

# Safe Trajectory Generation for Complex Urban Environments Using Spatio-temporal Semantic Corridor

Wenchao Ding<sup>1†</sup>, Lu Zhang<sup>1†</sup>, Jing Chen<sup>2</sup>, and Shaojie Shen<sup>1</sup>

**Abstract**—Planning safe trajectories for autonomous vehicles in complex urban environments is challenging since there are numerous semantic elements (such as dynamic agents, traffic lights and speed limits) to consider. These semantic elements may have different mathematical descriptions such as obstacle, constraint and cost. It is non-trivial to tune the effects from different combinations of semantic elements for a stable and generalizable behavior. In this paper, we propose a novel unified spatio-temporal semantic corridor (SSC) structure, which provides a level of abstraction for different types of semantic elements. The SSC consists of a series of mutually connected collision-free cubes with dynamical constraints posed by the semantic elements in the spatio-temporal domain. The trajectory generation problem then boils down to a general quadratic programming (QP) formulation. Thanks to the unified SSC representation, our framework can generalize to any combination of semantic elements. Moreover, our formulation provides a theoretical *guarantee* that the *entire* trajectory is safe and constraint-satisfied, by using the convex hull and hodograph properties of piecewise Bézier curve parameterization. We also release the code of our method to accommodate benchmarking.

**Index Terms**—Autonomous vehicle navigation, motion and path planning

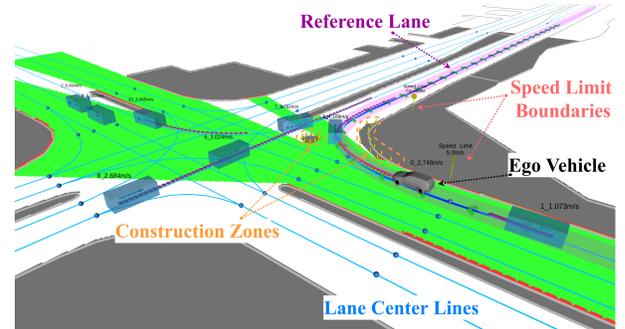
## I. INTRODUCTION

**T**RAJECTORY generation for autonomous vehicles (AVs) in complex urban environments is challenging since there are many semantic elements (e.g., dynamic agents, traffic lights, speed limits, stop signs, and lane geometry). Different types of semantic elements may have different mathematical descriptions such as obstacle, constraint and cost [1]. It is non-trivial to tune the effects from different combinations of semantic elements so that the formulation can generalize well to all combinations of semantic elements [2]. Therefore, it is essential to describe diverse kinds of semantic elements in a unified way such that the type and combination of semantic elements do not affect the planning performance.

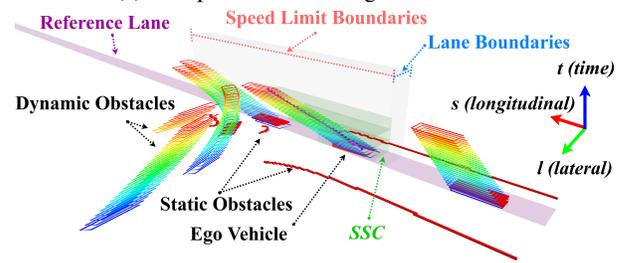
Accepted final version. To Appear in IEEE Robotics and Automation Letters. ©2019 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses. This work was supported by Hong Kong PhD Fellowship Scheme, HKUST-DJI Joint Innovation Laboratory, and HKUST Institutional Fund. (*Corresponding author: Wenchao Ding.*)

<sup>†</sup>W. Ding and L. Zhang contributed equally to this work. <sup>1</sup>W. Ding, L. Zhang, and S. Shen are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong 852, China (e-mail: wdingae@ust.hk; lzhangbz@ust.hk; eeshaojie@ust.hk).

<sup>2</sup>J. Chen is with DJI Technology Co., Ltd., Shenzhen 510810, China (e-mail: jing.chen@dji.com).



(a) Complex urban driving environments



(b) Spatio-temporal semantic corridor (SSC)

Fig. 1: Illustration of our trajectory generation framework. Complex semantic elements of the environment are projected to the spatio-temporal domain w.r.t. the reference lane. The SSC encodes the requirements given by the semantic elements and a safe trajectory is generated accordingly. Note that the visualization of the static obstacles is clipped to show the details of other components. More examples can be found in the video <https://www.youtube.com/watch?v=LrGmKaM3Iqc>.

Apart from the representation issue of the semantic elements, another issue is how to *guarantee* the safety and feasibility of the generated trajectory. Most existing optimization-based [3, 4] and lattice-based [5]–[7] motion planners try to check or enforce constraints at a series of sampled points. However, these methods may fail to detect or resolve infeasible points between two sample points, and thus cannot provide safety guarantee for the entire trajectory.

To overcome the above challenges, we propose a unified trajectory generation framework with a theoretical safety and feasibility guarantee. The key to the framework is a novel spatio-temporal semantic corridor (SSC) structure. The SSC is motivated by the fact that most semantic elements can be either rendered as spatio-temporal obstacles or constraints within a certain range of the spatio-temporal domain. The key feature of the SSC is its abstraction for different types of

semantic elements. Essentially, the SSC consists of a series of mutually connected collision-free cubes with dynamical constraints posed by the semantic elements. We propose an SSC generation process to generate and split the cubes so that the dynamical constraints can be correctly associated.

Given the unified SSC representation, the trajectory generation problem boils down to generating the optimal trajectory within the SSC while satisfying the dynamical constraints. In this paper, we contribute a quadratic programming (QP) formulation which guarantees the safety and feasibility of the generated trajectory by using piecewise Bézier curve parameterization. The proposed formulation is built on the top of the convex hull and hodograph properties of the Bézier curve. The contributions are summarized as follows:

- An SSC structure which provides a unified representation for diverse kinds of semantic elements in complex urban environments.
- An optimization-based trajectory generation formulation which ensures safety and feasibility for the entire generated trajectory.
- A complete and open-source trajectory generation framework and real-time implementation in a multi-agent urban simulation platform. Comprehensive experiments and comparisons are presented to validate the performance.

The related literature is reviewed in Sect. II. An overview of our trajectory generation framework is provided in Sect. III. Our SSC generation method and trajectory generation method are detailed in Sect. V and Sect. VI, respectively. Experimental results and benchmark analysis are elaborated in Sect. VII. Finally, a conclusion is drawn in Sect. VIII.

## II. RELATED WORKS

### A. Spatio-temporal motion planning for AVs

There is extensive literature on spatio-temporal motion planning for autonomous vehicles. Ziegler *et al.* [5] sample a spatio-temporal state lattice on a space-time manifold [8] and use a search-based approach to obtain an executable trajectory. McNaughton *et al.* [6] adopt a spatio-temporal state lattice, which can automatically conform to the lane geometry by numerical optimization. Due to an unacceptable blowup in the size of the search space (i.e., the *curse of dimensionality*), GPU-accelerated dynamic programming is adopted in [6]. However, the semantic elements in urban environments (such as speed limits, traffic lights, etc.) are not modeled in [5, 6, 8].

There are several approaches that attempt to model the semantic elements. Wolf *et al.* [9] associate semantic elements with specially designed cost functions and aggregate the cost terms as a potential field. However, this approach suffers from local minimums. Moreover, it is non-trivial to correctly balance the effects of different cost terms for different configurations of the semantic elements [2]. Hubmann *et al.* [10] render traffic lights and dynamic agents as obstacles in the longitudinal and time domain and apply a search-based method to obtain a generic driving strategy (i.e., a speed plan). Ajanovic *et al.* [1] extend the obstacle representation and render forbidden lane changes and solid lines as obstacles and speed limits as velocity constraints.

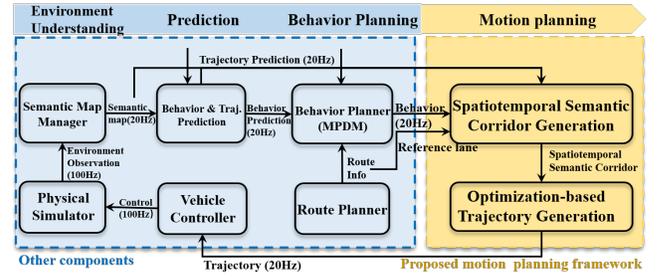


Fig. 2: Illustration of the proposed trajectory generation framework and its relationship with other system components.

Built on top of the obstacle-like and constraint-like representations in [1], we further propose the SSC structure to generally represent different types of semantic elements. The key feature of the SSC is that it provides a level of abstraction which encodes all the information needed for later optimization. Adding a new semantic element or combining different semantic elements does not affect the cost formulation and constraint specification, which renders a unified and generalizable trajectory generation framework.

### B. Corridor generation for AVs

The spatial corridor (i.e., convex free-space) is widely applied in trajectory generation. Zhu *et al.* [11] propose a convex elastic smoothing algorithm, which can generate a collision-free “tube” around the initial path and formulate the trajectory smoothing problem into a quadratically constrained quadratic programming (QCQP). Erlien *et al.* [12] consider not only spatial information but also vehicle dynamics to construct the convex tube. Both of these works, however, generate the corridor in a static environment and cannot deal with dynamic obstacles. Liu *et al.* [13] find a convex feasible set around the reference trajectory and leverage the convex feasible set to accelerate the non-convex optimization. However, the computation complexity is still prohibitively high for real-time applications. Moreover, collision-avoidance is their major concern and semantic elements are not considered. We are motivated by these corridor generation methods and further extend the spatial corridor to the spatio-temporal domain to cope with dynamic obstacles. Additionally, the proposed SSC can take various kinds of semantic elements into account.

## III. SYSTEM OVERVIEW

The proposed trajectory generation framework (Fig. 2) belongs to the motion planning layer of an autonomous vehicle, and it requires necessary inputs from the upper layers, e.g., the behavioral layer. Apart from the proposed trajectory generation, the other system components are also illustrated to clarify the input and output of our framework.

As depicted in Fig. 2, there are four phases for a single planning cycle. The first phase is the environment understanding obtained by a semantic map manager which takes the responsibility to manage the semantic elements (e.g., occupancy grid map, dynamic agents, lanes, traffic rules, etc.) for local planning purposes. The second phase is the prediction, which

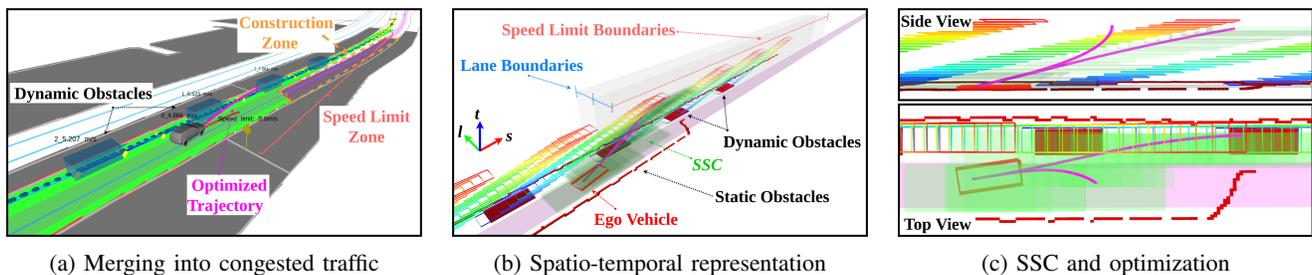


Fig. 3: Illustration of merging into congested traffic under a speed limit. For the two potential behaviors, i.e., lane change and lane keeping, the optimal trajectories are generated inside the SSC for each behavior.

not only provides high-level behavior anticipations (e.g., lane change, lane keeping, etc.) but also predicted trajectories for other dynamic agents. The third phase is the behavior planning, which is implemented using the multi-policy decision making (MPDM) method, as elaborated in Sect. IV. The fourth phase is our proposed motion planning, which takes discretized future simulated states from the behavior planner as seeds for corridor generation. Note that our trajectory generation framework can also work with other behavior planners, such as those in [10, 14, 15], as long as the behavior planner provides a preliminary initial guess about the future states.

To construct the SSC, four ingredients are needed, namely, a semantic map which consists of the semantic elements, predicted trajectories for dynamical agents, forward simulated states, and a reference lane given by the route information. Note that the trajectory prediction module may be optional if the forward simulated states already include the states for other agents such as the case of MPDM. In such case, we can use the simulated states of other vehicles as the predicted trajectories, which facilitates passing interaction anticipations from behavior planning to motion planning layer. However, since this is not a common feature in behavior planning, we still use the predicted trajectories from the trajectory prediction module in the experiments for generality, which may lose the interaction information from behavior planning. To summarize, the source of the seeds and the modeling of interaction depend on the choice of behavior planner.

#### IV. PRELIMINARIES ON MULTI-POLICY DECISION MAKING

In this paper, we adopt MPDM [16] as the behavioral layer. Recall that our trajectory generation method can also work with other behavior planning methods [10, 14, 15]. Since behavior planning is out of the scope of this paper, only preliminary information about MPDM is provided here.

The MPDM model formulates the behavior planning problem as a general multi-agent partially observable Markov decision process (POMDP) to model the interaction and uncertainty in dynamic environments. Since solving the POMDP quickly becomes computationally intractable when the number of vehicles increases, MPDM relaxes the problem and assumes that both our vehicle and the other agents are executing a finite set of closed-loop discrete policies (e.g., lane change, lane keeping, etc.). Moreover, for each closed-loop policy, the future situation is anticipated via forward simulating all the vehicle states using a simplified simulation model, such as

an idealized steering and speed controller. A comprehensive reward function is designed to assess the future situation and the best behavior is elected.

In this paper, we use the forward simulated states of the ego vehicle as the seeds in the corridor generation process. Although the initial seeds are collision-free, they can not be directly executed by the vehicle due to a coarse resolution (0.15 s in the experiments) and a simplified simulation model (e.g., piecewise linear control in the experiments).

Since MPDM provides the forward simulated states for multiple behaviors (e.g., lane change left, lane change right, and lane keeping) at the same time, we fully utilize this feature and generate candidate trajectories for all the potential behaviors to enhance the robustness of the framework. For example, while executing a lane change trajectory, our trajectory framework always prepares the trajectory for switching back to the original lane, as shown in Fig. 3.

#### V. SPATIO-TEMPORAL SEMANTIC CORRIDOR

##### A. Semantic Elements And Frenét Frame Representation

We deal with an  $slt$  3-D configuration space which consists of the longitudinal direction  $s$ , the lateral direction  $l$  and the time  $t$ . The longitudinal and lateral directions are with respect to a Frenét frame, which is a dynamical reference frame constructed from the reference lane. Typically, the reference lane is extracted from the route information provided by a route planner, as illustrated in Fig. 2. For an unstructured environment where there is no lane available, the reference lane can also be provided by a path planner [17].

Rather than generating the corridor in Cartesian coordinates, we adopt the Frenét frame representation since most of the semantic elements are associated with the lane geometry. For example, speed limits, traffic lights and stop signs are typically associated with a certain longitudinal range of a lane. Moreover, since human-like driving behavior can typically be decoupled into lateral movements and longitudinal movements, modeling the free-space in these two directions is a more natural representation than modeling free-space in Cartesian coordinates. Time is another necessary dimension since many semantic elements are time-indexed. For instance, the predicted trajectory is time-profiled and can be regarded as a series of spatio-temporal obstacles.

Two typical examples of projecting the semantic elements to a Frenét frame are depicted in Fig. 1 and Fig. 3, respectively. Diverse kinds of semantic elements can be generally divided

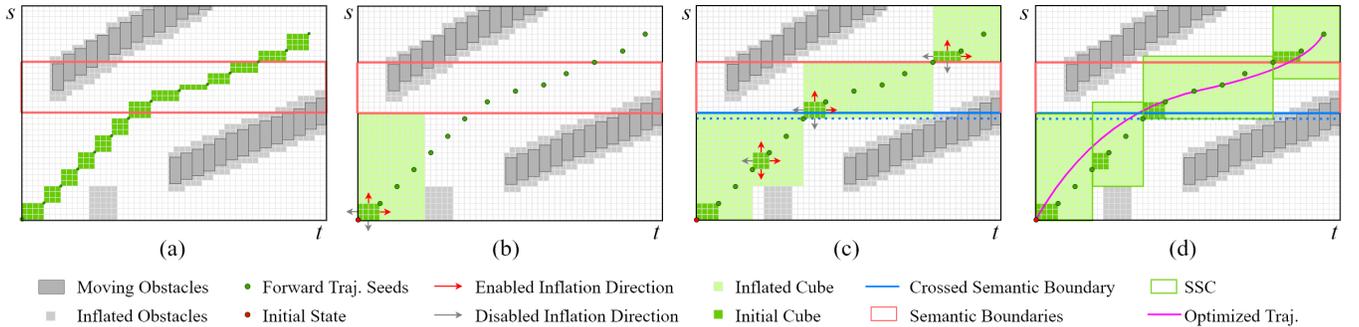


Fig. 4: Illustration of a toy example of the SSC generation algorithm in the  $st$  domain. There is a speed limit which takes effect between the two *orange* boundaries, as shown in (a). To begin, the first initial cube is inflated until the two inflation directions touch the semantic boundary and the obstacle, as shown in (b). Next, the last seed in the first cube and the first seed outside the first cube are picked out to construct the second initial cube, as shown in (c). The inflation for the second initial cube terminates at the semantic boundary. Then for the third initial cube, the inflation direction opposite to the entry direction is disabled. After the cube inflation, a cube relaxation process is applied depending on the constraints associated and the free-space, as shown in (d).

into two categories: obstacle-like and constraint-like semantic elements. We elaborate on this in the following.

1) *Obstacle-like semantic elements*: Many semantic elements have the physical meaning that a certain portion of the  $slt$  domain is not allowed to be driven in. For example, static obstacles can be viewed as obstacles across whole time axes, and dynamic obstacles can be viewed as a series of static obstacles in the time domain according to the predicted trajectory, while a red light can be rendered as an obstacle occupying a particular longitudinal position and time period. After rendering obstacle-like semantic elements to the  $slt$  domain, the configuration space is a 3-D occupancy grid.

2) *Constraint-like semantic elements*: Apart from the obstacle-like semantic elements, many semantic elements represent dynamical constraints or time constraints. For example, speed limits and stop signs can be viewed as velocity constraints. There are also semantic elements which pose time constraints. For instance, when crossing lanes, the total time of the lane change should not be unreasonably long.

We propose a unified representation, i.e., *semantic boundaries*, for all the constraint-like semantic elements. For instance, a speed limit can be regarded as the velocity constraint applied to a longitudinal range  $[s_{\text{begin}}, s_{\text{end}}]$ , where  $s_{\text{begin}}$  and  $s_{\text{end}}$  are the two semantic boundaries. The lane change constraint can be regarded as a time constraint applied to the lateral range  $[d_{\text{begin}}, d_{\text{end}}]$  of the current lane. Essentially, the semantic boundaries represent where a certain semantic element starts and stops taking effect.

Note that there is a minor difference in terms of the “hardness” of the constraints. Specifically, the constraints posed by traffic rules (e.g., speed limit) are hard constraints which should be followed without any compromise. Other constraints (e.g., lane change duration constraint) are required for a natural human-like behavior and there is no universal quantitative description of such constraints. We take the difference into account during the corridor generation process (Sect. V).

## B. Semantic Corridor Generation

As outlined in Algo. 1, the generation process consists of seed generation (Line 3), cube inflation (Line 4), constraint association (Line 5) and cube relaxation (Line 6).

---

### Algorithm 1: Semantic Corridor Generation

---

- 1 Inputs: forward simulated states  $\{x_0, x_1, \dots, x_t\}$ , initial state  $x_{\text{des}}$ , semantic boundaries  $\mathcal{B}$ ,  $slt$  configuration space  $\mathcal{E}$ ;
  - 2 Initializes: seeds  $\mathcal{S}^{\text{seed}} = \emptyset$ ;
  - 3  $\mathcal{S}^{\text{seed}} \leftarrow \text{SeedGeneration}(\{x_0, x_1, \dots, x_t\}, x_{\text{des}})$ ;
  - 4  $\mathcal{C}^{\text{infl}} \leftarrow \text{CubeInflation}(\mathcal{S}^{\text{seed}}, \mathcal{B}, \mathcal{E})$ ;
  - 5  $\mathcal{C}^{\text{infl}} \leftarrow \text{ConstraintAssociation}(\mathcal{C}^{\text{infl}}, \mathcal{B})$ ;
  - 6  $\mathcal{C}^{\text{final}} \leftarrow \text{CubeRelaxation}(\mathcal{C}^{\text{infl}}, \mathcal{E})$ ;
- 

1) *Seed Generation*: The seeds of the semantic corridor are generated by projecting the forward simulated states of the behavior planner to the  $slt$  configuration space. Since the forward simulated states are discretized, the feasibility of the corridor generation process depends on the complexity of environments and seed resolution. To guarantee the success of the corridor generation process, we require the initial cubes constructed from consecutive seeds to be collision-free (Fig. 4 (a) and Line 6, Algo. 2). In practice, this clearance requirement is reasonable and easy to achieve. For example, for a vehicle travelling at a longitudinal speed of 30  $m/s$  and a seed resolution of 0.15  $s$  (similar to [16]), the clearance required is roughly 4.5  $m$ , which is much shorter than the emergency braking distance at such a high speed. Therefore, it is reasonable to directly reject the cases which violate the proposed requirement.

The motivation for generating the corridor around the seeds is to fully model topologically equivalent free space, while preserving the same high-level behavior. For example, as shown in Fig. 4 (a), the semantic meaning of the seeds is to pass between the two dynamic obstacles, which is preserved by the corridor generation. Since the motion planner should work with any given initial state, the initial state should also be included in the seeds.

2) *Cube Inflation with Semantic Boundaries*: The corridor is generated by iterating over the seeds. The seeds which are already contained in the last inflated cube are skipped (Line 4, Algo. 2) since they are topologically equivalent. The initial cubes are generated based on two consecutive seeds, by

**Algorithm 2:** CubeInflation( $\mathcal{S}^{\text{seed}}, \mathcal{B}, \mathcal{E}$ )

---

```

1 Inputs: cube seeds  $\mathcal{S}^{\text{seed}}$ , semantic boundaries  $\mathcal{B}$ ,  $slt$ 
  configuration space  $\mathcal{E}$ ;
2 Initializes: inflated cubes  $\mathcal{C}^{\text{infl}} = \emptyset$ ;
3 for  $i = 2, \dots, |\mathcal{S}^{\text{seed}}|$  do
4   if !IfContainedInLastCube( $s_i^{\text{seed}}, \mathcal{C}^{\text{infl}}$ ) then
5      $c \leftarrow \text{GetInitialCubeBySeed}(s_i^{\text{seed}}, s_{i-1}^{\text{seed}})$ ;
6     if !IfInitialCubeFree( $c, \mathcal{E}$ ) then
7       return;
8     end
9      $\mathcal{D} \leftarrow \text{GetInflDirsBySemBoundaries}(c, \mathcal{B})$ ;
10     $c^{\text{infl}} \leftarrow \text{InflateCubeInDirs}(c, \mathcal{D}, \mathcal{B}, \mathcal{E})$ ;
11     $\mathcal{C}^{\text{infl}} \leftarrow \mathcal{C}^{\text{infl}} \cup c^{\text{infl}}$ 
12  end
13 end

```

---

regarding the two seeds as two cube vertices (Line 5, Algo. 2).

The key feature of the cube inflation is the consideration of the semantic boundaries (Line 9, Algo. 2). The goal of the cube inflation process is to generate cubes which match the semantic boundaries so that the constraints can be conveniently associated. Specifically, when the initial cube intersects with a certain semantic boundary, the inflation direction opposite to the entry direction is disabled, so that the inflated cube can almost match the semantic boundaries. The inflation alternates among three  $slt$  directions for one step of inflation and terminates if this step collides with an obstacle or intersects with a certain semantic boundary. A toy example is provided in Fig. 4 (b) and (c). Since in the optimization (Sect. VI) each cube corresponds to one piece of the trajectory and to preserve convexity we do not explicitly optimize the durations of the pieces, the time upper bound of the current cube should coincide with the time lower bound of the next cube. One may consider optimizing the durations (which is non-convex) and in such case, a further inflation to increase overlapping in the  $t$  dimension can be beneficial.

3) *Cube Relaxation*: After the cube inflation process, the inflated cubes almost match the semantic boundaries, as shown in Fig. 4 (c). However, as mentioned in V-A2, some constraints, such as the lane change duration constraint, are soft and extra space should be left for optimization. To this end, we adopt a cube relaxation process to relax the cube boundaries while preserving the hard constraints and collision-free property, as shown in Fig. 4 (d). The maximum margin allowed for the relaxation is systematically determined by the constraints applied to the two consecutive cubes. For example, in the longitudinal direction, the margin can be determined by velocity matching distance according to the velocity constraints. For the lateral direction (i.e., the lane change case), the margin can be calculated by the allowed fluctuation of lane change duration.

## VI. TRAJECTORY GENERATION WITH SAFETY AND FEASIBILITY GUARANTEE

Given the constraints specified by the SSC, we present an optimization-based trajectory generation method which can find the optimal trajectory within the SSC while satisfying the

dynamical constraints. The optimization problem is also formulated in the Frenét frame, which is consistent with the SSC representation. In [18], Werling *et al.* use a quintic monomial polynomial for both the longitudinal and lateral direction based on the optimal control theory. However, the quintic monomial polynomial is not suitable for the optimization in the SSC for the following two reasons: 1) one segment of the polynomial only has limited representation ability and may fail to represent a highly constrained maneuver required by the SSC, and 2) the monomial basis polynomial is not well suited to problems with complex configuration space obstacles and dynamical constraints. In previous works on monomial basis polynomial trajectories [18, 19], the constraints are only enforced/checked on a finite set of sampled points. However, this method may fail to detect collision between sample points, and thus cannot provide any guarantee on safety and feasibility.

In this paper, we remove the above two limitations by using a piecewise Bézier curve for the two-dimensional trajectory (i.e., the longitudinal direction  $s(t)$  and lateral direction  $l(t)$ ) along the reference lane. The reason for using the piecewise Bézier curve is its convex hull property and hodograph property [20].

### A. Bézier Basis and Its Properties

A degree- $m$  Bézier curve  $f(t)$  is defined on a fixed interval  $t \in [0, 1]$  by  $m + 1$  control points as follows,

$$f(t) = p_0 b_m^0(t) + p_1 b_m^1(t) + \dots + p_m b_m^m(t) = \sum_{i=0}^m p_i \cdot b_m^i(t), \quad (1)$$

where  $p_i$  denotes the control point and  $b_m^i(t) = \binom{m}{i} t^i \cdot (1-t)^{m-i}$  is the Bernstein basis. Denote the set of control points  $[p_0, p_1, \dots, p_m]$  as  $\mathbf{p}$ .

The *convex hull* property is suitable for the problem of constraining the curve in a convex free-space. Specifically, the Bézier curve  $f(t)$  is guaranteed to be entirely confined in the convex hull supported by the control points  $\mathbf{p}$ . In other words, by constraining  $\mathbf{p}$  inside the convex free-space, the resulting curve is guaranteed to be collision-free.

The *hodograph* property facilitates constraining high-order derivatives of the Bézier curve, which is useful for enforcing dynamical constraints. By the hodograph property, the derivative of a Bézier curve  $\frac{df(t)}{dt}$  is another Bézier curve with control point  $p_i^{(1)} = m \cdot (p_{i+1} - p_i)$ . By applying the convex hull property on the derivative Bézier curve, the entire dynamical profile of the original curve  $f(t)$  can be confined within a given dynamical range, as shown in Fig. 5.

### B. Piecewise Bézier Curve Representation

In this paper, we adopt a piecewise Bézier curve representation with each piece associated with one cube of the SSC. Accordingly, the  $j$ -th segment of an  $n$ -segment piecewise Bézier trajectory in one dimension  $\sigma \in \{s, l\}$  is given by

$$f_j^\sigma(t) = \begin{cases} \alpha_1 \cdot \sum_{i=0}^m p_i^1 \cdot b_m^i\left(\frac{t-t_0}{\alpha_1}\right), & t \in [t_0, t_1] \\ \alpha_2 \cdot \sum_{i=0}^m p_i^2 \cdot b_m^i\left(\frac{t-t_1}{\alpha_2}\right), & t \in [t_1, t_2] \\ \vdots & \vdots \\ \alpha_n \cdot \sum_{i=0}^m p_i^n \cdot b_m^i\left(\frac{t-t_{n-1}}{\alpha_n}\right), & t \in [t_{n-1}, t_n], \end{cases} \quad (2)$$

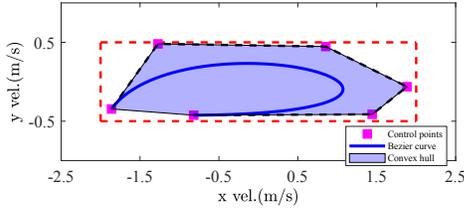


Fig. 5: Illustration of using the convex hull property to constrain a velocity profile inside a feasible region (dashed red lines).

where  $p_i^j$  denotes the  $i$ -th control point of the  $j$ -th segment and  $t_0, t_1, \dots, t_n$  are the time stamps of the start point and end point for each segment. Since the Bézier curve is defined on the fixed interval  $[0, 1]$  while the trajectory duration for each segment may vary, we introduce a scaling factor  $\alpha_j$  for each segment according to its duration, similar to [20].

Similar to [18], we minimize the cost function given by the time integral of the square of the jerk. Specifically, the cost  $J_j$  of the  $j$ -th segment can be written as,

$$J_j = w_s \int_{t_{j-1}}^{t_j} \left( \frac{d^3 f^s(t)}{dt^3} \right)^2 dt + w_l \int_{t_{j-1}}^{t_j} \left( \frac{d^3 f^l(t)}{dt^3} \right)^2 dt, \quad (3)$$

where  $w_s$  and  $w_l$  denote the weight for the control cost of the longitudinal direction and lateral direction, respectively. The objective is simple and invariant given different combinations of semantic elements thanks to the SSC, which allows the formulation to easily adapt to different traffic configurations.

Denote by  $y_j^\sigma(t)$  the non-scaled Bézier curve in the interval  $[0, 1]$  with  $\mathbf{p}_j$  as the control points. Let  $u = \frac{t-t_{j-1}}{\alpha_j}$  denote the normalized time of the non-scaled Bézier curve, the cost of the  $j$ -th segment on dimension  $\sigma$  can be rewritten using the non-scaled  $y_j^\sigma(t)$  as follows,

$$J_j^\sigma = \int_0^1 \alpha_j \cdot \left( \frac{d^3 (\alpha_j \cdot y_j^\sigma(t))}{d(u \cdot \alpha_j)^3} \right)^2 du = \frac{1}{\alpha_j^3} \cdot \mathbf{p}_j^T \mathbf{Q} \mathbf{p}_j,$$

where  $\mathbf{Q}$  is the Hessian matrix of the non-scaled Bézier curve. We omit the detailed calculation of  $\mathbf{Q}$  for brevity.

### C. Enforcing Safety and Dynamical Constraints

In this paper, we adopt a quintic ( $m=5$ ) piecewise Bézier curve as the trajectory parameterization. According to the hodograph property, the  $k$ -th derivative of the non-scaled Bézier curve  $\frac{d^k y_j^\sigma(t)}{dt^k}$  is supported by control points  $\mathbf{q}_j^{\sigma, (k)}$  which can be calculated by induction as follows,

$$q_{j,i}^{\sigma, (0)} = p_i^j, q_{j,i}^{\sigma, (k)} = \frac{m!}{(m-k)!} (q_{j,i+1}^{\sigma, (k-1)} - q_{j,i}^{\sigma, (k-1)}). \quad (4)$$

Based on this property, the  $k$ -th-order derivatives at the boundaries of  $f_j^\sigma(t)$  can be expressed as

$$\frac{d^k f_j^\sigma(t_{j-1})}{dt^k} = \alpha_j^{1-k} \cdot q_{j,0}^{\sigma, (k)}, \frac{d^k f_j^\sigma(t_j)}{dt^k} = \alpha_j^{1-k} \cdot q_{j,m}^{\sigma, (k)}, \quad (5)$$

respectively. Moreover, by further applying the convex hull property, we can constrain the entire derivative profile of  $f_j^\sigma(t)$  using the following sufficient condition,

$$\beta_{j,-}^{\sigma, (k)} \leq \alpha_j^{1-k} \cdot q_{j,i}^{\sigma, (k)} \leq \beta_{j,+}^{\sigma, (k)}, \forall i \Rightarrow \beta_{j,-}^{\sigma, (k)} \leq \frac{d^k f_j^\sigma(t)}{dt^k} \leq \beta_{j,+}^{\sigma, (k)}, \quad (6)$$

where  $\beta_{j,-}^{\sigma, (k)}$  and  $\beta_{j,+}^{\sigma, (k)}$  denote the lower and upper bound on dimension  $\sigma$  for the  $k$ -th derivative of the  $j$ -th segment.

1) *Desired state constraints*: First of all, the generated trajectory should start from the given initial state  $[\sigma_{t_0}^{(0)}, \sigma_{t_0}^{(1)}, \sigma_{t_0}^{(2)}]$  and terminate at the given goal state  $[\sigma_{t_n}^{(0)}, \sigma_{t_n}^{(1)}, \sigma_{t_n}^{(2)}]$  for  $\sigma \in \{s, l\}$ , where  $\sigma_t^{(k)}$  denotes the  $k$ -th-order derivative at time  $t$ . Specifically, this requires enforcing equality constraints for the first and last segment as follows,

$$\frac{d^k f_0^\sigma(t_0)}{dt^k} = \sigma_{t_0}^{(k)}, \quad \frac{d^k f_n^\sigma(t_n)}{dt^k} = \sigma_{t_n}^{(k)}, \quad (7)$$

where  $k = 0, 1, 2$ . By applying Eq. 5, these constraints can be written as linear equality constraints w.r.t.  $\mathbf{p}$ .

2) *Continuity constraints*: The generated trajectory should be continuous for all the derivatives up to the  $k$ -th order at all the connecting points between two consecutive pieces. The continuity constraints between the  $j$ -th segment and the  $j+1$ -th segment can be written as

$$\frac{d^k f_j^\sigma(t_j)}{dt^k} = \frac{d^k f_{j+1}^\sigma(t_j)}{dt^k}, \quad (8)$$

where  $k = 0, 1, 2, 3$ . By applying Eq. 5, these constraints can also be written as linear equality constraints w.r.t.  $\mathbf{p}$ .

3) *Free-space constraints*: To guarantee the generated trajectory is collision-free, we constrain each segment of the trajectory within the corresponding cube. The free-space constraint of the  $j$ -th segment on dimension  $\sigma$  can be enforced by using the sufficient condition (Eq. 6) under  $k = 0$ , where  $\beta_{j,-}^{\sigma, (0)}$  and  $\beta_{j,+}^{\sigma, (0)}$  represent the position bounds on dimension  $\sigma$  given by the shape of the cube.

4) *Dynamical constraints*: To comply with the environment semantics and dynamical feasibility constraint, we enforce the constraints on the derivatives of trajectories by using the sufficient condition (Eq. 6), where  $k = 1, 2$ . The physical meaning is that the maximum lateral/longitudinal velocity and acceleration is constrained. Summarizing all the linear equality and inequality constraints, the overall formulation can be written as a QP, which can be solved efficiently using off-the-shelf solvers (such as OQP). Although Eq. 6 is a sufficient condition, in practice we find it does not result in over-conservative behavior, as shown in Sect. VII. In the case that no feasible solution can be found, the error is fed back to the behavior layer for further reaction.

## VII. EXPERIMENTAL RESULTS

### A. Implementation Details

The experiments are conducted in a multi-agent simulation platform, as illustrated in Fig. 2. In the simulation, dynamic agents potentially interact with each other, but the interaction model is unknown to the planner. The ego (our) vehicle only has a limited sensing range for the environment semantics. The route planner finds a random route for the ego vehicle at the beginning of the mission, and the route information of other agents is also unknown to the planner. The prediction method is similar to that in [21] which decouples the problem into behavior prediction and trajectory prediction. A long prediction horizon facilitates accounting for long-term future

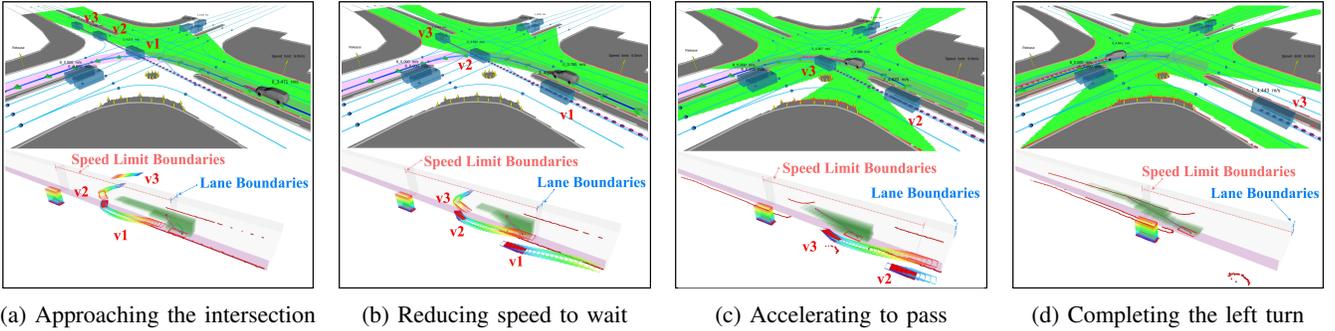


Fig. 6: Illustration of an unprotected left turn in a busy urban intersection. When the ego vehicle is approaching the intersection, it finds the left turn is not feasible and it reduces speed to wait. Once feasible, the vehicle quickly accelerates to complete the left turn.

rewards, which potentially results in a more consistent output compared to using a short prediction horizon. However, the uncertainty also scales with the prediction horizon. Therefore, it is beneficial to characterize the long-term prediction uncertainty, and we provide an attempt in [22]. All the test environments are annotated from real satellite maps via QGIS. The planning method proposed in this paper<sup>1</sup> is implemented in C++11. All the experiments are conducted on a desktop computer equipped with an Intel I7-8700K CPU, and our proposed method can run stably at 20 Hz.

### B. Qualitative Results

To verify that our proposed method can automatically adapt to different traffic configurations with different semantic elements, we choose three representative test cases.

1) *Merging into congested traffic due to road construction*: As illustrated in Fig. 3, this case is used to verify the capability of dealing with road construction, lane change (lane geometry), dynamic obstacles and the speed limit at the same time. The constructed SSC generally encodes the necessary information for optimization. The optimal trajectories are generated without explicitly caring about what types of semantic elements are present.

2) *Overtaking on an urban expressway*: This case is to validate the capability of dealing with high-speed traffic. The SSC is shown to be suitable for this time-critical scenario. As illustrated in Fig. 7, our method conducts a safe and smooth overtaking on an urban expressway with a speed of around 20  $m/s$ . The limitation is that the prediction uncertainty is not sufficiently considered in the current SSC generation process, which is left as important future work.

3) *An unprotected left turn at an intersection*: This case is used to verify the capability of quickly responding to complex interactions with other agents during traffic negotiation. There is also a speed limit which poses hard speed constraints for the whole interaction process. As shown in Fig. 6, our method efficiently finds safe and feasible trajectories so that the vehicle precisely follows the behavior plan and navigates smoothly.

### C. Comparisons and Analysis

We conduct a quantitative comparison with the seminal work [18], which is based on optimal primitives in the Frenét

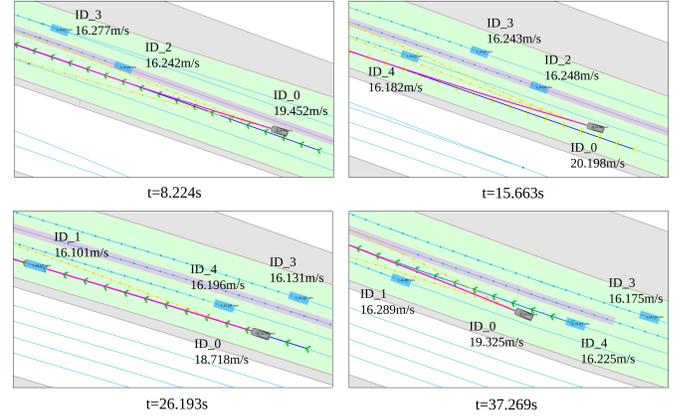


Fig. 7: Illustration of overtaking on an urban expressway.

frame. In [18], the primitives are regularly sampled around a local target state with a certain resolution in the  $s/t$  domain, and for different behaviors, the strategy for choosing the local target is different.

To conduct a fair comparison, we set up a benchmark track which is annotated from a real satellite map, as shown in Fig. 8b. To test the planner's response to semantic elements, we add a red light checkpoint and a speed limit to the track. Moreover, dense obstacles are placed on the track, as shown in Fig. 8a, to test the collision avoidance performance. Since the ego vehicle only has a limited sensing range (around 100  $m$ ), the collision avoidance task requires frequent replanning. The maximum acceleration and the maximum deceleration are set to 2  $m/s^2$  and 3  $m/s^2$ , respectively. We use the same behavior planner (MPDM) for both our method and [18] to generate the lane change command. The user-desired velocity is set to 15  $m/s$  for the behavior planner.

1) *Collision-avoidance in cluttered environments*: The first segment of the track is around 320  $m$  from the starting point to the red light. As shown in Fig. 8c, our method can fully utilize the maneuverability of the vehicle and arrive at the red light at 42  $s$ , about 14 seconds earlier than [18]. Moreover, our acceleration profile is smoother while staying within the dynamical limit. The reason is that our SSC representation models the continuous solution space, while the baseline method suffers from discretization and limited state space coverage. We observe that the benefit of using the corridor representation is more obvious in the cluttered environments since many primitives of the baseline method

<sup>1</sup>Source code is released at [https://github.com/HKUST-Aerial-Robotics/spatiotemporal\\_semantic\\_corridor](https://github.com/HKUST-Aerial-Robotics/spatiotemporal_semantic_corridor).

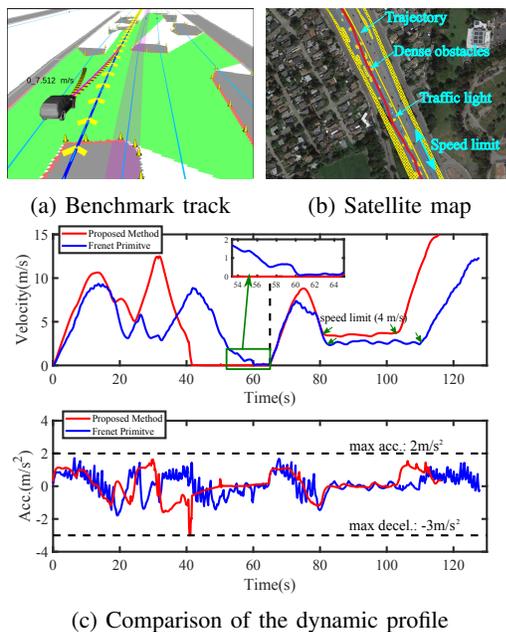


Fig. 8: Illustration of the comparison on a benchmark track.

become infeasible in this case.

2) *Precise stop with a high entry speed*: There is a red light checkpoint in the middle of the track, as shown in Fig. 8b, and the vehicle needs to complete a precise stop with a high entry speed. As shown in Fig. 8c, our method can reach the precise stop with an entry speed of  $13 \text{ m/s}$  while the max deceleration is strictly bounded inside the dynamic range. However, for the baseline method [18], a stopping mode is needed to fix the local target state so that the replanning process can consistently reach the target boundary condition. If we do not manually fix the local target state and dynamically calculate it based on zero desired velocity, the initial sampled stopping trajectory may not be sampled in later replanning due to a minor change of the target state. This may cause rolling, as shown in Fig. 8c. In contrast to [18], our method explicitly enforces the stopping boundary condition and achieves a precise stop.

3) *Collision avoidance under a low speed limit*: In addition to the study of high-speed collision-avoidance, we are also interested in the low-speed performance. To test this, a  $4 \text{ m/s}$  speed limit is placed on the track, as shown in Fig. 8b. As depicted in Fig. 8c, our method strictly follows the speed limit.

We also conduct experiments in which obstacles are placed in an online manner (see our video for details).

## VIII. CONCLUSION AND FUTURE WORK

In this paper, we propose a trajectory generation framework for complex urban environments. Our main contribution is twofold. First, we present an SSC structure which copes with an arbitrary combination of semantic elements in a unified way. Second, we present a trajectory optimization formulation which guarantees the safety and feasibility of the output trajectory. The proposed method is extensively analyzed using various traffic configurations and complex semantic elements. The main limitation is that the prediction uncertainty and interaction uncertainty are not sufficiently modeled, which

is the research direction we are currently working on [22]. Moreover, we find that the Bézier curve is also useful for non-linear trajectory optimization for AVs.

## REFERENCES

- [1] Z. Ajanovic, B. Lacevic, B. Shyrokau, M. Stolz, and M. Horn, "Search-based optimal motion planning for automated driving," in *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.* IEEE, 2018, pp. 4523–4530.
- [2] T. Gu, J. Atwood, C. Dong, J. M. Dolan, and J.-W. Lee, "Tunable and stable real-time trajectory planning for urban autonomous driving," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 250–256.
- [3] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for berthaa local, continuous method," in *IEEE Intl. Veh. Sym.* IEEE, 2014, pp. 450–457.
- [4] W. Xu, J. Wei, J. M. Dolan, H. Zhao, and H. Zha, "A real-time motion planner with trajectory optimization for autonomous vehicles," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2012, pp. 2061–2067.
- [5] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," IEEE, 2009.
- [6] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2011.
- [7] M. Ruffi and R. Siegwart, "On the design of deformable input- / state-lattice graphs," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2010, pp. 3071–3077.
- [8] M. Likhachev and D. Ferguson, "Planning long dynamically feasible maneuvers for autonomous vehicles," *Intl. J. Robot. Research*, 2009.
- [9] M. T. Wolf and J. W. Burdick, "Artificial potential functions for highway driving with collision avoidance," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2008, pp. 3731–3736.
- [10] C. Hubmann, M. Aeberhard, and C. Stiller, "A generic driving strategy for urban environments," in *Proc. of the Intl. Conf. on Intel. Trans. Syst.* IEEE, 2016, pp. 1010–1016.
- [11] Z. Zhu, E. Schmerling, and M. Pavone, "A convex optimization approach to smooth trajectories for motion planning with car-like robots," in *2015 IEEE Conference on Decision and Control*. IEEE, 2015, pp. 835 – 842.
- [12] S. M. Erlien, S. Fujita, and J. C. Gerdes, "Safe driving envelopes for shared control of ground vehicles," in *7th IFAC Symposium on Advances in Automotive Control*. Elsevier, 2013, pp. 831–836.
- [13] C. Liu, C.-Y. Lin, and M. Tomizuka, "The convex feasible set algorithm for real time optimization in motion planning," *SIAM Journal on Control and Optimization*, vol. 56, no. 4, pp. 2712–2733, 2018.
- [14] C. Hubmann, M. Becker, D. Althoff, D. Lenz, and C. Stiller, "Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles," in *IEEE Intl. Veh. Sym.* IEEE, 2017, pp. 1671–1678.
- [15] J. Chen, C. Tang, L. Xin, S. E. Li, and M. Tomizuka, "Continuous decision making for on-road autonomous driving under uncertain and interactive environments," in *IEEE Intl. Veh. Sym.* IEEE, 2018, pp. 1651–1658.
- [16] E. Galceran, A. G. Cunningham, R. M. Eustice, and E. Olson, "Multi-policy decision-making for autonomous driving via changepoint-based behavior prediction," in *Proc. of Robot.: Sci. and Syst.*, 2015.
- [17] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *Intl. J. Robot. Research*, vol. 29, no. 5, pp. 485–501, 2010.
- [18] M. Werling, S. Kammel, J. Ziegler, and L. Gröll, "Optimal trajectories for time-critical street scenarios using discretized terminal manifolds," *Intl. J. Robot. Research*, vol. 31, no. 3, pp. 346–359, 2012.
- [19] H. Fan, F. Zhu, C. Liu, L. Zhang, L. Zhuang, D. Li, W. Zhu, J. Hu, H. Li, and Q. Kong, "Baidu apollo em motion planner," *arXiv preprint arXiv:1807.08048*, 2018.
- [20] F. Gao, W. Wu, Y. Lin, and S. Shen, "Online safe trajectory generation for quadrotors using fast marching method and bernstein basis polynomial," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2018, pp. 344–351.
- [21] W. Ding and S. Shen, "Online vehicle trajectory prediction using policy anticipation network and optimization-based context reasoning," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2019.
- [22] W. Ding, J. Chen, and S. Shen, "Predicting vehicle behaviors over an extended horizon using behavior interaction network," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2019.