. 2026.

[14] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *Proc. IEEE Intell. Veh. Symp.*, Seoul, Korea, Jun./Jul. 2015, pp. 1094–1099.

[15] S. Di Cairano, H. E. Tseng, D. Bernardini, and A. Bemporad, "Vehicle yaw stability control by coordinated active front steering and differential braking in the tire sideslip angles domain," *IEEE Trans. Control Syst. Tech.*, vol. 21, no. 4, pp. 1236–1248, Jul. 2013.

[16] R. Yu, H. Guo, Z. Sun, and H. Chen, "MPC-based regional path tracking controller design for autonomous ground vehicles," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Hong Kong, China, Oct. 2015, pp. 2510–2515.

[17] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, "Review on model predictive control: An engineering perspective," *Int. J. Adv. Manuf. Tech.*, vol. 117, no. 5, pp. 1327–1349, 2021.

[18] M. Brown, J. Funke, S. Erlien, and J. C. Gerdes, "Safe driving envelopes for path tracking in autonomous vehicles," *Control Eng. Pract.*, vol. 61, pp. 307–316, 2017.

[19] H. Liu, J. Sun, and K. W. E. Cheng, "A two-layer model predictive path-tracking control with curvature adaptive method for high-speed autonomous driving," *IEEE Access*, vol. 11, pp. 89228–89239, 2023.

[20] Z. Sun, C. Li, J. Zhang, and Y. Xia, "Dynamic event-triggered MPC with shrinking prediction horizon and without terminal constraint," *IEEE Trans. Cybern.*, vol. 52, no. 11, pp. 12140–12149, Nov. 2022.

[21] F. Dang, D. Chen, J. Chen, and Z. Li, "Event-triggered model predictive control with deep reinforcement learning for autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, pp. 459–468, Jan. 2024.

[22] Z. Zhou, C. Rother, and J. Chen, "Event-triggered model predictive control for autonomous vehicle path tracking: Validation using CARLA simulator," *IEEE Trans. Intell. Veh.*, vol. 8, no. 6, pp. 3547–3555, Jun. 2023.

[23] J. Chen and Z. Yi, "Comparison of event-triggered model predictive control for autonomous vehicle path tracking," in *Proc. IEEE Conf. Control Technol. Appl.*, San Diego, CA, USA, Aug. 2021, pp. 808–813.

[24] H. Li and Y. Shi, "Event-triggered robust model predictive control of continuous-time nonlinear systems," *Automatica*, vol. 50, no. 5, pp. 1507–1513, 2014.

[25] J. Yoo and K. H. Johansson, "Event-triggered model predictive control with a statistical learning," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 51, no. 4, pp. 2571–2581, Apr. 2021.

[26] 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 Trans. Autom. Control*, vol. 62, no. 1, pp. 177–189, Jan. 2017.

[27] Z. Zhou, J. Chen, M. Tao, P. Zhang, and M. Xu, "Experimental validation of event-triggered model predictive control for autonomous vehicle path tracking," in *Proc. IEEE Int. Conf. Electro Inf. Technol.*, 2023, pp. 35–40.

[28] W. Jallet, E. Dantec, E. Arlaud, N. Mansard, and J. Carpentier, "Parallel and proximal constrained linear-quadratic methods for real-time nonlinear MPC," *Robot.: Sci. Syst.*, Jul. 2024.

[29] G. S. Watson, "Linear least squares regression," *Ann. Math. Statist.*, vol. 38, no. 6, pp. 1679–1699, Dec. 1967.

[30] J. Chen, X. Meng, and Z. Li, "Reinforcement learning-based event-triggered model predictive control for autonomous vehicle path following," in *Proc. Amer. Control Conf.*, Atlanta, GA, USA, Jun. 2022, pp. 3342–3347.

[31] E. V. Kumar and J. Jerome, "Robust LQR controller design for stabilizing and trajectory tracking of inverted pendulum," *Procedia Eng.*, vol. 64, pp. 169–178, 2013.

[32] B. T. Lopez, J.-J. E. Slotine, and J. P. How, "Dynamic tube MPC for nonlinear systems," in *Proc. Amer. Control Conf.*, 2019, pp. 1655–1662.

[33] X. Zhu, Z. Li, Y. Jiang, J. Xu, J. Wang, and X. Bai, "Real-time vehicle-to-vehicle communication-based network cooperative control system through distributed database and multimodal perception: Demonstrated in cross-roads," in *Proc. Int. Congr. Inf. Commun. Technol.*, Berlin, Germany: Springer, 2024, pp. 133–146.

[34] S. Huang and J. Chen, "Event-triggered model predictive control for autonomous vehicle with rear steering," SAE Tech. Paper, no. 2022-01-0877, 2022.