Journal of Mechatronics, Electrical Power, and Vehicular Technology 17 (2026) 68-88

Journal of Mechatronics, Electrical Power, and Vehicular Technology

e-ISSN: 2088-6985
p-ISSN: 2087-3379

mev.brin.go.id

An efficient motion planning framework for four-wheel steering autonomous vehicles using lazy edge-based A* and adaptive RK4-MPC

Deyndrawan Sutrisno [a], Subiyanto [a, *], Arimaz Hangga [a], Aldias Bahatmaka [a], Nur Azis Salim [a, b], Elfandy Yunus [b], Muhammad Hilmi Farras [a], Setya Budi Arif Prabowo [a]

[a] *Faculty of Engineering, Universitas Negeri Semarang*
Sekaran Campus, Gunungpati, Semarang, Central Java 50229, Indonesia

[b] *Graduate School of Engineering Science and Technology, National Yunlin University of Science and Technology*
No. 123, University Road, Section 3, Douliou, Yunlin 64002, Taiwan

Abstract

This work presents an efficient motion planning framework for four-wheel steering (4WS) autonomous vehicles operating in complex and unknown environments. To improve planning efficiency, the framework employs a lazy edge-based A* (LEA*) algorithm for global path planning, adaptive fourth-order Runge–Kutta model predictive control (RK4-MPC) for trajectory tracking and motion execution, and wheel force distribution control (WFDC) to ensure stable motion during steering maneuvers. Quantitative results show that the LEA* reduces planning time by 87.5 % edge evaluations by 96.1 % compared to conventional A*, while improving path smoothness by 51 %. The integration of adaptive RK4-MPC with WFDC achieves the lowest tracking error and heading error of 34.8 % and 37.5 % compared to OMNI, and 28.6 % compared to S-4WS. In addition, the proposed method reduces the wheel slip ratio 88.4 % better than OMNI and 46.7 % better than S-4WS, while also reducing yaw acceleration by 50 % compared to both baselines. For computational efficiency, the proposed framework achieves a search time of 0.5234 s, 83.1 % faster than OMNI, and 37.1 % faster than S-4WS, and an optimization time of 1.4892 s, 30.3 % faster than S-4WS. Overall, the proposed framework improves motion planning efficiency while maintaining smooth and stable motion in simulation.

Keywords: autonomous vehicles; four-wheel steering; lazy edge-based A*; motion planning; Runge-Kutta method.

I. Introduction

The rapid development in autonomous vehicle (AV) technology has significantly improved transportation by enhancing energy efficiency and reducing emissions [1]. In large and dynamic environments, such as urban and industrial areas, efficient motion planning is required for AV navigation [2]. However, existing motion planning algorithms often struggle in handling unpredictable terrains and dynamic obstacles, which often leads to high computational costs and suboptimal path planning [3][4][5].

The performance of motion planning in AV can be enhanced using advanced steering systems, such as a four-wheel steering (4WS) system, to improve

* Corresponding Author. subiyanto@mail.unnes.ac.id (Subiyanto)

https://doi.org/10.55981/j.mev.2026.1073

Received 2 September 2025; revised 11 December 2025; accepted 13 January 2026; available online 4 July 2026

How to Cite: D. Sutrisno *et al.*, "An efficient motion planning framework for four-wheel steering autonomous vehicles using lazy edge-based A* and adaptive RK4-MPC," *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 17, no. 1, pp. 68-88, July, 2026.

maneuverability, stability, and path-tracking [6][7]. The steering flexibility of 4WS enables diagonal and pivotal steering maneuvers that are not feasible with conventional Ackermann steering geometry [8]. For 4WS vehicles, path feasibility is governed by the instantaneous velocity center (IVC), which defines the kinematic relationship between steering angles and vehicle motion and directly constrains feasible path generation. However, under unknown and dynamic driving environments, 4WS systems pose challenges for path planning, motion control, and vehicle stability [9].

Path planning in unknown and dynamic environments places significant computational demands on graph-based planning methods, where edge evaluation, particularly collision checking, becomes a major bottleneck [10][11]. To mitigate this issue, A*-based variants with lazy search mechanisms have been developed to reduce the number of edge evaluations during planning, thereby improving computational efficiency over classical graph search methods such as breadth-first search (BFS), depth-first search (DFS), and Dijkstra's algorithm [12][13]. Recent lazy search algorithms are lazy weighted A* (LWA*) [14], lazy shortest path (LazySP) [15], lazy receding horizon A* (LRA*) [16], and lazy edged based A* (LEA*) [17]. Lazy search algorithms employ an edge cost heuristic that is inexpensive to compute and provides a lower bound on the true edge cost [18]. This heuristic guides the graph search so that expensive edge evaluations, such as collision checking, are performed only when necessary. Among these approaches, LazySP has been shown to be optimal in terms of minimizing the number of edge evaluations [15], while LEA* focuses on improving edge evaluation efficiency with lower computational overhead [17]. However, the application of lazy search algorithms to path planning for 4WS vehicles remains limited and has not been explicitly investigated. Moreover, despite an efficient path planning algorithm demonstrating significant improvements, the generated trajectories must still be executed reliably under vehicle dynamics and steering constraints, requiring an advanced motion control method [19][20].

Among advanced motion control research, model predictive control (MPC) has been widely used for trajectory optimization in AV systems [21][22]. MPC optimizes future control inputs by predicting vehicle states over a finite horizon while considering system dynamics and operational constraints [23]. However, conventional MPC applications use Euler discretization for state prediction, as a results limits prediction accuracy when dealing with nonlinear vehicle dynamics and steering constraints [24][25][26]. Thus, recent studies have integrated higher-order numerical integration methods, such as the fourth-order Runge–Kutta (RK4), into the MPC [27][28]. RK4 provides more accurate discretization of continuous vehicle dynamics and improves multi-stage prediction of nonlinear steering-coupled motion within each sampling interval [29]. Nevertheless, the application of RK4-based MPC to motion planning and control of 4WS vehicles has not been explicitly investigated. In addition, existing trajectory tracking strategies primarily focus on path tracking accuracy, yaw rate error, and real-time obstacle avoidance, they tend to neglect critical dynamic phenomena such as tire slip, load transfer, and wheel force limitations, making them less suitable for high speed or highly dynamic maneuvers [30][31][32][33]. Thus, a high-level planning framework is important to improve adaptation to complex environments through driving torque distribution.

Numerous studies have proposed various driving torque distribution algorithms to address uncoordinated force allocation and wheel slip issues, including average torque distribution, tire-load-based methods, and optimization-based strategies. Average torque distribution is simple but neglects vertical load transfer, which may lead to wheel slippage and insufficient yaw moment generation under varying adhesion conditions [34][35][36]. To improve stability, tire-load-based approaches incorporate vertical load transfer during acceleration, braking, and steering [37][38]. In addition, integrated and cooperative torque and force distribution strategies have been investigated, employing wheel force distribution control (WFDC) to optimally allocate vehicle-level yaw and roll moments to individual wheels in all-wheel independently actuated vehicles under complex steering maneuvers [39]. Therefore, WFDC is well-suited for optimizing the design of 4WS stability controllers in dynamic and complex environments.

Based on the literature, the complexity of modern AV systems, such 4WS system, demands integrated planning and control frameworks capable of handling dynamic environments and steering constraints. To answer this problem, this paper proposes an efficient motion planning for the 4WS system with the following contributions.

The global path-planning method based on the lazy search algorithm computes the feasible path, which evaluates wheel conditions and integrates travel cost into time consumption. As a result, the initial trajectory planner can search for efficient edge evaluation with lower computational time.

An adaptive motion control scheme based on RK4-MPC is developed. The controller integrates RK4-based state propagation within the MPC framework to ensure

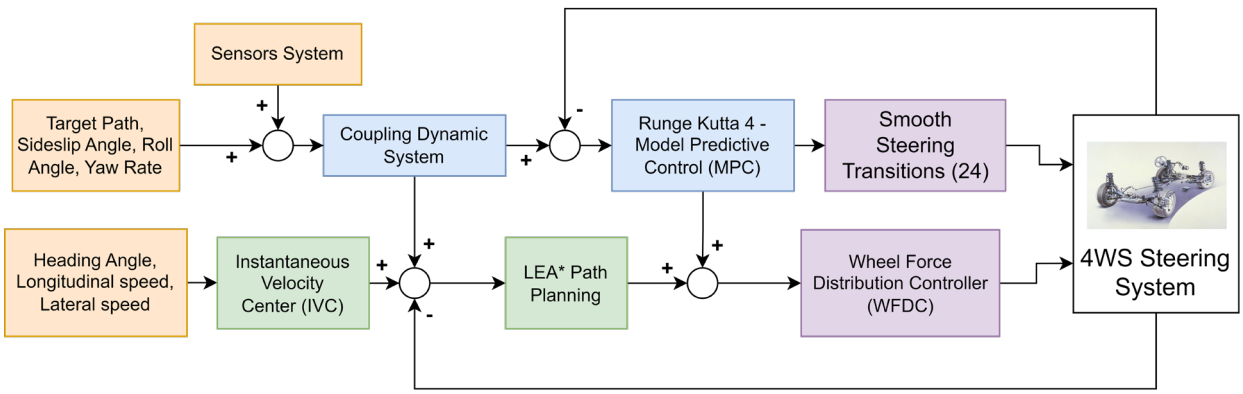


Figure 1. Proposed system block diagram.

accurate trajectory tracking, adaptive motion control, and smooth steering transitions under nonlinear steering-coupled dynamics and dynamic environmental conditions. The effectiveness of WFDC for 4WS vehicles is validated through simulation in unknown and complex driving environments along with the proposed framework.

The article is structured as follows: Section II presents the proposed motion planning framework for autonomous vehicles, including the 4WS system, the Lazy search algorithm for path planning, the adaptive RK4-MPC approach for motion control, and WFDC for tire force allocation and stability. Section III evaluates the proposed framework through simulations and compares it with baseline methods in terms of planning time, path smoothness, and vehicle stability. Section IV summarizes the main findings and explores prospects for future work.

II. Materials and Methods

The proposed framework consists of a hierarchical planning and control architecture, as illustrated in Figure 1. The system integrates global path planning, motion control, and force allocation modules to generate feasible and stable trajectories for 4WS vehicles in unknown and dynamic environments. Each module is designed to operate sequentially while exchanging state and control information to support real-time implementation.

As shown in Figure 1, sensor measurements and reference states, including vehicle velocity, yaw rate, and target path information, are processed to support real-time planning and control. Global path planning is performed using the LEA* algorithm, while the IVC is employed to ensure steering feasibility and forward motion constraints. The resulting reference trajectory is tracked using an adaptive RK4–MPC, which ensures smooth steering transitions and accurate state prediction. WFDC is subsequently applied to allocate tire forces and maintain vehicle stability during complex maneuvers.

A. Vehicle dynamic system modeling

A kinematic model is used to describe the planar motion and steering geometry of the four-wheel steering (4WS) vehicle. The model focuses on the relationship between wheel steering angles and vehicle motion, which is required for path feasibility analysis. Figure 2 illustrates the kinematic configuration of the 4WS vehicle and the associated coordinate system, where the direction Ω denotes positive yaw rotation.

The rotational motion of the vehicle is described using the IVC concept. The IVC defines the point about which the vehicle rotates during planar motion, and its position relative to the vehicle control center is represented by the radius vector $\boldsymbol{r}_{IVC}$ in equation (1). This representation establishes the geometric relationship between vehicle angular velocity and translational velocity.

$$\boldsymbol{r}_{IVC} = -\frac{1}{\dot{\theta}} K v \tag{1}$$

where $\dot{\theta}$ represents the angular velocity of the vehicle, K is a transformation matrix that defines the

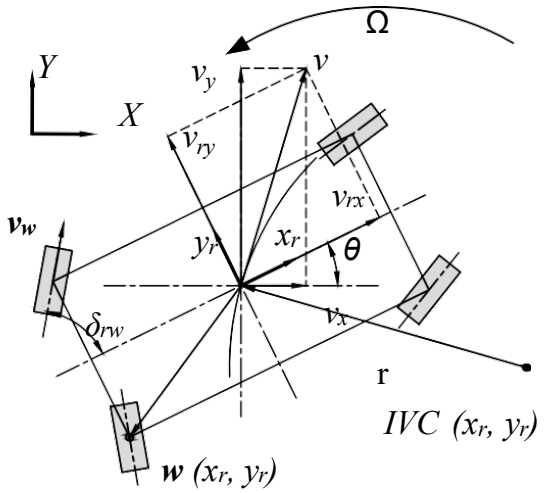


Figure 2. A 4WS vehicle kinematic model.

relationship between rotational and linear velocities, and v is the linear velocity of the vehicle. Once the IVC is defined, the velocity of each wheel in the world frame can be calculated by equation (2), considering the IVC velocity and the wheel's position relative to the vehicle's center.

$$\boldsymbol{r}_{IVC,w} = \boldsymbol{r}_{IVC} + w, \quad v_w = \Omega(\theta) \cdot r_{IVC,w} = v + \Omega(\theta)w \tag{2}$$

where $\boldsymbol{r}_{IVC,w}$ represents the position of the wheel relative to the IVC, and w is the position of the wheel in the vehicle's frame. The term $\Omega(\theta)$ is the rotation matrix that accounts for the vehicle's heading, and v_w is the velocity of the wheel in the world frame. The inverse of the rotation matrix is applied to convert the velocity of the wheel from the world frame to the robot frame. This transformation adjusts the velocity to align with the robot's reference frame, ensuring consistency in how the motion is analyzed in equation (3).

$$v_{r,w} = R(\theta)^{-1} v_w = R(\theta)^{\top} v_w \tag{3}$$

where $R(\theta)$ is the rotation matrix for the vehicle's heading, and $v_{r,w}$ is the velocity of the wheel in the robot frame. The transpose of the rotation matrix $R(\theta)^{\top}$ is applied to shift the velocity from the world frame to the robot frame. The steering angle of each wheel, denoted by δ_w, depends on the velocity components of the wheel in the x and y directions. This relationship in equation (4) helps determine how the wheel's motion influences steering behavior.

$$\delta_w = \arctan 2\left(v_{wy}, v_{wx}\right) = 2 \arctan\left(\frac{v_{wy}}{|v_w| + v_{wx}}\right) \tag{4}$$

where v_{wx} and v_{wy} are the x- and y- components of the wheel's velocity in the world frame. The equation expresses the wheel's steering angle based on its directional velocities. In the robot frame, the steering angle is adjusted by subtracting the vehicle's heading angle, which ensures that the steering is relative to the vehicle's orientation as defined in equation (5).

$$\delta_{r,w} = \delta_w - \theta \tag{5}$$

where θ is the vehicle's heading angle, and $\delta_{r,w}$ represents the adjusted steering angle in the robot frame. Lateral dynamics describe how the vehicle responds to lateral forces, which are critical for stability during turning. The forces generated by the front and rear tires are denoted as F_{yf} and F_{yf}, are essential for maintaining lateral stability, as described in equation (6). These forces help the vehicle balance during lateral movements, particularly during turns.

$$m(\dot{v} + u \cdot r) = F_{yf} + F_{yr} \tag{6}$$

with the total vehicle mass m is defined in equation (7),

$$m = m_s + m_u \tag{7}$$

With m_s represents the mass of the vehicle's sprung body and m_u is the mass of the unsprung body. Here $\dot{v}$ is the lateral velocity, u is the longitudinal velocity, r is the yaw rate. Yaw dynamics further explain the vehicle's rotational motion around its vertical axis. The yaw rate and the forces acting on the front and rear tires influence the vehicle's rotational stability. This relationship is captured by the yaw moment M_z, as described in equation (8). The yaw dynamics are crucial for ensuring that the vehicle maintains control while turning or performing complex maneuvers.

$$I_z \ddot{\theta} = 2r_f F_{yf} - 2r_r F_{yr} + M_z \tag{8}$$

where I_z represents the moment of inertia about the vehicle's yaw axis and $\ddot{\theta}$ is the yaw acceleration with r_f and r_r denoting the distances from the vehicle's center of gravity to the front and rear axles, respectively. The roll dynamics address the vehicle's tendency to tilt during lateral movements. Roll stability is managed through the vehicle's roll stiffness and damping, which counteract the effects of lateral forces and help maintain balance during sharp turns. The roll behavior is expressed in equation (9), where the roll moment, M_x, and other related parameters describe the vehicle's ability to resist tilting.

$$I_x \ddot{\phi} = m_s g h_s \sin(\phi) + m_s h_s \dot{y} \cos(\phi) - K_\phi \phi - C_\phi \dot{\phi} + M_x \tag{9}$$

In this equation, I_x is the moment of inertia about the roll axis, ϕ is the roll angle and M_x is the roll moment. The terms K_ϕ and C_ϕ represent the roll stiffness and damping coefficients, which help control the roll stability of the vehicle. The parameter h_s is the height of the center of gravity for the sprung mass, while g represents gravitational acceleration. These connected equations form the complete dynamic system model of the vehicle, providing a detailed understanding of its rigid body motion and its response to various dynamic forces.

B. Path planning algorithm

A feasible path planning strategy is implemented using the Lazy search algorithm combined with IVC-based grid sampling. The planner represents vehicle motion states using position, heading, and forward velocity, while reducing computational cost through lazy edge evaluation. To ensure steering feasibility for four-wheel steering (4WS) vehicles, forward motion is generated by sampling feasible Instantaneous Velocity Center (IVC) locations subject to steering angle constraints, as illustrated in Figure 3.

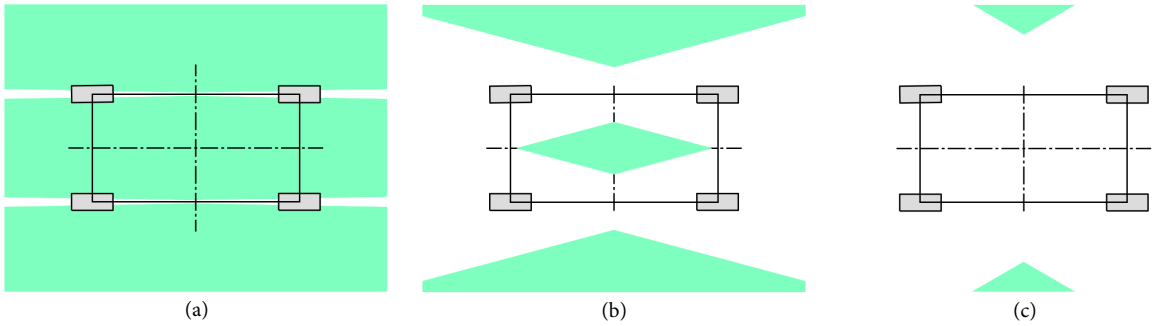


Figure 3. Feasible IVC regions for steering angle limits of (a) 90°; (b) 75°; (c) 60°.

As shown in Figure 3, feasible IVC regions shrink as steering angle limits decrease, directly constraining admissible vehicle motion. The vectors from the IVC to the vehicle control center, denoted by r, define the motion state and are obtained through uniform sampling of the parameters ε and ψ, as given in equation (10) and equation (11). For each sampled grid point, the corresponding vector r is computed using equation (12). While the steering feasibility is enforced by constraining the wheel steering angle δ_w within its physical limits, as expressed in equation (13). In addition, constraints on arc length and rotational speed Ω are applied to ensure smooth and feasible forward motion during path expansion.

$$\varepsilon = \left\{ \varepsilon_i \middle| \varepsilon_i = \frac{\pi i + \epsilon}{2N} + \epsilon, \forall i \in \{0,1,\dots,N\} \right\} \tag{10}$$

$$\psi = \left\{ \psi_i \middle| \psi_i = -\pi + \frac{2\pi i}{N}, \forall i \in \{0,1,\dots,N\} \right\} \tag{11}$$

$$r = [-\tan(\varepsilon) \cdot \cos(\psi), -\tan(\varepsilon) \cdot \sin(\psi)] \tag{12}$$

$$\underline{\delta_w} \leq \left| \arctan 2\left(r_{y,IVC,w}, r_{x,IVC,w}\right) \right| - \frac{\pi}{2} \leq \bar{\delta}_w \tag{13}$$

Figure 4 illustrates the sampling results across various steering limits. Based on the kinematic formulation in equation (1), the vehicle motion at each sampling step is propagated to obtain the next position and velocity state. The corresponding wheel velocity $v_{w,n}$ and steering angle $\delta_{w,n}$ at each node are computed using equations (3) and (5). For steering angles limited to 90°, a direct mapping between wheel velocity and feasible steering configuration can be established. For steering angle exceeds 90°, this relationship becomes non-unique, and additional kinematic constraints are required to ensure feasible maneuver generation.

The proposed planner integrates IVC-based grid sampling with the LEA* algorithm to perform global path planning with reduced computational cost, as illustrated in Figure 5. Feasible motion primitives are first generated by sampling IVC locations subject to steering angle and kinematic constraints, and these primitives are treated as candidate edges in the LEA* algorithm. Each edge is inserted into the priority queue using an estimated travel cost, while the actual edge cost is evaluated lazily only when the edge is selected for expansion. Path evaluation is performed by the cost function $c(v)$ defined in equation (14), which penalizes variations in wheel velocity $v_{w,n}$, steering angle $\delta_{w,n}$, translational displacement Δx, and orientation change $\Delta\theta$, normalized by their respective maximum allowable rates. If an evaluated edge satisfies feasibility constraints, its cost is updated, and successor edges are generated through further IVC sampling; otherwise, the edge is discarded. This process continues until a feasible path to the goal is identified, limiting cost computation to promising edges while preserving kinematic feasibility. The resulting path provides a kinematically feasible reference trajectory for the subsequent motion control stage.

$$c(v) = \max\left\{ k_v \cdot \left(\frac{|v_{w,n} - v_{w,n-1}|}{\dot{v}_{w,max}}\right)^2, k_\delta \cdot \left(\frac{|\delta_{w,n} - \delta_{w,n-1}|}{\dot{\delta}_{w,max}}\right)^2, \frac{\Delta x}{v_{max}}, \frac{\Delta\theta}{\dot{\theta}_{max}} \right\} \tag{14}$$

The LEA*-based planner will be assessed using planning scenarios with varying obstacle densities and environmental uncertainty. The evaluation metrics include planning time, number of evaluated edges, path length, and path smoothness, under steering-constrained motion primitives.

C. Predictive motion control

The motion control problem for the 4WS vehicle is formulated as a nonlinear MPC problem with finite-horizon optimization under vehicle dynamics and steering constraints. The reference path E generated by the LEA* algorithm planner is used as the initial guess and tracking reference for the MPC, while real-time feedback is employed to refine control execution under dynamic and uncertain driving conditions, as illustrated in Figure 6.

The system state vector $x(t)$ includes the vehicle kinematic and dynamic states, such as position, heading, velocity $\dot{v}(t)$, yaw rate $r(t)$, and steering angle $\delta(t)$, and their derivatives. The control input $u(t)$ represents the steering and longitudinal actuation

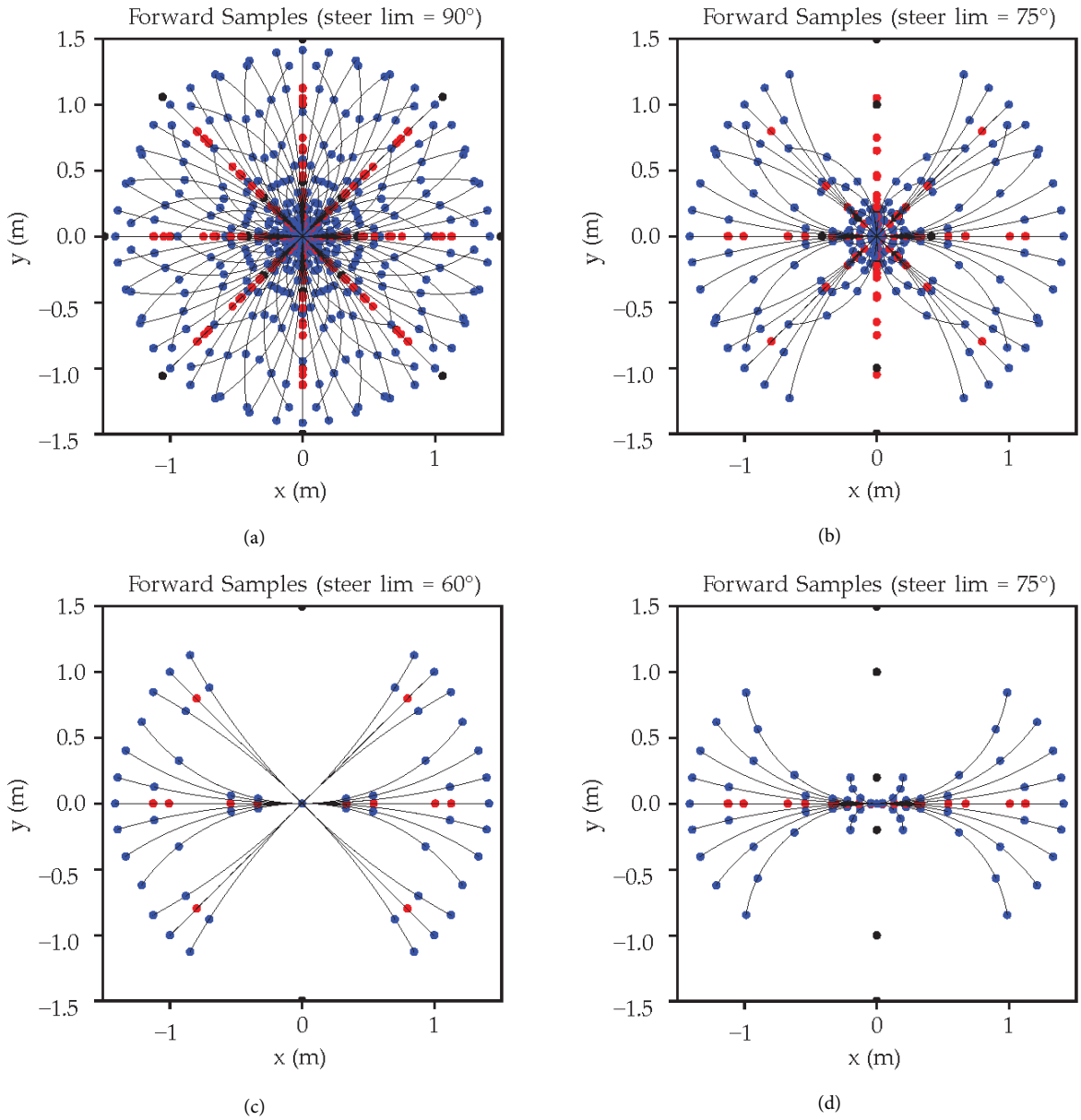


Figure 4. Sample forward maneuvers under different steering angle limits: (a) 90°; (b) 75°; (c) 60°; (d) front 75° - rear 0°

commands. To accurately propagate the nonlinear vehicle dynamics within the prediction horizon, the RK4 method is employed as the state discretization method expressed in equation (15).

$$x(t+dt) = x(t) + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \quad (15)$$

With $k_1, 2k_2, 2k_3, k_4$ are intermediate slopes computed from the continuous-time dynamics $\dot{x} = f(x,u)$ using the current control input $u[n]$. This formulation ensures state continuity and accurate prediction of steering-coupled vehicle motion. Accordingly, the RK4-based state transition constraint within the MPC prediction horizon is defined in equation (16).

$$G_c := x(t+dt) = RK4(x(t), u(t), dt), \quad \forall t \in [0,T] \quad (16)$$

To ensure the physical feasibility of the optimized trajectory, additional kinematic and steering constraints are imposed. Wheel velocity and acceleration limits are enforced through the constraint G_{vw} which bounds the admissible wheel speed profiles to prevent excessive slip and infeasible actuation. Steering angle limits are modeled by equation (17).

$$G_{\delta lim} := v_{r,w} \times \begin{bmatrix} \cos(\bar{\delta}_{lim}) \\ \sin(\bar{\delta}_{lim}) \end{bmatrix} \cdot v_{r,w} \times \begin{bmatrix} \cos(\underline{\delta}_{lim}) \\ \sin(\underline{\delta}_{lim}) \end{bmatrix} \leq 0, \quad \forall t \in [0,T], \forall w \in W \quad (17)$$

with $v_{r,w}$ denotes the wheel velocity vector in the vehicle frame and $\bar{\delta}_{lim}$ and $\underline{\delta}_{lim}$ is the maximum allowable steering angle. To avoid steering discontinuities caused by angular wrapping and mode switching, steering continuity is enforced using a vector-based constraint through equation (18).

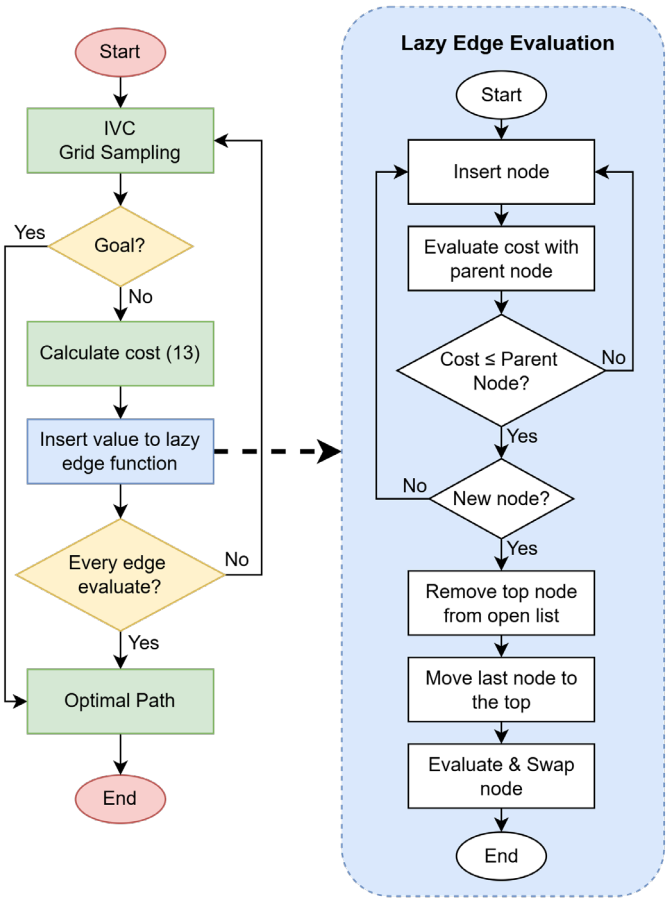


Figure 5. Flowchart of the LEA* path planning algorithm with IVC sampling.

$$G_{\delta_{lim}} \coloneqq \left(D_{w,t}v_{r,w,t} \cdot D_{w,t-1}v_{r,w,t-1}\right) - \left|v_{r,w,t}\right|\left|v_{r,w,t-1}\right| \cos\left(\dot{\delta}_{lim}dt\right) \geq 0 \quad (18)$$

With $D_{w,t} \in \{-1,1\}$ represents the wheel rotation direction and $v_{r,w,t}$ the wheel velocity at time t. In addition, mode transition constraints are imposed to handle steering mode switching in 4WS systems. Keyframes are defined as discrete time instants located at the boundaries between different steering modes, at which the vehicle velocity is constrained to zero to ensure continuity and stability during mode transitions, as defined in equation (19).

$$G_M \coloneqq v_t \cdot (|M(t) - M(t+dt)| + |M(t) - M(t-dt)|) = 0 \quad (19)$$

To improve the feasibility and reduce sensitivity to the initial guess, the time step dt is treated as an optimization variable with a bounded range as defined in equation (20).

$$G_t \coloneqq dt\{dt|dt \in \mathbb{R}^+, dt \in [\underline{dt}, \overline{dt}]\} \quad (20)$$

This dynamic adjustment of time steps improves feasibility and numerical stability during trajectory optimization. The MPC objective function is designed to minimize tracking error with respect to the reference path E, control effort, and total traversal time over the prediction horizon. In discrete form, the optimization problem is formulated through equation (21).

$$min\, J = \sum_{h=0}^{H} F(x[h], E[h]) + \sum_{h=0}^{H} u[h]^{\top} A u[h] dt(h) + \sum_{h=0}^{H} dt(h) \quad (21)$$

subject to the $G_c, G_{vw}, G_{\delta lim}, G_{\delta_{lim}}, G_M$, and G_t, with $F(\cdot)$ denotes the path-tracking cost, A is a diagonal weighting matrix, and $dt(h)$ is treated as an optimization variable within predefined bounds to improve feasibility and convergence. At each sampling instant, solving the MPC optimization problem in (21) yields an optimal control sequence over the prediction horizon as expressed in equation (22).

$$U^* = \{u^*[0], u^*[1], \ldots, u^*[H-1]\} \quad (22)$$

In accordance with the receding horizon principle, only the first control input of the optimal sequence is applied to the system and is defined in equation (23).

$$u_{MPC}[n] = u^*[0] \quad (23)$$

While the MPC optimizes the nominal control sequence over the prediction horizon, modeling uncertainties and external disturbances may cause deviations during execution. To compensate for this issue, an adaptive correction layer is used to compensate for prediction errors without modifying

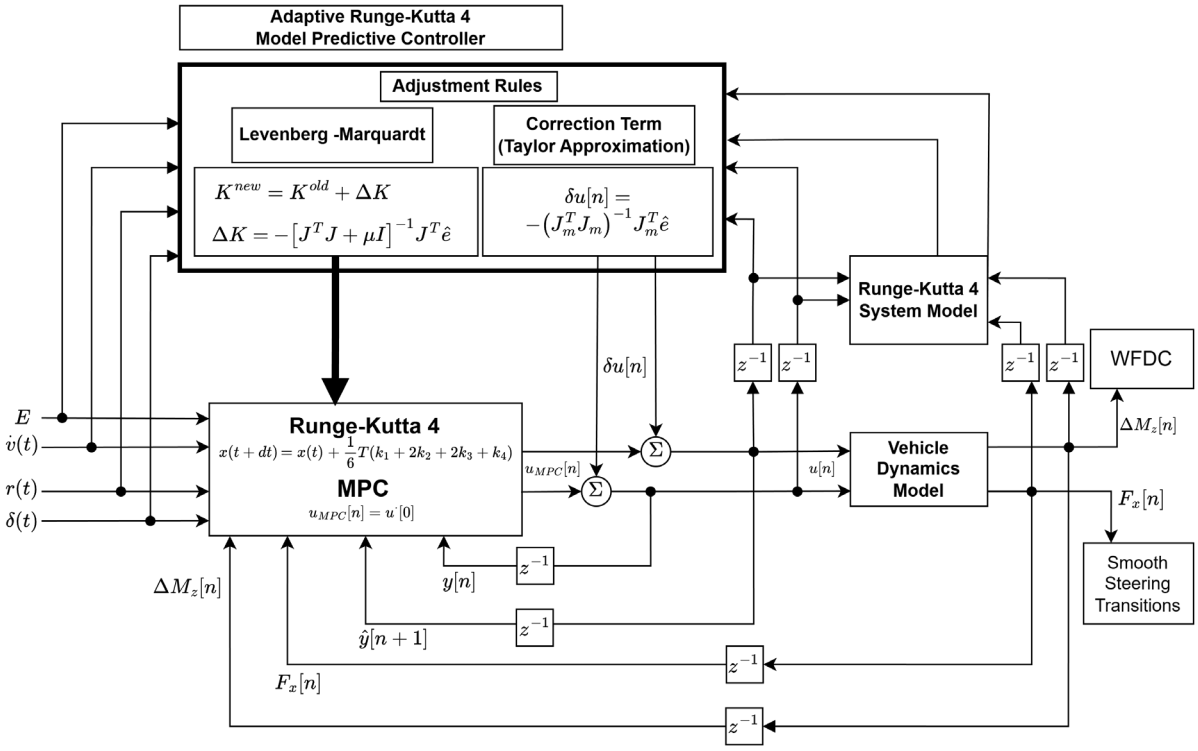


Figure 6. Proposed control block diagram.

the MPC objective function. Let $\hat{y}[n+1]$ denote the predicted output obtained from the RK4-based MPC model, and $y[n]$ he measured system output. The prediction error is defined in equation (24). The control gain matrix K is updated online using the Levenberg–Marquardt (LM) algorithm to minimize the prediction error with respect to the control gains given in equation (25). LM is selected due to its favorable trade-off between convergence speed and numerical robustness in nonlinear systems, providing improved stability compared to Gauss–Newton methods while avoiding the high computational burden of full second-order optimization.

$$\hat{e}[n] = \hat{y}[n+1] - y[n] \tag{24}$$

$$\Delta K = -[J^\top J + \mu I]^{-1} J \hat{\top} \hat{e} \tag{25}$$

with $J = \partial\hat{y}/\partial K$, is the Jacobian of the predicted output with respect to the gain matrix, μ is a damping factor, and I is the identity matrix. The gain matrix is then updated as expressed in equation (26).

$$K^{new} = K^{old} + \Delta K \tag{26}$$

In addition to gaining adaptation, a local control correction term is computed using a first-order Taylor approximation to further reduce residual tracking errors expressed in equation (27).

$$\delta u[n] = -(J_m^\top J_m)^{-1} J_m^\top \hat{e} \tag{27}$$

with $J_m = \partial\hat{y}/\partial u$ denotes the Jacobian of the predicted output with respect to the control input. This correction compensates for small deviations caused by unmodeled dynamics and disturbances. The final control input applied to the vehicle is obtained by superimposing the MPC control input and the correction term as defined in equation (28).

$$u[n] = u_{MPC}[n] + \delta u[n] \tag{28}$$

The control input $u[n]$ obtained from the predictive motion control framework, consisting of the longitudinal force command $F_x[n]$ and yaw moment correction $\Delta M_z[n]$, which are subsequently allocated to individual wheels through a wheel-level force distribution mechanism. The predictive motion control will be evaluated through trajectory tracking tasks executed under different steering models, including OMNI, S-4WS, and the proposed constrained 4WS framework. OMNI denotes an omnidirectional motion model that assumes holonomic motion and neglects steering angle constraints. S-4WS denotes a symmetrical four-wheel steering model in which the front and rear wheels steer with equal magnitude, representing a bicycle-model-equivalent non-holonomic baseline. The evaluation considers tracking error, steering continuity, and yaw response during confined maneuvers and curvature-varying trajectories.

D. Wheel force distribution

The WFDC is employed to allocate the vehicle-level control commands generated by the RK4-based MPC to individual wheel forces in a physically feasible

manner. As defined in Section III-C, the MPC outputs the longitudinal force command F_x and the yaw moment correction $\Delta M_z[n]$ at each sampling instant. The role of the WFDC is to distribute these commands to the four independently actuated wheels while satisfying force equilibrium and tire–road constraints.

Let F_{xij}, F_{yij}, and F_{zij} denote the longitudinal, lateral, and vertical forces at each wheel, respectively, where the indices $i \in \{1,2\}$ and $j \in \{1,2\}$ represent the front/rear and left/right wheels. The resultant longitudinal and lateral forces acting on the vehicle body are expressed in equations (29) and (30).

$$F_{x,des} = F_{x11} + F_{x12} + F_{x21} + F_{x22} \tag{29}$$

$$F_{y,des} = F_{y11} + F_{y12} + F_{y21} + F_{y22} \tag{30}$$

Equation (29) and equation (30) enforce force consistency between the wheel-level forces and the desired vehicle-level longitudinal and lateral force commands provided by the predictive motion controller. The roll moment around the longitudinal axis is generated by the imbalance of vertical tire forces between the left and right wheels and is given by equation (31).

$$M_{x,des} = \frac{B}{2}\left((F_{z11} - F_{z12}) - (F_{z21} - F_{z22})\right) \tag{31}$$

with B denotes the vehicle track width. This relationship is used to regulate roll behavior by appropriately distributing vertical loads among the wheels. The yaw moment about the vertical axis, which directly corresponds to the yaw correction command $\Delta M_z[n]$ generated by the RK4-MPC, is expressed in equation (32).

$$M_{z,des} = a \cdot \left(F_{y11} + F_{y12}\right) - b \cdot \left(F_{y21} + F_{y22}\right) - B \cdot \frac{F_{x11} - F_{x12}}{2} + B \cdot \frac{F_{x21} - F_{x22}}{2} \tag{32}$$

With a and b represent the distances from the vehicle center of gravity to the front and rear axles, respectively. Equation (32) establishes the mapping between wheel-level longitudinal and lateral forces and the desired yaw moment, ensuring that the yaw correction computed by the MPC can be physically realized by the wheels. To ensure tire–road feasibility, the wheel forces are constrained by the friction ellipse condition, as defined in equation (33).

$$F_{xij}^2 + F_{yij}^2 \leq \left(\mu_{ij} F_{zij}\right)^2 \tag{33}$$

with μ_{ij} denotes the tire–road friction coefficient at each wheel. This constraint limits the combined longitudinal and lateral forces within the available friction capacity and prevents wheel slip under varying surface conditions. In addition, actuator limitations are considered by bounding the longitudinal tire force through equation (34).

$$-\frac{T_{eij,max}}{R_e} \leq F_{xij} \leq \frac{T_{eij,max}}{R_e} \tag{34}$$

where $T_{eij,max}$ is the maximum drive or braking torque of the wheel actuator and R_e is the effective tire radius. This constraint ensures that the force allocation remains within the physical capabilities of the wheel motors. Through the constraints defined in equation (29) to equation (34), the WFDC computes a feasible set of wheel forces that satisfies the desired vehicle-level force and moment commands while respecting tire–road interaction limits and actuator constraints. The resulting wheel forces are then applied to the 4WS vehicle model to execute the control commands generated by the RK4-based MPC in a physically consistent manner. WFDC performance is assessed through the average tracking error $\Delta|d|$, average heading error $\Delta\theta$, maximum wheel slide ratio λ, and dynamic performance during constrained steering maneuvers.

III. Results and Discussions

This section presents the results of evaluating the proposed framework in various simulated evaluation environments. The evaluation assessed the system's performance, ensuring it operates as expected and meets the predefined requirements. The system was tested on a device with the specifications listed in Table 1.

The CasADi backbone [40] was employed to efficiently differentiate numerical optimization, while IPOPT [41] was used in solving a nonlinear programming formulation. The proposed framework was tested in simulations using raytracing and mapping using NVIDIA Isaac Sim, which accurately modeled real-world dynamics, while ROS2 ensured communication between vehicle systems, sensor data integration, and control actions.

A. Path planning algorithm evaluation

This section compares LEA*, A*, LWA*, LazySP, and LRA* algorithms in a 2D planning problem involving randomly generated rectangular obstacles, as shown in Figure 7. The environment is divided into sparse, medium, and cluttered settings with 8, 16, and 24 obstacles, respectively. Graph sizes range from 200

Table 1.
Device specification.

Description	Specifications
System operating	Ubuntu 22.04 LTS
Processor	Intel Core i7-10700F
Graphic card	NVIDIA RTX 3050
RAM	32GB DDR4
Library	ROS2, IPOPT, CasADi

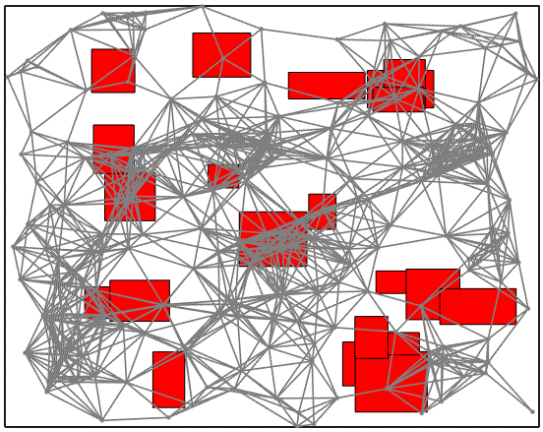


Figure 7. 2D planning problem.

to 20,000 vertices, with obstacle locations randomly distributed within the environment's boundaries. The connecting radius is determined as a function of N, following the lazy PRM algorithm [18]. Ten environments were sampled for each category, including sparse, medium, and cluttered settings. For each combination of environment and graph size, 50 start-goal vertex pairs were sampled, resulting in 7500 planning problems.

The performance comparison across different planning algorithms, as shown in Figure 8 and Figure 9, highlights the efficiency of LEA* in both $\varepsilon = 1$ and $\varepsilon = 2$ configurations. LEA* consistently outperforms A*, Lazy SP, LWA*, and LRA* in terms of average planning time, significantly as the number of obstacles increases. With $\varepsilon = 1$, LEA* reduces the planning time significantly, maintaining a stable performance even as the complexity of the environment increases, with times as low as 0.0167s for N=1.000, o=8. In contrast,

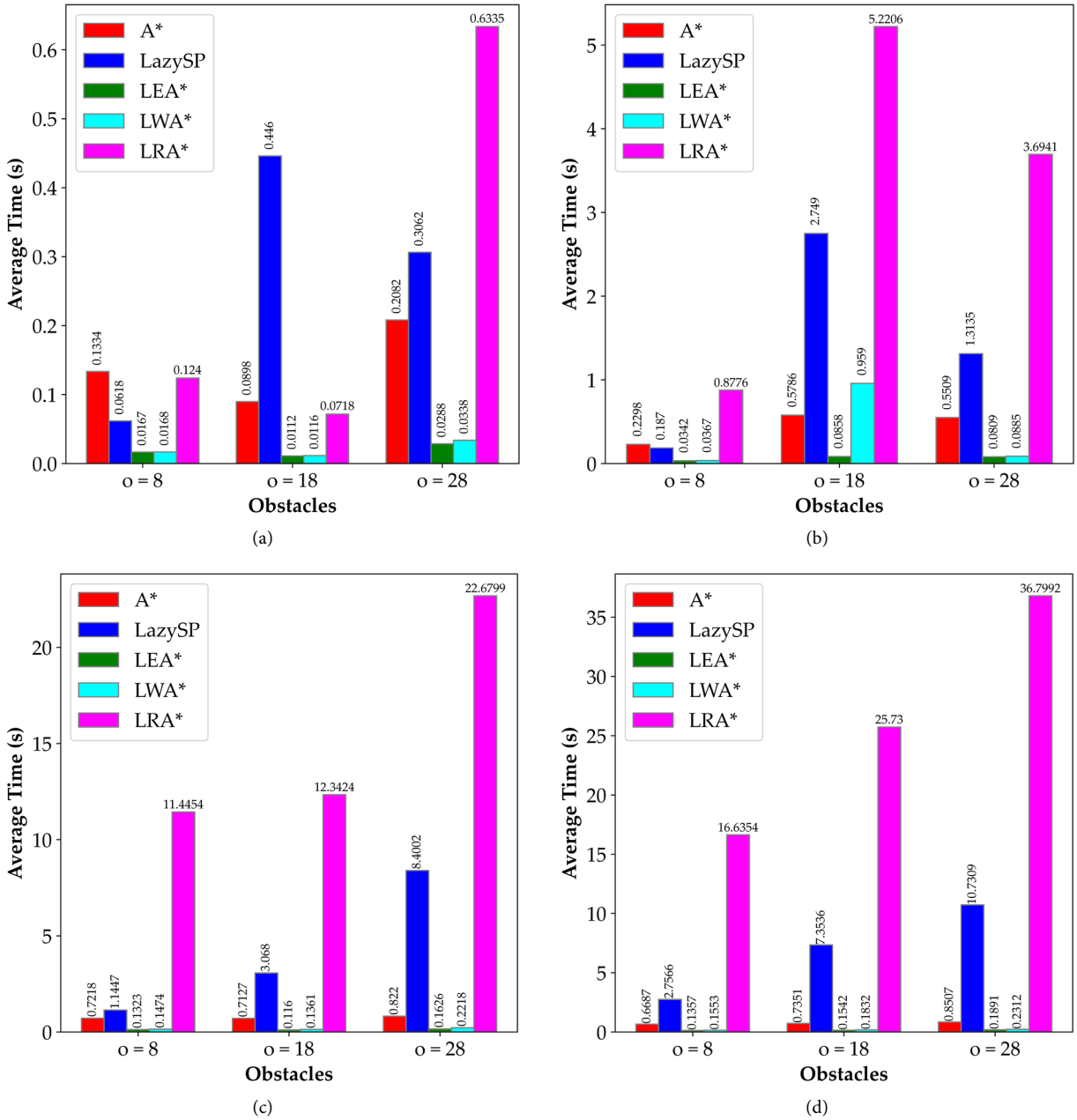


Figure 8. Planning time comparison with ε=1 across different N, (a) 1000; (b) 5000; (c) 10000; (d) 20000.

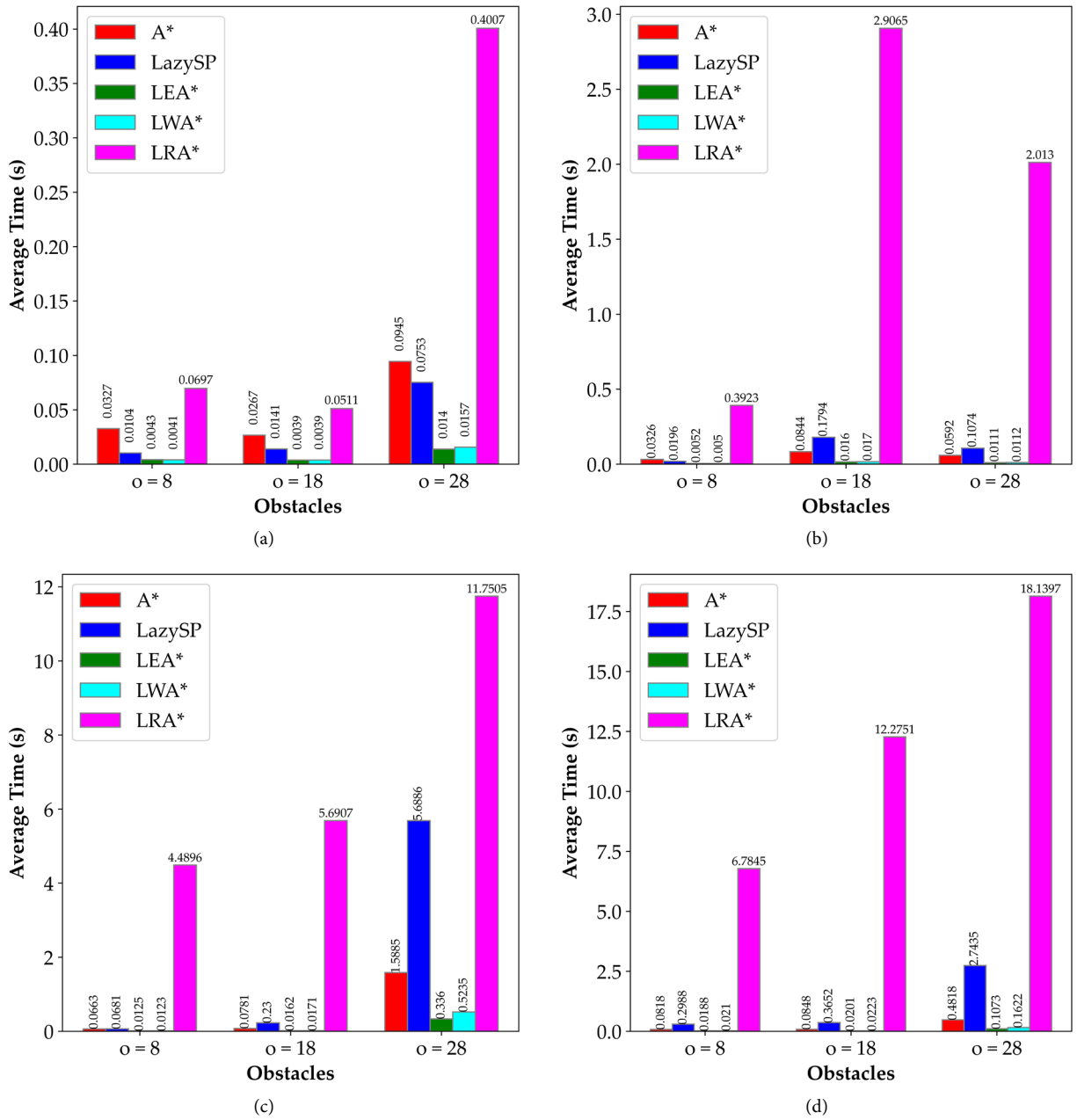


Figure 9. Planning time comparison with ε=2 across different N, (a) 1000; (b) 5000; (c) 10000; (d) 20000.

A* and Lazy SP exhibit significantly higher times, reaching 10.7309s under similar conditions for N=2.0000, o=28. Similarly, LEA* with $\varepsilon = 2$ showcases a further reduction in planning time, particularly in environments with more significant obstacle numbers, demonstrating its adaptive capability in more complex scenarios. This reduction, combined with its low time overhead, positions LEA* as a robust and scalable solution for dynamic motion planning in environments with varying complexities.

LEA* consistently outperforms other algorithms in edge evaluations across varying N values. Figure 10 and Figure 11 show that LEA* reduces edge evaluations by up to 95 % compared to A*, maintaining this advantage even with $\varepsilon = 2$. It also surpasses Lazy SP and LRA* in more complex obstacle setups like N=20000, making LEA* a highly efficient and adaptable solution for motion planning.

The comparison of path-planning algorithms reveals that LEA* consistently outperforms other methods in both planning time and edge evaluations. LEA* is significantly faster, completing tasks up to 87.5 % quicker than A*, such as 0.0167 seconds at N=1000 with 8 obstacles. For larger graphs with 28 obstacles and N=20000, LEA* reduces edge evaluations by 96.1% while maintaining 77.8 % faster planning times. Despite LazySP evaluating fewer edges, its longer planning time makes LEA* the more efficient option regarding computational effort and speed.

Further analysis focuses on the LEA* algorithm's ability to reduce runtime while preserving critical path characteristics, such as smoothness, clearance, and path

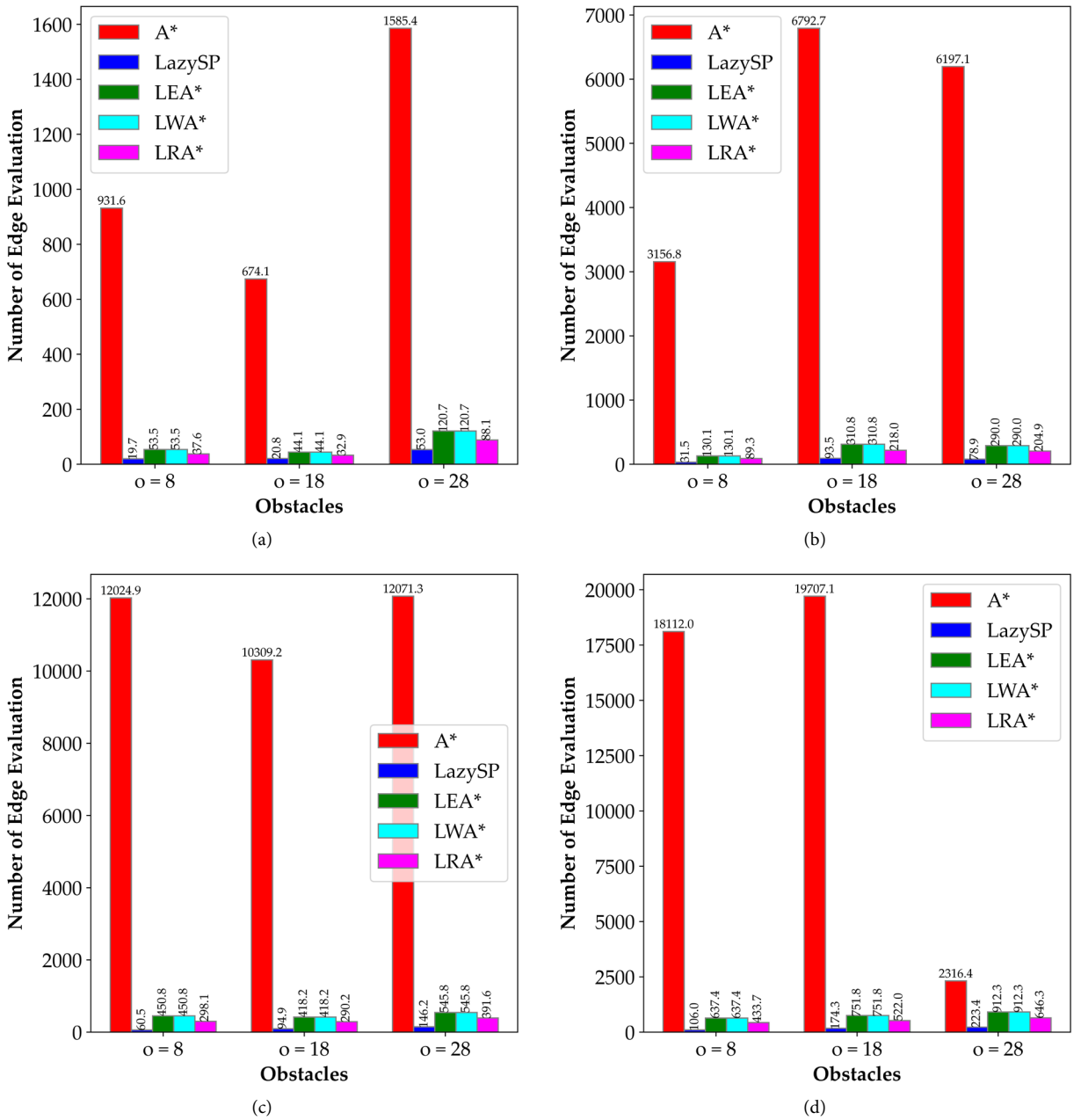


Figure 10. The number of edge evaluation comparisons with ε=1 across different N, (a) 1000; (b) 5000; (c) 10000; (d) 20000.

length. The two large-scale environments tested are presented in Figure 12. Figure 12(a) shows a previously known environment, while Figure 12(b) depicts the other environment, which was unknown. Table 2 compares A* and LEA* runtimes in known and unknown environments.

LEA* reduced cumulative runtime by 45 %, decreasing it from 0.215 seconds with A* to 0.12 seconds in known environments, while average runtime saw a 52 % reduction from 0.25 seconds to 0.012 seconds. In unknown environments, LEA* achieved even more significant improvements, cutting cumulative runtime by a factor of 4.8, from 4.2 seconds to 0.87 seconds, and reducing average runtime by a factor of 4.6, from 0.092 seconds to 0.020 seconds. These outcomes highlight LEA* efficiency in adapting to environmental changes, minimizing delays, and

Table 2.
Runtime comparison of A* and LEA* in known and unknown environments.

Case	Algorithm	execution	t_{max}	t_{cum} (s)	t_{avg}
Known	A*	1	-	0.59	0.012
	LEA*	35	0.21	0.25	-
Unknown	A*	32	0.32	4.2	0.092
	LEA*	108	0.15	0.87	0.020

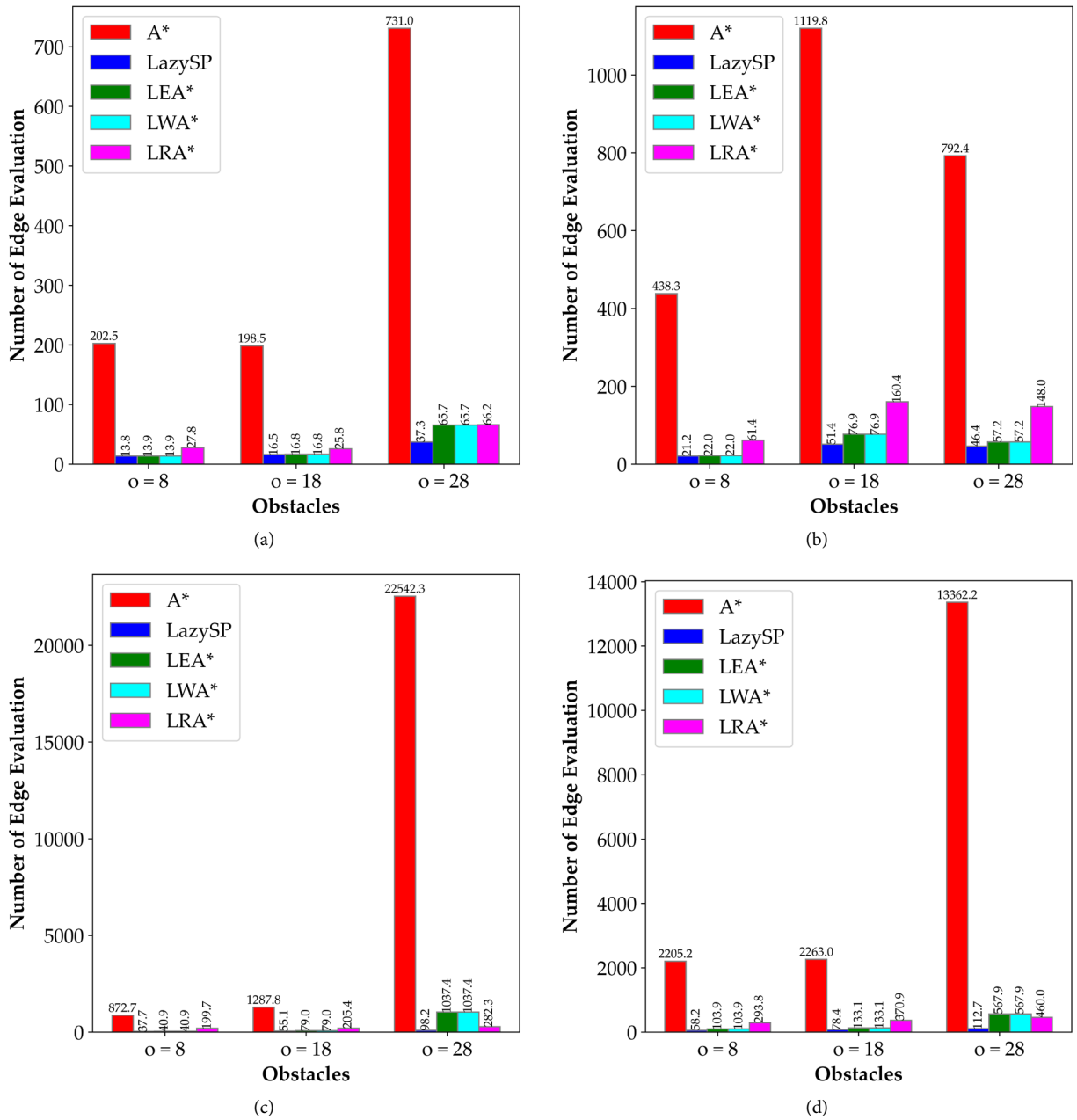


Figure 11. The number of edge evaluation comparisons with ε=2 across different N, (a) 1000; (b) 5000; (c) 10000; (d) 20000.

enhancing performance in challenging obstacle-dense settings.

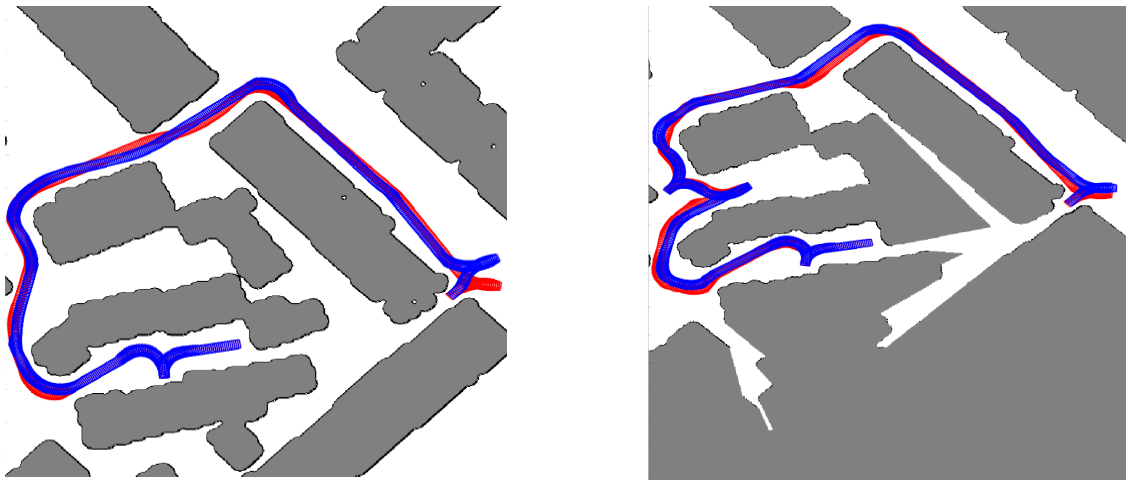
The analysis also compares path smoothness and obstacle clearance, as presented in Table 3. Smoothness is measured by the root mean square (RMS) of curvature change $(\varkappa)$, as defined by equation (35), which was introduced in [42] obstacle proximity is measured by the average and maximum distances p_{avg} and p_{max} using the generalized Voronoi diagram (GVD), which assigns values between 0 and 1 based on the distance to obstacles. The path length is also assessed after reaching the goal pose.

$$\varkappa_{RMS} = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1}\left(\frac{\varkappa_{i+1}-\varkappa_i}{ds}\right)^2} = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1}\varkappa^2} \quad (35)$$

LEA* achieves a κ_{RMS} of 0.20, offering a smoother path by 51 % compared to A* in known environments and showing a 20 % improvement in unexplored

Table 3.
Path comparison of A* and LEA* in known and unknown environments.

Environment	Method	$\varkappa_{RMS}$	p_{max}	p_{avg}	$l\ (m)$
Known	A*	0.41	0.65	0.0021	224
	LEA*	0.20	1.0	0.0042	220
Unknown	A*	0.55	1.0	0.0031	245
	LEA*	0.45	1.0	0.0053	241



(a) (b)

Figure 12. Motion records in known and unknown environments.

settings. While LEA* paths are slightly closer to obstacles on average, the proximity remains within safe limits. LEA* also reduces path length by 1.8 % in known environments and 1.6 % in unexplored areas, optimizing route planning. Overall, LEA* significantly enhances planning efficiency, delivering smoother paths with shorter lengths while maintaining safety and reducing runtime.

B. Motion planning performance comparison

The proposed framework was evaluated using the sparse collision-checking method in [43], which handles non-rotationally invariant vehicle shapes through disk-based footprint decomposition and supports on-demand collision checking on occupancy grids. This property aligns with the lazy edge evaluation strategy of LEA*, where collision checks are performed only for a limited subset of candidate edges. All experiments were conducted on the hardware listed in Table 1 using the vehicle parameters in Table 4.

The framework will be tested in various environments, including narrow spaces and complex large parking environments, to evaluate the motion planning performance and stability. The NVIDIA Isaac Sim [44] simulator was used to test the trajectory generator with its realistic physics engine and accurate vehicle modeling. Simulating exploration in unknown environments with ray tracing and mapping ensures the framework's adaptability in real-world conditions.

Table 4.
Vehicle evaluation parameters.

Notation	Description	Value
$l/w/h$	Dimensions	3.5/2.1/0.6 m
m	Vehicle total mass	1530 kg
m_s	Sprung mass	1210 kg
r_{xy}	Spatial rate of the LEA*	0.625 m
I_z	Yaw rate of the LEA*	10°
δ	Max steering angle	90°
r_f	Dist. front axle to IVC	1.40 m
r_r	Dist. rear axle to IVC	1.60 m

1) Extended system model

The LEA* and standard A* algorithms were compared to assess their performance under varying conditions. Table 5 presents the relative execution times for both algorithms, considering different parameter configurations, such as the frequency of extended motion primitives f_{ext} and steering adjustments δ. As expected, the execution time increased when finer motion primitives were applied due to the increased cumulative number of nodes explored. However, the increase remained within a similar order of magnitude to the standard A* algorithm. Notably, when $f_{ext} = 5$ and $\delta = 90°$, the execution time increased slightly, maintaining the algorithm's suitability for real-time applications. These parameters were chosen for further experiments because they balance computational costs and improve maneuverability.

The evaluation then shifted to analyzing the performance of the LEA* algorithm in narrow environments to determine whether it improves vehicle maneuverability over the standard A* algorithm. In this comparison, two environments were used: a narrow corridor with a circular plate of 8.6 meters in diameter and a narrower variant with a plate of 6.9 meters. The recorded motion planning in Figure 13 compares the driven paths in these environments. The red paths,

Table 5.
Relative values of the extended against the standard LEA* algorithm.

Method	f_{ext}	δ	Cum. of nodes	t_{avg}
A*	-	-	1.0	1.0
LEA*	1	45°	2.26	2.28
	5	45°	1.26	1.21
	1	90°	1.68	1.59
	5	90°	1.14	1.12

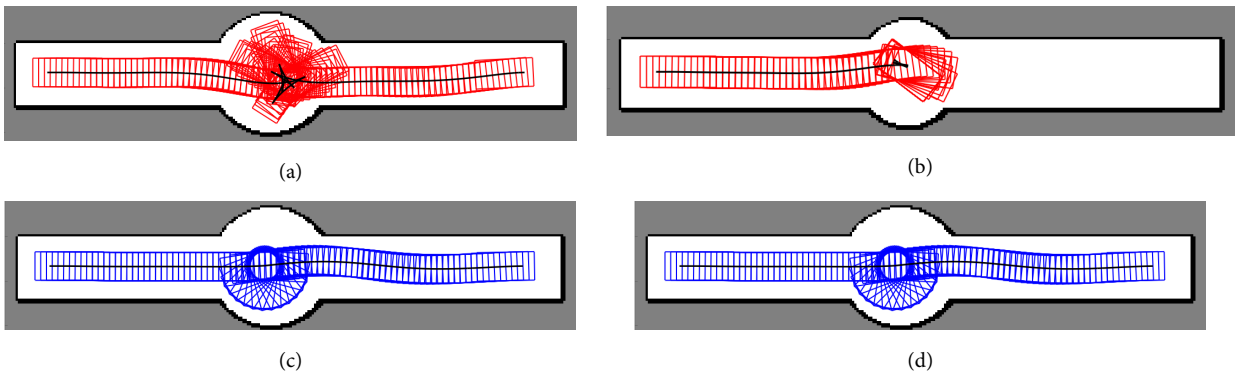


Figure 13. Path Comparison in two narrow environments. a) A* and c) LEA* in the narrow environment; b) A* and (d) LEA* in a narrower variant.

representing the standard A* algorithm, show multiple directional shifts in both scenarios, especially in the first environment. By contrast, the blue paths of the LEA* algorithm reveal a more optimized and continuous trajectory, especially in the narrower variant where the standard A* was unable to reach the goal.

Both algorithms successfully reached the goal pose in the first environment, but the standard A* required eight directional changes, indicating less efficient navigation. On the other hand, the LEA* algorithm achieved the same goal with fewer directional adjustments, demonstrating its capacity for smoother maneuvering in tight spaces. The second, narrower environment further exposed the limitations of the standard A* algorithm, as it failed to reach the goal. In contrast, the LEA* algorithm navigated the space effectively, reaching the target quickly. The LEA* algorithm's ability to navigate these confined spaces with minimal directional changes highlights its superior maneuverability and suitability for real-time path planning in complex environments, making it a robust alternative to the standard A* approach.

2) Dynamic performance comparison

The evaluation conducted in the parking environment, as shown in Figure 14, assesses the vehicle's agility and maneuvering precision. The proposed framework was adjusted with a reduced time horizon to fit the confined space of the parking scenario. Velocity commands were converted into specific angular and velocity instructions for each wheel through the vehicle control unit, operating at 10 Hz. Motion records compared three models: OMNI, S-4WS, and the proposed constrained 4WS framework. Wheel slippage was assessed using the slide ratio (λ), as defined in equation (36), which effectively measures traction control during vehicle turns. This analysis highlights the improvements in stability and precision under the proposed system approach.

$$\lambda = \max\left(v_w \cdot \cos\left(\delta_{real} - \delta_{ref}\right)\right) \tag{36}$$

The reference steering angle δ_{ref} for each wheel is derived from the ideal vehicle state at a given speed. The slide ratio (λ), a key indicator of traction, is shown over time in Figure 15. As defined by equation (34), (λ) is determined by the maximum product of wheel velocity (v_w) and the cosine of the difference between the actual steering angle (δ_{real}) and (δ_{ref}). This difference indicates the deviation from the optimal steering trajectory, with larger values pointing to more significant slippage and reduced traction. Tracking (λ) over time helps assess the vehicle's stability and handling, with lower (λ) values indicating better control and traction, and higher values suggesting instability.

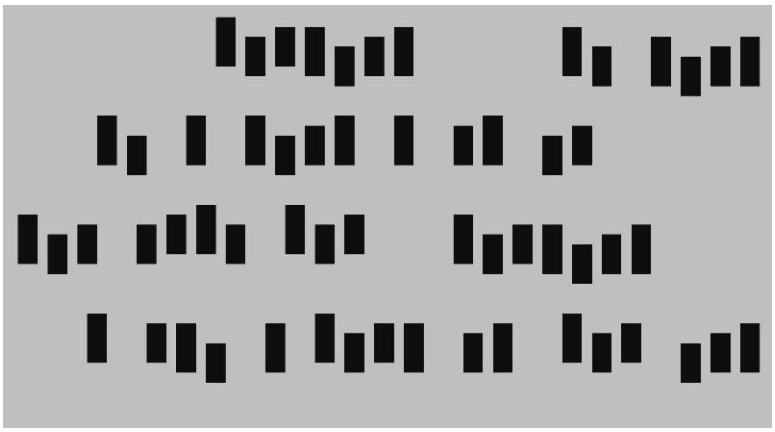


Figure 14. The map of the experiment parking scene.

Figure 15 compares the slide ratios and Figure 16 the motion records of the S-4WS, OMNI, and proposed framework. In Figure 16(a), S-4WS faces difficulties due to inadequate steering constraints, leading to erratic movements and significant wheel slippage. The slide ratio spikes repeatedly, peaking around 2.0, illustrating the system's instability in handling precise maneuvers as shown in Figure 16(a). The OMNI system, shown in the second row, offers better control by employing a bicycle-model chassis as shown in Figure 16(b), reducing the slide ratio peaks to approximately 1.0, representing a 50 % reduction in slippage compared to S-4WS as shown in Figure 15(b). However, this improved stability comes with a trade-off, OMNI requires more time for posture adjustments, prolonging the overall execution time while offering better control.

The proposed system demonstrates superior performance, achieving an 80 % reduction in slippage compared to OMNI and 90 % relative to S-4WS, maintaining a slide ratio below 0.2 throughout the motion as shown in Figure 15(c). This system ensures smoother, faster, and more precise movements, with greater flexibility to navigate complex paths and reduced tracking error, as shown in Figure 16(c).

3) Stability and force distribution performance

The stability, force distribution, and computational performance of the proposed framework were evaluated using 100 randomly sampled start–end point pairs in a simulated parking environment implemented in NVIDIA Isaac Sim, as illustrated in Figure 17. Three steering and execution methods were considered: OMNI, which neglects steering and wheel force constraints; S-4WS, which enforces symmetric front–rear steering equivalent to a bicycle model; and the proposed constrained 4WS framework, which integrates predictive motion control with WFDC for wheel-level force allocation. The evaluation focuses on motion execution accuracy, wheel slip behavior, and dynamic response under steering-constrained maneuvers. The average tracking error $\Delta|d|$, average heading error $\Delta\theta$, and the maximum slide ratio λ among wheels and dynamic performance are tabulated in Table 6.

Based on Table 6, the proposed constrained 4WS framework achieves the best overall ranking in terms of tracking accuracy, wheel slip reduction, and dynamic response, followed by S-4WS and OMNI. The proposed constrained 4WS framework yields the lowest tracking

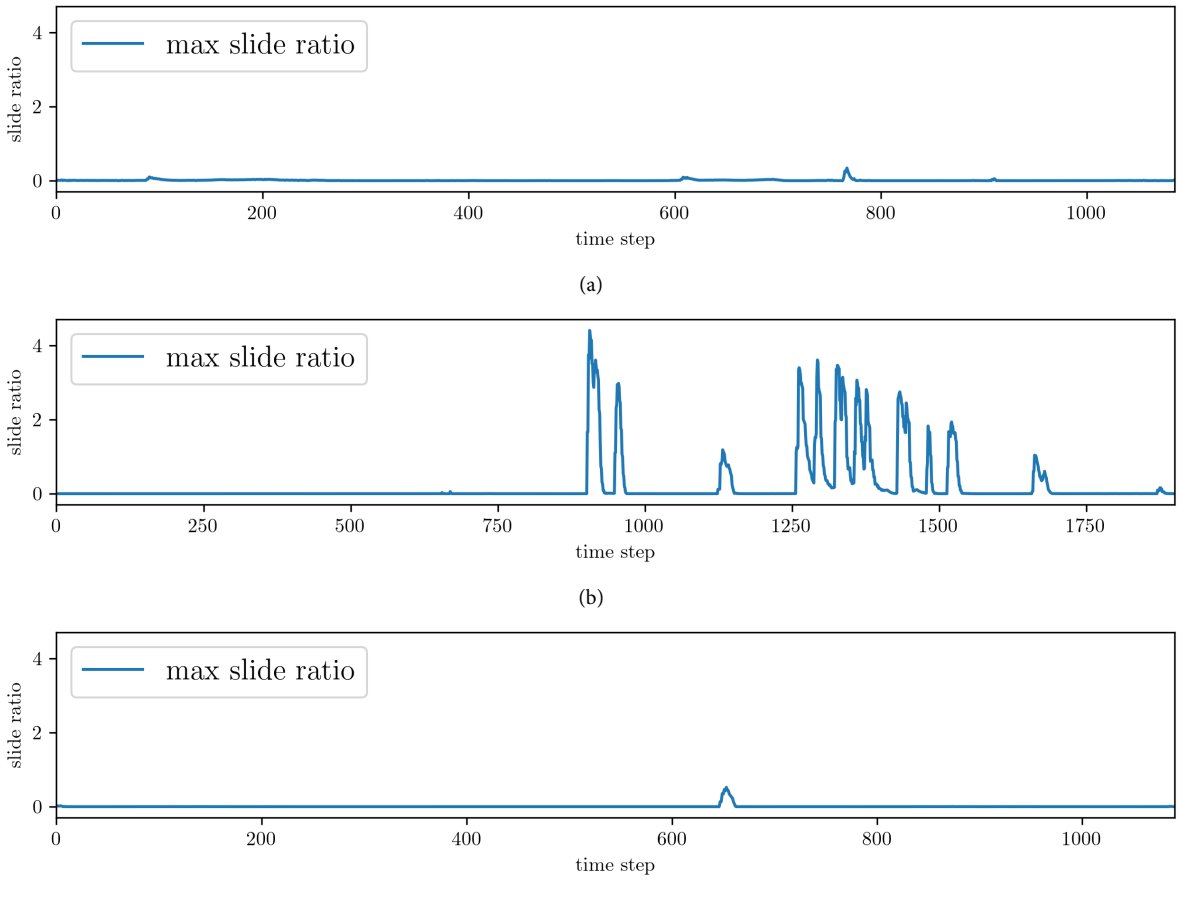
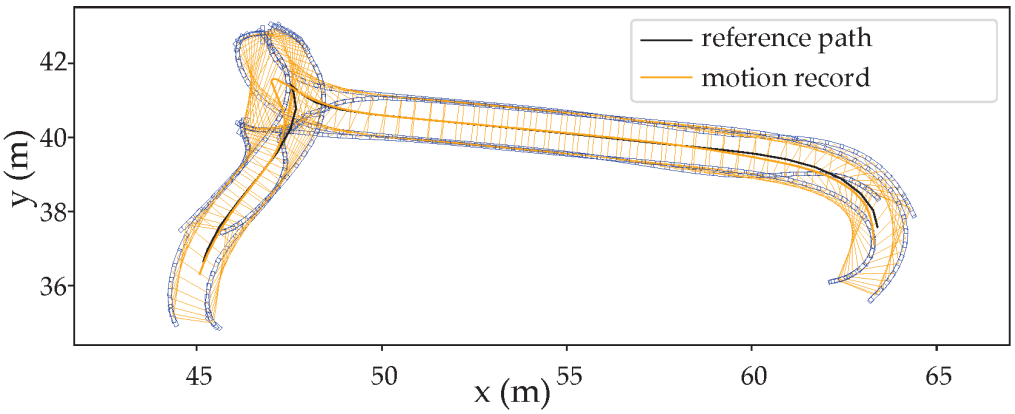
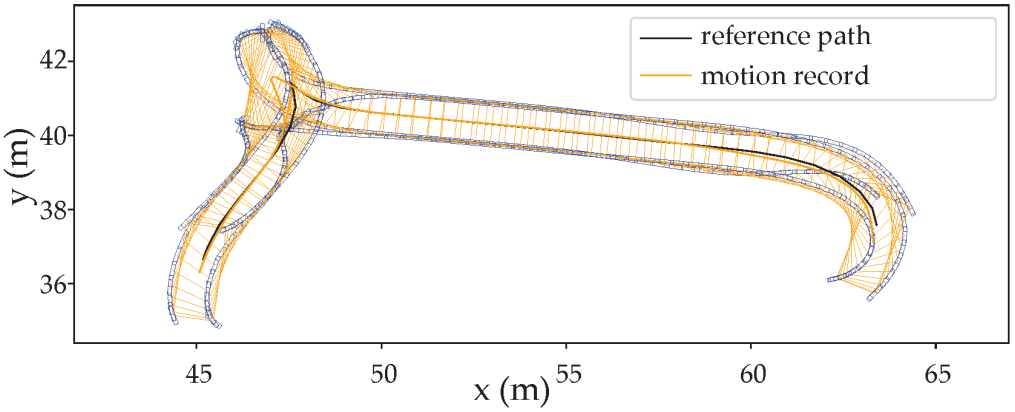


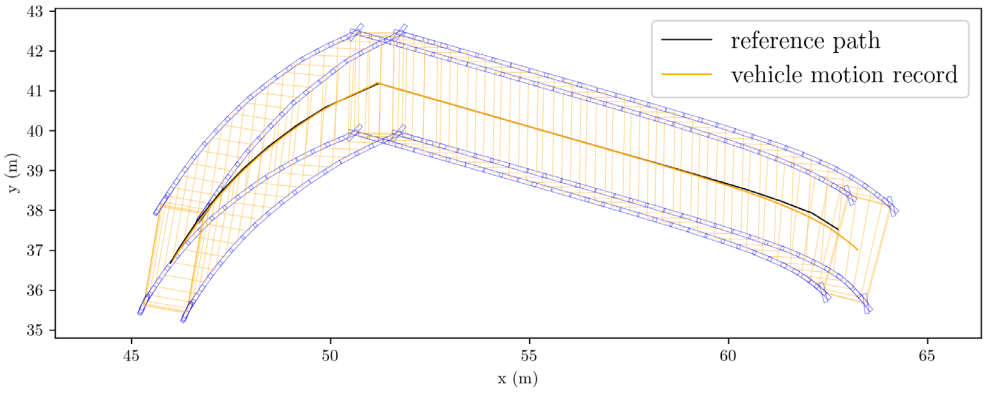
Figure 15. Slide ratio comparison: (a) S-4WS; (b) OMNI; (c) proposed.



(a)



(b)



(c)

Figure 16. Motion records comparison: (a) S-4WS; (b) OMNI; (c) proposed.

error of 0.15 m, corresponding to an improvement of 34.8 % compared to OMNI and 28.6 % compared to S-4WS. For heading accuracy, the proposed method also achieves the smallest heading error of 0.10 rad, representing a 37.5 % reduction relative to OMNI and a 28.6 % reduction relative to S-4WS. In terms of wheel slip, the proposed method records the lowest slip ratio of 0.008, which is reduced by approximately 88.4 % compared to OMNI and 46.7 % compared to S-4WS. Moreover, the dynamic response of the proposed framework shows a lower yaw acceleration of 0.07 m/s², corresponding to reductions of 53.3 % relative to OMNI and 50.0 % relative to S-4WS, while maintaining comparable longitudinal and lateral acceleration levels. Although the yaw jerk of the proposed method remains higher than that of S-4WS due to increased steering flexibility, it is reduced by 93.3 % compared to OMNI, reflecting smoother force transitions enabled by WFDC during dynamic maneuvers. These results conclude that the proposed constrained 4WS framework provides improved tracking accuracy and reduced wheel slip while maintaining balanced dynamic behavior compared to S-4WS and OMNI steering methods.

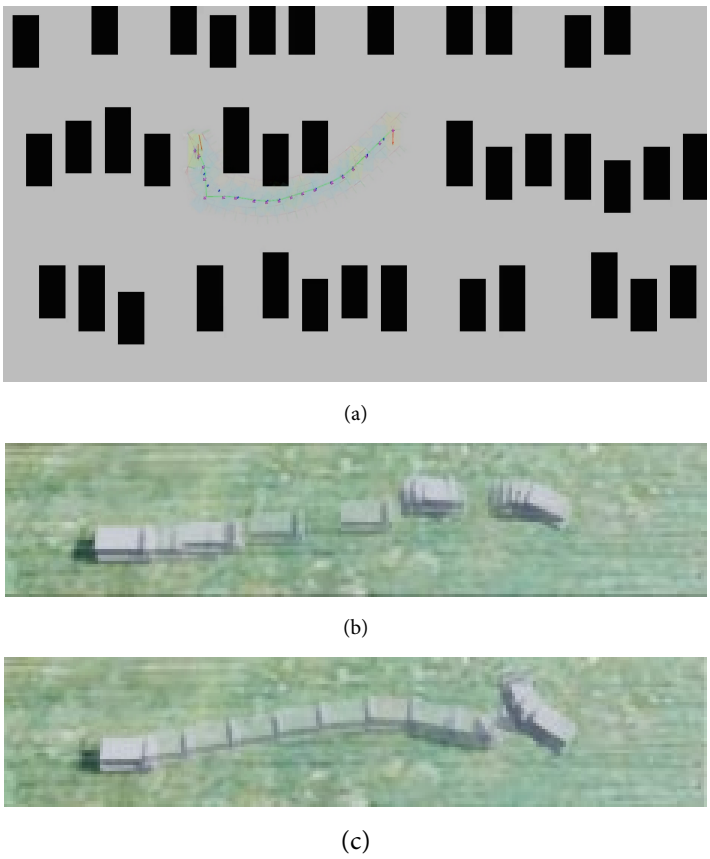


Figure 17. Proposed motion planning, (a) car parking environment; (b) 4WS vehicle mode switching; (c) movement optimization using the proposed system.

Further investigation is performed by examining the statistical computational performance of the proposed framework, as tabulated in Table 7. The proposed system achieves the fastest search time of 0.5234 s for 30 steps, which is 83.1 % faster than the OMNI method and 37.1 % faster than the S-4WS method. This reduction reflects the effectiveness of lazy edge evaluation combined with steering-constrained sampling in limiting unnecessary graph expansion. For optimization time, the proposed system requires 1.4892 s, which is 30.3 % faster than S-4WS, while OMNI exhibits a shorter optimization time of 0.2431 s due to its simplified holonomic formulation that neglects steering and force constraints. To further analyze the computational overhead, the optimization time excluding the initialization phase is evaluated and reported as the headless time. The proposed framework records a headless time of 0.9023 s, which is approximately 47.3 % faster than OMNI and 38.7 % faster than S-4WS, indicating that a significant portion

Table 6.
Comparison of slip ratio, acceleration, and jerk metrics under different steering methods.

Method	$\Delta\|\boldsymbol{d}\|$ (m)	$\Delta\boldsymbol{\theta}$ (rad)	Slide ratio $\boldsymbol{\lambda}$	Avg. velocity (m/s)			Avg. acceleration (m/s2)			Avg. jerk (m/s3)		
				$\boldsymbol{v_x}$	$\boldsymbol{v_y}$	$\boldsymbol{v_{yaw}}$	$\boldsymbol{a_x}$	$\boldsymbol{a_y}$	$\boldsymbol{a_{yaw}}$	$\boldsymbol{j_x}$	$\boldsymbol{j_y}$	$\boldsymbol{j_{yaw}}$
OMNI	0.23	0.16	0.069 ± 0.24	1.16	0.83	0.18	0.39	0.35	0.15	29.7	26.14	11.96
S-4WS	0.21	0.14	0.015 ± 0.08	1.46	0.06	0.18	0.30	0.15	0.14	1.26	1.20	0.640
Proposed	0.15	0.10	0.008 ± 0.04	1.33	0.56	0.12	0.29	0.21	0.07	1.73	1.29	0.801

Table 7.
Computation time comparison in 30 steps.

Method	Search (s)	Optimize (s)	Headless (s)
OMNI	3.1025	0.2431	0.1712
S-4WS	0.8315	2.1367	1.4721
Proposed	0.5234	1.4892	0.9023

of the computation can be further reduced through implementation-level optimization. Overall, the statistical computational results demonstrate that integrating WFDC with steering-constrained predictive motion control improved efficiency in both search and optimization stages compared to OMNI and S-4WS, making it more suitable for real-time trajectory planning under steering-constrained scenarios.

IV. Conclusion

This work presents an efficient motion planning framework for 4WS autonomous vehicles by integrating LEA* for path planning and adaptive RK4-MPC with WFDC for motion execution. The proposed framework reduces planning time by 87.5 % and edge evaluations by 96.1 % compared to A*, while improving path smoothness by 51 %. The proposed motion control demonstrates superior performance, achieving an 80 % reduction in slippage compared to OMNI and 90 % relative to S-4WS, maintaining a slide ratio below 0.2 throughout the motion. Furthermore, the proposed framework was validated using 100 randomly sampled start–end point pairs in a simulated parking environment implemented in NVIDIA Isaac Sim. Quantitative results show that the proposed constrained 4WS framework, integrating predictive motion control with WFDC, achieves the lowest tracking error of 0.15 m and heading error of 0.10 rad, corresponding to improvements of up to 34.8 % and 37.5 % compared to OMNI, and 28.6 % compared to S-4WS. By allocating longitudinal and lateral forces at the wheel level, WFDC reduces the slip ratio to 0.008, representing reductions of 88.4 % relative to OMNI and 46.7 % relative to S-4WS, while lowering yaw acceleration by more than 50 % compared to both baselines. In terms of computational efficiency, the proposed framework achieves a search time of 0.5234 s, which is 83.1 % faster than the OMNI method and 37.1 % faster than the S-4WS method, and an optimization time of 1.4892 s, which is 30.3 % faster than S-4WS, demonstrating its suitability for real-time trajectory planning and stable motion execution under steering-constrained conditions. These improvements demonstrate that the framework system excels in addressing challenges encountered in navigating complex and dynamic environments, significantly enhancing vehicle stability, maneuverability, and overall navigation efficiency compared to baseline approaches. The combined use of dynamic constraints, steering limits, and real-time adaptability highlights the system's applicability to real-world autonomous navigation. Future work could explore integrating learning-based environment perception to improve planning robustness and adaptability in more complex environments.

Declarations

Author contribution

All authors contributed equally as the main contributors of this paper. All authors read and approved the final paper.

Funding statement

This work was supported by UEESRG and funded by Direktorat Riset, Teknologi, dan Pengabdian Masyarakat, Direktorat Jenderal Pendidikan Tinggi, Riset, dan Teknologi, Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, under grant no. 070/E5/PG.02.00.PL/2024 with contract no. 153.12.6/UN37/PPK.10/2024.

Competing interest

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

Additional information

Reprints and permission: information is available at https://mev.brin.go.id/.

Publisher's Note: National Research and Innovation Agency (BRIN) remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

[1] R. Reddy, L. Almeida, P. M. Santos, H. Kurunathan, and E. Tovar, "Energy savings and emissions reduction of BEVs at an isolated complex intersection," *Transp Res D Transp Environ*, vol. 136, p. 104403, Nov. 2024.

[2] G. Xu *et al.*, "Distributed multi-vehicle task assignment and motion planning in dense environments," *IEEE Transactions on Automation Science and Engineering*, vol. 21, pp. 7027-7039, Oct 2023.

[3] S. Nie *et al.*, "Study on path planning for off-road autonomous vehicles in complex terrains | 考虑复杂地形的越野环境无人车辆路径规划研究," *Jixie Gongcheng Xuebao/Journal of Mechanical Engineering*, vol. 60, no. 10, pp. 261–272, 2024.

[4] Y. Zhao, C. Lei, Y. Shen, Y. Du, and Q. Chen, "Human-vehicle cooperative visual perception for autonomous driving under complex traffic environments," In: Karlinsky, L., Michaeli, T., Nishino, K. (eds) Computer Vision – ECCV 2022 Workshops. ECCV 2022. *Lecture Notes in Computer Science, vol 13801. Springer, Cham.*

[5] M. Seo, S. Shin, K. Kim, and K. Choi, "Reinforcement learning-based collision avoidance of a connected and automated vehicle at merging roads," in *Proceedings - IWIS 2023: 3rd International Workshop on Intelligent Systems*, 2023.

[6] T. A. Nguyen, "Establishing the dynamics model of the vehicle using the 4-wheels steering systems," *Mathematical Modelling of Engineering Problems*, vol. 7, no. 3, pp. 436–440, Sep. 2020.

[7] L. Zhong, X. Jiang, W. Jia, and W. Shi, "4WS intelligent fire-fighting robot trajectory tracking control based on adaptive cornering stiffness," in *IEEE Access*, vol. 12, pp. 36083–36095, 2024.

[8] Y. Chu, C. Wang, X. Zhou, Z. Zhang, and W. Zhao, "Instantaneous center of rotation tracking control of four-wheel independent steering vehicles under large-curvature turning conditions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 11, pp. 17965–17978, Nov. 2024.

[9] P. Hang and X. Chen, "Towards autonomous driving: Review and perspectives on configuration and control of four-wheel independent drive/steering electric vehicles," *Actuators*, vol. 10, no. 8, Aug. 2021.

[10] S. Venu and M. Gurusamy, "A comprehensive review of path planning algorithms for autonomous navigation," *Results in Engineering*, vol. 28, p. 107750, Dec. 2025.

[11] Y. Wang, Q. Yang, and W. Qu, "A collision-free transition path planning method for placement robots in complex environments," *Complex & Intelligent Systems*, vol. 10, no. 6, pp. 8481–8500, Sep. 2024.

[12] A. Alexander, K. Venkatesan, J. Mounsef, and K. Ramanujam, "A comprehensive survey of path planning algorithms for autonomous systems and mobile robots: Traditional and modern approaches," *IEEE Access*, vol. 13, pp. 176287–176326, 2025.

[13] J. P. Vásconez et al., "Comparison of path planning methods for robot navigation in simulated agricultural environments," *Procedia Comput Sci*, vol. 220, pp. 898–903, 2023.

[14] B. Cohen, M. Phillips, and M. Likhachev, "Planning single-arm manipulations with N-arm robots," *Proceedings of the International Symposium on Combinatorial Search*, vol. 6, no. 1, pp. 226–227, Sep. 2021.

[15] C. M. Dellin and S. S. Srinivasa, "A unifying formalism for shortest path problems with expensive edge evaluations via lazy best-first search over paths with edge selectors," in *ICAPS'16: Proceedings of the Twenty-Sixth International Conference on International Conference on Automated Planning and Scheduling*, London: AAAI Press, Jul. 2016, pp. 459–467. Accessed: Dec. 20, 2025. [Online].

[16] D. Zheng and P. Tsiotras, "LEA*: An a* variant algorithm with improved edge efficiency for robot motion planning," *ArXiv preprint*, Sep. 2023, Accessed: Dec. 21, 2025. [Online].

[17] R. Bohlin and L. E. Kavraki, "Path planning using lazy PRM," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), IEEE*, vol. 1, pp. 521–528 vol.1, 2000.

[18] G. Du, Y. Zou, X. Zhang, Z. Li, and Q. Liu, "Hierarchical path planning and motion control framework using adaptive scale based bidirectional search and heuristic learning based predictive control," *IEEE Trans Veh Technol*, vol. 74, no. 6, pp. 8647–8663, Jun. 2025.

[19] S. Jameel Al-Kamil and R. Szabolcsi, "Optimizing path planning in mobile robot systems using motion capture technology," *Results in Engineering*, vol. 22, p. 102043, Jun. 2024.

[20] K. Yang, X. Tang, Y. Qin, Y. Huang, H. Wang, and H. Pu, "Comparative study of trajectory tracking control for automated vehicles via model predictive control and robust H-infinity state feedback control," *Chinese Journal of Mechanical Engineering*, vol. 34, no. 1, pp. 74-, Aug. 2021.

[21] M. Charest-Finn and S. Pejhan, "Model predictive control used in passenger vehicles: an overview," *Machines*, Vol. 12, no. 11, Nov. 2024.

[22] J. Nan, Z. Ge, X. Ye, A. F. Burke, and J. Zhao, "Model predictive control for autonomous vehicle path tracking through optimized kinematics," *Results in Engineering*, vol. 24, p. 103123, Dec. 2024.

[23] Y. Chen, G. Feng, S. Wu, and X. Tan, "A new hybrid model predictive controller design for adaptive cruise of autonomous electric vehicles," *J Adv Transp*, vol. 2021, no. 1, p. 6626243, Jan. 2021.

[24] B. Zhao, N. Xu, H. Chen, K. Guo, and Y. Huang, "Design and experimental evaluations on energy-efficient control for 4WIMD-EVs considering tire slip energy," *IEEE Trans Veh Technol*, vol. 69, no. 12, pp. 14631–14644, Dec. 2020.

[25] L. Ge, Y. Zhao, S. Zhong, Z. Shan, and K. Guo, "Efficient nonlinear model predictive motion controller for autonomous vehicles from standstill to extreme conditions based on split integration method," *Control Eng Pract*, vol. 141, p. 105720, Dec. 2023.

[26] A. Parra, D. Tavernini, P. Gruber, A. Sorniotti, A. Zubizarreta, and J. Perez, "On nonlinear model predictive control for energy-efficient torque-vectoring," *IEEE Trans Veh Technol*, vol. 70, no. 1, pp. 173–188, Jan. 2021.

[27] E. Siampis, E. Velenis, S. Gariuolo, and S. Longo, "A real-time nonlinear model predictive control strategy for stabilization of an electric vehicle at the limits of handling," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 6, pp. 1982–1994, Nov. 2018.

[28] J. P. Allamaa, P. Listov, H. Van Der Auweraer, C. Jones, and T. D. Son, "Real-time Nonlinear MPC Strategy with Full Vehicle Validation for Autonomous Driving," in *2022 American Control Conference (ACC), IEEE*, pp. 1982–1987, Jun. 2022.

[29] J. Qu, H. Li, Z. Zhang, X. Xi, R. Zhang, and K. Guo, "Performance analysis and optimization for steering motion mode switching of an agricultural four-wheel-steering mobile robot," *Agronomy*, vol. 12, no. 11, p. 2655, Oct. 2022.

[30] Y. Chang, Z. Yang, M. Hu, Y. Bian, and Y. Li, "An optimal parking path planning method integrating motion mode decision-making for 4WIS vehicles," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 2570–2577, 2024.

[31] Y. Cho, K. Noh, E. Lee, and H. Jeong, "Introduction to a motion control algorithm for the autonomous 4-wheel steering vehicle," *2025 IEEE/SICE International Symposium on System Integration*, pp. 1090–1094, 2025.

[32] X. Zhang, Y. Huang, S. Wang, W. Meng, G. Li, and Y. Xie, "Motion planning and tracking control of a four-wheel independently driven steered mobile robot with multiple maneuvering modes," *Frontiers of Mechanical Engineering,* vol. 16, no. 3, pp. 504–527, Apr. 2021.

[33] D. Yin, J. Wang, J. Du, G. Chen, and J.-S. Hu, "A new torque distribution control for four-wheel independent-drive electric vehicles," *Actuators*, vol. 10, no. 6, p. 122, Jun. 2021.

[34] H. Ling and B. Huang, "Research on torque distribution of four-wheel independent drive off-road vehicle based on PRLS road slope estimation," *Math Probl Eng*, vol. 2021, pp. 1–11, Sep. 2021.

[35] R. Achdad, A. Rabhi, and J. Bosche, "Energy-efficient torque distribution strategy for four wheel drive electric vehicles based on Traffic zone," *IFAC-PapersOnLine*, vol. 58, no. 10, pp. 1–6, 2024.

[36] H. Chen, S. Chen, R. Zhou, X. Huang, and S. Zhu, "Research on four-wheel independent steering intelligent control strategy based on minimum load," *Concurr Comput*, vol. 33, no. 9, May 2021.

[37] G. Wang and Q. Song, "The control of handling stability for four-wheel steering distributed drive electric vehicles based on a phase plane analysis," *Machines*, vol. 12, no. 7, p. 478, Jul. 2024.

[38] Y. Zhang, J. Ni, H. Tian, W. Wu, and J. Hu, "Integrated robust dynamics control of all-wheel-independently-actuated unmanned ground vehicle in diagonal steering," *Mech Syst Signal Process*, vol. 164, Feb. 2022.

[39] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Math Program Comput*, vol. 11, no. 1, pp. 1–36, Mar. 2019.

[40] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math Program*, vol. 106, no. 1, pp. 25–57, May 2006.

[41] Y. J. Kanayama and B. I. Hartman, "Smooth local-path planning for autonomous vehicles1," *The International Journal of Robotics Research*, vol. 16, no. 3, pp. 263–284, Jun. 1997.

[42] J. Ziegler and C. Stiller, "Fast collision checking for intelligent vehicle motion planning," in *2010 IEEE Intelligent Vehicles Symposium, IEEE*, pp. 518–522, Jun. 2010.

[43] M. Mittal *et al.*, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robot Autom Lett*, vol. 8, no. 6, pp. 3740–3747, Jun. 2023.

[44] V. Makoviychuk *et al.*, "Isaac gym: High performance GPU-based physics simulation for robot learning," *ArXiv preprint*, Aug. 2021.