Integrated Graph Search and Model Predictive Control for Smooth and Efficient Path Planning in Autonomous Vehicles

Duc-Tien Bui[1], Ngoc Thinh Nguyen[2], Hung Duy Nguyen[3], Dong Bi[1], Tomislav Mihalj[1] and Arno Eichberger[1]

***Abstract*— Path planning is a fundamental component of autonomous vehicles, where achieving safe, comfortable, and dynamically feasible paths while ensuring computational efficiency remains a significant challenge. This paper presents a sequential path planning framework in which a rough path obtained from graph search is explicitly exploited to guide a Model Predictive Control (MPC)-based path refinement. A rough path is first obtained via Dijkstra search on a discretized grid and is then used to construct a spatially varying convex lateral safety corridor that explicitly captures obstacle avoidance constraints, transforming discrete obstacle avoidance decisions into continuous feasibility constraints for optimization. Within this corridor, an MPC problem is formulated to refine the path, enabling efficient optimization while maintaining path smoothness by penalizing the third-order spatial derivative of the lateral offset over a prediction horizon. The proposed algorithm is evaluated in multiple overtaking scenarios on both straight and curved roads, including cases with single and multiple target vehicles, using high-fidelity environment simulations (i.e., CarMaker). Compared with the previous study, which used polynomial fitting and a quadratic programming method, the proposed approach consistently achieves lower lateral acceleration, curvature, and jerk while reducing computational cost by 28.08% on straight roads and 29.52% on curved roads. These results demonstrate that exploiting graph-search structure within an MPC formulation provides an effective balance between path smoothness and computational efficiency for autonomous vehicles in structured driving environments.**

I. INTRODUCTION

The development of automated vehicles (AVs) has witnessed significant progress in recent years. As a core component of AVs, path planning is responsible for generating safe and dynamically feasible paths that guide the vehicle through complex environments while meeting requirements for passenger comfort and computational efficiency [1], [2].

*This work was supported by ASEAN-UNINET Project

[1]Duc-Tien Bui is with the Institute of Automotive Engineering, Graz University of Technology, 8010 Graz, Austria d.t.bui@tugraz.at

[2]Ngoc Thinh Nguyen is with Drone Engineering, University of Applied Science Kufstein Tirol, Austria ngoc-thinh.nguyen@fh-kufstein.ac.at

[3]Hung Duy Nguyen is with Automation and Control Institute (ACIN), Vienna University of Technology, Austria nguyen@acin.tuwien.ac.at

[1]Dong Bi is with the Institute of Automotive Engineering, Graz University of Technology, 8010 Graz, Austria dong.bi@tugraz.at

[1]Tomislav Mihalj is with the Institute of Automotive Engineering, Graz University of Technology, 8010 Graz, Austria tomislav.mihalj@tugraz.at

[1]Arno Eichberger is with the Institute of Automotive Engineering, Graz University of Technology, 8010 Graz, Austria arno.eichberger@tugraz.at

Path planning can be divided into global and local levels [2], [3], where global path planning creates a route from the starting point to the ending point and local path planning generates a concrete segment satisfying constraints, adapting to the changing dynamic environment [4], [5]. Both layers work cooperatively: the global planner provides macroscopic guidance, whereas the local planner ensures safety, feasibility, and smoothness in execution. Despite extensive research, key challenges in path planning remain in achieving smooth path profiles, computation cost and maintaining robustness across diverse driving scenarios [3].

To address these challenges, traditional algorithms such as Dijkstra's, A*, Artificial Potential Field (APF) and their variants have laid the theoretical foundation for early path planning [6], [7], [8], [9], [10]. Dijkstra's algorithm or A* [9] are employed to find the shortest path and the APF method [10] simulates attractive and repulsive forces to guide a car toward its goal while avoiding obstacles [11]. Although these methods are computationally efficient and easy to implement, they often suffer from local minima, limited adaptability, and difficulty handling dynamic obstacles [1], [2], [11].

To overcome the limitations of those algorithms, optimization-based methods are developed. These methods formulate the path or trajectory as a constrained linear programming problem, directly minimizing cost functions related to smoothness, safety, and comfort. Representative techniques include Sequential Quadratic Programming [12], Convex Feasible Set [13], Constrained Iterative LQR [14] and MPC [15], [16], which iteratively solve convex subproblems for faster convergence. While capable of producing smooth, feasible trajectories, these methods are sensitive to initial conditions and may converge to local optima, particularly in highly dynamic environments [4], [17].

Additionally, Sampling-Based Planning (SBP) is represented by Rapidly-exploring Random Tree (RRT) and its variants find the path between the start and endpoints with random tree-like branches [11], [18]. The paths are guaranteed to be found if given enough run-time. However, SBP approaches have several limitations: produce jerky paths, lack explicit geometric optimization, and require post-processing to ensure smoothness and dynamic feasibility [2], [11], [19].

In addition, Dynamic programming (DP) provides an efficient approach to addressing the shortest path planning problem. By breaking down the overall task into smaller, more manageable subproblems, it constructs a globally optimal path through the combination of their optimal solutions

[20]. In real-world implementations, Baidu Apollo employs DP within the EM and lattice planning frameworks, both of which are well-established in prior research [21]. The authors in [5], [3], [22] proposed a DP-based technique that discretizes the continuous state space combine with some constraints to achieve globally optimal results and improve path smoothness. Li et al. [23] employed DP in combination with QP for path planning and analyzed the impact of key parameters on trajectory planning in critical scenarios. However, when the resolution increases, computational costs of DP grow exponentially, making execution time challenging.

These above articles indicate that the development of a path planning capable of the optimality, real-time feasibility remains a critical and ongoing research challenge.

The main contributions of this paper are:

Graph guided convex corridor construction: A graph search guided convex lateral corridor construction method is proposed, where a rough collision-free path is explicitly used to define a spatially varying feasible region for optimization.

MPC-based path refinement: Within the constructed convex corridor, the path refinement problem is formulated as an MPC problem that penalizes the third-order spatial derivative of the lateral offset under feasibility constraints, thereby promoting smooth path profiles with reduced computational complexity.

The rest of this study is structured as follows: Section II presents the methodology, Section III describes the simulation setup and results while Section IV gives the conclusion.

II. Methodology

Algorithm 1 Integrated Graph Search - MPC Path Planning

Require: Map $\mathscr{M}$, obstacle set $\mathscr{O}$, initial state x_0
Ensure: Optimized trajectory $\{x_k\}_{k=0}^{N_p}$
1: Discretize $\mathscr{M}$ into grid with adaptive resolution (d_s, d_l)
2: Mark occupied cells based on obstacle set $\mathscr{O}$
3: Find rough path $\mathscr{P} = \{(s_i, l_i)\}$ using Dijkstra search
4: Construct convex lateral safety corridor $\mathscr{C}$
5: Initialize MPC state $x \leftarrow x_0$
6: **for** $k = 0$ to $N_p - 1$ **do**
7: Formulate and solve MPC problem
8: Apply first control input u_k
9: Update state x_{k+1}
10: **end for**
11: **return** optimized trajectory $\{x_k\}$

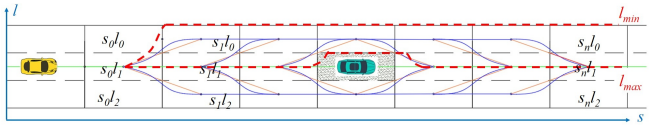

Fig. 1. Path planning generation: from the global path (green), the map is discretized into cells and occupied cells are marked. A rough reference path (orange) is searched and subsequently smoothed by MPC (blue), satisfying the boundary lines (red dot line).

Algorithm 1 illustrates the path planning flow in four steps. First, the map is discretized into cells, with target vehicles marked as occupied. Next, Dijkstra's algorithm [9] is used to find the shortest path through the unoccupied cells. Based on this path, a safe corridor is built by expanding the surrounding convex space to form a feasible region. Finally, MPC computes the path by optimizing vehicle motion under smoothness and acceleration constraints.

A. Grid map and Graph search

In this study, we use the Frenet coordinate system [3], [5], [23] to represent vehicle motion relative to the road. The considered map is discretized into cells, the cells occupied by obstacles are marked as unsafe, as illustrated in Fig 1.

1) Rough Path Generation: A Dijkstra-based search is performed on a discretized grid to compute a safe path (s,l) for the ego vehicle, where s and l denote the longitudinal and lateral coordinates in meters, respectively. The algorithm employs an adaptive resolution scheme for the grid construction, for complex maneuvers such as overtaking, the map is discretized with a high-resolution grid as:

$$\begin{aligned} s(i) &= s_0 + i \cdot ds \\ l(j) &= (n_{l,\text{left}} + 1 - j) \cdot dl \end{aligned} \tag{1}$$

where ds and dl represent the fine longitudinal and lateral grid resolutions, $n_{l,\text{left}}$ is the number of lateral cells to the left of the reference line, while i and j denote the respective grid indices. Conversely, the resolution is adaptively reduced in simpler driving states, such as, in the cruise control mode $l(i) = l_0 + i \cdot dl$ with $dl = 0$ to save the computational cost.

2) Obstacle Boundaries: For each static obstacle, the longitudinal and lateral extents are computed as:

$$s_{\min} = s_{obs} - \frac{obs_L}{2} \qquad l_{\min} = l_{obs} - \frac{obs_W}{2} \tag{2}$$

$$s_{\max} = s_{obs} + \frac{obs_L}{2} \qquad l_{\max} = l_{obs} + \frac{obs_W}{2} \tag{3}$$

where s_{obs}, l_{obs} are the obstacle center, obs_L, obs_W are its length and width, which are detected by the sensors.

3) Occupancy of Grid Cells: Each cell (i,j) in the grid map is marked as occupied if it intersects with an obstacle:

$$\begin{cases} (s_i + \frac{\Delta s}{2} > s_{min}) \wedge (s_i - \frac{\Delta s}{2} < s_{max}) \\ (l_j + \frac{\Delta l}{2} > l_{min}) \wedge (l_j - \frac{\Delta l}{2} < l_{max}) \end{cases} \tag{4}$$

4) Transition Cost Function: Defining (p_i, p_j) are the target grid cells that the vehicle intends to move to from the current cell i, j. The cost for a transition $(i,j) \to (p_i, p_j)$ is:

$$C(i, j \to p_i, p_j) = C_b(j, p_j) + C_p(p_i, p_j) \tag{5}$$

The path is designed such that the ego vehicle prioritises overtaking on the left. Therefore, the basic lane-change cost and the lateral penalty are defined as:

$$C_b(j, p_j) = \begin{cases} 1, & p_j = j \quad \text{(go straight)} \\ 1.1, & p_j < j \quad \text{(move left)} \\ 1.5, & p_j > j \quad \text{(move right)} \end{cases} \tag{6}$$

$$C_p(i,j) = \begin{cases} 10^5, & l(i,j) < l_0 - 0.1 \quad \text{(right overtake)} \\ 0.5, & l(i,j) > l_0 + 0.1 \quad \text{(left overtake)} \\ 0.1, & \text{otherwise (go straight)} \end{cases} \tag{7}$$

The cumulative distance is updated as:

$$d(p_i,p_j)=\min\left[d(p_i,p_j),d(i,j)+C(i,j\rightarrow p_i,p_j)\right] \quad (8)$$

with $d(i,j)$ represents the accumulated minimum cost to reach the cell (p_i,p_j).

The shortest path is obtained by backtracking from:

$$j^*=\arg\min_j \text{dist}(n_s,j) \quad (9)$$

with n_s is the index that indicates the longitudinal position of the path endpoint. The set of all path segments that satisfy the equation (9), the rough path is constructed as:

$$(s,l)=\left(s(i_1,j_1),\ldots,s(i_k,j_k);l(i_1,j_1),\ldots,l(i_k,j_k)\right) \quad (10)$$

B. Determination of the Lateral Safety Corridor

Given the path (s,l) from equation (10) and a set of obstacles positions, the lateral boundaries of the convex space are $l_{\min}(j),\quad l_{\max}(j),\quad j=1,\ldots,n$ in Figure 1.

For obstacle i, the longitudinal influence region is:

$$s_{\min}^{(i)}=s_{obs}(i)-\frac{obs_L}{2} \quad (11)$$

$$s_{\max}^{(i)}=s_{obs}(i)+\frac{obs_L}{2} \quad (12)$$

Mapping to indices corresponding to $s_{\min}^{(i)}$ and $s_{\max}^{(i)}$ as:

$$idx_s^{(i)}=\arg\min_j |s_j-s_{\min}^{(i)}| \quad (13)$$

$$idx_e^{(i)}=\arg\min_j |s_j-s_{\max}^{(i)}| \quad (14)$$

The avoidance side is determined by:

$$\bar{l}_{\text{path}}^{(i)}=\frac{l(idx_s^{(i)})+l(idx_e^{(i)})}{2} \quad (15)$$

with $\bar{l}_{\text{path}}^{(i)}$ specifies the lateral center of the safety corridor that the path should follow to avoid collisions. Update rules for the left and right avoidance:

$$l_{\min}(j)=\max\left(l_{\min}(j),l_{obs}^{(i)}+\frac{obs_W}{2}\right) \quad \text{if} \quad \bar{l}_{path}^{(i)}\geq l_{obs}^{(i)} \quad (16)$$

$$l_{\max}(j)=\min\left(l_{\max}(j),l_{obs}^{(i)}-\frac{obs_W}{2}\right) \quad \text{if} \quad \bar{l}_{path}^{(i)}< l_{obs}^{(i)} \quad (17)$$

for $j\in[idx_s^{(i)},idx_e^{(i)}]$. Now, the task is finding the optimal path (s,l) with $l\in[l_{min},l_{max}]$ such that the path satisfies constraints regarding safety and smoothness.

C. MPC-Based Optimal Path Generation

The previous studies [23], [24], [25], [26] built the QP problem to find the optimal path. Unlike those, in this paper, the path planning problem is formulated as an MPC problem.

Let $f(s_k)=l_k$, $f'(s_k)=\dot{l}_k$, $f''(s_k)=\ddot{l}_k$, $u_k=\dddot{l}_k$ represent the derivatives of f with respect to the longitudinal coordinate s. The path can be expanded at s_k up to the third order using Taylor expansion as follows [5]:

$$\begin{aligned} l_{k+1}&=l_k+\dot{l}_k\Delta s+\tfrac{1}{2}\ddot{l}_k\Delta s^2+\tfrac{1}{6}u_k\Delta s^3\\ \dot{l}_{k+1}&=\dot{l}_k+\ddot{l}_k\Delta s+\tfrac{1}{2}u_k\Delta s^2\\ \ddot{l}_{k+1}&=\ddot{l}_k+u_k\Delta s \end{aligned} \quad (18)$$

Here, l_k is the lateral offset, $\dot{l}_k$ and $\ddot{l}_k$ are its first and second derivatives with respect to longitudinal distance s, and u_k is the third-order spatial derivative of the lateral offset (control input). The lateral motion of the ego vehicle along the reference path is discretized with sampling step Δs.

Defining $N_p=20$ as the prediction horizon, for each prediction step $k=0,\ldots,N_p$, the state and input vectors are:

$$x_k=[l_k\ \dot{l}_k\ \ddot{l}_k]^\top,\qquad u_k=\dddot{l}_k \quad (19)$$

The dynamics in equation (18) can be written as the discrete-time state-space model:

$$x_{k+1}=Ax_k+Bu_k \quad (20)$$

with

$$A=\begin{bmatrix}1&\Delta s&\frac{1}{2}\Delta s^2\\0&1&\Delta s\\0&0&1\end{bmatrix},\qquad B=\begin{bmatrix}\frac{1}{6}\Delta s^3\\\frac{1}{2}\Delta s^2\\\Delta s\end{bmatrix} \quad (21)$$

The MPC problem is formulated as:

$$\begin{aligned}\min_{\substack{x_1,\ldots,x_{N_p},\\u_0,\ldots,u_{N_p-1}}}&\sum_{i=0}^{N_p-1}\left[(x_i-x_{\text{ref},i})^\top Q(x_i-x_{\text{ref},i})+u_i^\top Ru_i\right]\\&+(x_{N_p}-x_{\text{ref},N_p})^\top P(x_{N_p}-x_{\text{ref},N_p})\end{aligned} \quad (22)$$

subject to:

$$x_{i+1}=Ax_i+Bu_i,\quad \forall i=0,\ldots,N_p-1 \quad (23)$$

$$u_{\min}\leq u_i\leq u_{\max},\quad \forall i=0,\ldots,N_p-1 \quad (24)$$

$$A_{\text{sub}}x_i\leq b_{\text{sub}}(i),\quad \forall i=1,\ldots,N_p \quad (25)$$

$$\Delta_{\min}\leq G(x_{i+1}-x_i)\leq\Delta_{\max},\quad \forall i=0,\ldots,N_p-1 \quad (26)$$

$$x_0=x_{\text{start}} \quad (27)$$

In this formulation, the matrices A,B in equation (23) are presented in equation (21). The parameters ($u_{\min}=-0.05\ \text{m}^{-2}, u_{\max}=0.05\ \text{m}^{-2}$) in equation (24) denote the lower and upper bounds of the control input. The matrices $A_{\text{sub}},b_{\text{sub}}(i)$ in equation (25) represent the convex corridor constraint, which will be detailed hereinafter. The constraint in equation (26) imposes bounds on the variation of the first- and second-order spatial derivatives of the lateral offset between consecutive prediction steps to ensure smooth path transitions:

$$G=\begin{bmatrix}0&1&0\\0&0&1\end{bmatrix},\ \Delta_{\max}=\begin{bmatrix}\Delta_{\dot{l}}^{\max}\\\Delta_{\ddot{l}}^{\max}\end{bmatrix},\ \Delta_{\min}=-\Delta_{\max} \quad (28)$$

with ($\Delta_{\dot{l}}^{\max}$ = 0.2, $\Delta_{\ddot{l}}^{\max}$ =3.5 m^{-1}) denoting the maximum allowed variations of the first- and second-order spatial derivatives, respectively. Finally, $x_{\text{start}}\in\mathbb{R}^{3\times1}$ in equation (27) gives the initial condition of the ego car. The reference state $x_{\text{ref},i}\in\mathbb{R}^{3\times1}$ for time step i in the cost function in equation (22) is given by:

$$x_{\text{ref},i}=\begin{bmatrix}\frac{l_{\min}+l_{\max}}{2}&0&0\end{bmatrix}^\top,\quad \forall i=0,\ldots,N_p-1 \quad (29)$$

$$x_{\text{ref},N_p}=x^* \quad (30)$$

with $x^*=[l_{end},\dot{l}_{end},\ddot{l}_{end}]^T$ the desired terminal state. The weighting matrices are given by:

$$Q=\text{diag}\{w_l,w_{\dot{l}},w_{\ddot{l}}\},\quad P=w_{\text{end}}I_3,\quad R=w_u \quad (31)$$

where the weight factors w_l, $w_{\dot{l}}$, $w_{\ddot{l}}$, w_{end}, and w_u are selected empirically. Specifically, higher weights are assigned to the first- and second-order spatial derivatives $w_{\dot{l}} = 1500$ and $w_{\ddot{l}} = 200$, while lower weights for the lateral position $w_l = 3$, $w_{\text{end}} = 15$, control input $w_u = 20$ to refine the path and regularize the control effort.

Convex corridor constraints: For each step $i = 1, \ldots, N_p$, the ego vehicle must remain inside the convex safe corridor:

$$A_{\text{sub}} x_i \leq b_{\text{sub}}(i) \tag{32}$$

with:

$$A_{\text{sub}} = \begin{bmatrix} 1 & d_1 & 0 \\ 1 & d_1 & 0 \\ 1 & -d_2 & 0 \\ 1 & -d_2 & 0 \\ -1 & -d_1 & 0 \\ -1 & -d_1 & 0 \\ -1 & d_2 & 0 \\ -1 & d_2 & 0 \end{bmatrix}, \; b_{\text{sub}}(i) = \begin{bmatrix} l_{\max}(f_i) - \frac{w}{2} \\ l_{\max}(f_i) + \frac{w}{2} \\ l_{\max}(b_i) - \frac{w}{2} \\ l_{\max}(b_i) + \frac{w}{2} \\ -l_{\min}(f_i) + \frac{w}{2} \\ -l_{\min}(f_i) - \frac{w}{2} \\ -l_{\min}(b_i) + \frac{w}{2} \\ -l_{\min}(b_i) - \frac{w}{2} \end{bmatrix} \tag{33}$$

where f_i and b_i are the range of prediction forward and backward influence indices:

$$f_i = \min\big(i + \lceil d_1/\Delta s \rceil, N_p\big), \quad b_i = \max\big(i - \lceil d_2/\Delta s \rceil, 1\big).$$

By using the solver "mpcActiveSetSolver" in MATLAB to solve this problem under the constraints, a convex, smooth and dynamically feasible path is obtained.

III. Simulation and Results

In this study, the proposed algorithm is implemented using MATLAB/Simulink R2021b [27] for control development, IPG CarMaker 11.1.1 [28] and the BMW 5-Series vehicle model with calibrated parameters from [29] for high-fidelity vehicle dynamics simulation. All simulations and experiments are conducted on a workstation running Microsoft Windows 11 Pro. The hardware platform is an HP ZBook Fury 15.6" G8 Mobile Workstation, equipped with an 11th Gen Intel® Core™ i7-11800H processor (2.30 GHz, 8 cores, 16 threads) and 32 GB of DDR4 RAM.

To evaluate the performance of the path planning algorithm, we constructed three simulation scenarios: (1) the AV overtakes a single target car (TG) on a straight road; (2) the AV overtakes a single TG on a curved road; and (3) the AV drives on a straight road with multiple target cars. For comparisons, we employ the algorithm used in a previous study [23], which uses a discretization approach with sampled points combined with polynomial fitting and the QP method. The comparison aspects are lateral acceleration, jerk, curvature, steering angle and computational cost in scenarios 1 and 2. The video simulation results can be found at **link** [30].

A. Scenario 1: Straight road – single target vehicle

The AV drives at a speed of 110km/h and overtakes a TG moving at 90km/h on a straight road. The simulation duration for this scenario is set at 75s. The lateral acceleration, steering angle, curvature and jerk of the ego car are shown in Figure 2, 3 and Table I.

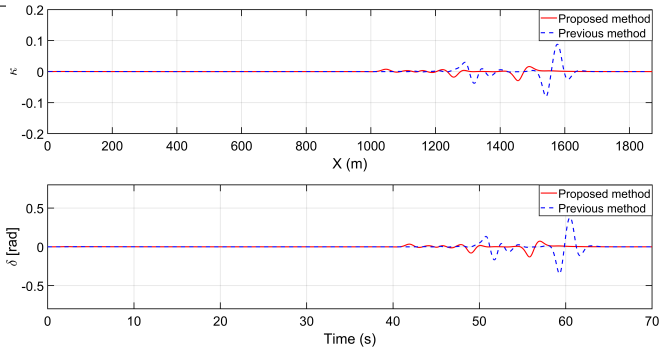

Fig. 2. Curvature and steering angle on the straight road with one TG

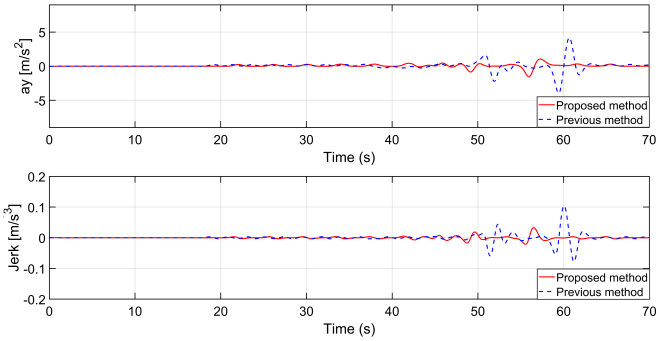

Fig. 3. Lateral acceleration and jerk on the straight road with one TG

B. Scenario 2: Curved road – single target vehicle

The AV moves at 100km/h on a curved road with a radius of R = 500m, overtaking a target car driving at 80km/h. The simulation setting time is 45s. The obtained results related to the lateral acceleration, steering angle, curvature, jerk and the computation cost when the ego car overtakes the TG on the curved road are presented in the Figure 4, 5 and Table I. As illustrated in Table I, for the scenarios 1 and 2, where the ego vehicle overtakes a single TG, the proposed algorithm exhibits smaller values of lateral acceleration, jerk, curvature and steering angle compared to the previous study [23]. Specifically, the maximum values of these parameters for the proposed algorithm are 1.028m/s^2, 0.034m/s^3, 0.016m^{-1} and 0.072rad, respectively, whereas the corresponding values in [23] are 4.140m/s^2, 0.103m/s^3, 0.087m^{-1} and 0.378rad. Similarly, the minimum values of lateral acceleration, jerk, curvature and steering angle in the proposed algorithm are -3.491m/s^2, -0.046m/s^3, -0.080m^{-1} and -0.349rad, compared to -4.322m/s^2, -0.077m/s^3, -0.096m^{-1} and -0.407rad in [23]. These results indicate the improved smoothness performance of the proposed algorithm.

The computational advantage of the proposed method is further underscored by the reduction in the average computation time per step, which decreased from 0.0021s in the previous study to 0.0017s. This reduction time plays a pivotal role in maintaining vehicle stability and minimizing algorithmic latency between sensing and actuation during high-speed maneuvers. The observed enhancement in efficiency arises from explicitly utilizing the Dijkstra-

generated rough path to construct a spatially varying convex safety corridor, whereby the complex problem of obstacle avoidance is transformed into a series of continuous feasibility constraints. This structural simplification allows the mpcActiveSetSolver to converge more rapidly compared to the polynomial fitting and traditional QP approaches used in [23]. Furthermore, the adoption of a prediction horizon ($N_p = 20$) instead of solving for the entire path simultaneously as a QP problem, significantly reduces the dimensionality of the optimization vector, thereby lowering the cumulative computational overhead while ensuring sufficient look-ahead capability for smooth path refinement. Consequently, the proposed method requires only 34.847s and 18.213s for the path planning module alone to execute scenarios 1 and 2, respectively. In contrast, the method reported in [23] requires 48.452s and 25.843s for the path planning module for the same scenarios. Therefore, the proposed algorithm not only achieves a 28.08% to 29.52% reduction in total execution time in scenario 1 and scenario 2 but also supports real-time applicability under the tested simulation conditions.

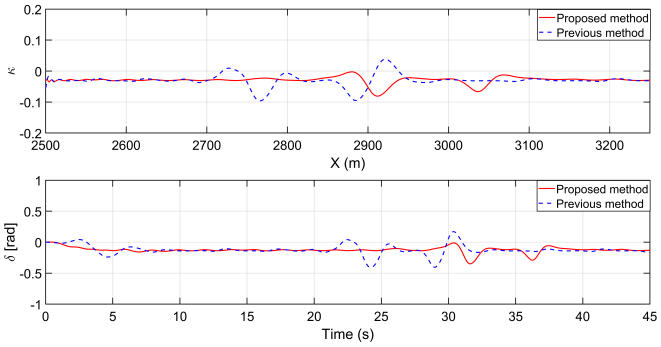

Fig. 4. Curvature and steering angle on the curved road with one TG

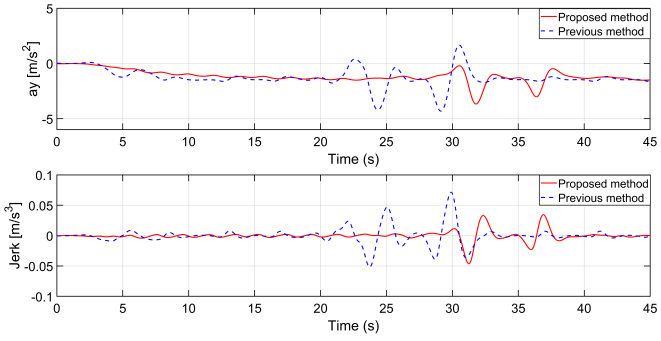

Fig. 5. Lateral acceleration and jerk on the curved road with one TG

C. Scenario 3: Straight road – multiple target vehicles

The AV drives at 70km/h on a straight road with multiple target vehicles, see Figure 6. Initially, two vehicles are present: TG1 (red) in the middle lane and TG2 (yellow) in the left lane, both moving at 50km/h. Since the safety distance between TG1 and TG2 is sufficient, the AV successfully overtakes TG1. TG2 then performs a deceleration and lane-change manoeuvre into the middle lane. Subsequently, the AV encounters TG3 (white) in the middle lane and TG4 (blue) in the right lane, both driving at 50km/h. While overtaking TG3, TG4 suddenly accelerates, cuts into TG3's lane, and continues in the same lane. The AV then re-plans path and overtakes both TG3 and TG4 safely.

TABLE I
COMPARISON BETWEEN PROPOSED ALGORITHM AND PREVIOUS STUDY

Features	Scenario	Algorithm	Max	Min	Unit
Lateral acceleration	**1**	Proposed Algorithm	1.028	-1.545	m/s^2
		Previous study	4.140	-3.966	m/s^2
	2	Proposed Algorithm	-0.210	-3.491	m/s^2
		Previous study	1.666	-4.322	m/s^2
Jerk	**1**	Proposed Algorithm	0.033	-0.021	m/s^3
		Previous study	0.103	-0.077	m/s^3
	2	Proposed Algorithm	0.034	-0.046	m/s^3
		Previous study	0.070	-0.051	m/s^3
Curvature	**1**	Proposed Algorithm	0.016	-0.029	m^{-1}
		Previous study	0.087	-0.080	m^{-1}
	2	Proposed Algorithm	-0.003	-0.080	m^{-1}
		Previous study	0.038	-0.096	m^{-1}
Steering angle	**1**	Proposed Algorithm	0.072	-0.130	*rad*
		Previous study	0.378	-0.346	*rad*
	2	Proposed Algorithm	-0.014	-0.349	*rad*
		Previous study	0.17	-0.407	*rad*
Simulation time setup in scenario 1			75		*s*
Simulation time setup in scenario 2			45		*s*
Average computation time per step		Proposed Algorithm	0.0017		*s*
		Previous study	0.0021		*s*
Execution time of path planning in scenario 1		Proposed Algorithm	34.847		*s*
		Previous study	48.452		*s*
Execution time of path planning in scenario 2		Proposed Algorithm	18.213		*s*
		Previous study	25.843		*s*

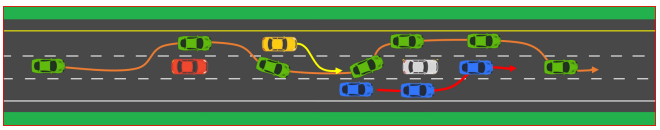

Fig. 6. Scenario 3: Straight road – multiple target vehicles

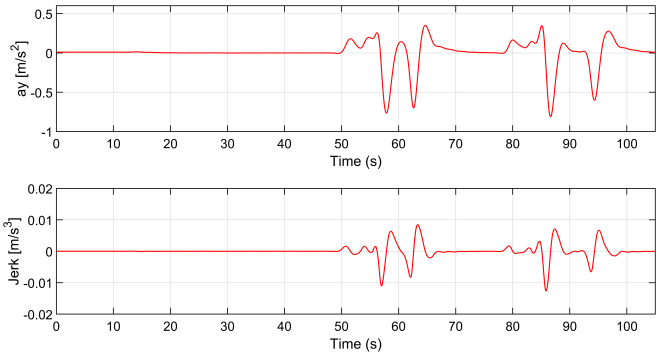

Fig. 7. Lateral acceleration and jerk on the straight road with multiple target cars

Figure 7 and 8 show the lateral acceleration, steering angle, curvature and jerk when the ego car drives on the straight road with multiple target cars. Specifically, the lateral acceleration is bounded between -0.812 and 0.341m/s^2, and the resulting steering angle is kept between -0.128 and 0.06rad. Similarly, the curvature spans from -0.028 to 0.013m^{-1} and the jerk is small, ranging from -0.013 to

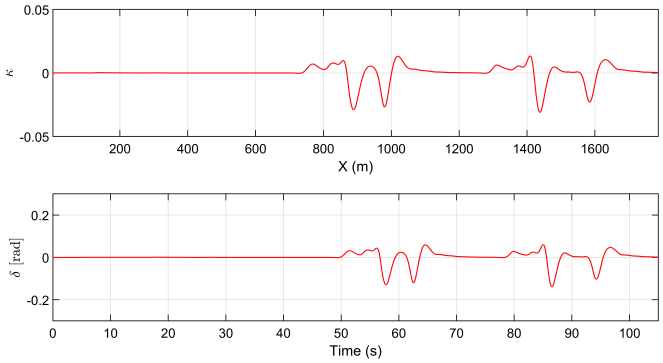

Fig. 8. Curvature and steering angle on the straight road with multiple target cars

$0.008\mathrm{m/s}^3$. These values confirm that the proposed algorithm can consistently output a smooth and feasible path that ensures driving comfort and stability.

IV. Conclusion

This paper introduced a new path-planning algorithm that integrates graph search, the Dijkstra algorithm and MPC. The proposed method discretizes the obstacle avoidance space according to varying driving environments, formulates constraints, and constructs a convex feasible region to represent vehicle–obstacle interactions. Within this region, an objective function is established and solved based on MPC algorithm to derive the optimal avoidance trajectory. This formulation enables efficient and reliable obstacle avoidance across a wide range of scenarios on both straight and curved roads. Simulation results demonstrate that the proposed method adapts effectively to diverse driving conditions, generating smooth and dynamically feasible paths while achieving a lower computational cost compared to existing approaches.

Future work will focus on further optimizing and testing the proposed algorithm under more complex traffic scenarios.

References

[1] S. Abdallaoui, E.-H. Aglzim, A. Chaibet, and A. Kribèche, "Thorough review analysis of safe control of autonomous vehicles: path planning and navigation techniques," *Energies*, vol. 15, no. 4, p. 1358, 2022.

[2] M. Reda, A. Onsy, A. Y. Haikal, and A. Ghanbari, "Path planning algorithms in the autonomous driving system: A comprehensive review," *Robotics and Autonomous Systems*, vol. 174, p. 104630, 2024.

[3] Y. Meng, Y. Wu, Q. Gu, and L. Liu, "A decoupled trajectory planning framework based on the integration of lattice searching and convex optimization," *IEEE Access*, vol. 7, pp. 130530–130551, 2019.

[4] L. Liu, X. Wang, X. Yang, H. Liu, J. Li, and P. Wang, "Path planning techniques for mobile robots: Review and prospect," *Expert Systems with Applications*, vol. 227, p. 120254, 2023.

[5] C. Zhang and W. Xu, "Intelligent vehicle path based on discretized sampling points and improved cost function: A quadratic programming approach," *IEEE Access*, vol. 12, pp. 24500–24515, 2024.

[6] H. Hongyu, Z. Chi, S. Yuhuan, Z. Bin, and G. Fei, "An improved artificial potential field model considering vehicle velocity for autonomous driving," *IFAC-PapersOnLine*, vol. 51, no. 31, pp. 863–867, 2018.

[7] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[8] W. Yijing, L. Zhengxuan, Z. Zhiqiang, and L. Zheng, "Local path planning of autonomous vehicles based on a* algorithm with equal-step sampling," in *2018 37th Chinese Control Conference (CCC)*, pp. 7828–7833, IEEE, 2018.

[9] D. Rachmawati and L. Gustin, "Analysis of dijkstra's algorithm and a* algorithm in shortest path problem," in *Journal of Physics: Conference Series*, vol. 1566, p. 012061, IOP Publishing, 2020.

[10] H. D. Nguyen, M. Choi, and K. Han, "Risk-informed decision-making and control strategies for autonomous vehicles in emergency situations," *Accident Analysis & Prevention*, vol. 193, p. 107305, 2023.

[11] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on intelligent transportation systems*, vol. 17, no. 4, pp. 1135–1145, 2015.

[12] W. Lim, S. Lee, M. Sunwoo, and K. Jo, "Hierarchical trajectory planning of an autonomous car based on the integration of a sampling and an optimization method," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 613–626, 2018.

[13] C. Liu, C.-Y. Lin, Y. Wang, and M. Tomizuka, "Convex feasible set algorithm for constrained trajectory smoothing," in *2017 American Control Conference (ACC)*, pp. 4177–4182, IEEE, 2017.

[14] J. Chen, W. Zhan, and M. Tomizuka, "Constrained iterative lqr for on-road autonomous driving motion planning," in *2017 IEEE 20th International conference on intelligent transportation systems (ITSC)*, pp. 1–7, IEEE, 2017.

[15] Z. Li, Y. Wang, Z. Zuo, R. Zhao, C. Hu, and Y. Shi, "Model predictive control-based trajectory optimization for autonomous vehicles using risk-aware corridors," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–16, 2026.

[16] M. Nezami, N. T. Nguyen, G. Männel, R. Kensbock, H. S. Abbas, and G. Schildbach, "Safe control architecture via model predictive control," *IEEE Transactions on Control Systems Technology*, 2024.

[17] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 416–442, 2015.

[18] Y. Zhang, H. Wang, M. Yin, J. Wang, and C. Hua, "Bi-am-rrt*: A fast and efficient sampling-based motion planning algorithm in dynamic environments," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 1282–1293, 2023.

[19] C. Zhou, B. Huang, and P. Fränti, "A review of motion planning algorithms for intelligent robots," *Journal of Intelligent Manufacturing*, vol. 33, no. 2, pp. 387–424, 2022.

[20] C. Wu, S. Zhou, and L. Xiao, "Dynamic path planning based on improved ant colony algorithm in traffic congestion," *IEEE Access*, vol. 8, pp. 180773–180783, 2020.

[21] H. Fan, F. Zhu, C. Liu, L. Zhang, L. Zhuang, D. Li, W. Zhu, J. Hu, H. Li, and Q. Kong, "Baidu apollo em motion planner," *arXiv preprint arXiv:1807.08048*, 2018.

[22] J. Ren and X. Huang, "Dynamic programming inspired global optimal path planning for mobile robots," in *2021 IEEE 4th International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pp. 461–465, 2021.

[23] H. Li, F. De Cristofaro, F. Orucevic, Z. Gu, and A. Eichberger, "Quantitative analysis of the impact of baidu apollo parameterization on trajectory planning in a critical scenario," *Transportation Research Procedia*, vol. 73, pp. 102–109, 2023.

[24] W. Lim, S. Lee, M. Sunwoo, and K. Jo, "Hierarchical trajectory planning of an autonomous car based on the integration of a sampling and an optimization method," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 613–626, 2018.

[25] W. Xu, J. Wei, J. M. Dolan, H. Zhao, and H. Zha, "A real-time motion planner with trajectory optimization for autonomous vehicles," in *2012 IEEE International Conference on Robotics and Automation*, pp. 2061–2067, 2012.

[26] C. Liu, C.-Y. Lin, Y. Wang, and M. Tomizuka, "Convex feasible set algorithm for constrained trajectory smoothing," in *2017 American Control Conference (ACC)*, pp. 4177–4182, 2017.

[27] The MathWorks, Inc., *MATLAB and Simulink Release R2021b*. The MathWorks, Inc., Natick, Massachusetts, United States, 2021. Available: https://www.mathworks.com/.

[28] "https://ipg-automotive.com/," tech. rep.

[29] D.-T. Bui, H. Li, F. De Cristofaro, and A. Eichberger, "Lateral control calibration and testing in a co-simulation framework for automated vehicles," *Applied Sciences*, vol. 13, no. 23, p. 12898, 2023.

[30] D.-T. Bui, "Integrated graph search and model predictive control for smooth and efficient path planning in autonomous vehicles," 2026. Available: https://www.youtube.com/watch?v=0SeYLDn2E6E, Accessed: Apr. 8, 2026.