Transportation Research Part B 173 (2023) 228–246

Contents lists available at ScienceDirect

Transportation Research Part B

journal homepage: www.elsevier.com/locate/trb

Trajectory Planning for an Autonomous Vehicle with Conflicting Moving Objects Along a Fixed Path – An Exact Solution Method

Xiaowei Shi [a], Xiaopeng Li [b,*]

[a] Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI, 48109, USA
[b] Department of Civil and Environmental Engineering, University of Wisconsin-Madison, Madison, WI, 53706, USA

ARTICLE INFO

Keywords:
Autonomous vehicle
Trajectory planning
Conflict area analysis
Dynamic programming

ABSTRACT:

Trajectory planning for autonomous vehicles (AVs) by considering conflicting moving objects (CMOs) is a challenging problem to AV operations. This paper investigates an AV trajectory planning problem where the AV follows a given spatial path, and the trajectories of CMOs are predictable. With the spatial path fixed, the two-dimensional trajectory planning problem is reduced to a one-dimensional speed planning problem that decides the optimal speeds and accelerations of the AV along the spatial path. This paper first analyzes the conflict area caused by a single CMO in the space-time diagram, which reveals upper and lower bounds to the conflict area. Then a multi-area fusion algorithm is proposed to extend the upper and lower bound analyses to a relatively complex traffic scenario with multiple CMOs. To facilitate the computation of the investigated problem, a customized dynamic programming (DP) algorithm is developed, which employs the revealed upper and lower bounds and arriving time constraints to cut invalid trajectories at each stage. With this, the number of stages and states, as well as the computational time for solving the proposed problem, are largely reduced. A set of numerical experiments are conducted to evaluate the performance of the customized DP-based algorithm. The results show that the proposed customized DP-based algorithm can solve the investigated problem within milliseconds, which enables applications to real-time AV control. This much outperforms a state-of-the-art commercial solver, Gurobi, especially for complex traffic scenarios. We further implemented the proposed method in a real-world case study, and the results show that the trajectory generated by the proposed model has a great potential to enhance the future traffic system.

1. Introduction

Autonomous vehicles (AVs) hold a great potential to improve traffic efficiency, safety, comfort, and energy optimization via optimal motion planning (Dresner and Stone, 2004; González et al., 2016; Li et al., 2018; Makridis et al., 2018; Rios-Torres and Malikopoulos, 2017; Shi et al., 2022; Shi and Li, 2021a). One outstanding challenge in AV motion planning is how to control an AV under complex traffic scenarios with multiple conflicting moving objects (CMOs), e.g., when the AV passes an intersection or drives on a multilane road (Hu and Sun, 2019).

To overcome this challenge, a number of studies attempted to optimize AV trajectories, e.g., selecting the best option over a large

* Corresponding author.
E-mail addresses: tomcee@umich.edu (X. Shi), xli2485@wisc.edu (X. Li).

https://doi.org/10.1016/j.trb.2023.05.001
Received 18 February 2022; Received in revised form 3 February 2023; Accepted 2 May 2023
Available online 15 May 2023

set of collision-free spatial paths in a two-dimensional (2D) space (Chen et al., 2019; Fulgenzi et al., 2009; Ji et al., 2017; Malikopoulos et al., 2018; Stachniss and Burgard, 2003; Ziegler et al., 2014a). Due to the large number of spatial path options, the computational load for obtaining the optimal trajectory is often intensive. To circumvent the excessive computational load, efforts have been made to improve solution efficiency, e.g., creating highly sophisticated algorithms (Katrakazas et al., 2015; Paden et al., 2016; Schmerling et al., 2015) or proposing heuristic algorithms while sacrificing the solution quality (Wu et al., 2021). For example, Karaman and Frazzoli (2011) solved the robot motion planning problem by proposing an algorithm that combines probabilistic roadmaps and rapidly explores random trees (RRT). Kasać et al. (2011) studied a conjugate gradient-based numerical algorithm for optimal control of nonlinear systems with control and state vector constraints. To jointly optimize the serving sequence and route choice for a large-scale autonomous intersection system in a timely manner, Wu et al. (2021) proposed a heuristic algorithm to tackle the computational load issue.

An alternative approach to reduce the computation complexity is limiting the spatial path candidate to a few candidates or fixing it to one reasonable candidate. A series of studies have investigated the AV trajectory planning problem over a fixed spatial path (Bobrow et al., 1985; Constantinescu and Croft, 2000; Dubowsky et al., 1986; Kunz and Stilman, 2013; Lipp and Boyd, 2014; Zhang et al., 2018; Ziegler et al., 2014b). In this case, since the AV follows a fixed spatial path, the trajectory planning problem reduces to determine the optimal speed (or acceleration) profile along the fixed path. For example, Lipp and Boyd (2014) studied the problem of optimizing the speed of a vehicle along a fixed path for the minimum travel time by considering multiple vehicle models (e.g., multi-wheel car model, aircraft model) and situations (e.g., vehicles with aerodynamic drag, banked turn). Zhang et al. (2018) investigated a spatially fixed trajectory planning problem that solves a smooth, safety-guaranteed, and time-efficient speed profile.

With a fixed spatial path, the location of the AV on a 2D spatial path can be reparametrized into one variable that represents the travel distance from the origin of the AV along the path (travel distance for short). Then the 2D trajectory planning problem along a fixed path is simplified into a one-dimensional (1D) problem representation, also known as a speed planning problem. Ziegler et al. (2014b) suggested this idea that the 2D problem can be converted into a 1D problem representation but did not describe the details for incorporating CMOs. To the best of the authors' knowledge, Zhan et al. (2017) and Liu et al. (2017b) are the only two studies incorporating CMOs into the speed planning problem. Zhan et al. (2017) identified the boundaries of the CMOs along the path and treated these boundaries as constraints to the speed planning problem. By proposing a method that mixes A-star search and quadratic programming, they solved the speed planning problem with CMOs under different scenarios. Liu et al. (2017b) studied the speed planning problem with CMOs by adopting a temporal optimization method that optimizes the time stamps along the path instead of the locations. Similarly, the boundaries of the CMOs served as the constraint to the speed planning problem. Then they solved the temporal optimization by approximating the problem into a sequence of quadratic programs. Despite these two successes on the speed planning problem with CMOs, both of them assume that the "passing priorities" of the AV and CMOs are given, which greatly restricts the mobility of the AV. We use a toy example to illustrate the difference between the problem with and without this assumption.

Consider a regular intersection with southbound and westbound traffic as shown in Fig. 1. A southbound AV aims to pass the intersection within 10 seconds. While a westbound CMO will pass through the intersection and thus occupy the intersection from the 4th through 6th seconds. To avoid collisions, without considering the passing priorities, the AV can pass the intersection either before the 4th second or after the 6th second. It means that the AV can avoid collisions by entering the intersection either before or after the CMO. However, if we specify the passing priorities of the AV and CMOs, say the AV can only pass the intersection after the 6th second, although the problem complexity can be reduced, the mobility and energy efficiency of the AV, as well as the CMO, may be suboptimal. Further, this assumption is not compatible with dynamic gap acceptance decisions made by a vehicle crossing or merging into conflict traffic only after observing the CMOs rather than being prescribed. Apart from the restrictive assumption, these two studies illustrate

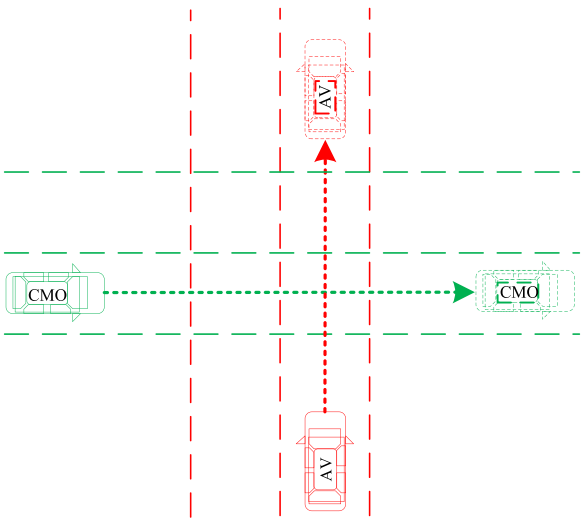

Fig. 1. Toy example for "passing priority" illustration.

the performance of the proposed methods with some fairly simple instances while the performance of the proposed methods on complex traffic scenarios remains unclear. Moreover, the AV trajectory planning problem is a time-sensitive problem that requires the planned trajectory must be generated within a relatively short time period, especially for complex traffic scenarios. Otherwise, the AV will be out of control and accidents may happen. The computation time reported in the two studies might still be not fast enough for real-time AV control in certain scenarios that require responses in less than 0.1 second (Lampariello et al., 2011).

Overall, we note that few studies investigate the AV trajectory planning problem over a fixed spatial path (i.e., speed planning problem) considering CMOs without a prescribed vehicle priority. This paper closes this research gap and makes the following contributions to the literature:

(1) This paper proposes a parsimonious and efficient model for investigating AV trajectory planning problem over a fixed spatial path considering CMOs. This fixed path setting converts the 2D trajectory planning problem to a 1D speed planning problem and thus significantly reduces the complexity of the studied problem.
(2) This paper innovatively proposes to explore the studied problem from a space-time diagram perspective. We explicitly study the potential conflict areas between the subject AV and its surrounding CMOs on a space-time diagram, which has never been studied by previous researchers. The investigations of the potential conflict areas reveal many fundamental properties of the studied problem (i.e., upper and lower bounds for feasible trajectories) and thus significantly tighten the regions of feasible trajectories for the AV.
(3) By utilizing the properties obtained from the potential conflict areas analysis, a customized dynamic-programming-(DP)-based algorithm is proposed. In the algorithm, the potential conflict areas and the upper and lower bounds help eliminate infeasible trajectories at each stage of the DP algorithm. With these customized elimination rules, the "curse of dimensionality" problem of the original DP algorithm is circumvented, and the optimal AV trajectory planning problem can be solved in milliseconds, which enables the possibility for real-world application.

The rest of this paper is organized as follows. Section 2 presents the proposed AV trajectory planning problem and the optimization model. The structure of this problem is analyzed in Section 3 to discover the properties of the potential conflicts between the subject AV and surrounding CMOs on a space-time diagram. This analysis leads to a customized dynamic-programming-(DP)-based algorithm presented in Section 4, which efficiently solves the optimal AV speed planning problem in milliseconds. Section 5 conducts a series of numerical experiments to illustrate the applicability and efficiency of the proposed model and algorithm. Section 6 concludes the paper and identifies directions for future research.

Table 1
Notation of this paper.

Name	Description
$\mathscr{S}$	Studied 2D infrastructure.
$[0, T]$	Studied time horizon.
$\mathscr{T}$	Set of studied time interval, $\mathscr{T} := \{1, 2, \cdots, \mathbb{T}\}$.
θ	Length of time intervals. $T = \theta \cdot \mathbb{T}$.
$\mathbb{A}$	Studied AV.
$\mathscr{N}^{\mathrm{o}}$	Set of CMOs in the infrastructure. $\mathscr{N}^{\mathrm{o}} := \{1, 2, \cdots, N\}$.
$\mathscr{N}$	Set of all objects in the infrastructure. $\mathscr{N} := \{\mathbb{A}\} \cup \mathscr{N}^{\mathrm{o}}$.
$\mathscr{P}$	AV path.
L	Length of the AV path.
$\mathscr{R}_n$	Trajectory of CMO . $n \in \mathscr{N}^{\mathrm{o}}$.
$\mathscr{X}_{t'l't''l''}$	Set of all feasible trajectories with respect to time-space windows $[(t', l'), (t'', l''), \forall t' < t'' \in \mathscr{T}, l' < l'' \in [0, L]]$.
${}^{\mathrm{e}}\mathscr{C}_n$	Conflict area of CMO n on the space-time diagram. $n \in \mathscr{N}^{\mathrm{o}}$.
${}^{\mathrm{m}}\mathscr{C}_n$	Conflict area of CMO n on the space-time diagram after extension. $n \in \mathscr{N}^{\mathrm{o}}$.
$p_n^+(t), p_n^-(t)$	Conflict area of CMO n on the space-time diagram after merging. $\gamma \in \{1, \cdots, N'\}$.
$p_{n,t}^+, p_{n,t}^-$	Upper and lower bounds of CMO n in continuous time horizon.
${}^{\mathrm{e}}p_{n,t}^+, {}^{\mathrm{e}}p_{n,t}^-$	Upper and lower bounds of CMO n in discrete time horizon.
${}^{\mathrm{m}}p_{n,t}^+, {}^{\mathrm{m}}p_{n,t}^-$	New upper and lower bounds of CMO n after conflict area extension.
${}^{\mathrm{e}}\mathscr{C}_n$	New upper and lower bounds of CMO n after merging operation.
$x(t), \dot{x}(t), \ddot{x}(t)$	Location, speed, and acceleration of the AV in continuous time horizon.
x_t, v_t, a_t	Location, speed, and acceleration of the AV in discrete time horizon.
$\overline{v}, \underline{a}, \overline{a}$	Maximum speed, minimum acceleration, and maximum acceleration of the AV.
v_0, a_0	Initial speed and initial acceleration of the AV.
w	Weight of the travel distance of the AV.
b_f, b_r	Front and rear buffer distance of CMOs.
a_t^{p}	Acceleration of the AV at Stage $t-1$.
Θ	Set of time interval length.
s_t, S_t	State and state space of the proposed algorithm at Stage t.

2. Problem Statement

2.1. Notation

For readers' convenience, the notation of this paper is summarized in Table 1.

2.2. Model formulation

Consider a 2D space $\mathscr{S}$ as the infrastructure where an AV and a set of CMOs are present as shown in Fig. 2a. The AV is indexed by $\mathbb{A}$, and the CMOs are indexed by $\mathscr{N}^{\circ} := \{1,2,\cdots,N\}$. Let $\mathscr{N} := \{\mathbb{A}\} \cup \mathscr{N}^{\circ}$ denote the set of all objects in the infrastructure. For example, in Fig. 2a, $\mathscr{N} = \{\mathbb{A}\} \cup \{1,2\}$.

Assume that the spatial path of the AV is fixed on path $\mathscr{P}$ (which could be a curve) in space $\mathscr{S}$. We consider a coordinate system that starts at the current location of the AV as location 0 and increases along Path $\mathscr{P}$ till the end of $\mathscr{P}$ marked as location L, i.e., the length of $\mathscr{P}$ is L. Let 0 denote the current time point and T denote a sufficiently long time for the AV to complete the path. Then the decisions are to determine the location $x(t')$, speed $\dot{x}(t')$, and acceleration $\ddot{x}(t'), \forall t' \in [0,T]$ of the AV along the path, where the dot and double dot operators the first- and second-order derivatives, respectively, e.g., $\dot{x}(t') = dx(t')/d_{t'}$ and $\ddot{x}(t') = d\dot{x}(t')/d_{t'}$. With this, the following trajectory planning problem is investigated over space-time diagram $[0, L] \times [0, T]$ as shown in Fig. 2b.

The environmental CMOs may conflict with the AV path. We assume there is an exogenous robust trajectory prediction algorithm to predict the possible trajectory of each environmental CMO n, $\forall n \in \mathscr{N}^{\circ}$ during time horizon $[0, T]$, which we denote as $\mathscr{R}_n \subset \mathscr{S} \times [0,T]$, $\forall n \in \mathscr{N}^{\circ}$. The CMOs trajectory prediction is another research topic in the literature. Interested readers can refer to Leon and Gavrilescu (2021) for a detailed review. Let $\mathscr{C}_n \subset [0,L] \times [0,T], \forall n \in \mathscr{N}^{\circ}$ denote the possible space-time area where the AV and $\mathscr{R}_n$ conflict (considering safety buffers of CMO n as detailed in Section 3). We specify $\mathscr{C}_n$ with two bounds $p_n^+(t')$ and $p_n^-(t')$ over its time range $[t_n^{'-}, t_n^{'+}] \subset [0,T]$ such that the conflict area of CMO n is enclosed within these two bounds as shown in the figure. Note that $p_n^+(t') \geq p_n^-(t')$, $\forall t' \in [t_n^{'-}, t_n^{'+}]$. Without loss of generality, we consider $p_n^+(t')$ and $p_n^-(t')$ as piece-wise quadratic functions.

The AV initial state is given as initial location $x(0) = 0$, initial speed $\dot{x}(0) = v$, and initial acceleration $\ddot{x}(0) = a$. Let $\underline{a} < 0$ and $\bar{a} > 0$ respectively denote the minimum and maximum accelerations allowed for the AV, and let $\bar{v}$ denote its speed limit. By the end of the time horizon (i.e., time T), the AV needs to pass location L. The dot-dash lines in Fig. 2b illustrate two feasible trajectories of the AV (i.e., $x(t')$).

For the convenience of the trajectory modeling, the continuous study time $[0, T]$ is discretized into a set of identical time intervals, indexed by $\mathscr{T} := \{0,1,2,\cdots,\mathbb{T}\}$, and the length of each time interval is $\theta := T/\mathbb{T}$, where time index $t \in \mathscr{T}$ maps to continuous time $t\theta$. With this, we redefine the time range of CMO n as $\{t_n^-, t_n^- + 1, \cdots, t_n^+ - 1, t_n^+\}$, where $t_n^- = \lfloor t_n^{'-}/\theta \rfloor \in \mathscr{T}$, $t_n^+ = \lceil t_n^{'+}/\theta \rceil \in \mathscr{T}$, and $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ are rounding down and rounding up operations, respectively. Then the two bounds of conflict area n are denoted by $\{t, p_{n,t}^+\}_{t \in \{t_n^-, t_n^- + 1, \cdots, t_n^+ - 1, t_n^+\}}$ and $\{t, p_{n,t}^-\}_{t \in \{t_n^-, t_n^- + 1, \cdots, t_n^+\}}$, where $p_{n,t}^+ \geq p_{n,t}^-, \forall t \in \{t_n^-, t_n^- + 1, \cdots, t_n^+ - 1, t_n^+\}$. The conflict area of CMO n (i.e., $\mathscr{C}_n$) can be generated regarding the two bounds, i.e., $\mathscr{C}_n := \{\{t, p_{n,t}^+\}_{t \in \{t_n^-, t_n^- + 1, \cdots, t_n^+\}}, \{t, p_{n,t}^-\}_{t \in \{t_n^-, t_n^- + 1, \cdots, t_n^+\}}\}$, as illustrated in Fig. 3.

With this, the motion characteristics for the AV at each time interval t, $\forall t \in \mathscr{T}$, including location x_t, speed v_t, and acceleration a_t, are planned subjecting to the following constraints (1)-(9). Note that constraints (9) are a set of logical constraints to help the AV avoid the conflict areas in the space-time diagram. That is, the AV avoids the conflict areas by passing the location either before the conflict

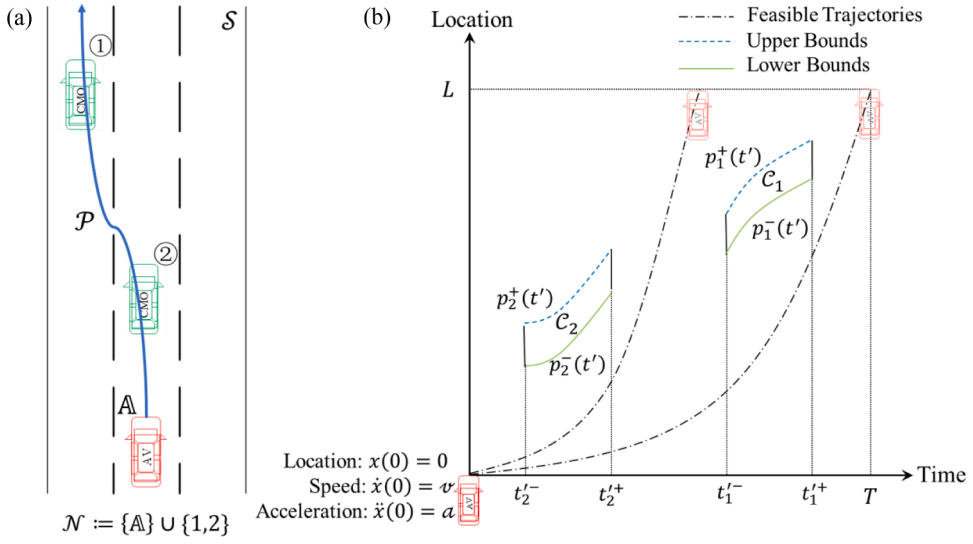

Fig. 2. Problem statement.

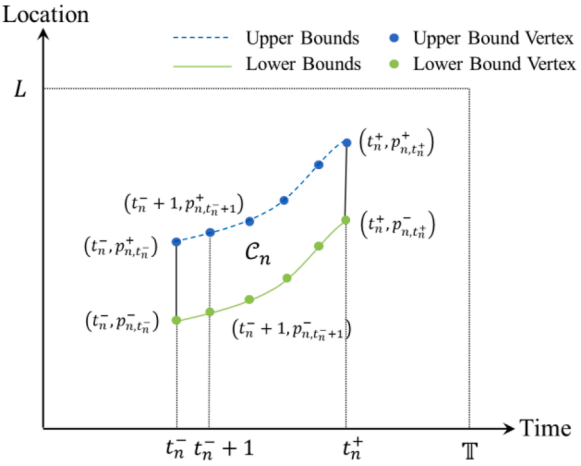

Fig. 3. Conflict area illustration.

areas appear or after they disappear.

$$0 \le v_t \le \overline{v}, \forall t \in \mathscr{T}, \tag{1}$$

$$\underline{a} \le a_t \le \overline{a}, \forall t \in \mathscr{T}, \tag{2}$$

$$x_0 = 0, \tag{3}$$

$$v_0 = \nu, \tag{4}$$

$$a_0 = \alpha, \tag{5}$$

$$v_t = (x_t - x_{t-1})/\theta, \ \forall t \in \mathscr{T} \backslash \{0\}, \tag{6}$$

$$a_t = (v_t - v_{t-1})/\theta, \ \forall t \in \mathscr{T} \backslash \{0, 1\}, \tag{7}$$

$$x_{\mathbb{T}} \ge L, \tag{8}$$

$$x_t \le p_{n,t}^- \text{ or } x_t \ge p_{n,t}^+, \forall t \in \left\{t_n^-, \cdots, t_n^+\right\}, n \in \mathscr{N}^{\circ}. \tag{9}$$

In general, for trajectory planning problems, the objective function is defined as the summation of certain polynomial functions of the location, velocity, acceleration and jerk of a vehicle along time (Li and Li, 2019). In this paper, to take both the driver experience and travel time of the AV into consideration, we minimize the summation of the acceleration variance square and the weighted travel distance in the objective function as shown in Eq. (10).

$$obj := \min_{x_t, v_t, a_t} \sum_{t \in \mathscr{T} \backslash \{0\}} \left[(a_t - a_{t-1})^2 - w x_t\right], \tag{10}$$

where w is the weight of the travel distance. The weighted travel distance is used for evaluating the travel time of the AV. The performance of the travel distance with different weights will be tested in the numerical experiments.

Note that due to the logical constraints (Eq. (9)) and the quadratic operation in the objective function (Eq. (10)), the proposed model is nonlinear in both objective function and constraints, which leads the model hard to be solved (Pei et al., 2022). As we mentioned earlier, the AV trajectory planning problem is a time-sensitive problem that requires the planned trajectory must be generated within a relatively short time period. Thus, there is a great need to study an efficient algorithm to solve the proposed model. In addition, we would like to point out that the logical constraints can be linearized into two set of "big M" constraints. The value of the big M can be set to $v_t \mathbb{T} + 0.5\overline{a}\mathbb{T}^2$, which is the maximum distance that the vehicle can travel within the given time horizon.

3. Conflict area identifications and analyses

3.1. Conflict area identifications

To generate a collision-free trajectory for the AV on space-time diagram $[0, L] \times [0, T]$, firstly the possible conflict areas caused by the CMOs on the space-time diagram (i.e., $\mathscr{C}_n$) need to be identified (Levin and Rey, 2017). Note that a CMO may have two types of

conflict with the AV. In the first type, the CMO path intersects with the AV path (e.g., the intersection scenario as shown in Fig. 4a). In the second type, the CMO path overlaps with the AV path (e.g., the car following scenario as shown in Fig. 4b).

Considering the trajectory following buffers for a CMO, the conflict area caused by the CMO on the space-time diagram usually is larger than the actual collision area. We denote the front and rear buffer distance of a CMO as b_f and b_r, respectively. For example, for the scenario shown in Fig. 4a, assuming CMO $n \in \mathcal{N}^{\circ}$ occupies location $[p^{-}_{n,t^{-}_{n}} + b_r, p^{+}_{n,t^{-}_{n}} - b_f]$ during time $t \in \{t^{-}_{n}, \cdots, t^{+}_{n}\}$, the conflict area caused by the CMO on the space-time diagram is illustrated in Fig. 5a. For the scenario shown in Fig. 4b, assuming the CMO moves from location $p^{+}_{n,t^{-}_{n}} - b_f$ to location $p^{+}_{n,\mathbb{T}} - b_f$ during time $t \in \{t^{-}_{n}, \cdots, \mathbb{T}\}$. Then the conflict area caused by the CMO on the space-time diagram is illustrated in Fig. 5b. In this paper, the conflict areas represent the infeasible areas for AV trajectories. In Fig. 5a and Fig. 5b, the conflict areas are the summation of the buffer areas and collision area, and the upper and lower boundaries of the conflict areas serve as the upper and lower bounds of the CMO (i.e., $p^{+}_{n,t}$ and $p^{-}_{n,t}$).

3.2. Conflict area analyses

To avoid the CMOs, the trajectory of the AV cannot intersect with the conflict areas in the space-time diagram. By taking advantages of this property, this subsection aims to further analyze the identified conflict areas by considering the kinematics constraints of the AV (Constraints (1) - (9)).

AV with a single CMO

We first analyze the conflict area on the space-time diagram if there is only one single CMO in the scenario (i.e., $\mathcal{N}^{\circ} := \{1\}$). To facilitate the math notation, we denote the set of all feasible trajectories with respect to time-space windows $[(t', l'), (t'', l'')], \forall t' < t'' \in \mathcal{T}, l' < l'' \in [0, L]$ by $\mathcal{X}_{t'l't''l''} := \{x^{t'l't''l''} := \{x^{t'l't''t''}_{t} \in [l', l'']\}_{t \in \{t', t'+1, \cdots, t''\}}$, s.t., Eqs.(1) – (9)$\}$, where (t', l') and (t'', l'') are two arbitrary time-space points in the diagram. By solving Eqs. (1)–(9) with (t', l') and (t'', l'') as input, the set of all feasible trajectories $\mathcal{X}_{t'l't''l''}$ can be obtained. For example, for the investigated problem, the two points should be (0, 0) and $(\mathbb{T}, L)$. The set of all feasible trajectories $\mathcal{X}_{00\mathbb{T}L}$ can be obtained by $\mathcal{X}_{00\mathbb{T}L} := \{x^{00\mathbb{T}L} := \{x^{00\mathbb{T}L}_{t} \in [0, L]\}_{t \in \mathcal{T}}$, s.t., Eqs. (1) – (9)$\}$. With this, two important terms for analyzing the conflict area on the space-time diagram are defined as follows.

Definition 1. We define *the slowest trajectory* with respect to time-space windows $[(t', l'), (t'', l'')], \forall t' < t'' \in \mathcal{T}, l' < l'' \in [0, L]$ by $\underline{x}^{t'l't''l''}$, $\underline{x}^{t'l't''l''} \in \mathcal{X}_{t'l't''l''}$, s.t., $\underline{x}^{t'l't''t''}_{t} \leq x^{t'l't''t''}_{t}$, $\forall x^{t'l't''l''} \in \mathcal{X}_{t'l't''l''}, t' \leq t \leq t'' \in \mathcal{T}, l' < l'' \in [0, L]$.

The movement process of the slowest trajectory $\underline{x}^{t'l't''l''}$ can be described as follows. The AV first decelerates at $\underline{a}$ as far as feasible (before coming to a full stop or having to accelerate, whichever comes first), and finally accelerates at $\overline{a}$ during the remaining time such that AV passes the destined location (i.e., l'') at the end of the time period (i.e., t'') within the speed range.

Definition 2. Similarly, we define *the fastest trajectory* with respect to time-space windows $[(t', l'), (t'', l'')], \forall t' < t'' \in \mathcal{T}, l' < l'' \in [0, L]$ by $\overline{x}^{t'l't''l''}$, $\overline{x}^{t'l't''l''} \in \mathcal{X}_{t'l't''l''}$, s.t., $\overline{x}^{t'l't''t''}_{t} \geq x^{t'l't''t''}_{t}$, $\forall x^{t'l't''l''} \in \mathcal{X}_{t'l't''l''}, t' \leq t \leq t'' \in \mathcal{T}, l' < l'' \in [0, L]$.

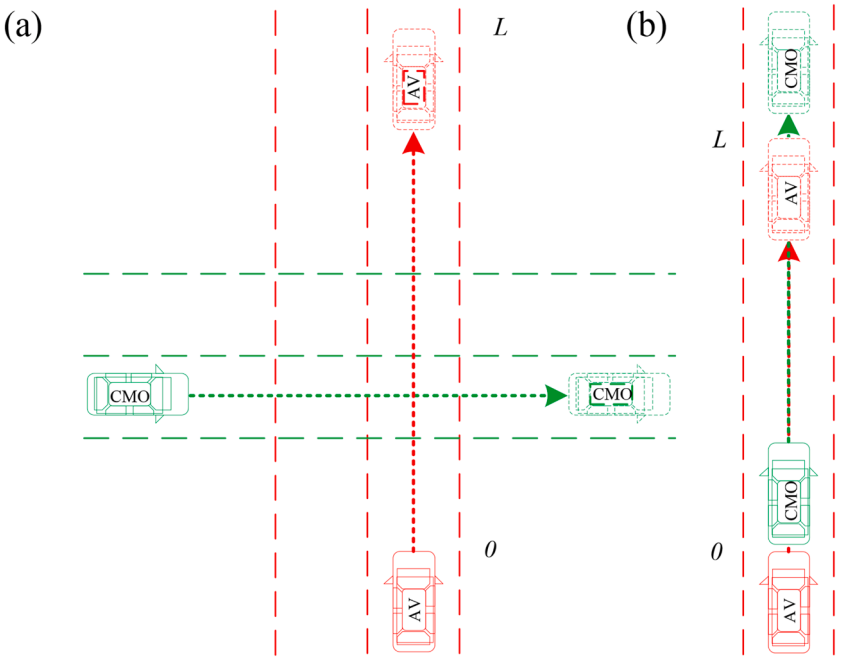

Fig. 4. Two types of conflict scenarios.

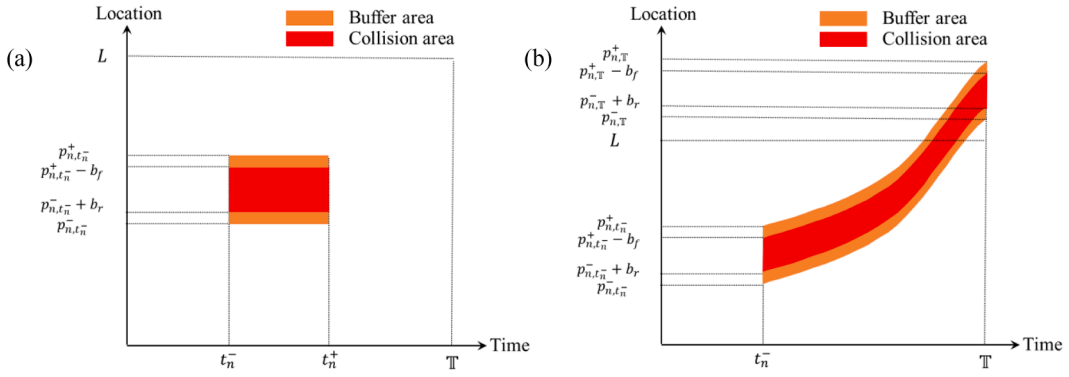

Fig. 5. Conflict area identifications.

The movement process of the fastest trajectory $\overline{x}^{t'l't''l''}$ can be described as follows. The AV accelerates at $\overline{a}$ as far as feasible (passing the destination before the speed reaches $\overline{v}$ or at speed $\overline{v}$, whichever comes first).

Then, we come to the conflict area analysis that aims to extend the conflict area regarding the kinematics of the AV. Assume that a generic CMO n occupies the path of the AV during time $\{t_n^-, \cdots, t_n^+\}$. The upper and lower bounds of the conflict area of the CMO in the time-space diagram are $\{t, p_{n,t}^+\}_{t\in\{t_n^-,\cdots,t_n^+\}}$ and $\{t, p_{n,t}^-\}_{t\in\{t_n^-,\cdots,t_n^+\}}$. To avoid CMO n in the time-space diagram, two reference envelopes, such as the upper and lower envelopes are defined as follows.

Definition 3. *The upper-envelope trajectory* for CMO n is defined as $\left\{\underline{x}^{00t_n^- p_{n,t_n^-}^+}, \{p_{n,t}^+\}_{t\in\{t_n^-,\cdots,t_n^+\}}, \underline{x}^{t_n^+ p_{n,t_n^+}^+ \mathbb{T}L}\right\}$.

The upper-envelope trajectory is composed of three trajectory segments, including $\underline{x}^{00t_n^- p_{n,t_n^-}^+}$, $\{p_{n,t}^+\}_{t\in\{t_n^-,\cdots,t_n^+\}}$, and $\underline{x}^{t_n^+ p_{n,t_n^+}^+ \mathbb{T}L}$. $\underline{x}^{00t_n^- p_{n,t_n^-}^+}$ is the trajectory barely passing the location at time t_n^- when the conflict area of the CMO appears, i.e., point $(t_n^-, p_{n,t_n^-}^+)$. It is the slowest trajectory regarding time windows $[(0,0),(t_n^-, p_{n,t_n^-}^+)]$. $\{p_{n,t}^+\}_{t\in\{t_n^-,\cdots,t_n^+\}}$ is the upper bound of the conflict area. $\underline{x}^{t_n^+ p_{n,t_n^+}^+ \mathbb{T}L}$ is the trajectory barely arriving the destination L at time $\mathbb{T}$. It is the slowest trajectory regarding time windows $[(t_n^+, p_{n,t_n^+}^+),(\mathbb{T},L)]$.

Definition 4. *The lower-envelope trajectory* for CMO n is defined as $\{\overline{x}^{00t_n^- p_{n,t_n^-}^-}, \{p_{n,t}^-\}_{t\in\{t_n^-,\cdots,t_n^+\}}, \overline{x}^{t_n^+ p_{n,t_n^+}^- t'_n L}\}$.

The lower-envelope trajectory is also composed of three trajectory segments, including $\overline{x}^{00t_n^- p_{n,t_n^-}^-}$, $\{p_{n,t}^-\}_{t\in\{t_n^-,\cdots,t_n^+\}}$, and $\overline{x}^{t_n^+ p_{n,t_n^+}^- t'_n L}$. $\overline{x}^{00t_n^- p_{n,t_n^-}^-}$ is the trajectory barely follows the location at time t_n^- when the conflict area of the CMO appears, i.e., point $(t_n^-, p_{n,t_n^-}^-)$. It is the fastest trajectory regarding time windows $[(0,0),(t_n^-, p_{n,t_n^-}^-)]$. $\{p_{n,t}^-\}_{t\in\{t_n^-,\cdots,t_n^+\}}$ is the lower bound of the conflict area. $\overline{x}^{t_n^+ p_{n,t_n^+}^- t'_n L}$ is the fastest trajectory regarding time windows $[(t_n^+, p_{n,t_n^+}^-),(t',L)]$, where $t'_n := \begin{cases} t_n^+ + \sqrt{2(L - p_{n,t_n^+}^-)/\overline{a}}, \text{if } L - p_{n,t_n^+}^- \le \overline{v}^2/2\overline{a}, \\ t_n^+ + \overline{v}/\overline{a} + (L - p_{n,t_n^+}^- - \overline{v}^2/2\overline{a})/\overline{v}, \text{otherwise}. \end{cases}$

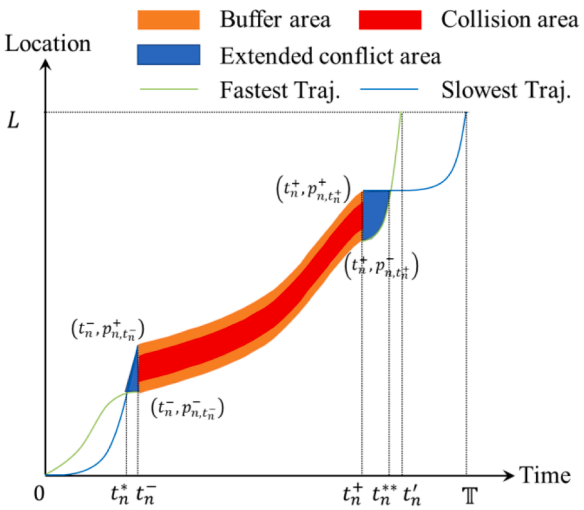

Fig. 6. Illustration of the extended conflict area for the AV and a single CMO.

To avoid collisions with CMO n, the AV trajectory shall be either faster than the upper-envelope trajectory or slower than the lower-envelope trajectory with respect to CMO n. Besides the original conflict area, we name the area enclosed by the two envelope trajectories and the original conflict area as the extended conflict area in this paper. As a result, any trajectories that enter the extended conflict area will also enter the original conflict area. Thus, the original conflict area can be enlarged by including the extended conflict area without loss of generality.

For illustrative purposes, one simple example is shown in Fig. 6. Assume that CMO n occupies the path of the AV, and the conflict area caused by the CMO in the space-time diagram is shown in Fig. 6. The AV stops at location 0 at time 0 and needs to pass location L by time $\mathbb{T}$. The upper-envelope reference trajectory is composed of trajectory segments $\{(0,0),(t_n^-,p_{n,t_n^-}^+)\}$, $\{(t_n^-,p_{n,t_n^-}^+),(t_n^+,p_{n,t_n^+}^+)\}$, and $\{(t_n^+,p_{n,t_n^+}^+),(\mathbb{T},L)\}$.

The lower-envelope is composed of trajectory segments $\{(0,0),(t_n^-,p_{n,t_n^-}^-)\}$, $\{(t_n^-,p_{n,t_n^-}^-),(t_n^+,p_{n,t_n^+}^-)\}$, and $\{(t_n^+,p_{n,t_n^+}^-),(t_n^{'},L)\}$. It can be observed in Fig. 6 that there is an area enclosed by the two envelope trajectories, which marks the extended conflict area as colored in blue. Any trajectory entering the extended conflict area will intersect with the original conflict area, and thus the original conflict area can be equivalently substituted by the extended conflict area. We denote the time coordinate of the intersection point of the two envelope trajectories with respect to CMO n as t_n^* and t_n^{**} as shown in Fig. 6.

Hence, the original conflict area caused by the CMO is enlarged by including the two extended conflict areas (i.e., the areas colored by blue in Fig. 6). The new upper bound for the conflict area n is denoted as $\{{}^{\mathrm{e}}p_{n,t}^+\}_{t\in\{{}^{\mathrm{e}}t_n^-,{}^{\mathrm{e}}t_n^-+1,\cdots,{}^{\mathrm{e}}t_n^+\}} := \{p_{n,t_n^*}^+,\cdots,p_{n,t_n^-}^+,\cdots,p_{n,t_n^+}^+,\cdots,p_{n,t_n^{**}}^+\}$, where ${}^{\mathrm{e}}t_n^- = t_n^*$, and ${}^{\mathrm{e}}t_n^+ = t_n^{**}$. Similarly, the new lower bound is denoted as $\{{}^{\mathrm{e}}p_{n,t}^-\}_{t\in\{{}^{\mathrm{e}}t_n^-,{}^{\mathrm{e}}t_n^-+1,\cdots,{}^{\mathrm{e}}t_n^+\}} := \{p_{n,t_n^*}^-,\cdots,p_{n,t_n^-}^-,\cdots,p_{n,t_n^+}^-,\cdots,p_{n,t_n^{**}}^-\}$. The new conflict area set is denoted as ${}^{\mathrm{e}}\mathscr{C} := \{{}^{\mathrm{e}}\mathscr{C}_n\}_{n\in\mathscr{N}^{\mathrm{o}}}$, where ${}^{\mathrm{e}}\mathscr{C}_n = \{\{{}^{\mathrm{e}}p_{n,t}^+\}_{t\in\{{}^{\mathrm{e}}t_n^-,{}^{\mathrm{e}}t_n^-+1,\cdots,{}^{\mathrm{e}}t_n^+\}},\{{}^{\mathrm{e}}p_{n,t}^-\}_{t\in\{{}^{\mathrm{e}}t_n^-,{}^{\mathrm{e}}t_n^-+1,\cdots,{}^{\mathrm{e}}t_n^+\}}\}$. For notation convenience, we assume ${}^{\mathrm{e}}p_{n,t}^+ = 0, {}^{\mathrm{e}}p_{n,t}^- = L, \forall t \in \mathscr{T} \setminus \{{}^{\mathrm{e}}t_n^-,{}^{\mathrm{e}}t_n^- + 1,\cdots,{}^{\mathrm{e}}t_n^+\}, n \in \mathscr{N}^{\mathrm{o}}$.

AV with multiple CMOs

The previous subsection describes the method for extending the conflict area of a single CMO on the space-time diagram. In practice, the operational scenario for the AV is usually stochastic and dynamic. That is, there may be multiple CMOs conflicting with the AV path and thus the conflict area on the space-time diagram may not be unique. Due to this, this subsection aims to analyze conflict areas on the space-time diagram for the AV with multiple CMOs in the scenario (i.e., $\mathscr{N}^{\mathrm{o}} := \{1,2,\cdots,N\},\ N \neq 1$).

Since we consider both buffer area and conflict area extension in the previous sections, the conflict areas on the time-space diagram may overlap with each other. The conflict areas of multiple CMOs are not the simple combinations of those of all single CMO. Instead, the overlapping conflict areas may form a larger irregular area in the space-time diagram. To study the conflict areas caused by multiple CMOs, by referring to the picture processing studies (Liu et al., 2017a; Peng et al., 2005; Rivero and Feito, 2000), this paper proposes a simple multi-area fusion (MAF) algorithm that merges all overlapping conflict areas in conflict area set ${}^{\mathrm{e}}\mathscr{C} := \{{}^{\mathrm{e}}\mathscr{C}_n\}_{n\in\mathscr{N}^{\mathrm{o}}}$.

Denote the conflict area set after processed by the MAF algorithm as ${}^{\mathrm{m}}\mathscr{C} := \{{}^{\mathrm{m}}\mathscr{C}_n\}_{n\in\{1,\cdots,N'\}}$, where ${}^{\mathrm{m}}\mathscr{C}_n$ includes the upper and lower bounds of the n-th new conflict area (i.e., ${}^{\mathrm{m}}p_{n,t}^+$ and ${}^{\mathrm{m}}p_{n,t}^-$), and the number of new conflict areas after merging is denoted by N'. ${}^{T}\mathscr{C}$ is a temporary set. The detailed MAF algorithm is described as follows.

Step 1: Input ${}^{\mathrm{e}}\mathscr{C}$; Initialize, ${}^{\mathrm{m}}\mathscr{C} \leftarrow \{\varnothing\}$, $n \leftarrow N$.

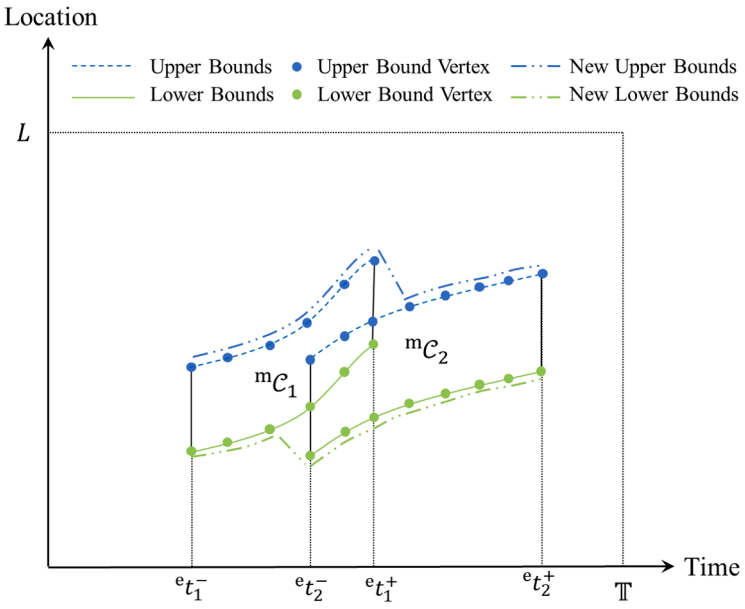

Fig. 7. Illustration of the MAF algorithm.

Step 2: If $n = 0$, output ${}^{\mathrm{m}}\mathscr{C}$; otherwise, check whether ${}^{\mathrm{e}}\mathscr{C}_n$ is intersected with the remaining conflict areas in ${}^{T}\mathscr{C}$ from back to forth (i.e., $\{n-1, \cdots, 2, 1\}$). If yes, we denote the index of the first conflict area that intersects with ${}^{\mathrm{e}}\mathscr{C}_n$ by n', and go to Step 3; otherwise, add ${}^{\mathrm{e}}\mathscr{C}_n$ into ${}^{\mathrm{m}}\mathscr{C}$ (i.e., ${}^{\mathrm{m}}\mathscr{C} = \{{}^{\mathrm{m}}\mathscr{C}, {}^{\mathrm{e}}\mathscr{C}_n\}$), $n = n-1$.

Step 3: The time range of the area formed by ${}^{\mathrm{e}}\mathscr{C}_n$ and ${}^{\mathrm{e}}\mathscr{C}_{n'}$ is $\{\min({}^{\mathrm{e}}t_n^-, {}^{\mathrm{e}}t_{n'}^-), \cdots, \max({}^{\mathrm{e}}t_n^+, {}^{\mathrm{e}}t_{n'}^+)\}$. The upper and lower bounds for the new formed area n' are as follows,

$$ {}^{\mathrm{e}}p_{n',t}^+ := \max\left\{{}^{\mathrm{e}}p_{n,t}^+, {}^{\mathrm{e}}p_{n',t}^+\right\},\ \forall t \in \left\{\min\left({}^{\mathrm{e}}t_n^-, {}^{\mathrm{e}}t_{n'}^-\right), \cdots, \max\left({}^{\mathrm{e}}t_n^+, {}^{\mathrm{e}}t_{n'}^+\right)\right\}, $$

$$ {}^{\mathrm{e}}p_{n',t}^- := \min\left\{{}^{\mathrm{e}}p_{n,t}^-, {}^{\mathrm{e}}p_{n',t}^-\right\},\ \forall t \in \left\{\min\left({}^{\mathrm{e}}t_n^-, {}^{\mathrm{e}}t_{n'}^-\right), \cdots, \max\left({}^{\mathrm{e}}t_n^+, {}^{\mathrm{e}}t_{n'}^+\right)\right\}. $$

Update ${}^{\mathrm{e}}\mathscr{C}_{n'}$ in ${}^{T}\mathscr{C}$. $n = n-1$, go to Step 2.

The methods for checking the intersection of two conflict areas are various, e.g., separating axis theorem (Boyd and Vandenberghe, 2004), point-in-polygon test (Polygon Collision, 2021). This paper adopts the hyperplane separation theorem to achieve this target (Boyd and Vandenberghe, 2004). To help readers understand the proposed MAF algorithm, a toy example is given as shown in Fig. 7. For this example, we have ${}^{\mathrm{e}}\mathscr{C} = \{{}^{\mathrm{e}}\mathscr{C}_1, {}^{\mathrm{e}}\mathscr{C}_2\}$. By following the steps of the MAF algorithm, ${}^{T}\mathscr{C} \leftarrow {}^{\mathrm{e}}\mathscr{C}$, ${}^{\mathrm{m}}\mathscr{C} \leftarrow \{\varnothing\}$, $n \leftarrow 2$. In Step 2, we check whether ${}^{\mathrm{e}}\mathscr{C}_2$ is intersected with ${}^{\mathrm{e}}\mathscr{C}_1$, and the result is yes. Thus, the index of the first conflict area that intersects with ${}^{\mathrm{e}}\mathscr{C}_2$ is 1. By following the equations in Step 3, we merge ${}^{\mathrm{e}}\mathscr{C}_2$ to ${}^{\mathrm{e}}\mathscr{C}_1$. The upper bound for the new formed ${}^{\mathrm{e}}\mathscr{C}_1$ is $\{{}^{\mathrm{e}}p_{1,{}^{\mathrm{e}}t_1^-}^+, \cdots, {}^{\mathrm{e}}p_{1,{}^{\mathrm{e}}t_1^+}^+, {}^{\mathrm{e}}p_{2,{}^{\mathrm{e}}t_1^+ + 1}^+, \cdots, {}^{\mathrm{e}}p_{2,{}^{\mathrm{e}}t_2^+}^+\}$, and the lower bound is $\{{}^{\mathrm{e}}p_{1,{}^{\mathrm{e}}t_1^-}^-, \cdots, {}^{\mathrm{e}}p_{1,{}^{\mathrm{e}}t_2^- - 1}^-, {}^{\mathrm{e}}p_{2,{}^{\mathrm{e}}t_2^-}^-, \cdots, {}^{\mathrm{e}}p_{2,{}^{\mathrm{e}}t_2^+}^-\}$, as shown in Fig. 7. Then, ${}^{\mathrm{e}}\mathscr{C}_1$ in ${}^{T}\mathscr{C}$ will be updated with the new upper and lower bounds, and $n = 1$. Next, since the remaining conflict area for $n = 1$ in ${}^{T}\mathscr{C}$ is $\{\varnothing\}$, we add ${}^{\mathrm{e}}\mathscr{C}_1$ into ${}^{\mathrm{m}}\mathscr{C}$. In the next iteration, n equals 0, and we output ${}^{\mathrm{m}}\mathscr{C} := \{{}^{\mathrm{e}}\mathscr{C}_1\}$, which includes the upper and lower bounds for the fused area of the original ${}^{\mathrm{e}}\mathscr{C}_1$ and ${}^{\mathrm{e}}\mathscr{C}_2$. Note that after being processed by the MAF algorithm, the conflict areas in ${}^{\mathrm{m}}\mathscr{C}$ are isolated in the space-time diagram.

4. Customized dynamic-programming-based algorithm

Based on the conflict area analyses we studied above, this section proposes a customized DP-based algorithm to generate the optimal trajectory for the AV along the path. If the trajectory of the AV intersects with the conflict areas in the space-time diagram, it indicates that the trajectory is infeasible. Thus, in the algorithm, the conflict areas are employed to cut invalid trajectories at each stage.

It is known that there is a relationship, $T = \theta\mathbb{T}$, between the length of a time interval (i.e., θ) and the number of time intervals ($\mathbb{T}$) that is related to the computational time for generating a trajectory. In a given study period $[0, T]$, the length of a time interval decreases with increasing the number of the discretized time intervals, and thus a longer computational time is needed to generate the motion parameters (e.g., acceleration, speed, location) for the time intervals. To obtain a collision-free trajectory without much loss of computational efficiency, in the proposed algorithm, a searching method is incorporated to decide the length of time intervals. For example, the searching method starts with relatively long time intervals. If no feasible trajectory can be obtained with the length of time intervals, the algorithm will gradually decrease the time interval duration until a feasible trajectory is found. Since we assume that T is a sufficiently long time for the AV to complete the path, the problem will finally be solved with the minimum length of a time interval denoted by θ_φ. In practice, the execution time interval for a state-of-the-art drive-by-wire control system is 20ms - 200ms (Wang et al., 2020). When the time interval length of the planned trajectory equals the length of the execution time interval, the further decrease of the planned time interval length cannot keep improving the control accuracy of the AV. Thus, in the proposed algorithm, the minimum length of time intervals for the planned trajectory is set to 20ms. If a feasible trajectory is found before the length of time intervals decreases to 20ms, to produce a curvature-continuous trajectory, an interpolation method with a cubic spline will be adopted to generate the trajectory points between any two consecutive locations (Gu et al., 2013).

With this, the basic elements of the proposed customized DP-based algorithm are described as follows.

- Stage: In the studied problem, each discrete time interval $t \in \mathscr{T}$ is a stage. Each stage constitutes a new problem to be solved to find the next stage with the minimum cost.
- States: There are three state variables in the customized DP algorithm at each Stage t, i.e., the location of the AV at Stage t (x_t), the speed of the AV at Stage t (v_t), and the acceleration of the AV at Stage $t-1$ (a_t^{p}). The initial values of these three states are set to 0, ν, and α, respectively (i.e., $x_0 = 0$, $v_0 = \nu$, $a_0^{\mathrm{p}} = \alpha$).
- Decision: The acceleration of the AV at each Stage t (i.e., a_t) is the decision variable. The range of the acceleration is $[\underline{a}, \overline{a}]$.
- Cost: The cost function of the algorithm till Stage t is denoted by $M(t)$ indicating the overall cost till Stage t. M_0 is set to 0.

The state space at each Stage t is denoted by $S_t := \{(x_t, v_t, a_t^{\mathrm{p}})\}$. The state transition function is shown in Eq. (11).

$$T\left(\begin{bmatrix} x_t \\ v_t \\ a_t^{\mathrm{p}} \end{bmatrix}, a_t\right) = \begin{bmatrix} x_{t+1} := x_t + v_t \\ v_{t+1} := \max\{\min\{v_t + a_t, \overline{v}\}, 0\} \\ a_{t+1}^{\mathrm{p}} := a_t \end{bmatrix}, \ \forall t \in \mathscr{T}. \tag{11}$$

With these fundamental concepts, the customized DP-based algorithm is described as follows.

Step 1: Input the set of time interval length $\Theta := \{\theta_1, \theta_2, \cdots, \theta_\varphi\}$, we have $\theta_i \langle \theta_j, \forall i \rangle j \in \{1, \cdots, \varphi\}$. Input the length of the fixed path L, the end of time horizon T, conflict area set $\mathscr{C}'$, and vehicle kinematic limits $\overline{v}, \underline{a}, \overline{a}$. Input $i = 1$.
Step 2: Generate the discretized time intervals $\mathscr{T}_i := \{1, 2, \cdots, \mathbb{T}_i\}$ based on θ_i. Start from Stage $t = 0$ with state space $S_0 := \{x_0, v_0, a_0^{\mathrm{p}}\}$ and Cost $M_0 := 0$.
Step 3: Set state space $S_{t+1} = \varnothing$.
Step 3.1: For each state $s_t := (x_t, v_t, a_t^{\mathrm{p}}) \in S_t$, state $s_{t+1} := (x_{t+1}, v_{t+1}, a_{t+1}^{\mathrm{p}}) \in S_{t+1}$ is obtained by the transition function (i.e. $s_{t+1} = T(s_t, a_t)$). If any of the following conditions (i.e., Eqs. (12) – (14)) is met for this state, delete this state from S_{t+1} and repeat Step 3.1; otherwise, if $s_{t+1} = \varnothing$, $i = i + 1$, go to Step 2; if $s_{t+1} \neq \varnothing$, go to Step 3.2.

$${}^{\mathrm{m}}p_{n,t}^{-} \leq x_t \leq {}^{\mathrm{m}}p_{n,t}^{+}, \tag{12}$$

$$x_t + \overline{v} \cdot (\mathbb{T}_i - t) < L, \tag{13}$$

$$v_t + a_t \cdot \theta_i \langle 0 \ or \ v_t + a_t \cdot \theta_i \rangle \overline{v}. \tag{14}$$

Step 3.2: Calculate the cost of $M'_{t+1}(s_{t+1} := T(s_t, a_t)) := M_t^*(s_t) + (a_t - a_t^{\mathrm{p}})^2 - wx_t$. If $s_{t+1} \notin S_{t+1}$, add s_{t+1} to S_{t+1}, and set mappings $M_{t+1}^*(s_{t+1}) = M'_{t+1}(s_{t+1})$, $x_t^*(s_{t+1}) = a_t$, and $S_t^*(s_{t+1}) = s_t$; otherwise, if $M'_{t+1}(s_{t+1}) < M_{t+1}^*(s_{t+1})$, update mappings $M_{t+1}^*(s_{t+1}) = M'_{t+1}(s_{t+1})$, $x_t^*(s_{t+1}) = a_t$, and $S_t^*(s_{t+1}) = s_t$.
Step 3.3: If $t < \mathbb{T}_i$, $t = t + 1$, go to Step 3; otherwise, go to Step 3.4.
Step 3.4: Interpolate the AV location with a cubic spline to make the length of a time interval equal to θ_φ. If $\exists t'' \in \mathscr{T}_\phi$, we have ${}^{\mathrm{m}}p_{n,t''}^{-} \leq x_{t''} \leq {}^{\mathrm{m}}p_{n,t''}^{+}$, then $i = i + 1$, and go to Step 2; otherwise, go to Step 4.

Algorithm
Customized DP-based algorithm.

Input $\Theta; L; T; x_0, v_0, a_0^{\mathrm{p}}; \overline{v}, \underline{a}, \overline{a}; \mathscr{C}'; i \leftarrow 1$.
While 1 **do**
  $S_0 \leftarrow \{x_0, v_0, a_0^{\mathrm{p}}\}, M(0) \leftarrow 0$;
  For $t = 0$ to $\mathbb{T}_i$ **do**
    $S_{t+1} \leftarrow \varnothing$;
    For each state $s_t := \{x_t, v_t, a_t^{\mathrm{p}}\} \in S_t$ **do**
      $s_{t+1} \leftarrow T(s_t, a_t)$;
      If s_{t+1} does not meet the conditions in Eqs. (12) - (14)
        Calculate the cost of $M'_{t+1}(s_{t+1} := T(s_t, a_t)) := M_t^*(s_t) + (a_t - a_t^{\mathrm{p}})^2 - wx_t$;
        If $s_{t+1} \notin S_{t+1}$
          Add s_{t+1} to S_{t+1}, and set mappings $M_{t+1}^*(s_{t+1}) \leftarrow M'_{t+1}(s_{t+1})$, $x_t^*(s_{t+1}) \leftarrow a_t$, and $S_t^*(s_{t+1}) \leftarrow s_t$;
        Else if $M'_{t+1}(s_{t+1}) < M_{t+1}^*(s_{t+1})$
          Update mappings $M_{t+1}^*(s_{t+1}) \leftarrow M'_{t+1}(s_{t+1})$, $x_t^*(s_{t+1}) \leftarrow a_t$, and $S_t^*(s_{t+1}) \leftarrow s_t$;
        End if
        End if
      End for
    End for
  Get $\min\{M_{\mathbb{T}_\phi}^*(s_{\mathbb{T}_\phi})\}, \ \forall s_{\mathbb{T}_\phi} \in S_{\mathbb{T}_\phi}$. Denote the state for this optimal cost by $s_{\mathbb{T}_\phi}^+$;
  For $t = \mathbb{T}_i$ to 1 **do**
    $x_{t-1}^* \leftarrow x_{t-1}^*(s_t^*), s_{t-1}^* \leftarrow S_{t-1}^*(s_t^*)$;
  End for
  Interpolate the optimal AV trajectory $\{x_t^*\}$ with a cubic spline to make the length of a time interval equal to θ_φ;
  If $\forall t'' \in \mathscr{T}_\phi, \ x_{t''} > {}^{\mathrm{m}}p_{n,t''}^{+}$ *or* $x_{t''} < {}^{\mathrm{m}}p_{n,t''}^{-}$
    Break;
  End if
  $i \leftarrow i + 1$
End while
Output The optimal AV trajectory $\{x_t^*\}$, speed $\{v_t^*\}$, and acceleration $\{a_t^*\}$;

Step 4: Get mappings at Stage $\mathbb{T}_\phi$ with the minimum cost in the state space (i.e., $\min\{M^*_{\mathbb{T}_\phi}(s_{\mathbb{T}_\phi})\},\ \forall s_{\mathbb{T}_\phi} \in S_{\mathbb{T}_\phi}$), and set it as the optimal cost. Denote the state for this optimal cost by $s^*_{\mathbb{T}_\phi}$. Then the optimal accelerations of the AV at Stage $t \in \mathcal{T}$ (i.e., $\{a^*_t\}$) are obtained by tracking back from Stage $\mathbb{T}_\phi$ to Stage 1 according to $x^*_{t-1} = x^*_{t-1}(s^*_t)$ and $s^*_{t-1} = S^*_{t-1}(s^*_t)$.

Note that due to Step 2 and Step 3.1, the sizes of stages and states at each stage can be largely reduced, respectively, which significantly facilitates the computational efficiency (see Section 5 for demonstrations). The pseudo code of the proposed customized DP-based algorithm is shown in Algorithm.

5. Numerical experiments

In this section, we conduct a set of experiments to demonstrate the performance of the proposed model and the customized DP-based algorithm under different scenarios. A well-known commercial solver, Gurobi, has capability to solve optimization models with logical constraints. Thus, the performance of Gurobi serves as a benchmark for the proposed algorithm, and the comparisons between the original DP algorithm, the customized DP-based algorithm, and Gurobi are given. All algorithms are coded in Visual Studio Code C++ at a Window 10 PC with E3-1275 CPU and 32.0 GB RAM.

5.1. Single-CMO experiments

To test the effectiveness of the proposed model and algorithm, this subsection conducts two simple single-CMO experiments. Specifically, one type of conflict scenario is tested in each experiment, e.g., the scenarios that AV passes an intersection or follows a car. The parameter settings of the AV are as follows. The maximum speed (i.e., $\bar{v}$), minimum acceleration (i.e., $\underline{a}$), and maximum acceleration (i.e., $\bar{a}$) of the AV are set to 12 m/s, -2 m/s^2 and 1 m/s^2 (Bae et al., 2019; Milanés and Shladover, 2014), respectively. The front and rear safety buffers (i.e., b_f and b_r) are respectively set to 1 meter considering vehicle safety. Note that in the real-world implementation, the settings of safety buffer are usually a tricky problem that needs to consider a series of factors, e.g., AV current speed, AV control accuracy, and customer acceptance to the minimum following distance. Interested readers can refer to (Li et al., 2021; Shi and Li, 2021b) to understand the AV car-following design of commercial AVs. For illustration purposes, we set the safe buffers to 1 meter in the numerical experiments. To test the impacts of different travel distance weights (i.e., w) on optimal trajectories, we solve the optimal trajectories with different values of w, i.e., $w := \{0.04, 0.02, 0.1, 0.5\}$. The set of the length of a time intervalΘ is set to {2000, 1000, 500, 200, 100, 50, 20} milliseconds.

With this, for the intersection scenario as shown in Fig. 4a, we assume that the length of the AV path (i.e., L) is 25 meters. The AV stops at location 0 at time 0 and thus the initial speed (i.e., v_0) and acceleration (i.e., a_0) of the AV are set to 0, respectively. The AV needs to arrive at location L within 10 seconds. A CMO will pass through the intersection from the horizontal direction. The spatial range and period occupied by the CMO on the AV path are [15, 20] meters and [3, 6] seconds.

Then, based on the conflict area analyses described in Section 3.2, the conflict area caused by the CMO on spatial-time diagram is shown in Fig. 8a. With this, the optimal trajectories with different weights of the travel distance (i.e., w) for the AV to safely arrive at location L before time T are obtained by the proposed DP-based algorithm, as highlighted by different colors and markers in Fig. 8a. It can be observed that as the values of w gradually increase, the optimal trajectories shift. When w equals 0.004 or 0.02, the optimal trajectories are the same, shown as the lower trajectories in Fig. 8a. When w increases to 0.1, the optimal trajectory shifts to the green dash line. When w keeps increasing to 0.5, to balance the time efficiency and comfort of the AV, the trajectory shifts to the pink line with star markers. Fig. 8b shows the speed profiles for the optimal trajectories. We can find that as w increases, more speed fluctuations can be observed in the figure due to the variations of vehicle accelerations. That is, when the speed profiles with w equal to 0.004 or

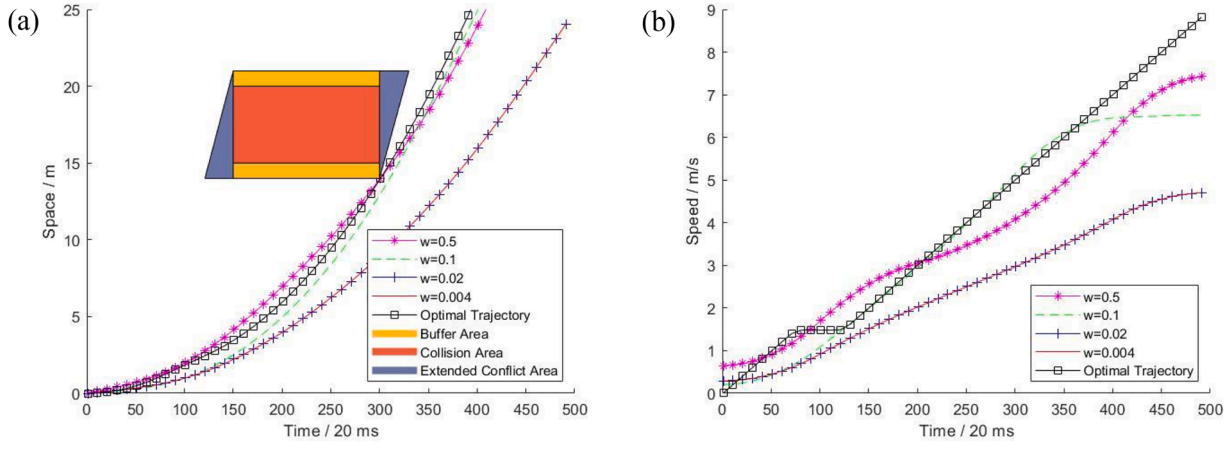

Fig. 8. Conflict area, optimal trajectories, and corresponding speed profiles for the intersection scenario with a CMO.

0.02, the slopes of the speed profiles are almost constant between [1, 9] seconds. When w equal to 0.1, the slope of the speed profile is almost constant between [2, 8] seconds. However, when w equal to 0.5, the slope of the speed profile is varying in the time horizon since the importance of comfort is further reduced.

For the car following scenario shown in Fig. 4b, we assume that the length of the AV path (i.e., L) also is 25 meters. The AV stops at location 0 at time 0 and needs to arrive at location L within 10 seconds. There is a CMO before the AV stops at location 15 m, and the length of the CMO is 5 m. To make the motion of the CMO become more general, this CMO will accelerate in [0,2] seconds at a rate of 2 m/s^2; then the CMO will decelerate in [2,6] seconds at a rate of 1 m/s^2; finally, the CMO will accelerate at a rate of 1 m/s^2 until the end of the time horizon (i.e., 10 seconds).

With these settings, the conflict area and the optimal trajectories for the AV with different values of w in the space-time diagram are shown in Fig. 9a, and the speed profiles for the optimal trajectories are shown in Fig. 9b. The results are consistent with what we observed in Fig. 8. That is, as w increases, the optimal trajectory shifts from the lowest red line and blue line with plus signs to the middle green dash line, and then shifts to the highest pink line with star markers, which indicates that the weights of travel distance in the objective are increasing. Also, as can be observed in Fig. 9b, as w increases, more speed fluctuations can be observed in the figure due to the variations of vehicle accelerations. The speed profile with the least value of w is about a straight line. As w increases, the fluctuations of the speed profiles become significant. This is also related to the driving behavior of the AV. That is, as w increases, AV's driving behaviors become more aggressive. The AV is expected to travel as long as possible at each time interval. Thus, travel comfort is less important as w increases, which can be supported by the significant speed variations shown in Fig. 9b.

To evaluate the computational performance of the proposed DP-based algorithm, Table 2 compares the results among the original DP algorithm, the customized DP-based algorithm, and Gurobi, including the solution time, objective, and time interval length (i.e., θ_i). Since θ_i is not a given value in the proposed DP-based algorithm, to make the comparison fair, we first use the customized DP-based algorithm to solve the instance. Then the θ_i obtained by the proposed DP-based algorithm will serve as the input of Gurobi. Thus, the instance for different methods will be exactly the same in terms of the number of decision variables, objective function, and constraints. Moreover, since the cubic interpolation will be adopted if $\theta_i \neq 20$ milliseconds, we are interested in the difference between the solutions with or without interpolations. To achieve this, if $\theta_i \neq 20$ milliseconds, Gurobi will be used to solve the optimal trajectory with $\theta_i = 20$ milliseconds. The optimal trajectories for each scenario can also be observed in Fig. 8 and Fig. 9 for making comparisons. The maximum solution time for the original DP algorithm, the customized DP-based algorithm, and Gurobi is set to 30 seconds, after which the results of all experiments will be displayed. The maximum solution time for calculating the benchmark trajectory with $\theta_i = 20$ milliseconds is unlimited.

As can be seen in Table 2, for some instances, the objective and solution time for the original DP algorithm are empty. This is because the original DP algorithm cannot obtain a feasible solution within the time limit due to the rapidly increased number of states, also known as the "curse of dimensionality" of the original DP algorithm. However, the customized DP-based algorithm circumvents this problem by the proposed upper and lower bounds and arriving time constraints (i.e., Step 3.1 in the proposed algorithm), and thus the number of states at each stage is largely reduced. As a result, the customized DP-based algorithm obtains solutions within the time limit. Comparing the results of the customized DP-based algorithm and Gurobi, it can be found that the objectives of these two methods are the same. It indicates that the two methods achieve the same optimal solutions with $\theta_i = 2$ (intersection scenario) or $= 1$ seconds (car-following scenario). However, comparing the solution time of these two methods, it is found that though Gurobi can solve the instance within the time limit, Gurobi always requires a much longer solution time than that of the proposed customized DP-based algorithm.

Although the proposed customized DP-based algorithm has an extremely short solution time, we would like to point out that this excellent performance is achieved by sacrificing the solution quality, which is revealed by the differences between the solutions obtained by the proposed algorithm and the optimal solution shown in Fig. 8 and Fig. 9. The sparse number of time intervals is the main

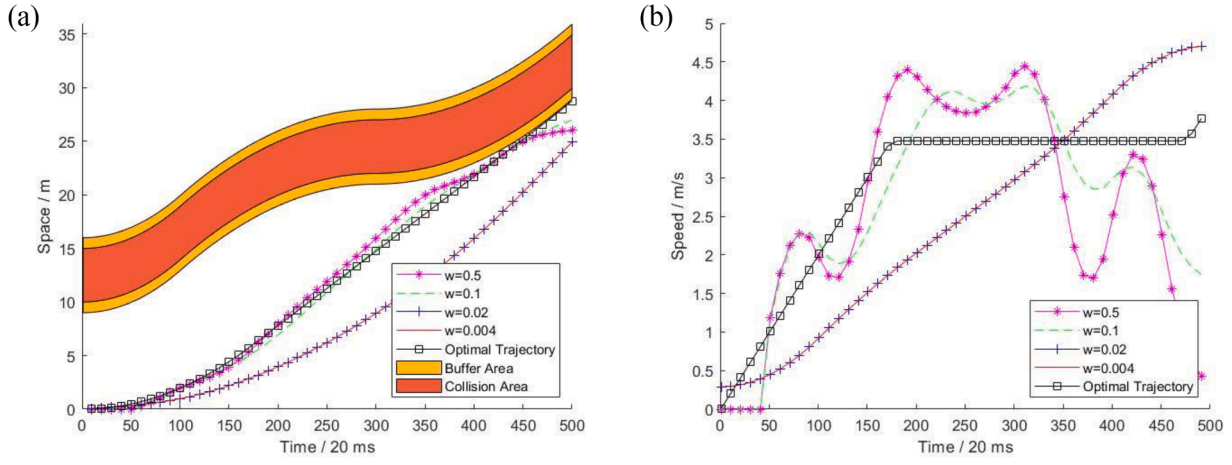

Fig. 9. Conflict area, optimal trajectories, and corresponding speed profiles for the car following scenario with a CMO.

Table 2
Computational result comparisons among different methods.

	Method	w	θ_i (s)	Objective		Solution time (ms)	BM objective	BM solution time (s)
Intersection	Original DP	0.004	2	0.13	170	0.08		8.5
		0.02	2	-	-	-0.57		10.3
		0.1	2	-	-	-4.23		10.6
		0.5	2	-	-	-23.17		12.9
	Customized DP-based	0.004	2	0.13	10	0.08		8.5
		0.02	2	-0.38	7	-0.57		10.3
		0.1	2	-3.90	4	-4.23		10.6
		0.5	2	-22.14	4	-23.17		12.9
	Gurobi	0.004	2	0.13	82	0.08		8.5
		0.02	2	-0.38	83	-0.57		10.3
		0.1	2	-3.90	82	-4.23		10.6
		0.5	2	-22.14	87	-23.17		12.9
Car-following	Original DP	0.004	1	0.13	200	0.05		14.1
		0.02	1	-	-	-0.74		15.4
		0.1	1	-	-	-15.94		17.7
		0.5	1	-	-	-60.73		19.4
	Customized DP-based	0.004	1	0.13	4	0.05		14.1
		0.02	1	-0.35	6	-0.74		15.4
		0.1	1	-9.13	6	-15.94		17.7
		0.5	1	-53.03	5	-60.73		19.4
	Gurobi	0.004	1	0.13	92	0.05		14.1
		0.02	1	-0.35	90	-0.74		15.4
		0.1	1	-9.13	87	-15.94		17.7
		0.5	1	-53.03	95	-60.73		19.4

reason causing this difference. That is, to facilitate computational efficiency, a searching method is incorporated into the proposed algorithm to decide the length of time intervals. A long time interval benefits the number of stages for solving the instance, but it also misses the detail of the solution. On the other hand, to produce a curvature-continuous trajectory, the interpolation method with cubic spline we adopted further enlarges the gap. Thus, one can find that there is a tradeoff between solution quality and solution efficiency. Instead of finding an optimal trajectory with a long solution time, the proposed DP-based algorithm prefers to find an optimality-based trajectory (e.g., the optimal trajectory for long time intervals, then interpolating it with cubic spline) with the minimized solution time, which renders the capability for implementing the proposed algorithm to practice.

To help readers understand the improvements of the proposed upper and lower bounds and arriving time constraints (i.e., Step 3.1 in the proposed algorithm), Fig. 10 plots out the number of states at each stage for both the original DP algorithm and the proposed customized DP-based algorithm for the car following scenario with $w = 0.5$ and $\theta_i = 200$ milliseconds. It can be observed that the number of states of the original DP algorithm increases rapidly due to the "curse of dimensionality" problem. For the customized DP-based algorithm, on the other hand, the states at each stage have a low increase rate because a large number of invalid states (i.e., infeasible trajectories) are eliminated by the proposed upper and lower bounds at each stage (i.e., Eq. (12)). In addition, as the stage approaches the end, the number of states is largely reduced due to the arriving time constraints (i.e., Eq. (13)). With these two customized cut rules, the "curse of dimensionality" problem of the original DP algorithm is circumvented and thus the optimal solution

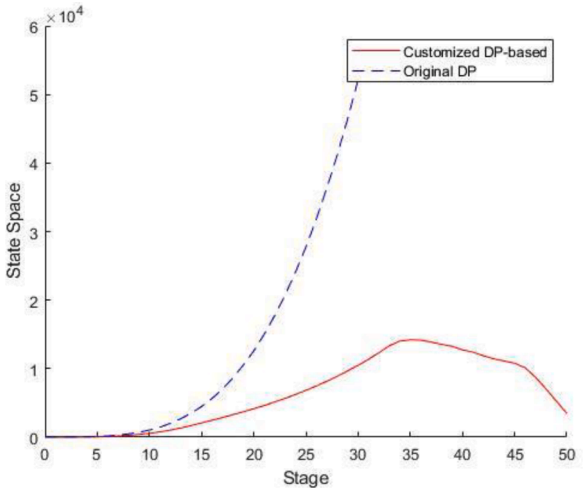

Fig. 10. Comparison between the number of states for the customized DP-based algorithm and original DP algorithm.

can be obtained within a relatively short solution time.

5.2. Multiple CMOs experiments

This subsection conducts experiments that aim to illustrate the capability of the proposed model and algorithm under complex traffic scenarios. As shown in Fig. 11, we assume that the AV starts its journey at the red point and heads for the orange point by following the spatial path colored by green. The length of the path (i.e., L) is 30 meters. On the path, there is one intersection, and the AV will turn left at the intersection to continue its motion. The length of each segment is shown in Fig. 11. Assume that the AV needs to arrive at the destination (i.e., orange point) within 16 seconds, and the intersection is occupied during [8,10] seconds because of the horizontal direction CMOs passing through. Besides the intersection conflicts, a 5-meter-length HV will make the lane changing from lane 2 to lane 1 at location 10 meters in 6 seconds. The initial speed of the HV is 5 m/s, the acceleration of the HV is 2 m/s^2, and this HV will go through the intersection. There are two pedestrians considered in the scenario. A pedestrian will cross the intersection with occupying location 25 meters during [13,16] seconds, and the other pedestrian will cross the street with occupying location 15 meters during [2,5] seconds.

With these settings, the conflict area and optimal trajectories for the AV with different values of w in the space-time diagram are shown in Fig. 12. It can be observed that the conflict areas of the CMOs are intersected in the space-time diagram. By using the proposed MAF algorithm, the intersected conflict areas are fused into one larger conflict area that serves as cut rules to the customized DP-based algorithm. Note that the instances with different values of w obtain the same optimal trajectories as shown in Fig. 12. One possible reason for this result is that in the complex traffic scenario, the space-time areas to feasible trajectories are relatively narrow. Thus, although the values of w vary, the optimal trajectory keeps the same. To further support this possibility, two benchmark trajectories of the AV are provided in Fig. 12. These two trajectories are generated by considering either the travel distance (i.e., $obj = \sum_{t\in\mathcal{T}} x_t$,s.t., equations (1) – (9), denoted by the selfish trajectory) or the travel comfort (i.e.$obj = \sum_{t\in\mathcal{T}\setminus\{0\}} |a_t - a_{t-1}|$,s.t., equations (1) – (9), denoted by the comfortable trajectory). We can see that the optimal trajectory must be within the area covered by these two trajectories. Considering the conflict areas in which the trajectory cannot pass through, the feasible areas for the AV are pretty limited.

The computational result comparisons between the different solution methods are shown in Table 3. Because the original DP algorithm still is unable to obtain a feasible solution within the time limit for most of the instances, Table 3 just compares the computational results between the customized DP-based algorithm and Gurobi. It can be seen that the objective values of the customized DP-based algorithm and Gurobi are the same for the different w, which indicates that the optimal solutions obtained by the two methods are the same. However, the proposed customized DP-based algorithm always requires much less solution time than that of Gurobi. The results suggest that the proposed customized DP-based algorithm has better capability to generate the optimal trajectory under the complex traffic scenario than Gurobi. Particularly, due to the nature of the proposed algorithm mentioned above, the larger the conflict area in the space-time diagram is (e.g., the more complex the traffic scenario is), the more invalid trajectories will be eliminated at each stage, and the shorter the solution time is needed for generating the optimal solution for the proposed DP-based algorithm. On the contrary, more CMOs in the environment indicate that more variables and constraints will be added into the model, and the model will be more complex. As a result, the Gurobi solver requires more solution time to solve the model to the optimal, which can be supported by comparing the solution time between Table 2 and Table 3.

Further, we would like to compare the performance of the trajectory generated by the proposed method with those generated by state-of-the-art AV car-following models to provide insights into the traffic benefits of the proposed method. Here we provide the

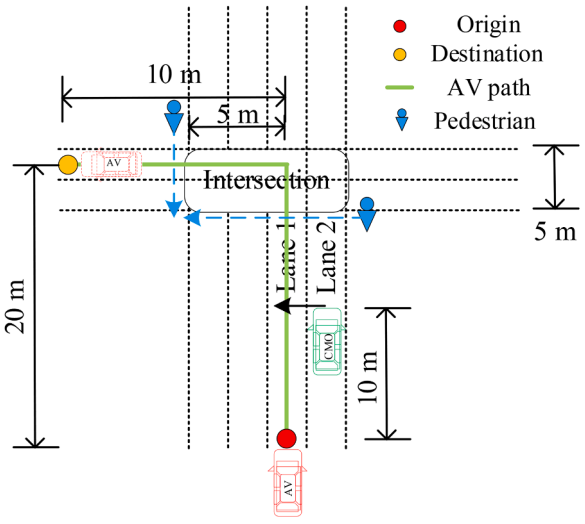

Fig. 11. Experiment settings for the multiple CMOs experiments.

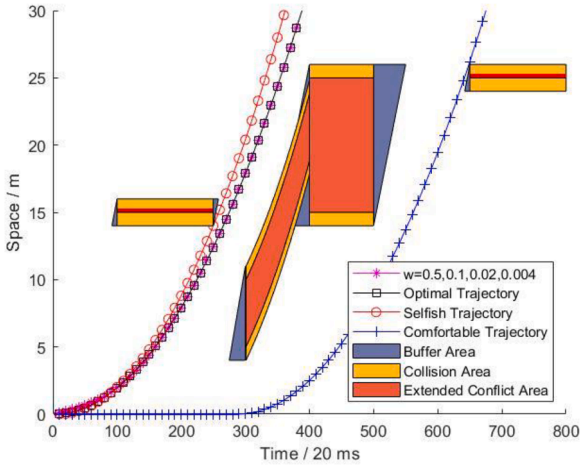

Fig. 12. Conflict area and optimal trajectories for the scenario with multiple CMOs.

Table 3
Computational result comparisons for the multiple CMOs in environments.

Method	w	θ_i (s)	Solution time (ms)	Objective	BM objective	BM solution time (s)
Customized DP-based	0.004	1	7	0.15	0.15	10.3
	0.02	1	7	-4.25	-4.63	10.2
	0.1	1	6	-26.25	-29.04	9.8
	0.5	1	6	-137.02	-151.26	9.9
Gurobi	0.004	1	229	0.15	0.16	10.3
	0.02	1	219	-4.25	-4.25	10.2
	0.1	1	209	-26.25	-29.04	9.8
	0.5	1	203	-137.02	-151.26	9.9

trajectories generated by the intelligent driver model (IDM) (Milanés and Shladover, 2014) and the commercial adaptive cruise control (ACC) model (Gunter et al., 2020) as benchmarks as shown in Fig. 13. Since the AV car-following models can only make the decisions according to its preceding vehicle, it is fairly easy for the models to generate collision trajectories if a CMO suddenly appears and blocks the AV's path. To help the AV car-following models generate collision-free trajectories, we add several virtual CMOs in the environment. For example, if we want the AV to avoid the CMOs by following the lane-changing vehicle (e.g., the green line shown in Fig. 14), we add a virtual CMO that blocks the path of the AV within [0,6] seconds. It is equivalent to drawing a line in Fig. 13 at the location of 4 meters within [0,6] seconds. Then, the collision-free trajectories generated by the IDM and commercial ACC models are shown in the figure.

We can observe that the trajectory generated by the proposed model avoids the CMOs before the appearance of the lane-changing CMO. Thus, it is the trajectory with the earliest arriving time to the destination. However, none of the car-following models can generate collision-free trajectories before the appearance of the lane-changing CMO. It is because the car-following models make trajectory planning only referring to its preceding vehicle. Both of the AV car-following models can successfully avoid the pedestrian at 15 meters. However, they have no knowledge about the lane-changing CMO. Due to the insufficient acceleration after avoiding the pedestrian, both of them conflict with the lane-changing CMO and thus can only avoid the lane-changing CMO after it appears.

We can also see that the IDM model generated a trajectory that can arrive at the destination by the end of the time horizon (green dash line), but the commercial ACC model failed to do so (blue solid line). One possible reason is that the commercial ACC model has more conservative operation settings than the IDM model. It is reasonable because safety is always the most important criterion for vehicle vendors. Thus, in their current ACC settings, they tend to sacrifice AV's mobility to gain safety. Due to this, the ACC model missed opportunities to arrive at the destination within the time horizon.

Since the trajectory provided by the proposed model is the optimal trajectory to avoid the CMOs in the environment, it has the best performance in terms of travel time and travel comfort. Overall, we can conclude that the trajectories generated by the proposed model and algorithm not only can provide safe, comfortable, and time-efficient references for the AV traveling from the origin to the destination, but also outperforms some state-of-the-art AV car-following models, which indicates that the proposed model has a great potential to further enhance the future traffic.

6. Conclusions

This paper studies the optimal trajectory planning problem for an AV with CMOs along a fixed path. The conflict areas for a possible

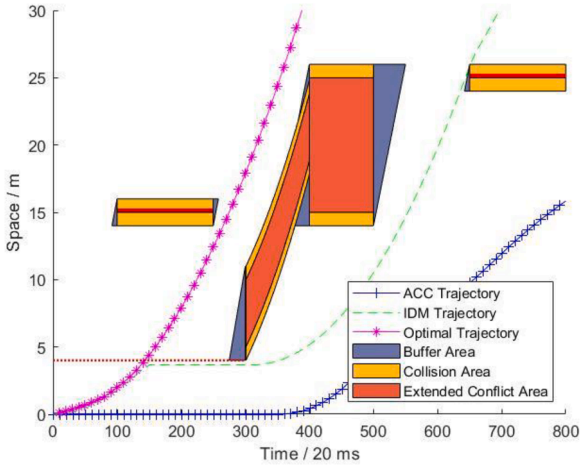

Fig. 13. Trajectories comparison among the proposed model, IDM model, and ACC model.

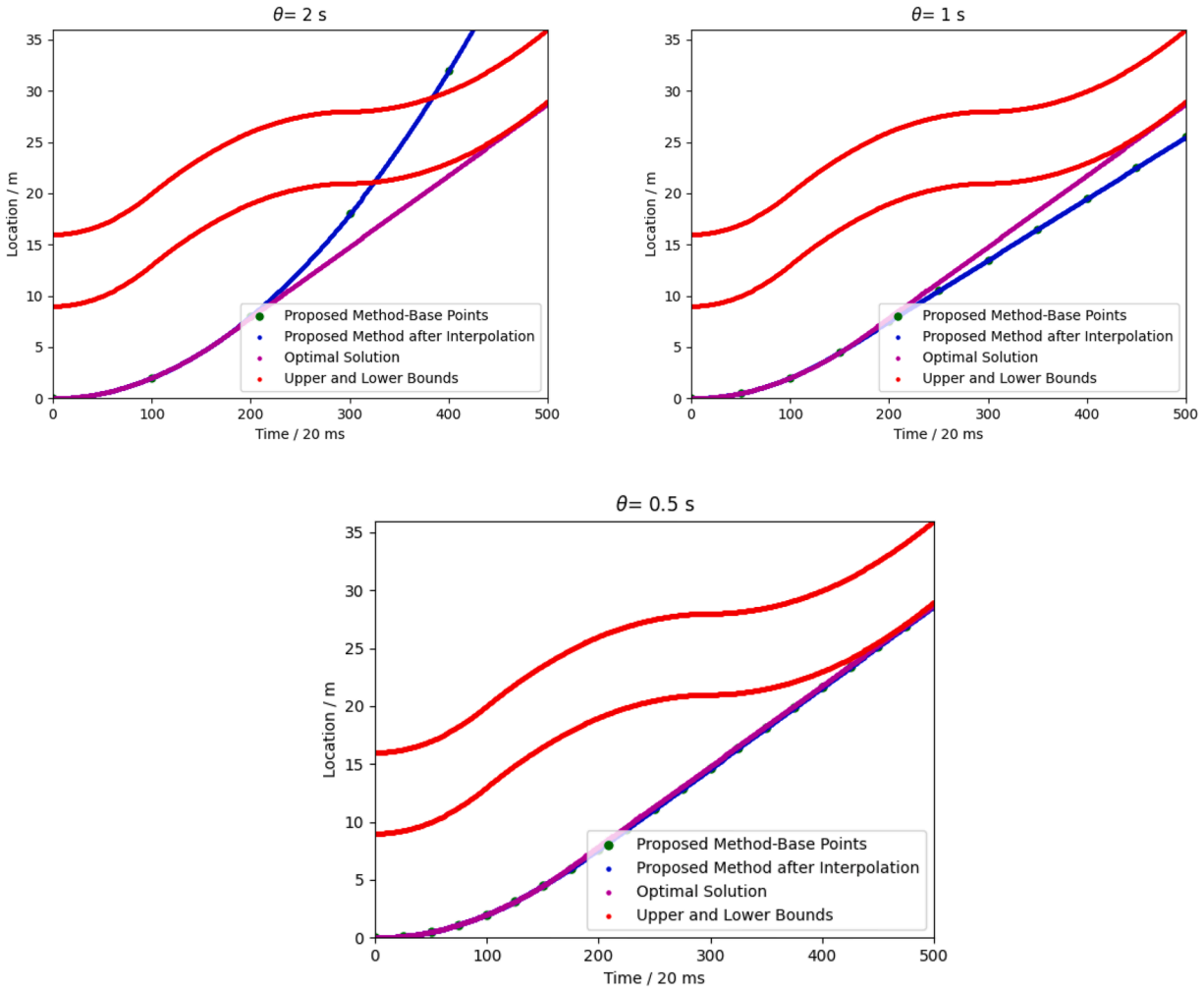

Fig. 14. Tradeoff between the solution quality and θ_i.

trajectory are characterized using analytical operations of piece-wise quadratic functions. A customized DP-based algorithm is constructed where the number of stages and states are largely reduced based on the results of the conflict area analyses. A set of numerical experiments are conducted to demonstrate the effectiveness of the proposed model and the customized DP-based algorithm. The result shows that the proposed customized DP-based algorithm can solve the problem efficiently with a solution time much less than that of a commercial solver, especially for the complex traffic scenarios. The research outcomes can be used to design safe, comfortable, and time-efficient speed profiles for AV operations in an environment with CMOs.

Future research can be conducted in a few directions. In this paper, we studied the conflict areas for a feasible trajectory, which are served as the cut rules for the proposed customized DP-based algorithm. It would be interesting to investigate the feasible regions of all AV trajectories on the space-time diagram and thus the optimal trajectory can be obtained easily regarding the feasible regions. Then, as stated in Section 5.2, although the proposed searching method for the time interval length can benefit the solution time, it also will degrade the solution accuracy due to the sparse time intervals and the adopted cubic interpolation method. Thus, some heuristics algorithms can be studied to further enhance the solution quality. Moreover, the trajectory planning for an AV platoon would also be interesting and has practical meanings on the future development of AV technologies. In addition, in this paper, we assume that the trajectory prediction of CMOs is robust and thus the CMOs will exactly follow the predicted path. However, the CMOs' behaviors are uncertain, particularly when the CMOs are human-driven vehicles or pedestrians. As a result, it would be interesting to incorporate stochasticity and uncertainty into the model in future research.

CRediT authorship contribution statement

Xiaowei Shi: Methodology, Validation, Formal analysis, Investigation, Data curation, Visualization, Writing – original draft. **Xiaopeng Li:** Conceptualization, Methodology, Validation, Writing – review & editing, Supervision, Funding acquisition.

Data availability

Data will be made available on request.

Acknowledgment

This research is sponsored by NSF through grants CMMI # 1558887 and # 1932452. The preprint of this work has received the Neville A. Parker – Science & Technology Award in 2021 from the Council of University Transportation Centers (CUTC).

Appendix

To understand the computational performance of the proposed customized-DP algorithm under different values of θ_i, a sensitivity analysis to θ_i is conducted. The values of θ_i will vary in {2, 1, 0.5, 0.2, 0.1, 0.05, 0.02} seconds. The results are respectively {6, 10, 15, 29, 48, 97, 313} milliseconds. The solution time is the average solution time across all experiments. We can find that as the time discretization step decreases, the solution time exhibits increasing trend. This is because of the nature of the dynamic programming algorithm. As the number of stages (i.e., number of time intervals) increase, the number of state spaces will increase exponentially, which is also known as the "curse of dimensionality". However, benefiting from the proposed valid cuts (i.e., Step 3.1), even for the shortest time discretization step setting (i.e., 0.02 s), all instances can be solved with an average solution time of 313 ms, which is a fairly good result compared to 10.8 seconds for the Gurobi solver and further demonstrates the superior performance of the proposed customized-DP algorithm.

Then, the car-following scenario with different time discretization steps is studied as follows to illustrate the tradeoff between the solution quality and time discretization steps. In Fig. 14, the solutions generated by the proposed algorithm with time discretization step θ equals to {2s, 1s, 0.5s} are plotted, respectively.

We can observe that when θ equals 2s, since the discretization step is too rough, the proposed algorithm generates an invalid trajectory, which passes through the conflict area (blue dot line in the figure). Thanks to the trajectory validation step in the proposed algorithm (i.e., Step 3.4), this invalid trajectory will be rejected, and θ will decrease to 1s to search for a new solution. When θ equals 1s, we can see that since the duration for each movement is shorter, the previous issue does not happen. However, there still is quite obvious gap between the optimal solution and the solution generated by the proposed algorithm. The objective for the generated solution is -0.67 and that for the optimal solution is -0.74, and the gap is 9.38%. If we further decrease the discretization step to 0.5s, we can observe in the third figure that the solution generated by the proposed algorithm is almost overlapped with the optimal solution, and the solution gap becomes 1.35%. In the current version of the proposed algorithm, for the studied scenario, the algorithm will output the solution when θ equals 1s. Considering the real-vehicle control and tracking errors, the trajectories generated by the current version of the algorithm are more conservative and safer for conducting real-world experiments.

References

Bae, I., Moon, J., Seo, J., 2019. Toward a comfortable driving experience for a self-driving shuttle bus. Electron. 8, 1–13. https://doi.org/10.3390/electronics8090943.

Bobrow, J.E., Dubowsky, S., Gibson, J.S., 1985. Time-optimal control of robotic manipulators along specified paths. Int. J. Rob. Res. 4, 3–17. https://doi.org/10.1177/027836498500400301.

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press.

Chen, L., Qin, D., Xu, X., Cai, Y., Xie, J., 2019. A path and velocity planning method for lane changing collision avoidance of intelligent vehicle based on cubic 3-D Bezier curve. Adv. Eng. Softw. 132, 65–73. https://doi.org/10.1016/j.advengsoft.2019.03.007.

Constantinescu, D., Croft, E.A., 2000. Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. J. Robot. Syst. 17, 233–249. https://doi.org/10.1002/(SICI)1097-4563(200005)17:5<233::AID-ROB1>3.0.CO;2-Y.

Dresner, K., Stone, P., 2004. Multiagent traffic management: a reservation-based intersection control mechanism. In: Autonomous Agents and Multiagent Systems, International Joint Conference on. IEEE Computer Society, pp. 530–537. Vol. 3.

Dubowsky, S., Norris, M., Shiller, Z., 1986. Time optimal trajectory planning for robotic manipulators with obstacle avoidance: a CAD approach. In: Proceedings. 1986 IEEE International Conference on Robotics and Automation. Institute of Electrical and Electronics Engineers, pp. 1906–1912. https://doi.org/10.1109/ROBOT.1986.1087434.

Fulgenzi, C., Spalanzani, A., Laugier, C., 2009. Probabilistic motion planning among moving obstacles following typical motion patterns. 2009 IEEE/RSJ Int. Conf. Intell. Robot. Syst. IROS 2009 4027–4033. https://doi.org/10.1109/IROS.2009.5354755.

González, D., Pérez, J., Milanés, V., Nashashibi, F., 2016. A review of motion planning techniques for automated vehicles. IEEE Trans. Intell. Transp. Syst. 17, 1135–1145. https://doi.org/10.1109/TITS.2015.2498841.

Gu, T., Snider, J., Dolan, J.M., Lee, J.W., 2013. Focused Trajectory Planning for autonomous on-road driving. IEEE Intell. Veh. Symp. Proc. 547–552. https://doi.org/10.1109/IVS.2013.6629524.

Gunter, G., Gloudemans, D., Stern, R.E., McQuade, S., Bhadani, R., Bunting, M., Monache, M.L.D., Lysecky, R., Seibold, B., Sprinkle, J., Piccoli, B., Work, D.B., 2020. Are commercially implemented adaptive cruise control systems string stable? IEEE Trans. Intell. Transp. Syst. 1–12. https://doi.org/10.1109/TITS.2020.3000682.

Hu, X., Sun, J., 2019. Trajectory optimization of connected and autonomous vehicles at a multilane freeway merging area. Transp. Res. Part C Emerg. Technol. https://doi.org/10.1016/j.trc.2019.02.016.

Ji, J., Khajepour, A., Melek, W.W., Huang, Y., 2017. Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. IEEE Trans. Veh. Technol. 66, 952–964. https://doi.org/10.1109/TVT.2016.2555853.

Karaman, S., Frazzoli, E., 2011. Sampling-based algorithms for optimal motion planning. Int. J. Rob. Res. 30, 846–894. https://doi.org/10.1177/0278364911406761.

Kasać, J., Deur, J., Novaković, B., Kolmanovsky, I.V., Assadian, F., 2011. A conjugate gradient-based BPTT-like optimal control algorithm with vehicle dynamics control Application. IEEE Trans. Control Syst. Technol. 19, 1587–1595. https://doi.org/10.1109/TCST.2010.2084088.

Katrakazas, C., Quddus, M., Chen, W.H., Deka, L., 2015. Real-time motion planning methods for autonomous on-road driving: state-of-the-art and future research directions. Transp. Res. Part C Emerg. Technol. 60, 416–442. https://doi.org/10.1016/j.trc.2015.09.011.

Kunz, T., Stilman, M., 2013. Time-optimal trajectory generation for path following with bounded acceleration and velocity. Robot. Sci. Syst. 8, 209–216. https://doi.org/10.7551/mitpress/9816.003.0032.

Lampariello, R., Nguyen-Tuong, D., Castellini, C., Hirzinger, G., Peters, J., 2011. Trajectory planning for optimal robot catching in real-time. In: Proceeding of the - IEEE International Conference on Robotic Automation. https://doi.org/10.1109/ICRA.2011.5980114.

Leon, F., Gavrilescu, M., 2021. A review of tracking and trajectory prediction methods for autonomous driving. Mathematics. https://doi.org/10.3390/math9060660.

Levin, M.W., Rey, D., 2017. Conflict-point formulation of intersection control for autonomous vehicles. Transp. Res. Part C Emerg. Technol. https://doi.org/10.1016/j.trc.2017.09.025.

Li, L., Li, X., 2019. Parsimonious trajectory design of connected automated traffic. Transp. Res. Part B Methodol. 119, 1–21. https://doi.org/10.1016/j.trb.2018.11.006.

Li, T., Chen, D., Zhou, H., Laval, J., Xie, Y., 2021. Car-following behavior characteristics of adaptive cruise control vehicles based on empirical experiments. Transp. Res. Part B 147, 67–91. https://doi.org/10.1016/j.trb.2021.03.003.

Li, X., Ghiasi, A., Xu, Z., Qu, X., 2018. A piecewise trajectory optimization model for connected automated vehicles: exact optimization algorithm and queue propagation analysis. Transp. Res. Part B Methodol. 118, 429–456. https://doi.org/10.1016/j.trb.2018.11.002.

Lipp, T., Boyd, S., 2014. Minimum-time speed optimisation over a fixed path. Int. J. Control 87, 1297–1311. https://doi.org/10.1080/00207179.2013.875224.

Liu, C., Lin, C.Y., Wang, Y., Tomizuka, M., 2017a. Convex feasible set algorithm for constrained trajectory smoothing. Proc. Am. Control Conf. 4177–4182. https://doi.org/10.23919/ACC.2017.7963597.

Liu, C., Zhan, W., Tomizuka, M., 2017b. Speed profile planning in dynamic environments via temporal optimization. IEEE Intell. Veh. Symp. Proc. 154–159. https://doi.org/10.1109/IVS.2017.7995713.

Makridis, M., Mattas, K., Ciuffo, B., Raposo, M.A., Toledo, T., Thiel, C., 2018. Connected and automated vehicles on a freeway scenario. Effect on traffic congestion and network capacity. 7th Transp. Res. Arena TRA.

Malikopoulos, A.A., Cassandras, C.G., Zhang, Y.J., 2018. A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections. Automatica. https://doi.org/10.1016/j.automatica.2018.03.056.

Milanés, V., Shladover, S.E., 2014. Modeling cooperative and autonomous adaptive cruise control dynamic responses using experimental data. Transp. Res. Part C Emerg. Technol. 48, 285–300. https://doi.org/10.1016/j.trc.2014.09.001.

Paden, B., Čáp, M., Yong, S.Z., Yershov, D., Frazzoli, E., 2016. A survey of motion planning and control techniques for self-driving urban vehicles. IEEE Trans. Intell. Veh. 1, 33–55. https://doi.org/10.1109/TIV.2016.2578706.

Pei, H., Zhang, Yuxiao, Zhang, Yi, Feng, S., 2022. Optimal Cooperative Driving at Signal-Free Intersections With Polynomial-Time Complexity. IEEE Trans. Intell. Transp. Syst. https://doi.org/10.1109/TITS.2021.3118592.

Peng, Y., Yong, J.H., Dong, W.M., Zhang, H., Sun, J.G., 2005. A new algorithm for Boolean operations on general polygons. Comput. Graph. https://doi.org/10.1016/j.cag.2004.11.001.

Rios-Torres, J., Malikopoulos, A.A., 2017. A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps. IEEE Trans. Intell. Transp. Syst. https://doi.org/10.1109/TITS.2016.2600504.

Rivero, M., Feito, F.R., 2000. Boolean operations on general planar polygons. Comput. Graph. 24, 881–896. https://doi.org/10.1016/S0097-8493(00)00090-X.

Schmerling, E., Janson, L., Pavone, M., 2015. Optimal sampling-based motion planning under differential constraints: the driftless case. In: Proceeding of the - IEEE International Conference on Robotic Automation 2015-June, pp. 2368–2375. https://doi.org/10.1109/ICRA.2015.7139514.

Shi, X., Li, X., 2021a. Constructing a fundamental diagram for traffic flow with automated vehicles: methodology and demonstration. Transp. Res. Part B Methodol. 150, 279–292. https://doi.org/10.1016/j.trb.2021.06.011.

Shi, X., Li, X., 2021b. Empirical study on car-following characteristics of commercial automated vehicles with different headway settings. Transp. Res. Part C Emerg. Technol. 128, 103134 https://doi.org/10.1016/j.trc.2021.103134.

Shi, X., Yao, H., Liang, Z., Li, X., 2022. An empirical study on fuel consumption of commercial automated vehicles. Transp. Res. Part D Transp. Environ. 106 https://doi.org/10.1016/j.trd.2022.103253.

Stachniss, C., Burgard, W., 2003. An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments 508–513. doi:10.1109/irds.2002.1041441.

Wang, Z., Zhao, X., Xu, Z., Li, X., Qu, X., 2020. Modeling and field experiments on autonomous vehicle lane changing with surrounding human-driven vehicles. Comput. Civ. Infrastruct. Eng. 1–13. https://doi.org/10.1111/mice.12540.

Wu, W., Liu, Y., Liu, W., Zhang, F., Dixit, V., Waller, S.T., 2021. Autonomous Intersection Management for Connected and Automated Vehicles: a Lane-Based Method. IEEE Trans. Intell. Transp. Syst. https://doi.org/10.1109/TITS.2021.3136910.

Zhan, W., Chen, J., Chan, C.Y., Liu, C., Tomizuka, M., 2017. Spatially-partitioned environmental representation and planning architecture for on-road autonomous driving. IEEE Intell. Veh. Symp. Proc. 632–639. https://doi.org/10.1109/IVS.2017.7995789.

Zhang, Y., Chen, H., Waslander, S.L., Yang, T., Zhang, S., Xiong, G., Liu, K., 2018. Toward a more complete, flexible, and safer speed planning for autonomous driving via convex optimization. Sensors (Switzerland) 18. https://doi.org/10.3390/s18072185.

Ziegler, J., Bender, P., Dang, T., Stiller, C., 2014a. Trajectory planning for Bertha - a local, continuous method. IEEE Intell. Veh. Symp. Proc. 450–457. https://doi.org/10.1109/IVS.2014.6856581.

Ziegler, J., Bender, P., Schreiber, M., Lategahn, H., Strauss, T., Stiller, C., Dang, T., Franke, U., Appenrodt, N., Keller, C.G., Kaus, E., Herrtwich, R.G., Rabe, C., Pfeiffer, D., Lindner, F., Stein, F., Erbs, F., Enzweiler, M., Knoppel, C., Hipp, J., Haueis, M., Trepte, M., Brenk, C., Tamke, A., Ghanaat, M., Braun, M., Joos, A., Fritz, H., Mock, H., Hein, M., Zeeb, E., 2014b. Making bertha drive-an autonomous journey on a historic route. IEEE Intell. Transp. Syst. Mag. 6, 8–20. https://doi.org/10.1109/MITS.2014.2306552.