Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Autonomous Intersection Management over Continuous Space: A Microscopic and Precise Solution via Computational Optimal Control

Bai Li* Youmin Zhang Ning Jia*** Xiaoyan Peng***

** College of Mechanical and Vehicle Engineering, Hunan University, Changsha, China (e-mail: libai@hnu.edu.cn, xiaoyan_p@126.com).*
*** Department of Mechanical, Industrial and Aerospace Engineering, Concordia University, Montreal, Canada (e-mail: ymzhang@encs.concordia.ca).*
**** Institute of Systems Engineering, Tianjin University, Tianjin, China (e-mail: jia_ning@tju.edu.cn).*

Abstract: Autonomous intersection management (AIM) refers to planning cooperative trajectories for multiple connected and automated vehicles (CAVs) when they pass through an unsignalized intersection. In modeling a generic AIM scheme, the predominant network-level or lane-level methods limit the cooperation potentiality of a multi-CAV team because 1) lane changes are forbidden or only allowed at discrete spots in the intersection, 2) each CAV's travel path is fixed or selected among a few topological choices, and 3) each CAV's travel velocity is fixed or set to a specified pattern. To overcome these limitations, this work models the intersection as a continuous free space and describes an AIM scheme as a multi-CAV trajectory optimization problem. Concretely, a centralized optimal control problem (OCP) is formulated and then numerically solved. To derive a satisfactory initial guess for the numerical optimization, a priority-based decentralized framework is proposed, wherein an *x*-*y*-time A* algorithm is adopted to generate a coarse trajectory for each CAV. To facilitate the OCP solution process, 1) the collision-avoidance constraints in the OCP are convexified, and 2) a stepwise computation strategy is adopted. Simulation results show the efficacy of the proposed offline AIM method.

Keywords: Autonomous intersection management (AIM), connected and automated vehicles (CAVs), trajectory planning, numerical optimization, computational optimal control

1. INTRODUCTION

An intersection is a typical scenario that reflects the inherent conflicts among multiple vehicles when their nominal routes intersect. Connected and automated vehicles (CAVs) have brought about promising chances to resolve the conflicts through their cooperative driving capability. This paper is focused on the autonomous intersection management (AIM) scheme, which is about planning the cooperative trajectories for multiple CAVs when they pass through an unsignalized intersection (Rios-Torres and Malikopoulos, 2016). The predominant AIM methodologies cannot sufficiently exploit the cooperation potentiality of a CAV team. This work aims to overcome this limitation.

AIM related studies began from (Dresner and Stone, 2008), wherein the trajectory of each CAV is sequentially planned with a first-come-first-serve (FCFS) strategy. Several rules are defined to plan the trajectory for each CAV, which support lane changes and velocity changes. These rules, compared with some subsequent studies, bring about flexibility in the vehicles' mobility. Vehicle travel behaviors under such rules appear to be near-optimal, i.e., resemble the optimal solutions derived by optimal control. Most of the subsequent studies, nonetheless, only focus on how to derive an optimal passing order and/or how to plan optimal velocities for the fixed-path vehicles. Mirheli et al. (2019) proposed an iterative framework to determine the velocity of each CAV along a specified path through solving a mixed-integer non-linear programming (MINLP) problem. The iterative process continues until consensus is achieved. Mirheli et al. (2018) formulated the multi-vehicle velocity planning problem as a mixed-integer linear programming (MILP) problem and solved it via Monte Carlo tree search. Similarly, Levin et al. (2017) derived the passing order and intersection entrance time simultaneously for all the CAVs through solving an MILP problem. Xu et al. (2018) projected the vehicles into a virtual lane, determined the passing order through a geometry topology method, and then designed a distributed controller to generate the time-continuous trajectories. Malikopoulos et al. (2018) decentralized the multi-vehicle velocity planning scheme into a sequential form and provided analytical solutions to the corresponding single-vehicle planning problems. Zohdy et al. (2012) defined two preparation zones for each CAV before it would enter the intersection, and then planned the velocity via a simulator-in-the-loop optimizer. Kamal et al. (2013) developed a model predictive control (MPC) approach to determine the cooperative velocity profiles as well as the passing order. Similar works include (Lin et al., 2017; Liu et al., 2019).

The aforementioned AIM methods are featured by 1) the CAVs' paths are fixed or restricted to predefined patterns, 2)

the CAVs' velocities are set to be constant or restricted to typical patterns, and 3) trajectories of the whole team are not simultaneously planned. These features, although may enhance the real-time performance of an AIM method, render that the spatial-temporal intersection space is not sufficiently utilized.

This work aims to maximize the cooperation capability of a multi-CAV team and the spatial-temporal potentiality of the intersection in a generic AIM problem. To that end, we describe the AIM scheme at a microscopic level through a centralized optimal control problem (OCP), which is about simultaneously planning trajectories for all the CAVs. In the OCP, the CAVs are allowed to change their lanes and adjust the velocities flexibly. Through efficiently solving this OCP offline, we expect to exploit the ideally cooperative performance in a generic AIM scheme.

2. OPTIMAL CONTROL PROBLEM FORMULATION

In this section, an AIM scheme is described as an OCP, which is about minimizing a cost function, subject to several types of constraints.

2.1 Vehicle Kinematics

Suppose there are N_V CAVs simultaneously passing through an intersection. A bicycle model is adopted to describe the mobility of vehicle i $(i = 1,\ldots,N_V)$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_i(t) \\ y_i(t) \\ \theta_i(t) \\ v_i(t) \\ \phi_i(t) \end{bmatrix} = \begin{bmatrix} v_i(t)\cdot\cos\theta_i(t) \\ v_i(t)\cdot\sin\theta_i(t) \\ v_i(t)\cdot\tan\phi_i(t)/L_{Wi} \\ a_i(t) \\ \omega_i(t) \end{bmatrix},\ t\in[0,t_f], \qquad (1)$$

where t_f stands for the fixed completion time, the other variables are defined according to (Li et al., 2018), and L_{Wi} denotes the wheelbase of the ith CAV (Fig. 1).

Boundaries are imposed on some variables throughout $[0,t_f]$:

$$|a_i(t)| \le a_{\max i}, \qquad (2a)$$

$$0 \le v_i(t) \le v_{\max i}, \qquad (2b)$$

$$|\phi_i(t)| \le \Phi_{\max i}, \qquad (2c)$$

$$|\omega_i(t)| \le \Omega_{\max i}. \qquad (2d)$$

Herein, $a_{\max}$, $v_{\max}$, $\Phi_{\max}$, and $\Omega_{\max}$ respectively represents the upper bounds on $|a_i(t)|$, $v_i(t)$, $|\phi_i(t)|$, and $|\omega_i(t)|$.

2.2 Drivable Region Constraints

Each type of CAV is restricted to travel in partial regions of the entire intersection space. Before presenting the details, we classify all the CAVs according to the directions they enter and exit the intersection. As depicted in Fig. 2, 12 types of travel behaviors are defined. Herein, each index set records the IDs of CAVs that fall in those groups.

As an example, the drivable regions of vehicles A1, A2, and

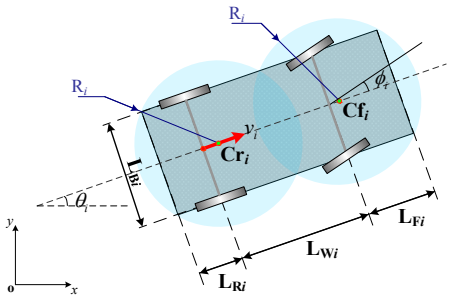

Fig. 1. Kinematic model of a front steering CAV. L_{Bi}, L_{Ri}, L_{Wi}, and L_{Fi} determine the geometric shape of CAV i. We use two discs to evenly cover the rectangular vehicle body. The disc centres are denoted as C_{ri} and C_{fi}.

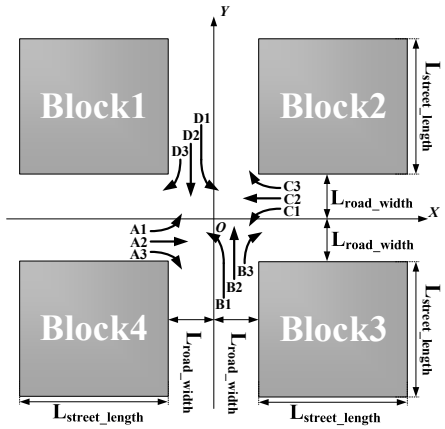

Fig. 2. Schematics on intersection setup and CAV classification. L_{road_width} denotes the length of each road (a road consists of multiple lanes), and L_{street_length} denotes the length of each street block.

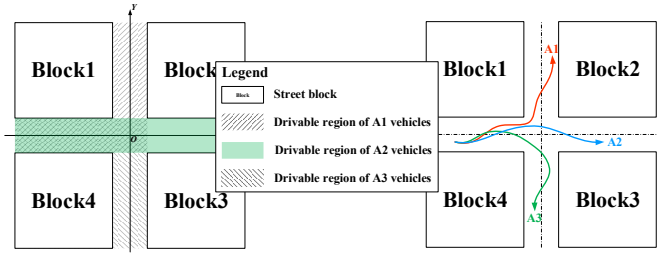

Fig. 3. Drivable regions and allowable trajectories for A1, A2, and A3 vehicles.

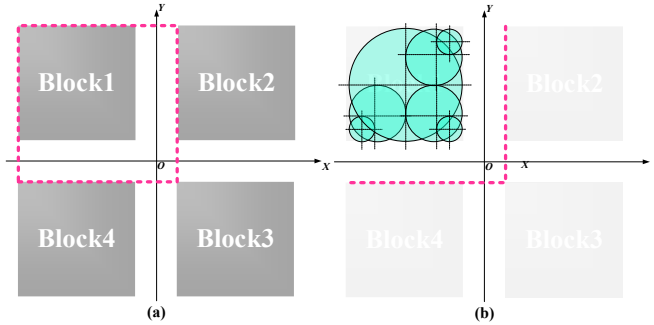

Fig. 4. Schematics on within-drivable-region constraints for A1 vehicles: (a) standard drivable region formulation; (b) simplified street block region via inscribed circles.

A3 are plotted in Fig. 3. A1 vehicles denote the ones that enter from west and exit towards north. The drivable regions of A1 vehicles additionally contain the regions on the opposite roads, which is different from the current traffic laws. The settings for A2 and A3 are also expanded, thus the trajectories shown in Fig. 2 may appear. The expanded regions render extra flexibilities for better cooperative performances of the multi-CAV team.

Next, we would introduce how to formulate the drivable-region constraints. Let us take A1 vehicles as an example. For each CAV $i \in \mathrm{A1}$, staying within the defined drivable regions is identical to 1) vehicle *i* stays within the dashed-line box region plotted in Fig. 4a, and 2) vehicle *i* does not collide with Block 1. Herein, the second condition, which calls for a precise collision-avoidance constraint between a rectangular CAV and a rectangular street block, is overly complicated (Li and Shao, 2015), and thus is simplified via convexification. Concretely, we use a couple of inscribed circles to approximate the rectangular region of Block 1 (Fig. 4b). Also, we use two equal-sized discs to evenly cover the rectangular body of the vehicle (Fig. 1). With such approximations, the collision-avoidance constraint between vehicle *i* and Block 1 becomes that neither disc overlaps with any circle in the environment. In contrast with the rectangle-to-rectangle constraints which are highly nonlinear and almost non-differentiable (Li and Shao, 2015), the convexified circle-to-circle constraints are easier. We use inscribed circles to approximate one street block. The radii of the circles form a geometric series (the common ratio is 0.5), and the centers of these circles can be easily determined offline.

Now that vehicle *i* has been represented by two discs, the within-box-region constraint (see Fig. 4a) is identical to the condition that both disc centers keep above the horizontal line $y = -\mathrm{L}_{\mathrm{road\_width}} + \mathrm{R}_i$, and keep left to the vertical line $x = \mathrm{L}_{\mathrm{road\_width}} - \mathrm{R}_i$, wherein R_i denotes the radius of either disc (Li and Zhang, 2018). The drivable regions for the rest 11 categories can be defined similarly. The details are omitted.

2.3 Collision-Avoidance Constraints

While each CAV travels in its specific drivable regions, it needs to avoid collisions with other moving CAVs as well. This is achieved through requiring that either disc of one CAV does not overlap with the discs of another CAV at every moment during $[0, \mathrm{t_f}]$.

2.4 Initial and Terminal Moment Constraints

At the initial moment $t = 0$, the driving status of CAV *i*, i.e., $[x_i(0), y_i(0), \theta_i(0), v_i(0), a_i(0), \phi_i(0), \omega_i(0)]$, is set as per the ground truth. At the terminal moment $t = \mathrm{t_f}$, each vehicle is required to travel stably for safety:

$$[v_i(\mathrm{t_f}), a_i(\mathrm{t_f}), \phi_i(\mathrm{t_f}), \omega_i(\mathrm{t_f})] = [\mathrm{v_{commom}}, 0, 0, 0],\ i = 1, \ldots, \mathrm{N_V}, \quad (3)$$

where $\mathrm{v_{common}}$ is a common velocity being safe for all the CAVs when they all have exited the intersection. Besides that, each CAV should reach a specific region at the terminal moment. Let us take A1 as an example again, the A1 vehicles should stay within the red corridor marked in Fig. 5, and travel along the direction of the corridor. The terminal region and orientation of the rest 11 categories are defined similarly.

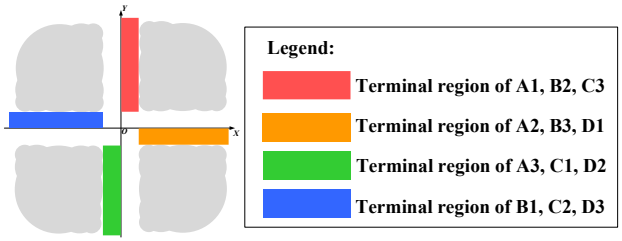

Fig. 5. Terminal corridor for each category of CAVs.

2.5 Cost Function

We expect that the CAVs to travel smoothly during $[0, \mathrm{t_f}]$. This is achieved by minimizing each $|a_i(t)|$ and $|\omega_i(t)|$:

$$J_1 = \int_{t=0}^{\mathrm{t_f}} \left(\sum_{j=1}^{\mathrm{N_V}} \left(a_j^2(t) + \omega_j^2(t) \right) \right) \cdot \mathrm{d}t. \quad (4a)$$

Also, we hope the CAVs would go far in their terminal directions, thus a distance cost function J_2 is formulated:

$$J_2 = -\sum_{i \in \mathrm{A1} \cup \mathrm{B2} \cup \mathrm{C3}} y_i(\mathrm{t_f}) - \sum_{i \in \mathrm{A2} \cup \mathrm{B3} \cup \mathrm{D1}} x_i(\mathrm{t_f}) + \sum_{i \in \mathrm{A3} \cup \mathrm{C1} \cup \mathrm{D2}} y_i(\mathrm{t_f}) + \sum_{i \in \mathrm{B1} \cup \mathrm{C2} \cup \mathrm{D3}} x_i(\mathrm{t_f}). \quad (4b)$$

The overall cost function is a weighted sum of J_1 and J_2:

$$J = J_1 + \mathrm{w} \cdot J_2, \quad (5)$$

wherein $\mathrm{w} \geq 0$ is a weighting parameter that balances the smoothness penalty and the going-far encouragement.

2.6 Overall Formulation

With the aforementioned elements summarized, our concerned multi-CAV cooperative trajectory planning scheme is described as the following OCP:

$$\begin{aligned} &\text{Minimize (5),} \\ &\text{s.t. Kinematic constraints;} \\ &\quad\ \text{Drivable-region constraints;} \\ &\quad\ \text{Boundary constraints;} \\ &\quad\ \text{Collision-avoidance constraints.} \end{aligned} \quad (6)$$

3. NUMERICAL SOLUTION TO OPTIMAL CONTROL PROBLEM

3.1 Basic Solution Procedures

Since an analytical solution to (6) is not available, we aim to find a numerical solution instead. Concretely, (6) is discretized into a nonlinear program (NLP) problem, and then solved by an NLP solver. In forming the NLP problem, a collocation-based approach is adopted, which requires that both the control and state profiles are discretized and taken as the decision variables. A collocation-based strategy has some merits w.r.t. solution stability and high order accuracy

(Biegler, 2010). This work adopts the explicit first-order Runge-Kutta (EFRK) method to convert (6) into an NLP problem.

The interior-point method (IPM) is utilized to solve the formed NLP problem. As a barrier-function based optimizer, IPM converts the constraints into interior-penalty polynomials, which are added to the objective function thereafter, thereby building an unconstraint optimization problem. Through solving that problem with the barrier multiplier iteratively converges to 0^+, a local optimum can be derived finally. Compared with the active-set based NLP solvers, IPM is featured by consistently bearing all the constraints "in the mind", thus is particularly suitable for the NLP problems with really complicated constraints.

In general, an NLP solution process largely relies on the initial guess (also known as the starting point) from which the iteration begins. An initial guess production approach is proposed in the next subsection.

3.2 Initial Guess Generation via A Search*

An initial guess refers to a solution from which an NLP optimization process begins. A near-optimal or even a near-feasible initial guess could largely ease the NLP solution difficulty while a poor initial guess may mislead the convergence towards infeasibility.

An initial guess, in our concerned scheme, is represented by N_V coarse trajectories as well as the corresponding state/control profiles for the CAVs. The coarse trajectories are computed sequentially, wherein each coarse trajectory is produced by an *x-y-time* A* search algorithm. Herein, all the CAVs are prioritized by their expected time to exit the intersection. For each CAV, a lower priority would be set if it is likely to exit the intersection at a later moment. In each coarse trajectory search, the CAVs with already searched trajectories are fixed as moving obstacles in the environment.

The rest part of this subsection elaborates on the principle of the *x-y-time* A* search algorithm. It is an extension of the standard *x-y* A* algorithm by adding the time profile as the third search dimension. The new algorithm differs from the standard A* algorithm in the following aspects: 1) node expansion in the time dimension is strictly monotonous because time flows forward in a uniform pace; 2) the goal node is not a specified point but a manifold with specified $t = \mathrm{t_f}$, which means the vehicle must at least "survive" until the terminal moment $\mathrm{t_f}$; 3) a nominal goal node for each CAV is set as the location it should reach if it travels along its reference line in an empty intersection until $t = \mathrm{t_f}$, then the Manhattan distance function still works to measure the cost-to-go value with such a nominal goal node; and 4) each node additionally records the orientation angle of the vehicle, which is coarsely estimated according to the locations of the current node and its parent in the 2D Cartesian space.

As a preliminary step, the continuous *x-y-time* space is abstracted uniformly in each dimension so as to form a grid map *G*, wherein each grid is called a node. The nodes occupied by the moving/static obstacles are marked. The initial node and a nominal goal node should be specified. Denoting the cost-to-come function as $g(\cdot)$, the cost-to-go function as $h(\cdot)$, the gross cost function as $f = g + h$, and the Manhattan distance function as $L(\cdot, \cdot)$, we present the pseudo-code of *x-y-time* A* search algorithm as follows.

Algorithm 1. *x-y-time* A* Search Algorithm

***Input*:** Gridmap with occupied grids marked, initial node $node_{\text{init}}$, and nominal goal node $node_{\text{end}}$;
Output*:** An initially guessed coarse trajectory ***Traj;

1. Initialize $\text{OPEN} = \varnothing$, and $\text{CLOSED} = \varnothing$;
2. Set $node_{\text{init}}.g = 0$, $node_{\text{init}}.parent = \text{Null}$, calculate $node_{\text{init}}.h$ and $node_{\text{init}}.f$, add $node_{\text{init}}$ to OPEN;
3. Initialize $node_{\text{best\_so\_far}} = \text{Null}$, and $flag = 0$;
4. **while** $(\text{OPEN} \neq \varnothing) \cap (flag == 0)$, **do**
5. Find $node_{\text{current}}$ such that $\arg\min_{node_{\text{current}} \in \text{OPEN}} node_{\text{current}}.f$;
6. Move $node_{\text{current}}$ from OPEN to CLOSED;
7. **for** each expanded child (denoted as $node_{\text{child}}$) of $node_{\text{current}}$, **do**
8. **if** $node_{\text{child}} \in$ CLOSED, **then**
9. **continue;**
10. **end if**
11. **if** $node_{\text{child}} \in$ OPEN, **then**
12. Calculate $g^* = node_{\text{current}}.g + L(node_{\text{current}}, node_{\text{child}})$;
13. **if** $g^* < node_{\text{child}}.g$, **then**
14. Reset $node_{\text{child}}.parent = node_{\text{current}}$, $node_{\text{child}}.g = g^*$;
15. Update $node_{\text{child}}.f$ and $node_{\text{child}}.\theta$;
16. Update $node_{\text{child}}$ in OPEN;
17. **end if**
18. **else**
19. Calculate $node_{\text{child}}.g$, $node_{\text{child}}.h$, $node_{\text{child}}.f$, $node_{\text{child}}.\theta$;
20. **if** $node_{\text{child}}$ is subject to collisions, **then**
21. Move $node_{\text{child}}$ to CLOSED;
22. **continue;**
23. **else**
24. Add $node_{\text{child}}$ to OPEN;
25. **if** $node_{\text{child}} == node_{\text{end}}$ or $node_{\text{child}}.time == \mathrm{t_f}$, **then**
26. Set $flag = 1$ and $node_{\text{best\_so\_far}} = node_{\text{child}}$;
27. **break;**
28. **end if**
29. **end if**
30. **end if**
31. **end for**
32. **end while**
33. Backtrack the ancestors of $node_{\text{best\_so\_far}}$ recursively until Null is found, inversely place them to form ***Traj***, and then output ***Traj***;
34. **return.**

3.3 NLP Solution Facilitation Strategy

The NLP solution process begins from the initial guess derived by the preceding subsection. A strategy is adopted in this subsection for further easing the NLP solution difficulties.

Intuitively, a generic way to handle a difficult optimization problem is to decompose it into easier ones. In our concerned cooperative trajectory planning task, the primary difficulties lie in the large-scale collision-avoidance constraints. We

temporarily remove all the vehicle-to-vehicle collision-avoidance constraints and add them back incrementally until a feasible solution is derived. The detailed procedures are presented in (Li et al., 2019).

3.4 Overall AIM Method

As a summary of the aforementioned two sections, the complete AIM principles are depicted in Fig. 6.

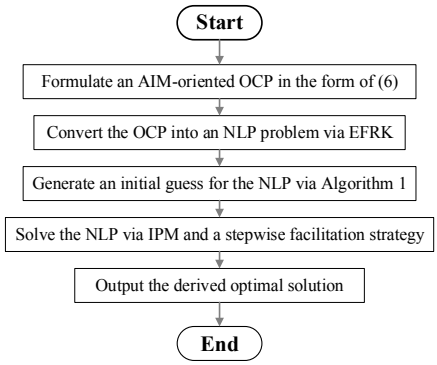

Fig. 6. Flowchart of the proposed AIM method.

4. SIMULATION RESULTS AND DISCUSSIONS

Simulations were performed in MATLAB 2019a and executed on an i5-7200U CPU with 32 GB RAM that runs at 2.50×2 GHz. A benchmark set containing 50 cases is formed. In each case, the CAVs' initial configurations and driving intensions are randomly specified. Basic parametric settings are listed in Table 1. More details about the parametric settings and the benchmark case setups are provided at https://github.com/libai1943/AIM_COCP.

Table 1. Parametric settings for model and approach.

Parameter(s)	Description	Setting(s)
N_V	Number of CAVs	24
L_{Wj}, L_{Rj}, L_{Fj}, L_{Bj}	Geometric size of each CAV ($j=1,...,N_V$)	2.80 m, 0.929 m 0.96 m, 1.942 m
t_f	Specified terminal moment	10.0 s
$a_{max\,j}$, $v_{max\,j}$, $\Phi_{max\,j}$, $\Omega_{max\,j}$	Bounds on $\lvert a_j(t)\rvert$, $v_j(t)$, $\lvert\phi_j(t)\rvert$, and $\lvert\omega_j(t)\rvert$	2 m/s^2, 25 m/s 0.7 rad, 0.3 rad/s
v_{common}	Common terminal velocity for each CAV	20 m/s
L_{road_width}, L_{street_length}	Road width and street block length	12 m, 176 m
w	Weight in (5)	1.0
N_{fe}	Number of finite elements in forming the NLP problem	100

According to our simulations, 82% out of the entire 50 benchmark AIM problems are solved successfully by our proposed method, and the average CPU time for each problem is 2274.03 sec. There may be potentials to further promote the solution capability and time efficiency.

The NLP facilitation approach adopted in Section 3.3 is efficacious to ease the computational difficulties. Among the successfully solved benchmark problems, 6.5% of the vehicle-to-vehicle collision-avoidance constraints are safely discarded when the optimum is derived on average (max. 67%, min. 1%). By contrast, the success rate to solve the benchmark problems declines from 82% to 36% if the NLP solution facilitation strategy is disabled.

As depicted in Fig. 3, the drivable regions for the left-turn, through, and right-turn vehicles are expanded in our formulated AIM-oriented optimal control problem. To investigate the effect of expanding the drivable regions, we define a comparative algorithm (denoted as Algorithm 2) which is the same with our proposal except that the drivable regions are set according to Fig. 7, and use Algorithm 2 to solve the 50 benchmark problems. Consequently, the average throughput grows from 3.3858 to 3.4572 sec, which means the expanding the drivable regions leads to better cooperative driving performances.

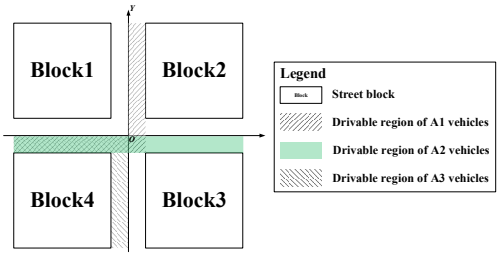

Fig. 7. Drivable region settings for A1, A2, and A3 vehicles in a comparative algorithm.

A video containing the typical simulation results is provided at https://www.bilibili.com/video/BV1DA411b7Q2/. As an example, the initial guess, optimized cooperative trajectories derived by this work and Algorithm 2 for Case #50 are illustrated in Figs. 8–10, respectively.

5. CONCLUSIONS

This paper has introduced a computational optimal control based autonomous intersection management (AIM) method, which is featured by regarding the intersection as a continuous free-space and expanding the vehicles' drivable regions for improvements in the travel efficiency. Although our proposed AIM method only provides offline solutions, they are useful to measure the solution optimality of any online AIM method. Also, the appearances of the offline solutions may inspire the proposal of smart online AIM methods in the future.

ACKNOWLEDGMENTS

This work was supported by the Fundamental Research Funds for the Central Universities, the Natural Sciences and Engineering Research Council of Canada, and the National Key R&D Program of China under Grant 2016YFB0100903-2.

REFERENCES

Biegler, L. T. (2010). Nonlinear programming: Concepts, algorithms, and applications to chemical processes. SIAM, 10.

Dresner, K., and Stone, P. (2008). A multiagent approach to

autonomous intersection management. *Journal of Artificial Intelligence Research*, 31, 591–656.

Kamal, M. A. S., Imura, J. I., Ohata, A., Hayakawa, T., & Aihara, K. (2013). Coordination of automated vehicles at a traffic-lightless intersection. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)* (pp. 922–927). IEEE.

Levin, M. W., & Rey, D. (2017). Conflict-point formulation of intersection control for autonomous vehicles. *Transportation Research Part C: Emerging Technologies*, 85, 528–547.

Li, B., & Shao, Z. (2015). A unified motion planning method for parking an autonomous vehicle in the presence of irregularly placed obstacles. *Knowledge-Based Systems*, 86, 11–20.

Li, B., & Zhang, Y. M. (2018). Fault-tolerant cooperative motion planning of connected and automated vehicles at a signal-free and lane-free intersection. *IFAC-Papers OnLine*, 51(24), 60–67.

Li, B., Zhang, Y. M., Zhang, Y., Jia, N., & Ge, Y. (2018). Near-optimal online motion planning of connected and automated vehicles at a signal-free and lane-free intersection. In *2018 IEEE Intelligent Vehicles Symposium (IV)* (pp. 1432–1437). IEEE.

Li, B., Jia, N., Li, P., & Li Y. (2019). Incrementally constrained dynamic optimization: A computational framework for lane change motion planning of connected and automated vehicles. *Journal of Intelligent Transportation Systems*, 23(6), 557–568.

Lin, P., Liu, J., Jin, P. J., & Ran, B. (2017). Autonomous vehicle-intersection coordination method in a connected vehicle environment. *IEEE Intelligent Transportation Systems Magazine*, 9(4), 37–47.

Liu, B., Shi, Q., Song, Z., & El Kamel, A. (2019). Trajectory planning for autonomous intersection management of connected vehicles. *Simulation Modelling Practice and Theory*, 90, 16–30.

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*, 93, 244–256.

Mirheli, A., Hajibabai, L., & Hajbabaie, A. (2018). Development of a signal-head-free intersection control logic in a fully connected and autonomous vehicle environment. *Transportation Research Part C: Emerging Technologies*, 92, 412–425.

Mirheli, A., Tajalli, M., Hajibabai, L., & Hajbabaie, A. (2019). A consensus-based distributed trajectory control in a signal-free intersection. *Transportation Research Part C: Emerging Technologies*, 100, 161–176.

Rios-Torres, J., & Malikopoulos, A. A. (2016). A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps. *IEEE Transactions on Intelligent Transportation Systems*, 18(5), 1066–1077.

Xu, B., Li, S. E., Bian, Y., Li, S., Ban, X. J., Wang, J., & Li, K. (2018). Distributed conflict-free cooperation for multiple connected vehicles at unsignalized intersections. *Transportation Research Part C: Emerging Technologies*, 93, 322–334.

Zohdy, I. H., Kamalanathsharma, R. K., & Rakha, H. (2012). Intersection management for autonomous vehicles using iCACC. In *2012 15th International IEEE Conference on Intelligent Transportation Systems* (pp. 1109-1114). IEEE.

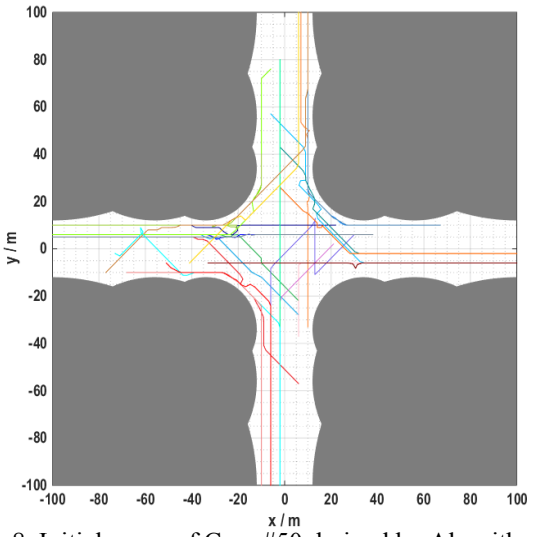

Fig. 8. Initial guess of Case #50 derived by Algorithm 1.

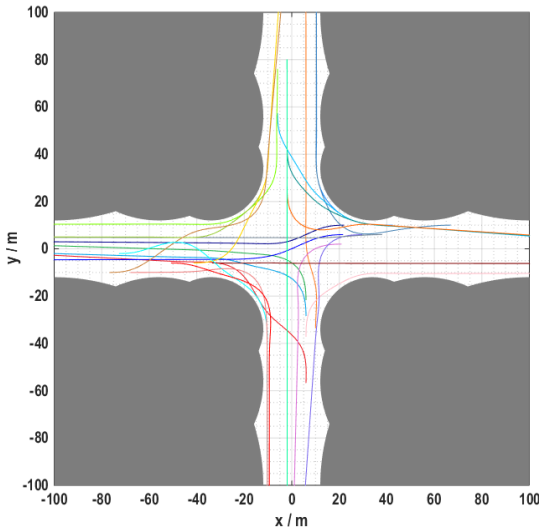

Fig. 9. Optimized cooperative trajectories of Case #50 derived by the proposed method in this work.

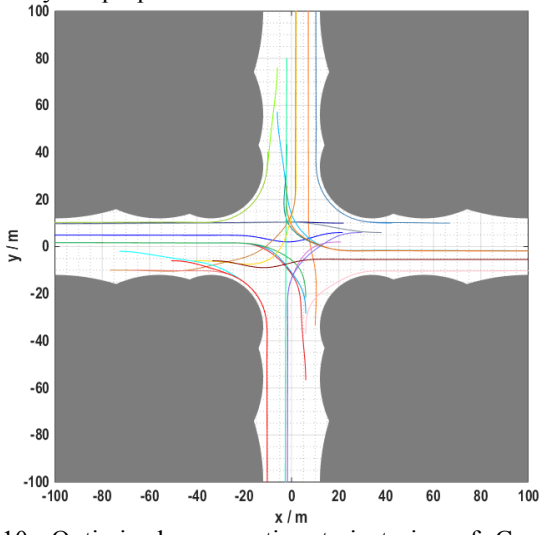

Fig. 10. Optimized cooperative trajectories of Case #50 derived by Algorithm 2.