2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Continuous Decision Making for On-road Autonomous Driving under Uncertain and Interactive Environments

Jianyu Chen[1], Chen Tang[1], Long Xin[1,2], Shengbo Eben Li[2] and Masayoshi Tomizuka[1]

Abstract— **Although autonomous driving techniques have achieved great improvements, challenges still exist in decision making for variety of different scenarios under uncertain and interactive environments. A good decision maker must satisfy the following requirements: (1) Be in a generic and unified form to cover as more scenarios as possible. (2) Be able to interact properly with other moving obstacles under the uncertainty of their motions. In this paper, the continuous decision making (CDM) framework is proposed to formulate different driving scenarios in a unified way, which encodes the high level decision making information into a continuous reference trajectory that can be naturally combined with a lower level trajectory planner. Within the framework, a maximum interaction defensive policy (MIDP) is proposed, which calculates the best action to interact with stochastic moving obstacles while guaranteeing safety. The method is applied to a ramp merging scenario and the stochastic behavior models of the surrounding vehicles are learned from the NGSIM dataset. Simulations are shown to visualize and analyze the results.**

I. Introduction

In the past years, progresses in robotics and breakthroughs in machine learning boosted the development of autonomous driving systems. With the enhanced techniques designed for autonomous driving, such as stable and precise vehicle control [1], accurate perception and localization [2], and efficient motion planning techniques [3], [4], some prototypes developed by universities or companies have already been demonstrated in public. People are more optimistic that autonomous driving will become reality in the near future.

Despite the inspiring achievements, there is still a gap between the state-of-the-art systems and a full level autonomous driving system defined by SAE [5]. The challenges mainly exist in the decision making part and they come in two folds. First, the decision maker should be in a generic and unified form to cover as more scenarios as possible. Currently most autonomous driving systems are designed on a case-by-case basis. When a new scenario is encountered, a new specific module will be added to take care of this scenario. However, such design mechanism lacks the ability for generalizing to new scenarios. Moreover, the cumulatively added modules will make it extremely difficult to maintain the software. Second, the decision maker should be able to interact properly with other moving obstacles under the uncertainty of their motions. Failure cases are likely to appear when this issue is not solved well. For example, a Google self-driving car collided with a bus in February 2016 [6]. The Google car tried to merge left to the lane where a bus was running on. It predicted that the bus would yield but the bus did not.

As discussed above, we believe that full level autonomous driving is not tractable by rule-based method [7]–[9], which is dominated even nowadays. In highway and urban driving, autonomous vehicles will encounter different scenarios with different kinds of moving obstacles. It is impossible to design specific planners for each scenario and switch among them. Therefore, a more generic decision maker is necessary.

Efforts have been made to enable generalization in decision making. For example, semantic-based methods [10], [11] have been proposed for traffic scene modeling and high-level maneuver planning. However, these methods, together with the rule-based methods, can be defined as "discrete" decision making methods. They all output a discrete high-level maneuver command and then let a low level planner to execute the maneuver. This mechanism asks us to enumerate all possible maneuvers to ensure generalization, which is also impossible because there are infinitely many combinations of maneuvers.

Instead of planning discrete maneuvers, some methods such as [12], [13] plan a continuous speed profile as the output of the decision maker. We define this kind of methods as "continuous" decision making and we believe they have larger potential for generalization. Nevertheless, these methods are not able to consider lateral decisions such as lane changing. More importantly, they do not consider motion uncertainty of moving obstacles as well as the interaction with them. Motions of all moving obstacles are assumed deterministic that are known in advance, and more specifically, with constant speed. Mechanism to address the uncertainty and interaction must be embedded in the framework.

There are some works to address the motion uncertainty issue in autonomous driving. The non-conservatively defensive strategy [14] enables autonomous vehicles to consider different intentions of the surrounding vehicles and act properly. Although it guarantees safety while preventing overcautious behaviors, it does not consider the interaction, e.g, the motions of others are not influenced by the host vehicle. S. Dorsa et al [15] model other vehicles as optimal controllers, and then use optimization to solve the best control sequence to act. The work considers interaction with others, but the formulated optimization problem is almost intractable, multiple local optimums exist and it is limited to short preview horizon given the limited computation power.

[1]Department of Mechanical Engineering, University of California, Berkeley, CA 94720, USA [2]State Key Lab of Automotive Safety and Energy, Department of Automotive Engineering, Tsinghua University, Beijing 100084, China. Correspondence to: Jianyu Chen (jianyuchen@berkeley.edu).

This work is supported by DENSO International at America.

In this paper, the continuous decision making (CDM) method is proposed, which formulates different driving scenarios in a unified way. The high level decision making information is encoded into a continuous reference trajectory that can be naturally combined with a lower level trajectory planner. To address the uncertainty and interaction issue, a maximum interaction defensive policy (MIDP) is proposed, which calculates the best action to interact with stochastic moving obstacles while guaranteeing safety. The stochastic models of surrounding vehicles are built and learned from NGSIM dataset [16].

II. System Architecture

In this section, an overview of the system architecture is provided. As shown in Fig.1, the system contains three main modules: perception, motion prediction and planning. The contributions they make to the system and the roles they play in this paper are described in the following paragraphs.

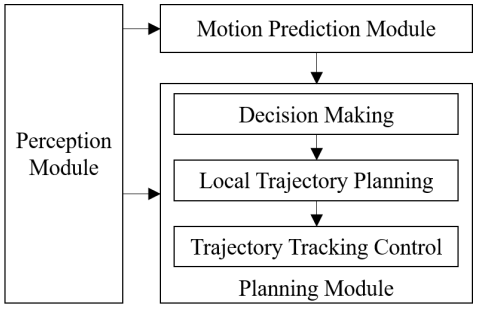

Fig. 1. The system structure

The perception module handles sensor data processing. It receives raw data from multiple sensors, and then processes and transforms the data into useful information such as position and velocity of the host and surrounding vehicles. Then the information is passed to the motion prediction and planning modules. In this paper, the output of the sensor data is assumed ready to use, and the perception module will no longer be discussed in the rest of the paper. The details of the sensor setup of our system is described in [17].

The motion prediction module predicts the probable future motion of the surrounding vehicles. The predicted motion is in a stochastic form instead of a maximum likelihood form. In other words, the future motion is represented by a probability distribution over multiple possible trajectories instead of a single trajectory. Furthermore, the candidate trajectories correspond to multiple possible distinct behaviors of the driver. The details of the motion prediction module will be described in Section V-B.

The planning module takes inputs from both the perception module and the motion prediction module, and outputs the control command for the vehicle to execute. It contains three parts: a decision maker, a local trajectory planner and a trajectory tracking controller. The decision maker provides high level driving strategies, which will be discussed in detail in Section III and Section IV.

The local trajectory planner plans an optimal trajectory considering the vehicle kinematics under the given strategy. The tracking controller lets the vehicle track the trajectory generated by the local trajectory planner. Their details can be found in the previous work [18].

III. Continuous Decision Making

In this section, the continuous decision making method is introduced. The unified formulation for common driving scenarios is established using speed profile. Then the algorithm to choose the best speed profile is introduced.

A. Continuous Decision Making Concept

Generally speaking, a decision maker takes the observations as input, runs the algorithm and then gives an output as the decision result. The process can be formulated as a function mapping the input to the output:

$$d = p_{decision}\left(o\right) \tag{1}$$

where o is the the observation, which can be either single current step or a historic sequence of observations (e.g, position and velocity of host and surrounding vehicles). $p_{decision}$ represents the decision process and d is the decision result.

A low-level planning and control module can be represented in a similar way as a function mapping from the decision to the control command:

$$c = p_{pc}\left(d\right) \tag{2}$$

where p_{pc} represents the process of planning and control. c is the control command (e.g, acceleration and steering angle).

For a classic discrete autonomous driving decision maker, d is in a discrete domain set D such that $d \in D$. Moreover, the semantic meanings of elements in set D are driving maneuvers such as stopping, car following, lane changing, turn right/left and so on. This formulation not only asks us to specify all elements in D, but also requires us to design a specific planning and control process p_{pc} for each $d \in D$.

A continuous decision maker, in contrast to a discrete one, generates a continuous mapping f instead of a decision command d:

$$f : [0, T] \rightarrow S \tag{3}$$

where $[0, T]$ is the time period and S is a set. If elements s in S are constituted of unified form of states in the environment (e.g, position and velocity), then we do not need to enumerate all possible values of s.

Note that the mapping (3) is equal to a trajectory in state space S. Furthermore, the classic discrete maneuver information is contained in the homotopy of the trajectories.

B. Unified Formulation of Different Driving Scenarios

The authors believed that the decision mainly exists in the longitudinal motion for on-road driving scenarios. Indeed, for scenarios such as car following, stopping, lane merging, turning and roundabout, the vehicle just need to keep the centerline of its reference lane and no lateral decision is needed. Note here the lateral "decision" is different from

the lateral "motion". We refer making a lateral decision as to changing the reference lane. The vehicle do need to perform some lateral motions such as avoiding the parked vehicles (which can be handled by the low level planning and control module), but their lateral decision (reference lane) can remain the same. Under this definition, the overtaking scenario, although containing significant lateral motion, can be categorized as a longitudinal decision problem if the vehicle is supposed to get back to its original lane after overtaking. Therefore, the only type of lateral decision for on-road driving is lane changing.

A speed profile is adequate to fully represent the longitudinal motion, as it can be integrated to a longitudinal position profile or differentiated to an acceleration profile. A speed profile can be denoted as $v(t) \in V$ with $t \in [0, T]$, and V is the set of feasible velocity candidates. The corresponding longitudinal position profile can be denoted as $x(t)$ and the corresponding acceleration profile can be denoted as $a(t)$.

The lateral decision can be denoted as $l(t) \in L = \{-1, 0, 1\}$ where $l = 0$ means keeping the original lane, $l = -1$ means turning left and $l = 1$ means turning right. Integrating the lateral decision with the longitudinal decision, the decision mapping in (3) becomes:

$$f : [0, T] \rightarrow V \times L \tag{4}$$

Then the decision making process can be formulated as the following optimization problem:

$$\begin{aligned} &\min_{f} \int_{t=0}^{T} \left\{ J_v\left(v(t), l(t)\right) + J_a\left(a(t)\right) \right\} dt \\ &s.t. \quad g_v\left(v(t), l(t), t\right) < 0, \quad t \in [0, T] \\ &\qquad g_x\left(x(t), l(t), t\right) < 0, \quad t \in [0, T] \\ &\qquad g_a\left(a(t), t\right) < 0, \quad t \in [0, T] \end{aligned} \tag{5}$$

where J_v and J_a represent the costs on velocity and acceleration, which are mainly constituted of desired velocity tracking costs and acceleration penalty costs. g_a is the constraint on the acceleration profile, which can be formulated as a bound limit $\underline{a} \leq a \leq \bar{a}$. g_v is the constraint on the speed profile, which is mainly about bounding the lateral acceleration for passenger comfort. The speed constraint will be described in Section III-C.

The constraint on the longitudinal position profile g_x is also called the event constraint. It is mainly generated by traffic events such as surrounding vehicles, traffic lights and merging. We now give some examples to illustrate the event constraint under a deterministic assumption, which means that the future states of the traffic (e.g, positions of other vehicles, traffic light time...) are known in advance and deterministic. The stochastic and interactive version will be discussed in Section IV.

Fig.2 (a) shows a ramp merging scenario where the host vehicle tries to merge into the lane occupied by a surrounding vehicle. Fig.2 (a) is a bird view of the scenario, where the host vehicle is in red and the surrounding vehicle is in yellow, x_0 is longitudinal position of the merge point. Fig.2 (c) shows the longitudinal position profile of the host vehicle. The yellow bar represents the predicted future space

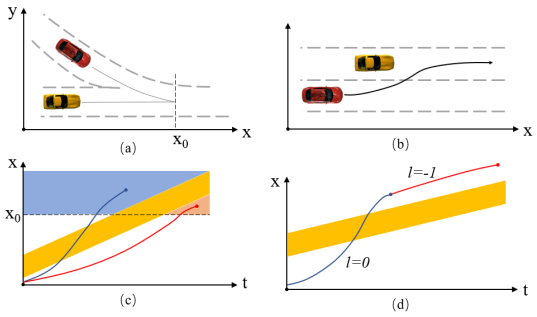

Fig. 2. Example scenarios

occupied by the surrounding vehicle, which is assumed to maintain constant speed in this case. The yellow bar area is not accessible when $x > x_0$ (after finishing merging). This split the trajectory into two homotopy classes. One class contains the trajectories ended in the blue area, where the blue trajectory is an example. Another class contains the trajectories ended in the red area, with the red trajectory as an example. Note that the two classes represent the decisions for pass and yield respectively.

Fig.2 (b) shows a lane changing scenario where the host vehicle (in red) tries to change the lane to the left lane which is occupied by a surrounding vehicle (in yellow). Fig.2 (b) is a bird view of the scenario, in this case the host vehicle accelerates to pass the surrounding vehicle and then changes the lane. Fig.2 (d) is the longitudinal position profile. The yellow bar again represents the surrounding vehicle. In this scenario the host vehicle executes a lateral decision (lane changing), where at first it keeps the original lane ($l = 0$, blue trajectory) and then changes to the left lane ($l = -1$, red trajectory). The yellow bar is not accessible when $l = -1$.

After generating the longitudinal position profile and the lateral decision commands (equal to the lane index), the augmented trajectory (4) can be transformed to a 2D trajectory $\mathbf{x}^r$ according to the road coordinate, where the lateral position is set to the centerline of its corresponding lane index. $\mathbf{x}^r$ is then passed to the local trajectory planner as a reference trajectory.

C. Passenger Comfort Awareness

Along the generated trajectory $\mathbf{x}^r$, lateral acceleration needs to be limited to ensure passenger comfort. Since the lateral acceleration has relationship with the trajectory curvature and speed. Moreover, the curvature is determined given the road centerline, which means that limiting lateral acceleration is equivalent to limiting speed.

The Bezier curve is used to represent the road centerline because it has analytical form and is efficient. The formulation of Bezier curve is:

$$B(\tau) = (1-\tau)^2 P_0 + 2(1-\tau)\tau P_1 + \tau^2 P_2, \quad 0 \leq \tau \leq 1 \tag{6}$$

where P_0, P_1 and P_2 are key points on the road centerline. With this analytical form, the first and second derivatives of

the curve can be calculated analytically:

$$\begin{aligned} B'(\tau) &= 2(1-\tau)(P_1 - P_0) + 2\tau(P_2 - P_1) \\ B''(\tau) &= 2(P_2 - 2P_1 + P_0) \end{aligned} \tag{7}$$

Then the curvature can be calculated by:

$$\kappa(\tau) = \frac{|B'(\tau) \times B''(\tau)|}{\|B'(\tau)\|^3} \tag{8}$$

and the speed limit is given by:

$$v_{\max}(\tau) = \sqrt{\frac{a_{\max}}{\kappa(\tau)}} \tag{9}$$

where a_{max} is the maximum lateral acceleration allowed.

By integrating the norm of $B'(\tau)$, the longitudinal position can be calculated as:

$$x(\tau) = \int_0^{\tau} \|B'(\tau)\| d\tau \tag{10}$$

The index τ bridges a relationship between the longitudinal position and the speed limit $v_{\max}(x)$. Therefore, the constraint on speed profile $g_v(v(t), t) < 0$ is formulated as:

$$g_v(v(t), t) = v(t) - v_{\max}(x(t)) < 0 \tag{11}$$

A remaining question is how to calculate the speed limit when the lane index l changes, which will cause an abrupt change in the lateral position. We believe that its influence is negligible because the curvature of the adjacent lanes at the same longitudinal position is approximately the same, and the low-level trajectory planner will smooth the abrupt lateral change.

D. Speed Profile Sampling and Searching

To plan an optimal speed profile, a speed profile sampling and searching method similar to that in [19] is used. First, multiple possible speed profiles are sampled in the speed-time graph. To initialize the sampling, a discretization process is applied to both the speed and the time dimensions. For example, as shown in Fig.3, the time is discretized by 2 seconds and the speed is discretized by 2 m/s. We call each time discretization point a "station", e.g, 2 s, 4 s and 6 s are the three stations in Fig.3. Furthermore, we call the speed-time joint discretization points "nodes", which are shown as green dots in the figure. Note that on the initial time station (0 s), there is only one node which is the current speed.

In order to construct the speed profiles, every node is connected to all of its "feasible" adjacent nodes. Here "feasible" means satisfying the basic acceleration constraint, which is formulated as

$$\underline{a} \le \frac{v(t_1) - v(t_0)}{t_1 - t_0} \le \bar{a} \tag{12}$$

where t_0 and t_1 are adjacent stations with $t_0 < t_1$. Note this constraint is not equal to the acceleration constraint in (5). It is just a fast method to prune the speed profile samples for computational efficiency.

The connections between nodes are represented by third order polynomials:

$$v(t) = \rho_0 + \rho_1 t + \rho_2 t^2 + \rho_3 t^3, \quad t_0 \le t \le t_1 \tag{13a}$$

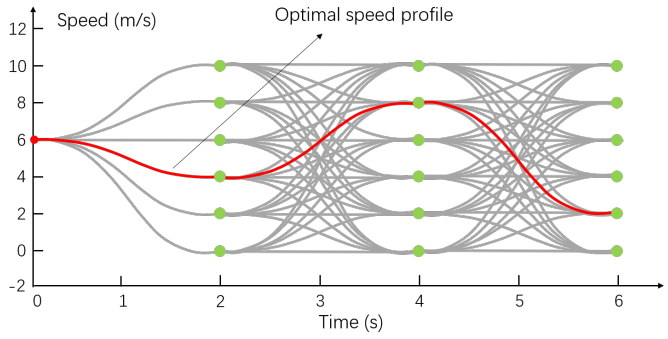

Fig. 3. Speed profile sampling

$$\begin{aligned} s.t. \quad & v(t_0) = v_0 \\ & v(t_1) = v_1 \\ & a(t_0) = \left.\frac{dv(t)}{dt}\right|_{t=t_0} = 0 \\ & a(t_1) = \left.\frac{dv(t)}{dt}\right|_{t=t_1} = 0 \end{aligned} \tag{13b}$$

where ρ_0, ρ_1, ρ_2 and ρ_3 are parameters which can be determined by the conditions (13b). t_0, t_1, v_0 and v_1 are constant values from the nodes. The zero acceleration conditions ensure the smoothness of the speed profile. (13) also constitutes a piece of speed profile. By connecting pieces through the whole preview horizon, an entire speed profile is established:

$$v_i(t), \quad 0 \le t \le T, \quad i \in I \tag{14}$$

where i is the index of the speed profile and I is the set of all indexes.

With $v_i(t)$, the longitudinal position profile $x_i(t)$ and the acceleration profile $a_i(t)$ can be calculated. The lateral decision profile $l(t)$ is sampled by defining a decision time t_{dec} such that $l = 0$ when $t < t_{dec}$ and $l \neq 0$ when $t \ge t_{dec}$. Here we assume that the decision will only change once in the preview horizon. Then the costs and constraints of each speed profile will be calculated according to (5). The speed profile with the minimal cost without violating the constraints is selected as the optimal speed profile.

However, calculating the costs and constraints in the continuous analytical form as described in (5) is intractable. In order to address this issue, the speed profile is discretized to several checkpoints. The cost of (5) is approximated by the sum of costs of all checkpoints, and the constraint violation is approximated by the violation of the checkpoints.

The resulting speed profile will not be accurately optimal because of several discretization processes. However, this will not influence the final planning result because the speed profile is only a reference to the low level planning module. The planning module will smooth and optimize the trajectory. The most important effect of the speed profile is to give a decision pattern.

IV. Maximum Interaction Defensive Policy

Until now, we have been discussing under a deterministic environment. In the real world, the motions of moving

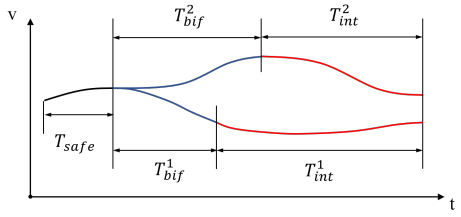

Fig. 4. The Maximum Interaction Defensive Policy

obstacles are uncertain, and they can be influenced by the motion of the host vehicle. In this section, the maximum interaction defensive policy is introduced to take into account the uncertain and interactive environments.

A. Three-stage Framework

Considering full uncertainty and interaction is extremely difficult and computationally intractable. In order to simplify the problem, we analogize the pattern of human drivers' decision process. When a human driver is encountered with some other vehicle, he/she will keep estimating its intention while guaranteeing safety, and then decide how to interact with the vehicle. In this paper, we will only discuss the situation with one obstacle. Cases with multiple obstacles can be solved by sequentially applying the algorithm for one obstacle. The decision pattern is formulated as a three-stage framework described as follows:

1) Safety Stage: The first stage is a short term safety stage. In this stage, safety (e.g, no collision) must be guaranteed in respect to all possible behaviors of the moving obstacle. Without loss of generality, assume that the obstacle has two possible behaviors, whose models can be written as:

$$\xi_m = b_i\left(\xi_h, s_m\right), \quad i \in \{1,2\} \tag{15}$$

where ξ_h is the future trajectory (a piece of longitudinal position profile) of the host vehicle and ξ_m is the predicted future trajectory of the moving obstacle. s_m is the moving obstacle's state which contains useful information for prediction. Furthermore, each model comes with a probability:

$$P_i\left(s_m\right), \quad i \in \{1,2\} \tag{16}$$

The models (15) can be either interactive or not, depending on whether ξ_h has influence on ξ_m. Denote the longitudinal position of the host vehicle and the moving obstacle at time t be $\xi_h(t)$ and $\xi_m(t)$, respectively. Actually $\xi_h(t) = x(t)$. Define a safe set as

$$d\left(\xi_m(t), x(t)\right) < 0 \tag{17}$$

Then if at time t, the host vehicle and the moving obstacle are able to have physical contact (e.g, finished merging or lane changing), $d\left(\xi_h(t), x(t)\right) = d\left(b\left(x(t), s_m(t)\right), x(t)\right)$ makes up part of the constraint function $g_x\left(x(t), l(t), t\right)$ in (5).

In the safety stage, both the two safe set (17) generated by the two behavior models (15) need to be considered as constraints if time t is physical contactable, unless one of them has zero probability. The underneath logic is that in the short term, safety needs to be guaranteed regardless of the type of the surrounding vehicle behavior.

2) Uncertainty Bifurcation Stage: The second stage is called uncertainty bifurcation stage. In this stage, the two different behaviors need to be considered separately. As shown in Fig.4, after the safety stage T_{safe}, the speed profile bifurcates to two different trajectories in the uncertainty bifurcation stage denoted by T^1_{bif} and T^2_{bif}. The safety constraint (17) is not considered because the host vehicle and the obstacle are not able to have physical contact in this stage.

Denote the time interval of the safety stage be $[0, T_{safe}]$. Denote the time intervals of the uncertainty bifurcation stage by $\left[T_{safe}, T_{safe} + T^1_{\text{bif}}\right]$ and $\left[T_{safe}, T_{safe} + T^2_{\text{bif}}\right]$ for the two behaviors respectively. Then the predicted states at the end of this stage are $s^1_m\left(T_{safe} + T^1_{\text{bif}}\right) = s\left(\xi_m\left(T_{safe} + T^1_{\text{bif}}\right)\right)$ and $s^2_m\left(T_{safe} + T^2_{\text{bif}}\right) = s\left(\xi_m\left(T_{safe} + T^2_{\text{bif}}\right)\right)$ for the two behaviors respectively, where s represents the process to generate the state from the speed profile.

3) Interaction Stage: The third stage is the interaction stage. The start time of this stage is determined by the host vehicle. For a merging scenario, the stage starts when the host vehicle exceeds the longitudinal position of the merge point. For a lane changing scenario, the stage starts when the value of l changes from 0 to -1 or 1. As shown in Fig.4, T^1_{int} and T^2_{int} represent the interaction stage for the two behaviors respectively.

In this stage, a unified interaction model is used to predict the surrounding vehicle. For example, before the host vehicle merges to the lane, the surrounding vehicle can either yield or pass. However, after the host vehicle merges in front of the surrounding vehicle, the surrounding vehicle can only yield the host vehicle. Despite the unified interaction model, we still need to consider two different trajectories, because the uncertainty bifurcation stage introduces two different predictions of the surrounding vehicle motion. The interaction model is written as:

$$\xi_m = b_{\text{int}}\left(\xi_h, s_m\right) \tag{18}$$

Since now the host vehicle is able to have physical contact with the moving obstacle, the safety constraints need to be considered for the two behaviors:

$$d\left(b_{\text{int}}\left(x(t), s_m\left(T_{safe} + T^i_{\text{int}}\right)\right), x(t)\right) < 0, \quad i \in \{1,2\} \tag{19}$$

B. The Speed Profile Planning Algorithm

Under the three stage framework, the cost function of the optimization problem (5) is modified to:

$$\begin{aligned} &\min_f \int_{t=0}^{T_{safe}} J dt \\ &+ \sum_{i=1}^{2} P_i \left\{ \int_{T_{safe}}^{T_{safe}+T^i_{bif}} J dt + \int_{T_{safe}+T^i_{bif}}^{T_{safe}+T^i_{bif}+T^i_{\text{int}}} J_{\text{int}} dt \right\} \end{aligned} \tag{20}$$

where J is the abbreviation of $J_v\left(v(t), l(t)\right) + J_a\left(a(t)\right)$. J_{int} is the cost in the interaction stage, which is different

from J by adding an additional courtesy awareness term J_{court}, which is proportional to the trajectory change of the moving obstacle caused by the interaction with the host vehicle. For example, if before the interaction stage the moving obstacle runs with model b_i, then according to this model its future trajectory will be ξ_m. Then when using the interaction model b_{int}, the future trajectory is ${\xi_m}'$. The trajectory change is calculated by $\left\|{\xi_m}' - \xi_m\right\|_2^2$. The courtesy awareness term makes the host vehicle less aggressive. Otherwise, the host vehicle will immediately start the interaction (e.g, change the lane) if it will not collide with the surrounding vehicle, but this might cause the surrounding vehicle change its action abruptly (e.g, perform a sudden brake).

When solving the optimization problem, the constraints for the velocity and acceleration are the same as (5), and the event constraints are specified in Section IV-A. In order to handle the uncertainty bifurcation, we first check the speed profiles from T_{safe} to T, and find the optimal speed profiles for both $i = 1$ and $i = 2$. Then their optimal costs are summed weighted by their probabilities. Finally, the costs of speed profiles from the initial time to T_{safe} are added to calculate the total costs. The speed profile with the smallest total costs is the optimal one. Note that this speed profile is bifurcated in $[T_{safe}, T]$. We average the bifurcated profiles to get a single speed profile. The resulting profile is then transformed to a reference trajectory and sent to the low level trajectory planner.

V. Simulations

In this section, several simulations are presented to show the capability of the proposed method. We first show the system performance under environments with static obstacles and deterministic obstacles. Then we show a case study on a ramp merging scenario, where a stochastic surrounding vehicle behavior model is learned from NGSIM dataset.

The simulation program is written in Matlab. A bicycle kinematic model is used as the vehicle dynamic model. The simulation runs closed loop with a lower level trajectory planner and controller, whose details can be found in [18]. The sampling time $T_s = 0.2s$ and the preview horizon is $T = 6s$. For stochastic and interactive moving obstacles, the safety stage time is $T_{safe} = 1s$. The lateral acceleration limit is $a_{\max} = 2.6m/s^2$. The longitudinal acceleration limits are $\bar{a} = 3m/s^2$ and $\underline{a} = -5m/s^2$. The rectangles represent vehicles, where the blue rectangle represents the host vehicle and the red rectangles represent the surrounding vehicles.

A. Static and Deterministic Obstacles

For static and deterministic obstacles, the longitudinal position constraints are known in advance. In this case, tools in Section III are enough to make a good decision. Two cases are shown here to illustrate the planning results.

1) Static Obstacles: This case tests the system's capability of collision avoidance for static obstacles and passenger comfort awareness on curve road. As shown in Fig.5 (a), the blue curve represents the the road centerline which the host vehicle needs to track. There are two static vehicles (in red) parking on the road side. Fig.5 (a) shows the trajectory of the host vehicle, where the deeper color represents the later time step. The host vehicle avoids the parked vehicles and turns right smoothly. Fig.5 (b) shows the speed profile of the host vehicle, from which we see that the host vehicle starts from a zero speed, accelerates when passing the parked vehicles, and then decelerates to turn right comfortably.

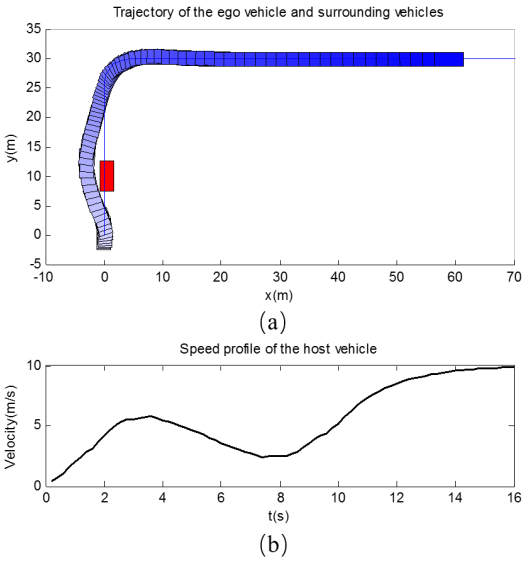

Fig. 5. A right turning scenario with street side parked vehicles

2) Deterministic Moving Obstacles with constant velocity: This case shows the system's capability of dealing with multiple deterministic moving obstacles. As shown in Fig.6 (a), the two blue lines represent the centerlines of two opposite-direction adjacent lanes. The host vehicle is driving on the downside lane with a slow vehicle in front of it. The front vehicle maintains a speed of $3m/s$. A vehicle driving towards the opposite direction is in the adjacent lane.

In Fig.6, the oncoming vehicle maintains a speed of $5m/s$. From the trajectory plots Fig.6 (a) and the host vehicle speed profile Fig.6 (b), we can see that since the oncomming vehicle is slow, the system decides to accelerates to overtake the front vehicle before the oncoming vehicle arrives.

In Fig.7, the oncoming vehicle maintains a speed of $15m/s$. From the trajectory plots Fig.7 (a) and the host vehicle speed profile Fig.7 (b), we can see that since the oncomming vehicle is fast, the system decides to decelerates to wait the oncoming vehicle pass, and then overtake the front vehicle.

B. Stochastic and Interactive Motion Prediction

Before showing the results for cases with stochastic moving obstacles, we describe how to predict their motions first. The objective of the motion prediction is to obtain the surrounding vehicle behavior models (15) and the interaction model (18), as well as the probabilities (16) for the behavior models. Here we give an example about how we predict the motion for a merging scenario. As shown in Fig. 2 (a), the

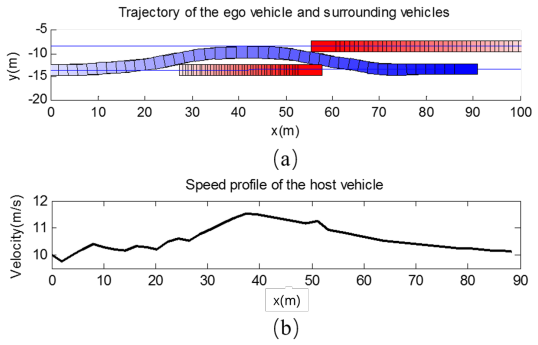

Fig. 6. An overtaking scenario when the up coming vehicle is slow

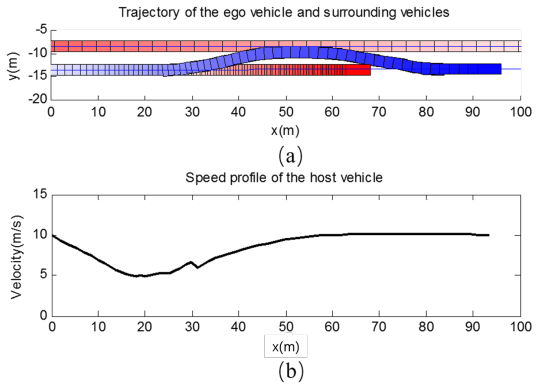

Fig. 7. An overtaking scenario when the up coming vehicle is fast

host vehicle (in red) is on the ramp, and the motion of the surrounding vehicle (in yellow) on the main road needs to be predicted. The built models will be used in Section V-C.

1) Vehicle Models: Before the host vehicle merges to the lane, there are two different behaviors for the surrounding vehicle: pass or yield. For the pass case, we assume it is a constant speed model. For the yield case, we assume it is a car following model. Mathematically, let b_1 be a constant speed model and b_2 be a car following model.

The constant speed model is easy. For the car following model, we learn it from the NGSIM dataset. Several car following trajectories are found in the dataset and features are extracted. Then a neural network is trained to predict the next time step speed of the host vehicle from the current speed/position of the host and front vehicle. The predicted speed is further filtered and integrated to get the future position. The interaction model b_{int} is assumed to be also a car following model, which is same with b_2.

2) Model Recognition: Before merging, b_1 and b_2 are the two possible models with probabilities P_1 and P_2 respectively. Trajectories of vehicles for both the passing case and the yielding case are found in the NGSIM dataset and features such as speed, position, time to merge point are extracted. A logistic model is trained with the data and used to predict P_1 and P_2. After merging, the only possible model is the interaction model so no model recognition process is needed.

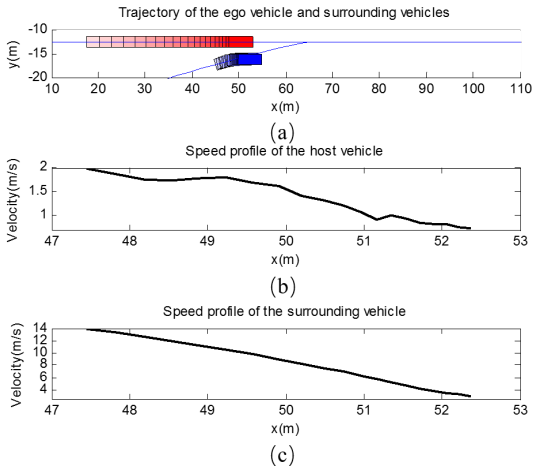

Fig. 8. Deterministic planning in a ramp merging scenario with the surrounding vehicle yielding

C. A Ramp Merging Scenario Case Study

A ramp merging scenario is studied in this subsection, where the surrounding vehicle is considered stochastic and interactive. As shown in Fig. 8 (a), the blue lines represent the two merging lanes' centerlines. The host vehicle (in blue) is trying to merge to the lane occupied by a surrounding vehicle (in red).

For the surrounding vehicle, we use the models learned in Section V-B. In the first case we use the car following model b_2 and in the second case we use the constant speed model b_1. After the host vehicle arrives at the merge point, the model changes to a car following model.

1) Yielding host vehicle: In this case, the surrounding vehicle will yield the host vehicle, but the host vehicle does not know that in advance. Fig.8 shows the result of adopting deterministic planning with constant speed assumption. The host vehicle fails to merge even though the surrounding vehicle decelerates to yield. On the contrary, Fig.9 shows the result of adopting MIDP. The host vehicle recognizes the yielding behavior and succeeds finishing merging.

2) Passing host vehicle: In this case, the surrounding vehicle will not yield the host vehicle. Fig.10 shows the results of adopting MIDP without considering interaction. Since the probability of the surrounding vehicle not yielding is very high, in order to guarantee safety, the host vehicle decelerates to wait. This is acceptable when there is only one surrounding vehicle. However, when there are multiple surrounding vehicles coming consequently, then the host vehicle can never finish merging. Fig.11 shows the results of adopting MIDP considering interaction. Since the host vehicle predicts that the surrounding vehicle will yield after it merges, it succeeds merging the lane.

VI. Conclusion

In this paper, the continuous decision making (CDM) framework is proposed to formulate different driving scenarios in a unified way. Within the framework, a maximum interaction defensive policy (MIDP) is proposed, which

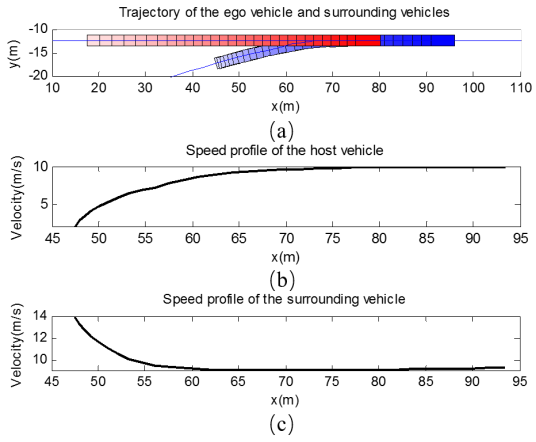

Fig. 9. MIDP in a ramp merging scenario with the surrounding vehicle yielding

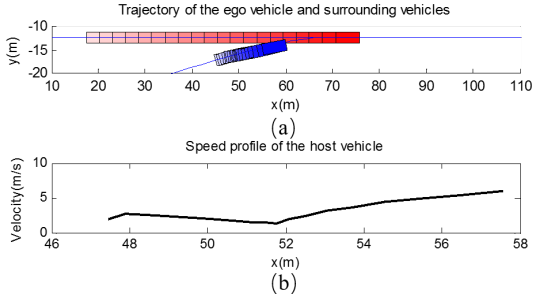

Fig. 10. MIDP without interaction stage in a ramp merging scenario with the surrounding vehicle passing

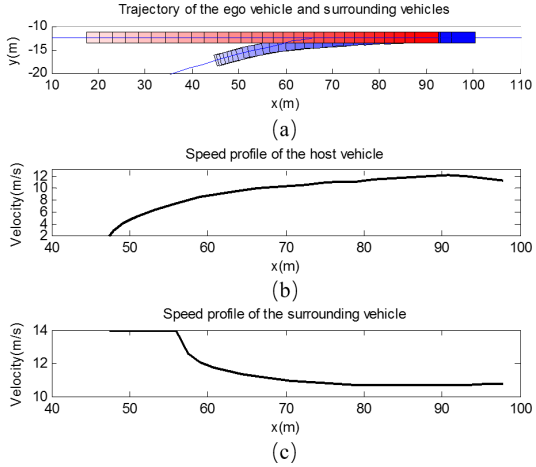

Fig. 11. MIDP in a ramp merging scenario with the surrounding vehicle passing

calculates the best action to interact with stochastic moving obstacles while guaranteeing safety. The method is applied to a ramp merging scenario and the stochastic behavior behavior models of the surrounding vehicles are learned from the NGSIM dataset. In the future, different complex driving scenarios with multiple surrounding vehicles will be studied.

REFERENCES

[1] Rajamani, R., Tan, H. S., Law, B. K., & Zhang, W. B. (2000). Demonstration of integrated longitudinal and lateral control for the operation of automated vehicles in platoons. IEEE Transactions on Control Systems Technology, 8(4), 695-708.

[2] Montemerlo, M., Thrun, S., Koller, D., & Wegbreit, B. (2002, July). FastSLAM: A factored solution to the simultaneous localization and mapping problem. In Aaai/iaai (pp. 593-598).

[3] Liu, C., Lin, C. Y., Wang, Y., & Tomizuka, M. (2017, May). Convex feasible set algorithm for constrained trajectory smoothing. In American Control Conference (ACC), 2017 (pp. 4177-4182). IEEE.

[4] Chen, J., Zhan, W., & Tomizuka, M. (2017, October). Constrained Iterative LQR for On-road Autonomous Driving Motion Planning. In Intelligent Transportation Systems (ITSC), 2017 IEEE 20th International Conference on. IEEE.

[5] SAE On-Road Automated Vehicle Standards Committee. (2014). Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. SAE Standard J3016, 01-16.

[6] Available Online: https://www.theguardian.com/technology/2016/mar/09/google-self-driving-car-crash-video-accident-bus.

[7] Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M. N., ... & Gittleman, M. (2008). Autonomous driving in urban environments: Boss and the urban challenge. Journal of Field Robotics, 25(8), 425-466.

[8] Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., ... & Johnston, D. (2008). Junior: The stanford entry in the urban challenge. Journal of field Robotics, 25(9), 569-597.

[9] Miller, I., Campbell, M., Huttenlocher, D., Kline, F. R., Nathan, A., Lupashin, S., ... & Garcia, E. (2008). Team Cornell's Skynet: Robust perception and planning in an urban environment. Journal of Field Robotics, 25(8), 493-527.

[10] Kohlhaas, R., Bittner, T., Schamm, T., & Zllner, J. M. (2014, October). Semantic state space for high-level maneuver planning in structured traffic scenes. In Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on (pp. 1060-1065). IEEE.

[11] Kohlhaas, R., Hammann, D., Schamm, T., & Zllner, J. M. (2015, September). Planning of high-level maneuver sequences on semantic state spaces. In Intelligent Transportation Systems (ITSC), 2015

[12] Hubmann, C., Aeberhard, M., & Stiller, C. (2016, November). A generic driving strategy for urban environments. In Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on (pp. 1010-1016). IEEE.

[13] Zhan, W., Chen, J., Chan, C. Y., Liu, C., & Tomizuka, M. (2017, June). Spatially-partitioned environmental representation and planning architecture for on-road autonomous driving. In Intelligent Vehicles Symposium (IV), 2017 IEEE (pp. 632-639). IEEE.

[14] Zhan, W., Liu, C., Chan, C. Y., & Tomizuka, M. (2016, November). A non-conservatively defensive strategy for urban autonomous driving. In Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on (pp. 459-464). IEEE.

[15] Sadigh, D., Sastry, S., Seshia, S. A., & Dragan, A. D. (2016, June). Planning for Autonomous Cars that Leverage Effects on Human Actions. In Robotics: Science and Systems.

[16] Next Generation Simulation (NGSIM). [Online]. Available: https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm.

[17] C. Liu, J. Chen, T. Nguyen and M. Tomizuka. The Robustly-Safe Automated Driving System for Enhanced Active Safety. No. 2017-01-1406. SAE Technical Paper, 2017.

[18] J. Chen, C. Liu and M. Tomizuka. FOAD: Fast optimization-based autonomous driving motion planner. submitted to American Control Conference, 2018.

[19] Gu, T., Atwood, J., Dong, C., Dolan, J. M., & Lee, J. W. (2015, September). Tunable and stable real-time trajectory planning for urban autonomous driving. In Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on (pp. 250-256). IEEE.