ASME Letters in Dynamic Systems and Control
Online journal at:
https://asmedigitalcollection.asme.org/lettersdynsys

Comparison of Two-Wheel and Four-Wheel Steering Using Event-Triggered Predictive Motion Control and Scale Vehicles

Zhaodong Zhou
Department of Electrical
and Computer Engineering,
Oakland University,
Rochester, MI 48309
e-mail: zhaodongzhou@oakland.edu

Christopher Rother
Department of Electrical
and Computer Engineering,
Oakland University,
Rochester, MI 48309
e-mail: crother@oakland.edu

Jun Chen[1]
Department of Electrical
and Computer Engineering,
Oakland University,
Rochester, MI 48309
e-mail: junchen@oakland.edu

This study compares the trajectory tracking performance of two- and four-wheel steering systems, especially under normal driving conditions. Specifically, the lateral motion is controlled by an event-triggered model predictive control (MPC), which activates either when consecutive control steps surpass the predictive horizon or when tracking error exceeds a predetermined lateral offset. Using a modified 1/10th scale Tamiya TT-02 RC car as a test platform, the tracking performance of both two- and four-wheel systems are evaluated. Results from the experiments highlight the better tracking performance of the four-wheel steering system over the traditional two-wheel systems and demonstrate the benefit of using event-triggered MPC for lateral motion control even under normal driving conditions, contrary to common belief that four-wheel steering systems are beneficial only in tight steering maneuvers. [DOI: 10.1115/1.4065093]

Keywords: event triggered MPC, four-wheel steering system, scale vehicle, autonomous vehicle, path tracking, automotive systems, control applications, model predictive control

1 Introduction

With the rising popularity of electric vehicles, autonomous driving technology is also advancing rapidly [1,2]. In the field of autonomous vehicles, steering control plays a key role in ensuring precise navigation and safety [3–5]. While traditional two-wheel steering systems have been standard for decades, the advent of advanced technologies has brought four-wheel steering systems to the forefront as an alternative. Four-wheel steering aims to enhance handling, stability, and maneuverability compared to conventional two-wheel steering [6]. In four-wheel steering systems, both the front and rear wheels can be steered simultaneously. This provides an additional degree-of-freedom and control authority over the vehicle's lateral dynamics, which enhances the vehicle's maneuverability at slower speeds and increases stability at higher speeds [7].

A variety of control algorithms have been proposed for four-wheel steering vehicles, including linear quadratic control [8,9], decoupling control [10], and H^∞ control [11]. Moreover, model predictive control (MPC) has been widely investigated in the context of autonomous driving, with the main advantage being its capability to solve complex optimization problems and constraints [12–15]. MPC can handle the highly nonlinear and complex dynamical processes during driving maneuvers. However, the optimization problem solved at each control step requires significant computation. Therefore, techniques to reduce the computation time are essential for real-time feasibility. To this regard, event-triggered MPC has been proposed to decrease the computation requirements of MPC [16,17]. Rather than performing optimization and updating the control at every step, the optimization is only solved when needed based on the system states satisfying a triggering condition. The stability and feasibility of the event-triggered MPC are extensively addressed in detail in papers [18,19]. These studies provide conditions under which the event-triggered MPC is feasible and stable. For example, the stability can be established by utilizing the Lyapunov stability theory with the cost function as Lyapunov functions. The efficacy of event-triggered MPC has been demonstrated in path-tracking problems using simulation [5] and vehicle experiments [20], demonstrating its benefits of reducing computation without major performance degradation.

However, these prior works have been focused on two-wheel steering systems only. In the authors' previous work [21], MPC was implemented in a four-wheel steering system to test its advantage over two-wheel steering systems. However, Ref. [21] only considered tight steering maneuver, where the benefit of using four-wheel steering is rather obvious. This article builds upon the previous work and focuses on driving conditions that *do not require excessive steering*, examining whether four-wheel steering can still offer benefits. To enable experimental validation, a customized scaled vehicle platform was developed based on a 1/10th scale Tamiya TT-02 RC model car [22]. The major addition was an actuated rear-wheel steering mechanism to transform the vehicle from

[1] Corresponding author.

Manuscript received December 18, 2023; final manuscript received March 12, 2024; published online April 3, 2024. Assoc. Editor: Wen-Chiao Lin.

its original two-wheel steering system into one with a four-wheel steering system. For autonomous operation, the platform was outfitted with an NVIDIA Jetson Nano single board computer to run control algorithms and a Marvelmind ultrasonic-based indoor positioning system for real-time location measurements. Experimental results demonstrate that even in normal driving conditions, four-wheel steering systems can still offer control performance improvement and computation reduction in the event-triggered control framework. Moreover, efficiency improvement in four-wheel steering vehicles is also an interesting research topic. See, for example, Refs. [23,24], where significant energy improvement is reported by using optimal torque distribution methods, without compromising stability. In the future, this could be a potential area for testing on this platform.

The remainder of this article is organized as follows. Section 2 discusses the vehicle model for the MPC and the algorithm of event-triggered MPC. Section 3 introduces the setting up of the experimental platform. The numerical results are presented in Sects. 4, and Sec. 5 concludes this article.

2 Predictive Motion Control

2.1 Kinematic Bicycle Model. In this article, we conducted tests on a scaled vehicle at low speeds. Given these conditions, the lateral tire forces have a minimal impact and can be reasonably omitted from our analysis. Please note that, for low speeds, kinematic models can be more effective for MPC, while dynamic models are generally better for high-speed maneuvers. This is due to the fact that at low speed, lateral tire forces are usually hard to accurately model, which can result in unrealistic prediction for MPC. It is also our observation through experiments that MPC with dynamic model would require a computational time that is too long for effective steering control, resulting in worse tracking performance. Therefore, in this article, we utilize a kinematic model for MPC.

The vehicle model is then represented through the kinematic bicycle model, as illustrated in Fig. 1. The state vector x is located at the vehicle's center of gravity (CG) and is defined as $x=[p_x,\ p_y,\ \psi]^T$, where p_x and p_y are the vehicle global coordinates of CG, and ψ is the heading angle in a counterclockwise direction. Then the set of difference equations of the kinematic model is given as follows:

$$p_{x,t+1}=p_{x,t}+V\cos(\psi_t+\beta_t)T_s \tag{1a}$$

$$p_{y,t+1}=p_{y,t}+V\sin(\psi_t+\beta_t)T_s \tag{1b}$$

$$\psi_{t+1}=\psi_t+\frac{V\cos(\beta_t)}{L_{xf}+L_{xr}}(\tan(\delta_{f,t})-\tan(\delta_{r,t}))T_s \tag{1c}$$

where T_s is the sampling time, V is the velocity of the vehicle's CG, L_{xf} and L_{xr} are the distances from the CG to the front and rear axle, respectively, $\delta_{f,t}$ and $\delta_{r,t}$ are the front and rear steering angles, respectively, and $\beta_t=\arctan\big((L_{xf}\tan(\delta_{r,t})+L_{xr}\tan(\delta_{f,t}))/(L_{xf}+L_{xr})\big)$ is the vehicle slip angle. The control vector can be compactly denoted as $u_t=[\delta_{f,t},\ \delta_{r,t}]^T$. Note that for the two-wheel steering system, the rear steering angle is fixed to zero, and the control vector reduces to $u_t=\delta_{f,t}$.

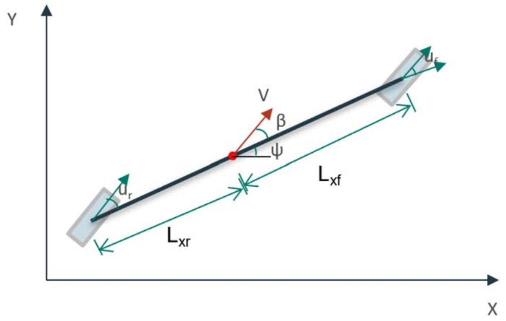


Fig. 1 Schematic of the kinematic bicycle model

2.2 Optimal Control Problem. In the context of MPC-based path-tracking control, the general MPC algorithm undertakes the following steps at a given time instance t. First, it gauges the system's current state. Then, using the system model, constraints, and the present state, it solves the optimal control problem and identifies the optimal state sequence $X_t=[x_{t+1},\ x_{t+2},\ldots,x_{t+p}]$ and the optimal control sequence $U_t=[u_t,\ u_{t+1},\ldots,u_{t+p-1}]$, where p denotes the prediction horizon. Finally, the actuators receive the initial element from the optimal control sequence. The optimal control problem (OCP) is given as follows:

$$\min_u \quad \sum_{k=1}^{p}\|x_{t+k}-x^r_{t+k}\|^2_{Q_x}+\sum_{k=0}^{p-1}\|u_{t+k}\|^2_{Q_u} + \sum_{k=0}^{p-1}\|u_{t+k}-u_{t+k-1}\|^2_{Q_d} \tag{2a}$$

$$\text{s.t.}\quad x_t=\hat{x}_t \tag{2b}$$

$$\text{System dynamic (1)},\ \ k=1,\ \ldots,p \tag{2c}$$

$$u_{\min}\le u_{t+k}\le u_{\max},\ \ k=0,\ \ldots,p-1 \tag{2d}$$

$$\Delta_{\min}\le u_{t+k}-u_{t+k-1}\le\Delta_{\max},\ \ k=0,\ \ldots,p-1 \tag{2e}$$

In the case of path-tracking control, the primary objective is to minimize the offset between the vehicle's actual and desired position. Therefore, in the cost function (2a), the first term penalizes deviation from the desired path. Meanwhile, to ensure stability, the second and third terms of cost function minimize the steering angles and the steering angle rate, respectively. The matrices Q_x, Q_u, and Q_d are the weights of path following error, steering efforts, and control activity respectively. In Eq. (2b), $\hat{x}_t$ denotes the current state feedback, meaning at time t, the initial states of the optimization problem are equal to the state feedback at time t. Equations (2c), (2d), and (2e) are the constraints of the optimal control problem, where Eq. (2c) requires that at each step the state vector follows the kinematic model (1), Eq. (2d) limits the range of steering angle, and Eq. (2e) limits the steering angle rate. In Eq. (2d) when k is equal to 0, u_{t-1} denotes the control action applied at the previous time-step. To address the optimal control problem mentioned earlier, the MPCTOOLS package along with CASADI are used, both of which are open-source optimization tools as described in Refs. [25,26]. The MPCTOOLS package is divided into three key components: an estimator, a target calculator, and a regulator. Each of these elements provides access to the solvers within CASADI.

2.3 Event-Triggered Control. To minimize the computation complexity of MPC, event-triggered MPC is investigated in this article. Unlike the typical MPC, event-triggered MPC solves the OCP (2) only when an event is triggered. This article considers the threshold-based event-trigger mechanism adopted by Ref. [27], as follows:

$$e=\begin{cases}1 & \text{if } d_y>\theta \text{ or } k>k_{\max}\\ 0 & \text{Otherwise}\end{cases} \tag{3}$$

In other words, the condition that causes the MPC to be triggered in this mechanism depends on k and θ, where k is the consecutive times that the OCP has not been solved, and θ is the maximum tolerable tracking error. It is crucial to remember that $k_{\max}$ should not be more than the prediction horizon p. In other words, the MPC is activated if the current step number k exceeds the threshold $k_{\max}$ or if the distance d_y that the vehicle deviates from the closest point on the planned path exceeds a predetermined threshold θ. In the case that $e = 1$, the current control action u is determined by solving OCP (2). In the case that $e = 0$, selecting the control action will be based on the optimal sequence U_t computed during the previous event.

Please note that, since only input constraints are consider in OCP (2), the feasibility of event-triggered MPC is automatically guaranteed if the original OCP is feasible. The stability of the event-triggered MPC has been established in the literature [18,19], by utilizing the Lyapunov stability theory with the cost function as Lyapunov functions. Please also note that though other control methods, such as proportional-integral-derivative (PID), linear quadratic regulator (LQR), or time-triggered MPC, can also be applied to demonstrate the difference between two-wheel and four-wheel steering control, event-triggered MPC is selected for investigation due to its capability to handle constraints compared to PID and LQR and its relatively low computational requirement compared to time-triggered MPC.

3 Experimental Setup

3.1 JETRACER With Rear Steering. In this article, the scale vehicle platform based on the open-source JETRACER [22] is used to validate four-wheel steering systems with event-triggered MPC, which is built from a 1/10th scale Tamiya TT-02 RC car (Fig. 2). Compared with the original Tamiya TT-02 RC car, the rear wheel steering system needs to be added. In particular, the added rear-wheel steering system is based on the front-wheel steering design with slight modification to fit the different architecture around the rear axles. Mounting points on the rear bumper originally used for cosmetic parts are repurposed to fasten replacement copies of the front-wheel bell cranks using a 3D-printed bracket with mounting pins. Because the bell cranks are positioned farther apart, the bell crank linkage design also has to be lengthened to match the separation of the mounting points and 3D printed. Replacement copies of both the steering links and drag links have to be shortened and reconnected with a 3D-printed bracket. The final lengths of the steering links must result in the wheels pointing straight forward when the steering linkage assembly is centered. After drilling holes in the rear bumper and fastening a second steering servo in place, the custom rear knuckles are attached to the suspension arms, and the aforementioned steering linkage components are assembled as shown in Fig. 3. The vehicle is driven by an electric motor powering all four wheels. Motor speed is controlled by a Tamiya electronic speed controller (ESC), and a servo multiplexer (switched from the RC transmitter) is used to select if the RC transmitter/receiver or the Jetson Nano/servo driver module supplies the control signals for the ESC and steering servos. More details about the setup of the platform can be found in Ref. [21].

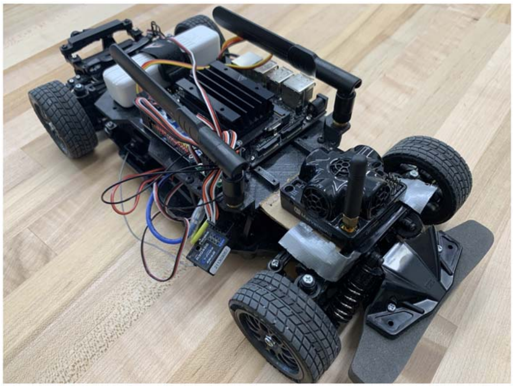


Fig. 2 The scale vehicle platform based on Tamiya TT-02 RC car

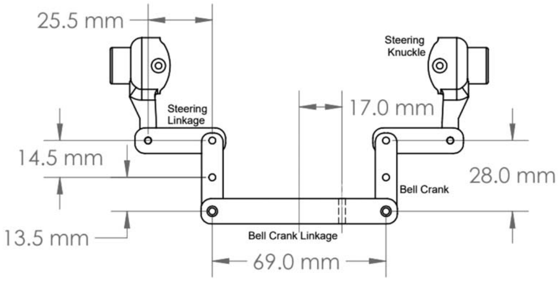
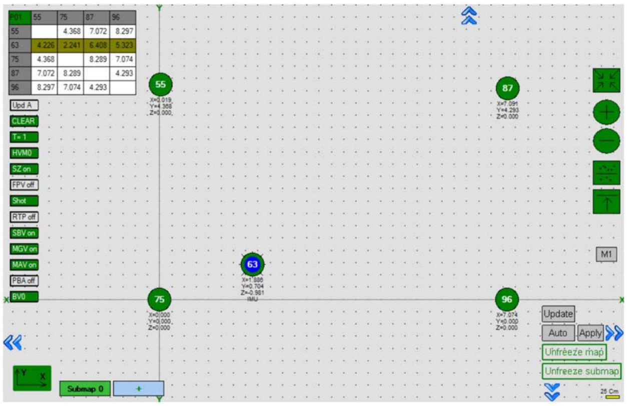


Fig. 3 Custom rear-wheel steering assembly annotated drawing



P01	55	75	87	96
55		4.368	7.072	8.297
63	4.226	2.241	6.408	5.323
75	4.368		8.289	7.074
87	7.072	8.289		4.293
96	8.297	7.074	4.293	

55 X=0.019 Y=4.368 Z=0.000

87 X=7.091 Y=4.293 Z=0.000

63 X=1.886 Y=0.704 Z=-0.981 IMU

75 X=0.000 Y=0.000 Z=0.000

96 X=7.074 Y=0.000 Z=0.000

Fig. 4 MARVELMIND DASHBOARD software showing the stationary/mobile beacon locations

To calculate vehicle state information for the MPC-based motion controls, a MARVELMIND indoor positioning system[2] is used, which consists of five beacons and a modem. One of the beacons is mounted on the scale vehicle, and the other four beacons are placed around the perimeter of the testing area, which are set as stationary beacons. Figure 4 shows the beacon positions during a typical test in the MARVELMIND software.

3.2 Delay Compensation. The system delay causes the actual position of the vehicle to be slightly ahead when control actions are applied. To address this issue, delay compensation is incorporated into the motion controls. The main causes of the delay are the time taken by the processor to update the control sequence and the latency of the MARVELMIND position data caused by filtering methods designed to reduce noise. Several measures can be taken to significantly reduce, but not entirely eliminate, latency in the MARVELMIND system. For example, including IMU data from the mobile beacon in the system results in a notable increase in latency. In this article, delay compensation is achieved by using the vehicle model to estimate the future vehicle state based on the measured vehicle state, vehicle velocity, and the current control set point. The MPC system uses this estimated vehicle state as the initial state x_0, expecting it to reflect the real vehicle state when the newly calculated control actions are applied more accurately.

3.3 PID Velocity Generation. The vehicle speed is controlled by a PID-based velocity control system, with all three (proportional, integral, and derivative) terms. However. as is typical with PID controller, the derivative term results in an amplification of noise from the error signal. To minimize this issue, a discrete low-pass filter is implemented using the form of the single-pole infinite impulse

[2] https://marvelmind.com/

response filter in Ref. [28] for both the signal and for the calculation of the derivative term. In particular, let $V = bv + aV_{-1}$, where V_{-1} is the previous filtered velocity, v is the raw velocity signal before filtering, and a and b are the filter coefficients. For this application, in addition to the standard PID terms, an additional control term, $S = K_s \delta_f^2$, is added to increase the throttle value as steering angles increase to account for sources of resistance such as reduced efficiency of u-joints.

3.4 Track Generation. The scale vehicle testing included in this article uses an oval track (two half circles connected by straight sections) defined as an array of coordinates. To simplify the process of modifying the track, a PYTHON code was written to generate the track based off of several inputs. Because the MARVELMIND data are generated using one of the stationary beacons as the origin, it is useful to specify shifts in the x and y directions, as well as a rotation angle for the track (θ). In addition, the code allows the user to set the radius of the turns (R), the length of the straight sections (L), and the number of points included in each half circle section (N).

First, coordinates for the upper half circle are generated. For each of the points from $n = 0$ to $n = N - 1$, the coordinates are calculated as $x = R \times \cos(n\pi/(N-1))$ and $y = R \times \sin(n\pi/(N-1))$. The lower half circle coordinates are the same, but with negated x and y values. Then, the distance between points (d) is calculated as $d = (\pi r)/(N-1)$. To ensure that the value of d remains constant throughout the track, the requested length of the straight sections l is modified to be the first multiple of d that exceeds the requested length. The top/bottom half circles are shifted up/down by $L/2$. The left and right straight section coordinates are generated to fill the space between the half circle sections while ensuring that the order of points results in a continuous track when the four sections are concatenated into a single array. After concatenation, the track can be rotated by multiplying each coordinate pair by a rotation matrix $R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$. Finally, all coordinates in the rotated track are shifted by the specified values to create the final array of track points used for motion controls testing.

4 Experimental Results

4.1 Controller Calibration. The calibration process presented in this article focuses on the weights Q_x, Q_u, and Q_d for the cost function. In addition, only the relative magnitude of the weight affects the solution of Eq. (2a). Since Q_x is held as constant, this study only focuses on adjusting the values of Q_u and Q_d. In the two-wheel steering mode, both the control input $u = \delta_f$ and the matrices Q_u and Q_d are scalars. Consequently, there are two calibration parameters Q_u and Q_d that need to be determined. In the scenario of four-wheel steering, the control input $u = [\delta_f, \ \delta_r]^T$ spans a two-dimensional space. Therefore, in this case, both Q_u and Q_d are two-dimensional diagonal matrices and have four calibration parameters that need to be determined, which are Q_{uf}, Q_{ur}, Q_{df}, and Q_{dr}. To generate the parameter matrix for testing, the design of experiment (DoE) method is taken into consideration. However, the factorial design method, which is the most common approach in DoE, is inadequate for problems involving four variables. Consequently, in this article, the Latin hypercube method [29], an alternative DoE approach better suited for multiple variables, is employed. The sets of parameters for the two-wheel and four-wheel steering systems are presented in Figs. 5 and 6, respectively. Both DoEs show a reasonable distribution over the design space. Note that more points are generated in the four-wheel steering system due to its large design space.

To collect data for these calibration sets, we implemented the two-wheel and four-wheel MPC controllers in separate scripts, with a target vehicle speed of 1.6 m/s. The scripts automatically iterate through the DoEs, apply the calibration for a set time, and generate data files. Between each iteration, the first calibration (known to function reasonably well from hand calibration or

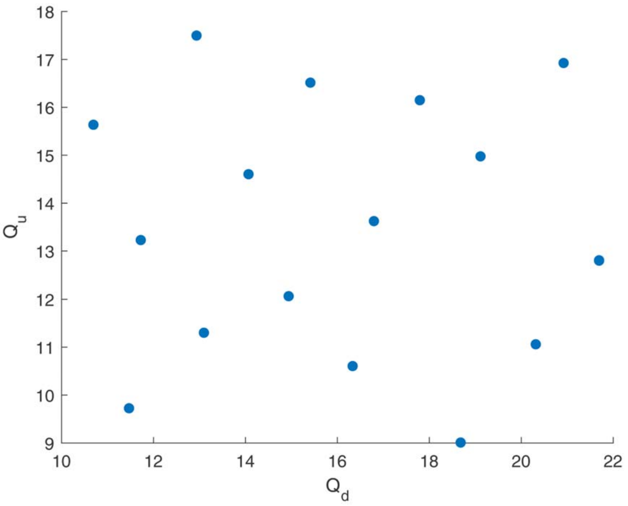


Fig. 5 Latin hypercube calibration set for the two-wheel steering mode test

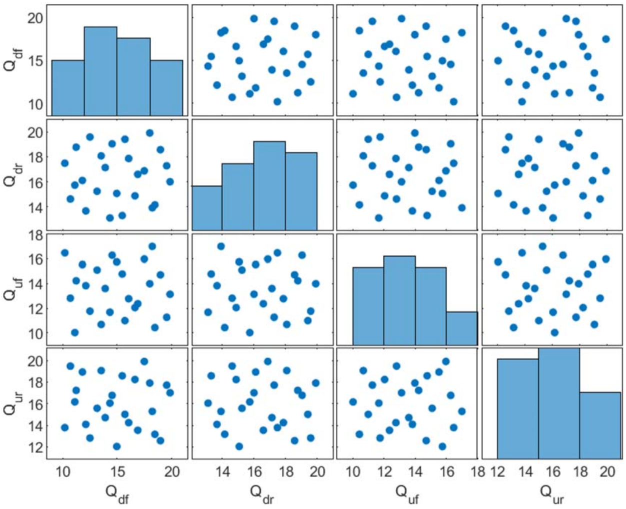


Fig. 6 Latin hypercube calibration set for the four-wheel steering mode test

Table 1 DoE results for two-wheel steering MPC track (unit: CM)

Q_d	Q_u	RMSE	Max error	I
11.47	9.73	5.7	14.6	2.65
10.69	15.64	7.5	18.0	3.37
21.69	12.81	7.6	15.7	3.15
14.94	12.07	7.0	17.1	3.17
20.92	16.93	9.0	21.4	4.03
19.11	14.98	8.9	18.3	3.68
12.94	17.50	8.6	18.7	3.66
13.10	11.30	6.1	14.9	2.77
11.72	13.23	6.1	10.3	2.28
20.32	**11.06**	**5.0**	**11.0**	**2.15**
16.79	13.62	6.5	18.0	3.18
18.68	9.01	5.7	10.9	2.28
15.41	16.51	6.3	16.4	2.97
16.33	10.61	6.2	15.7	2.86
17.79	16.15	6.4	12.8	2.60
14.07	14.60	5.3	12.1	2.31
15.60	13.20	5.8	14.7	2.69

Note: The calibrations with the best performance according to the cost index are highlighted in bold.

Table 2 DoE results for four-wheel steering MPC

Q_{df}	Q_{dr}	Q_{uf}	Q_{ur}	RMSE	Max error	I
14.00	16.00	12.00	14.00	5.3	12.0	2.32
12.50	19.58	11.76	12.83	6.3	10.4	2.33
14.95	15.08	15.75	12.05	6.0	14.0	2.66
17.98	19.90	13.98	17.91	7.3	15.8	3.11
16.87	16.60	12.36	13.54	6.6	11.5	2.51
11.81	16.11	15.53	18.93	5.0	10.1	2.05
17.47	16.88	15.97	19.90	6.2	13.8	2.67
16.05	17.86	12.77	14.25	6.9	15.7	3.02
13.53	18.08	10.68	19.06	7.0	17.8	3.26
12.12	13.68	13.83	14.09	7.0	14.9	2.95
18.21	13.93	17.00	15.29	7.9	18.0	3.46
11.24	18.77	14.23	17.23	7.1	15.2	3.01
14.33	13.12	11.67	16.05	6.3	13.5	2.67
13.16	15.25	15.09	15.58	7.5	13.8	2.93
13.90	**17.14**	**13.60**	**14.72**	**5.1**	**9.60**	**2.02**
10.71	14.63	12.81	19.47	7.2	17.3	3.24
11.12	15.74	10.02	16.17	7.4	13.8	2.91
15.70	19.40	11.00	15.01	6.6	15.6	2.94
18.96	18.57	14.71	12.58	8.0	17.8	3.46
16.62	14.88	12.05	18.23	8.4	19.0	3.65
19.53	17.28	11.26	17.72	7.3	15.8	3.11
19.83	16.01	13.13	17.01	7.6	19.2	3.52
18.45	14.17	10.43	13.18	7.6	14.4	3.01
15.46	13.31	14.77	18.58	7.3	15.1	3.03
10.20	17.49	16.47	13.79	6.4	15.4	2.89
14.53	19.04	16.28	16.77	8.0	16.4	3.30

Note: The calibrations with the best performance according to the cost index are highlighted in bold.

previous DoE results) is reapplied to allow the system to recover if the previous DoE calibration performs poorly. In addition, data collection for a DoE calibration is ended early if the vehicle deviates from the desired path beyond a predefined threshold. The results for the two-wheel steering DoE shown in Fig. 5 are presented in Table 1, and the results for the four-wheel steering DoE shown in Fig. 6 are presented in Table 2. The cost index value I is the normalization of $\overline{\text{RMSE}} + \overline{\text{Max Error}}$ with respect to the smallest value and is used to evaluate the performance of the sets of weights. The calibrations with the best performance according to the cost index are highlighted in bold, with the corresponding path-tracking performance shown in Fig. 7.

4.2 Event-Triggered Motion Control.

Table 3 compares the tracking performance of the two- and four-wheel steering systems under different values of θ (the maximum tolerable error; see Sec. 2.3). From this table, it is evident that regardless of the threshold value applied, the four-wheel steering MPC consistently achieves a lower RMSE and Max tracking error. For instance, at $\theta = 0.0$, the RMSE and Max error for the two-wheel steering system are 3.9 and 11.0, respectively, while for the four-wheel steering system, they are 3.7 and 8.6, respectively. This trend persists for higher values of θ. At $\theta = 3.5$, the RMSE and Max error for the two-wheel system rise to 12.3 and 26.8, whereas for the four-wheel system, they only increase to 9.6 and 20.3, respectively. In addition, the event-triggered mechanism in the four-wheel steering system reduces computational requirements more than in the two-wheel steering system. The results demonstrate that four-wheel steering MPC offers performance improvements over two-wheel steering MPC, and the event-trigger control mechanism provides the capability to lessen the computational load, even in normal driving conditions. This supplements the previous results [21] that focus on tight steering maneuvers only.

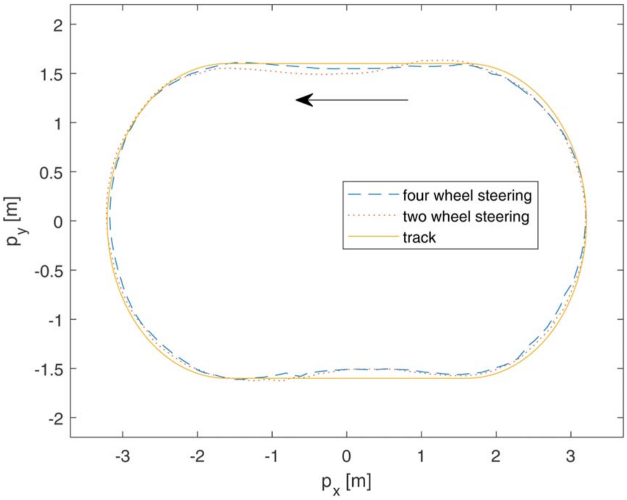


Fig. 7 Path-tracking performance for two- and four-wheel steering systems

Table 3 Tracking performance of two-wheel and four-wheel steering system (unit: CM)

θ	Two-wheel steering			Four-wheel steering		
	RMSE	Max error	Freq.	RMSE	Max error	Freq.
0.0	3.9	11.0	100	3.7	8.6	100
1.5	8.8	19.2	92.2	4.8	10.7	80.6
2.5	9.5	17.4	93.4	6.8	13.3	83.6
3.5	12.3	26.8	90.1	9.6	20.3	82.5

5 Conclusion

This article investigates the advantages of a four-wheel steering system under normal driving conditions, where the event-triggered MPC is applied to both two- and four-wheel steering systems. A modified 1/10th scale Tamiya TT-02 RC car with ultrasonic-based indoor positioning system is used for experiments. Contrary to the conventional belief that four-wheel steering systems are beneficial only in tight steering maneuvers, the results reported here show that, under identical settings of the event-triggered MPC, the four-wheel steering system offers better tracking performance while using less computational resources.

Acknowledgment

This work was supported in part by the Faculty Startup Fund at Oakland University and in part by National Science Foundation (Award No. 2237317).

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Fagnant, D. J., and Kockelman, K., 2015, "Preparing a Nation for Autonomous Vehicles: Opportunities, Barriers and Policy Recommendations," Transp. Res. Part A: Policy Practice, **77**, pp. 167–181.

[2] Chen, L., Li, Y., Huang, C., Xing, Y., Tian, D., Li, L., Hu, Z., Teng, S., Lv, C., Wang, J., and Cao, D., 2023, "Milestones in Autonomous Driving and Intelligent Vehicles—Part 1: Control, Computing System Design, Communication, Hd Map, Testing, and Human Behaviors," IEEE. Trans. Syst. Man. Cybernet.: Syst., **53**(9), pp. 5831–5847.

[3] Yu, R., Guo, H., Sun, Z., and Chen, H., 2015, “MPC-Based Regional Path Tracking Controller Design for Autonomous Ground Vehicles,” IEEE International Conference on Systems, Man, and Cybernetics, Oct. 9–12, Hong Kong, China, pp. 2510–2515.

[4] Yang, L., Lu, C., Xiong, G., Xing, Y., and Gong, J., 2022, “A Hybrid Motion Planning Framework for Autonomous Driving in Mixed Traffic Flow,” Green Energy Int. Transp., **1**(3), p. 100022.

[5] Zhou, Z., Rother, C., and Chen, J., 2023, “Event-Triggered Model Predictive Control for Autonomous Vehicle Path Tracking: Validation Using CARLA Simulator,” IEEE Trans. Int. Veh., **8**(6), pp. 3547–3555.

[6] Hang, P., Xia, X., and Chen, X., 2021, “Handling Stability Advancement With 4WS and DYC Coordinated Control: A Gain-Scheduled Robust Control Approach,” IEEE Trans. Veh. Tech., **70**(4), pp. 3164–3174.

[7] Zhao, W., Qin, X., and Wang, C., 2018, “Yaw and Lateral Stability Control for Four-Wheel Steer-by-Wire System,” IEEE/ASME Trans. Mechatron., **23**(6), pp. 2628–2637.

[8] Hang, P., and Chen, X., 2021, “Path Tracking Control of 4-Wheel-Steering Autonomous Ground Vehicles Based on Linear Parameter-Varying System With Experimental Verification,” Proc. Inst. Mech. Eng., Part I: J. Sys. Contr. Eng., **235**(3), pp. 411–423.

[9] Zhu, S., Wei, B., Liu, D., Chen, H., Huang, X., Zheng, Y., and Wei, W., 2022, “A Dynamics Coordinated Control System for 4WD-4WS Electric Vehicles,” Electronics, **11**(22), p. 3731.

[10] Skarpetis, M. G., Koumboulis, F. N., Barmpokas, F. S., and Chamilothoris, G. E., 2006, “Decoupling Control Algorithms for 4WS Vehicles,” IEEE International Conference on Mechatronics, Budapest, Hungary, July 3–5, pp. 499–504.

[11] Lv, H.-M., Chen, N., and Li, P., 2004, “Multi-Objective H^{∞} Optimal Control for Four-Wheel Steering Vehicle Based on Yaw Rate Tracking,” Proc. Inst. Mech. Eng., D: J Auto. Eng., **218**(10), pp. 1117–1123.

[12] Rawlings, J. B., 2000, “Tutorial Overview of Model Predictive Control,” IEEE Control Syst. Mag., **20**(3), pp. 38–52.

[13] Chen, J., Liang, M., and Ma, X., 2021, “Probabilistic Analysis of Electric Vehicle Energy Consumption Using MPC Speed Control and Nonlinear Battery Model,” IEEE Green Tech Conference, Denver, CO, Apr. 7–9, pp 181–186.

[14] Yoo, J., and Johansson, K. H., 2021, “Event-Triggered Model Predictive Control With a Statistical Learning,” IEEE. Trans. Syst. Man. Cybernet.: Syst., **51**(4), pp. 2571–2581.

[15] Bao, H., Kang, Q., Shi, X., Zhou, M., Li, H., An, J., and Sedraoui, K., 2023, “Moment-Based Model Predictive Control of Autonomous Systems,” IEEE Trans. Intell. Vehicles, **8**(4), pp. 2939–2953.

[16] Chu, X., Liu, Z., Mao, L., Jin, X., Peng, Z., and Wen, G., 2022, “Robust Event Triggered Control for Lateral Dynamics of Intelligent Vehicle With Designable Inter-Event Times,” IEEE Trans. Circuits Syst. II: Exp. Briefs, **69**(11), pp. 4349–4353.

[17] Chen, J., Meng, X., and Li, Z., 2022, “Reinforcement Learning-Based Event-Triggered Model Predictive Control for Autonomous Vehicle Path Following,” American Control Conference, Atlanta, GA, June 8–10, pp 3342–3347.

[18] Li, H., and Shi, Y., 2014, “Event-Triggered Robust Model Predictive Control of Continuous-Time Nonlinear Systems,” Automatica, **50**(5), pp. 1507–1513.

[19] Yu, L., Xia, Y., and Sun, Z., 2018, “Robust Event–triggered Model Predictive Control for Constrained Linear Continuous System,” J. Robust. Nonlinear. Control., **29**(5), pp. 1216–1229.

[20] Zhou, Z., Chen, J., Tao, M., Zhang, P., and Xu, M., 2023, “Experimental Validation of Event-Triggered Model Predictive Control for Autonomous Vehicle Path Tracking,” 2023 IEEE International Conference on Electro Information Technology, Romeoville, IL, May 18–20, pp. 35–40.

[21] Rother, C., Zhou, Z., and Chen, J., 2023, “Development of a Four-Wheel Steering Scale Vehicle for Research and Education on Autonomous Vehicle Motion Control,” IEEE RA-L, **8**(8), pp. 5015–5022.

[22] Jetracer, 2019, “An Autonomous AI Racecar Using NVIDIA Jetson Nano,” https://github.com/NVIDIA-AI-IOT/jetracer, Accessed November 2021.

[23] Wang, J., Gao, S., Wang, K., Wang, Y., and Wang, Q., 2021, “Wheel Torque Distribution Optimization of Four-Wheel Independent-Drive Electric Vehicle for Energy Efficient Driving,” Control. Eng. Pract., **110**, p. 104779.

[24] Goodarzi, A., and Mohammadi, M., 2014, “Stability Enhancement and Fuel Economy of the 4-Wheel-Drive Hybrid Electric Vehicles by Optimal Tyre Force Distribution,” Vehicle Syst. Dyn., **52**(4), pp. 539–561.

[25] Rawlings, J., “Model Predictive Control (MPC) Tools Package,” https://sites.engineering.ucsb.edu/ jbraw/software.html, Accessed March 2022.

[26] Andersson, J. A. E., Gillis, J., Horn, G., Rawlings, J. B., and Diehl, M., 2019, “CasADi—A Software Framework for Nonlinear Optimization and Optimal Control,” Math. Programm. Comput., **11**(1), pp. 1–36.

[27] Chen, J., and Yi, Z., 2021, “Comparison of Event-Triggered Model Predictive Control for Autonomous Vehicle Path Tracking,” IEEE Conference on Control Technology and Applications, San Diego, CA, Aug. 8–11, pp. 808–813.

[28] Isen, F. W., 2009, “DSP for MATLAB and Labview I: Fundamentals of Discrete Signal Processing,” Synthesis Lectures on Signal Processing, J. Moura, ed., Springer Nature, Berlin, Germany, pp. 169–188.

[29] Viana, F. A. C., 2016, “A Tutorial on Latin Hypercube Design of Experiments,” Qual. Reliab. Eng. Int., **32**(5), pp. 1975–1985.