Reachability-based Decision Making for City Driving

Ahn, H.; Berntorp, K.; Di Cairano, S.

TR2018-091 July 13, 2018

Abstract

This paper presents the design of a discrete decision making algorithm for vehicles with advanced driver-assistance and automated features. We model the system as a hybrid automaton, where transitions between discrete modes in the automaton correspond to driving mode decisions, and develop a method to determine the timing of mode transitions based on backward and forward reachable sets. The algorithm can be used either as a stand-alone component or as a method to guide an underlying motion planner to safe reference trajectories. Under certain assumptions, the algorithm guarantees safety and liveness, which can be validated through computer simulations on a city driving scenario that requires going through multiple discrete modes and includes several surrounding moving obstacles.

American Control Conference (ACC)

Reachability-based Decision Making for City Driving

Heejin Ahn, Karl Berntorp, and Stefano Di Cairano

***Abstract*—This paper presents the design of a discrete decision making algorithm for vehicles with advanced driver-assistance and automated features. We model the system as a hybrid automaton, where transitions between discrete modes in the automaton correspond to driving mode decisions, and develop a method to determine the timing of mode transitions based on backward and forward reachable sets. The algorithm can be used either as a stand-alone component or as a method to guide an underlying motion planner to safe reference trajectories. Under certain assumptions, the algorithm guarantees safety and liveness, which can be validated through computer simulations on a city driving scenario that requires going through multiple discrete modes and includes several surrounding moving obstacles.**

I. Introduction

As more sophisticated advanced driver-assistance systems (ADAS) and eventually autonomy are introduced in vehicles, more complicated decision making systems are required. To design decision making systems, several approaches have been recently proposed in the literature. A common approach is to employ *forward reachable sets*, which are sets of states reachable from a set of initial states over a finite horizon. For instance, the work [1] finds cubic splines that represent the vehicle's paths and keeps a probability of colliding with obstacles below a threshold; the work [2] makes a decision by comparing forward reachable sets of the ego vehicle associated with each high-level decision; and the work [3] formulates an optimization problem to compute an input that actively corrects the driver's estimated behaviors, at every time step without mode switching, to prevent unintended lane departures. Another approach is to leverage Monte-Carlo methods, which are sampling-based methods that provide the full coverage of possible decisions and respective safety as the number of samples goes to infinity. Threat levels are determined for the ego vehicle to make a decision by computing potential future trajectories of surrounding vehicles based on samples of their control inputs [4], [5].

However, these approaches, which restrict predicted trajectories to a finite horizon, usually do not ensure persistent feasibility [6], which means that even if the current decision making problem is feasible, some subsequent problems may be infeasible as a result of decisions made in the current time step. An approach that can address this issue is based on computing *backward reachable sets* of a given set, which are sets of initial states that can reach the given set. As long as the state of the ego vehicle is controlled to be inside the backward reachable sets of a goal set, it can reach the goal set at some future time. The same concept has been used for collision avoidance applications [7], [8]. As the state is controlled to be outside the set of initial states that cannot avoid entering a collision set, collisions can be averted.

All the authors are with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA. `hjahn@mit.edu`,`{karl.o.berntorp, dicairano}@ieee.org`

Our approach to solving the decision making problem exploits both backward and forward reachable sets. The system of a vehicle driving in cities is modeled as a hybrid automaton, where transitions between discrete modes of the automaton correspond to different decisions, triggered by discrete inputs. We formulate the decision making problem as the task of finding a discrete input controller such that a mode transition occurs when there exists a continuous input that makes the continuous state reach a goal of the subsequent mode (liveness), while not colliding with other moving obstacles (safety). We design an algorithm implementing such a discrete input controller, which determines the timing of mode transitions based on the membership of the continuous state in forward reachable sets and in backward reachable sets of a goal. Under assumptions detailed in the paper, the algorithm ensures safety and liveness, which is validated as a stand-alone system in simulations. Our algorithm can be preferably used as an add-on system to a motion planner [9], [10], which can utilize the information from the decision making system to discard or give a higher cost to trajectories. While primarily developed for city driving, the approach can be applied to other scenarios, such as highway driving.

The rest of this paper is organized as follows. In Section II, we describe city driving scenarios considered in this paper and model the system as a hybrid automaton. We state the decision making problem in Section III and explain preliminaries in Section IV. We present an algorithm that exploits reachable sets in Section V and validate the algorithm using computer simulations in Section VI. We conclude the paper with suggestions of future work in Section VII.

II. Problem Setup

Suppose a route of the ego vehicle is determined by a navigation system based on the Global Positioning System (GPS). Each route involves a series of discrete decisions, which correspond to transitions between discrete modes. In this paper, for the ease of description, we restrict our attention to four decisions on unsignalized intersections, which are changing lanes, staying in lane, braking, and crossing intersections, because these decisions enable most of the city driving scenarios. Also, these decisions are the main components considered in driving behavioral systems in DARPA Urban Challenge [12]. Other decisions, such as arriving at a destination, overtaking, or U-turn, can be included in scenarios by applying the same approach presented in this paper.

We focus on the scenario illustrated in Fig. 1, which contains all four decisions, to study decision making processes. Given a route that indicates a left turn, we construct a series of

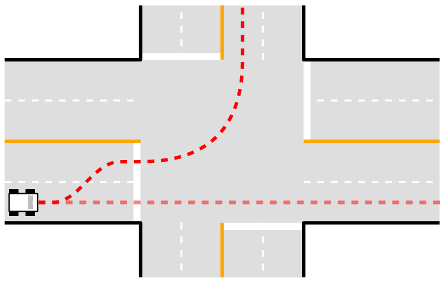


Fig. 1: Representative scenario considered in this paper. If changing lanes is infeasible due to other vehicles, the vehicle has to maintain the lane and obtain a new route.

$$\begin{tikzcd}
q_1 = LF_1 \arrow[r, bend left, "\varepsilon_1"] \arrow[dr, bend left, "\varepsilon_1'"'] & q_2 = LF_2 \arrow[r, bend left, "\varepsilon_2"] & q_3 = S_2 \arrow[r, bend left, "\varepsilon_3"] & q_4 = LF_3 \\
 & q_5 = S_1 \arrow[r, bend left, "\varepsilon_5"] & q_6 = LF_4 &
\end{tikzcd}$$

Fig. 2: Mode transitions for the scenario in Fig. 1.

required decisions for the left turn, such as changing lanes if a left turn is not allowed on the current lane, and stopping before the intersection. If changing lanes is not feasible due to the presence of other vehicles, the ego vehicle instead maintains the lane. In this case, a navigation system can find a new route for the vehicle. The ego vehicle must fully stop before unsignalized intersections, irrespective of whether it stopped earlier in the lane due to heavy traffic.

The control system of a vehicle driving in cities involves discrete mode changes and continuous dynamics of vehicles, bicycles, and/or pedestrians and thus, is modeled as a hybrid automaton. In discrete parts, we have a set of modes $Q = \{LF_1, LF_2, LF_3, LF_4, S_1, S_2\}$ as illustrated by the automaton in Fig. 2, where the subscript indicates the index of the associated lane. For example, LF_1 indicates the ego vehicle following lane 1, and S_1 the vehicle stopping before intersections in lane 1. Mode transitions are triggered by decisions, called discrete inputs ε. For example, the decision of lane changing (ε_1) is to transition the mode from LF_1 to LF_2, and the decision of stopping (ε_1') from LF_1 to S_1. A set of discrete inputs $\mathcal{E}$ is $\{\varepsilon_0, \varepsilon_1, \varepsilon_1', \varepsilon_2, \varepsilon_3, \varepsilon_5\}$ where ε_0 is a void input that does not affect the mode transition. The mode transition function R is a map $Q \times \mathcal{E} \to Q$, and $R(q_i, \varepsilon_i) = q_{i+1}$ means that mode q_i changes to q_{i+1} by discrete input ε_i.

We model the vehicle dynamics using a planar kinematic model, with the dynamical state $x = (p_x, p_y, v, \theta)$ where (p_x, p_y) is the position vector, v is the speed in the vehicle's orientation (heading angle) θ:

$$\begin{aligned}
p_x(t_{k+1}) &= p_x(t_k) + T_s v(t_k)\cos(\theta(t_k)), \\
p_y(t_{k+1}) &= p_y(t_k) + T_s v(t_k)\sin(\theta(t_k)), \\
v(t_{k+1}) &= v(t_k) + T_s u_v(t_k), \\
\theta(t_{k+1}) &= \theta(t_k) + T_s u_\theta(t_k).
\end{aligned} \tag{1}$$

The speed is bounded by $[0, v_{\max}]$, the continuous input is $u = (u_v, u_\theta)$ in the space $U = U_v \times U_\theta$ where $U_v = [u_{v,\min}, u_{v,\max}]$ and $U_\theta = [u_{\theta,\min}, u_{\theta,\max}]$, and T_s denotes the sampling time where $t_{k+1} - t_k = T_s$ for all k. For short, we write the dynamics (1) as $x(t_{k+1}) = f(x(t_k), u(t_k))$. Here, we assume that f is independent of discrete modes for notation simplicity of reachable sets in Section IV, but the dependency on discrete modes can easily be included. An underlying motion planner and subsequent vehicle control typically uses more advanced vehicle models and can be made robust to modeling uncertainty, thereby handling errors between the unicycle vehicle model (1) and more sophisticated models.

The whole system that combines the discrete mode transitions with the continuous dynamics is modeled as a hybrid automaton $H = (Q, X, \mathcal{E}, U, f, R)$. We define a hybrid trajectory $(\mathbf{q}, \mathbf{x}, \boldsymbol{\varepsilon}, \mathbf{u})$ that evolves according to f and R. A sequence of discrete modes $\mathbf{q}$ is $\{q(t_k)\}_{k=0}^{N_{\text{total}}}$ where N_{total} is the total number of time steps throughout the travel of the ego vehicle. The sequence $\mathbf{q}$ is partitioned into N_{mode} subsequences $\mathbf{q}_i : I_i \to Q$, where N_{mode} is the number of discrete modes in a given route, such that $\mathbf{q}_i$ is constant over I_i. Let τ_i denote the minimum entry of I_i, and τ_i' denote the maximum entry of I_i. A sequence of continuous states $\mathbf{x}$ is a family of sequences $\{\mathbf{x}_i\}_{i=0}^{N_{\text{mode}}}$ where $\mathbf{x}_i : I_i \to X$. Similarly, $\boldsymbol{\varepsilon} = \{\boldsymbol{\varepsilon}_i\}$ and $\mathbf{u} = \{\mathbf{u}_i\}$ where $\boldsymbol{\varepsilon}_i : I_i \to \mathcal{E}$ and $\mathbf{u}_i : I_i \to U$. In this paper, boldface font, such as $\mathbf{x}_i$, denotes a sequence of values x, and $\mathbf{x}_i(t_k)$ denotes its component at the (time) index t_k. Our objective is to design a discrete input controller

$$\pi_\varepsilon : Q \times X \to 2^{\mathcal{E}}$$

that works in conjunction with any continuous input controller.

III. Problem Statement

In this paper, we seek a discrete input controller π_ε such that there exists a continuous input that makes the continuous state reach a goal while avoiding collisions with other vehicles. In this section, we first specify goals of each mode and models of other vehicles, and then formally state the problem.

A. Goals of Each Mode

Each discrete mode $q_i \in Q$ is associated with a goal $\mathcal{G}(q_i) \subset X$ that the continuous state must reach. We express the goals with respect to the $p_x - p_y$ coordinate frame given in Fig. 3. We denote by $(c_x(q_i), c_y(q_i))$ the center position of the goal of mode q_i. For example, $c_y(q_1) = c_y(q_5)$ because both modes indicate the same horizontal lane.

For the lane following mode ($q_1 = LF_1$), the goal is a set of states that satisfy $p_y \in c_y(q_1) + [-\epsilon_p, \epsilon_p]$, $v \in [v_{\min}, v_{\max}]$ where $v_{\min} > 0$ to ensure that vehicles do not stop in the middle of lane following, and $\theta \in 0 + [-\epsilon_\theta, \epsilon_\theta]$. Here, ϵ_p and ϵ_θ are

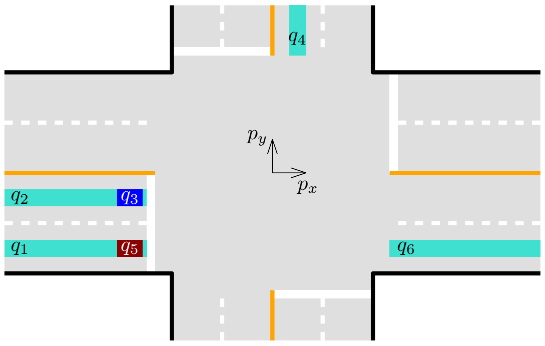


Fig. 3: Goals $\mathcal{G}(q_i)$ projected on the $p_x - p_y$ plane where q_i indicates the modes in Fig. 2.

error margins for the position and heading angle, respectively. We let $\epsilon_p = v_{\min} T_s$ and $\epsilon_\theta = \max(u_{\theta,\max} T_s, -u_{\theta,\min} T_s)$ to account for the errors of time discretization. For the stop mode ($q_5 = S_1$), the goal is a set of states that satisfy $p_x \in c_x(q_5) + [-\epsilon_p, \epsilon_p], p_y \in c_y(q_5) + [-\epsilon_p, \epsilon_p], v = 0$, and $\theta \in 0 + [-\epsilon_\theta, \epsilon_\theta]$.

B. Modeling of Other Vehicles

In the same lane, there can exist a lead vehicle or a following vehicle, which are vehicles right in front of and at the back of the ego vehicle, respectively. We denote their states by x^l and x^f, respectively. The states of vehicles that are going straight or turning left through intersections are denoted by x^{cross}. Here, by abuse of notation, we denote the states of multiple vehicles by $x^{\text{cross}} \in X$ for simplicity. Vehicles that turn right do not interfere with the ego vehicle inside intersections under the U.S. driving rules. We denote by $\mathcal{I} \subset X$ the set of states whose position is inside the intersection area.

We make several assumptions regarding the behaviors of surrounding vehicles. First, at time t_0, the estimates of $\{x^l(t_k)\}_{k=0}^N, \{x^f(t_k)\}_{k=0}^N$, and $\{x^{\text{cross}}(t_k)\}_{k=0}^N$ are available by using vehicle to vehicle communications or a sensing system in combination with motion predictors (e.g. [13]). Second, other vehicles also try to avoid collisions and do not actively seek collisions with the ego vehicle. More specifically, the vehicle behind the ego vehicle maintains a safety distance if it is given a finite time horizon N to adjust its maneuver. Lastly, other vehicles move according to the unicycle dynamics (1) but with different input bounds.

Suppose the ego vehicle is in the lane following mode of q_1. For the state sequence of the ego vehicle $\mathbf{x} = \{x(t_k)\}_{k=0}^N$, we can define errors $e^l(t_k) = x^l(t_k) - x(t_k)$ and $e^f(t_k) = x(t_k) - x^f(t_k)$, where $e^l = (e^l_x, e^l_y, e^l_v, e^l_\theta)$ and $e^f = (e^f_x, e^f_y, e^f_v, e^f_\theta)$. The error states lie in the error space $E := \{x_1 - x_2 : x_1, x_2 \in X\}$. The lead vehicle has an input u^l in the set $U^l_v \times U^l_\theta$ where $U^l_v \subset U_v$, which means that the ego vehicle can apply a larger braking than the lead vehicle to avoid collisions. The error dynamics of the position along the x-axis and the speed are

$$\begin{aligned} e^l_x(t_{k+1}) &= e^l_x(t_k) + T_s e^l_v(t_k) \cos(\theta(t_k)), \\ e^l_v(t_{k+1}) &= e^l_v(t_k) + T_s(u^l_v(t_k) - u_v(t_k)), \end{aligned} \tag{2}$$

where $e^l_y(t_k) = 0$ and $e^l_\theta(t_k) = 0$ because the vehicles are following the same horizontal lane. We write this dynamical equation as $e^l(t_{k+1}) = f_e(e^l(t_k), u_v(t_k), u^l_v(t_k))$. In case of curvy roads, the coordinate system is defined by curvilinear coordinates with respect to the centerline of the road [14], which also yields linear error dynamics.

We say that a rear-end collision occurs if $e^l_x < d_{\min}$ or $e^f_x < d_{\min}$. The set of such rear-end collision points is called a *bad set* $B := \{e \in E : e_x < d_{\min}\}$. A collision inside intersections occurs if $x(t_k) \in \mathcal{I}$ and $x^{\text{cross}}(t_k) \in \mathcal{I}$.

C. Decision Making Problem

The problem of designing π_ε is stated as follows.

Problem 1: Given (q_0, x_0), find a discrete input controller π_ε that guarantees the existence of a continuous input $\mathbf{u}$ such that for all hybrid trajectories $(\mathbf{q}, \mathbf{x}, \boldsymbol{\varepsilon}, \mathbf{u})$ where $\mathbf{x}_i(t_{k+1}) = f(\mathbf{x}_i(t_k), \mathbf{u}_i(t_k))$ and $\boldsymbol{\varepsilon}_i(t_k) \in \pi_\varepsilon(\mathbf{q}_i(t_k), \mathbf{x}_i(t_k))$ for $t_k \in I_i$,

- Liveness: $\mathbf{x}_i(\tau'_i) \in \mathcal{G}(\mathbf{q}_i(\tau'_i))$ for all i;
- Safety: $\forall t_k \in I_i, \mathbf{x}_i(t_k) - x^f(t_k) \notin B, x^l(t_k) - \mathbf{x}_i(t_k) \notin B$, and $\mathbf{x}_i(t_k) \notin \mathcal{I}$ if $x^{\text{cross}}(t_k) \in \mathcal{I}$ for all i.

IV. Preliminaries

The definitions presented in this section provide a foundation for the design and analysis of the solution to Problem 1.

A. Reachable Sets, Capture Sets, and Control Invariant Sets

A backward reachable set (or a predecessor) of a set $K \subset X$ is a set of initial states from which there is an input sequence to reach K and denoted by $\text{Pre}(K)$. A forward reachable set of a set $K \subset X$ is the set of states that can be reached with an input sequence starting from K and denoted by $\text{Reach}(K)$. The formal definitions can be found in [6]. We let $\text{Pre}^0(K) = K$ and $\text{Pre}^{k+1}(K) = \text{Pre}\left(\text{Pre}^k(K)\right)$.

Using the backward reachable set, we redefine the goals of discrete modes, when a route is associated with a series of modes $(q_1, q_2, \ldots, q_{N_{\text{mode}}})$, as $\mathcal{G}^*(q_{N_{\text{mode}}}) := \mathcal{G}(q_{N_{\text{mode}}})$ and

$$\mathcal{G}^*(q_i \to q_{i+1}) := \mathcal{G}(q_i) \cap \bigcup_{k=0}^{\infty} \text{Pre}^k(\mathcal{G}^*(q_{i+1})). \tag{3}$$

If q_{i+1} is not necessary to specify, we simply write $\mathcal{G}^*(q_i)$. Note that if $x(t_0) \in \mathcal{G}^*(q_i \to q_{i+1})$, there is an input sequence that makes $x(t_{N'}) \in \mathcal{G}^*(q_{i+1})$. Because of the geometry of intersections, we have $\mathcal{G}^*(S \to LF) = \mathcal{G}(S)$, which implies that there always exists an input sequence to cross intersections from the goal of the stop mode in the absence of other vehicles.

For rear-end collision avoidance, we define a *capture set* as the set of states that reach the bad set B for any admissible inputs. If the state is inside the capture set, there is no input to avoid a future rear-end collision.

Definition 1: The *one-step capture set* is $\mathcal{C}(B) := \{e^l : \forall u_v \in U_v, \exists u^l_v \in U^l_v, f_e(e^l, u_v, u^l_v) \in B\}$. Let $\mathcal{C}^0(B) = B$ and $\mathcal{C}^{k+1}(B) = \mathcal{C}\left(\mathcal{C}^k(B)\right)$.
The capture set refers to $\bigcup_{k=0}^{\infty} \mathcal{C}^k(B)$. The definition of the capture set only takes the lead vehicle into account, because the following vehicle, by assumption, adjusts its maneuver to

maintain a safety distance $d_{\min}$ to the ego vehicle for all future times after a finite horizon.

If the error state is outside the capture set, there exists an input u_v that keeps the error state outside the capture set for all future times. Hence, the complement set of the capture set is *control invariant* [6] for the error system (2). The stop goal $\mathcal{G}^*(S)$ is control invariant for the system (1) because the state can stay inside the set with the continuous inputs $u_v = 0$ and $u_\theta = 0$. Control invariant sets play an essential role in achieving the liveness property.

B. Motion Primitives for Heading Angles

For computationally feasible computation of the goals $\mathcal{G}^*(q_i \to q_{i+1})$, we predetermine a set of heading angle profiles, $\Theta(q_i \to q_{i+1})$, for each mode transition by relying on motion primitives with the extreme inputs. Here, we use Fig. 1 as an illustrative example, but can generate heading angle profiles for any (known) transition.

For the mode transition from LF to S, the ego vehicle maintains its heading angle to 0. Thus, $\Theta(LF \to S) = \{\bar{\boldsymbol{\theta}}\}$ where $\bar{\boldsymbol{\theta}}(t_k) = 0, \forall k \in \{0, 1, \ldots, N\}$. Let $N_{\text{leave}} = 0$, which is the time step when the vehicle leaves the current goal, and $N_{\text{reach}} = N$, which is the time step when the vehicle reaches the next goal. For the mode transition from S to LF (intersection), the heading angle changes from 0 to $\pi/2$. For a positive integer N_0, we define

$$\bar{\boldsymbol{\theta}}(t_k) = \begin{cases} 0 & \text{if } k \le N_0, \\ \bar{\boldsymbol{\theta}}(t_{k-1}) + T_s u_{\theta,\max} & \text{if } N_0 < k \text{ and } \\ & \bar{\boldsymbol{\theta}}(t_{k-1}) + T_s u_{\theta,\max} < \pi/2, \\ \pi/2 & \text{otherwise.} \end{cases}$$

We define a set of integers $\mathcal{N}_{S \to LF}(v_0)$ as $\{N_0 \in \mathbb{Z}_+ : v(t_0) = 0, v(t_{k+1}) = v(t_k) + u_{v,\max} T_s, v(t_k) \le v_0, c_x(S) + \sum_{k=0}^{N} v(t_k) \cos(\bar{\boldsymbol{\theta}}(t_k)) = c_x(I)\}$, where $v_0 \in [v_{\min}, v_{\max}]$. This is the set of integers N_0 that make the state enter $\mathcal{G}^*(LF)$. By using numerical methods, we can compute $\mathcal{N}_{S \to LF}(v_0)$. The set $\Theta(S \to LF)$ contains the heading angle profiles dependent on different $N_0 \in \mathcal{N}_{S \to LF}(v_0)$. We have $N_{\text{leave}} = 0$ and $N_{\text{reach}} = N_0 + \lfloor \pi/(2T_s u_{\theta,\max}) \rfloor$.

For the mode transition from LF_1 to LF_2, the initial heading angle is the same as the final heading angle. In the case of changing to the left lane, the heading angle increases in time and then decreases to the final value. For some positive integers N_0, we design a heading angle profile $\bar{\boldsymbol{\theta}}$ such that

$$\bar{\boldsymbol{\theta}}(t_k) = \begin{cases} \bar{\boldsymbol{\theta}}(t_{k-1}) + T_s u_{\theta,\max} & \text{if } 1 \le k - N_{\text{leave}} \le N_0, \\ \bar{\boldsymbol{\theta}}(t_{k-1}) + T_s u_{\theta,\min} & \text{if } N_0 < k - N_{\text{leave}} \le 2N_0, \\ 0 & \text{otherwise.} \end{cases}$$

We define a set of integers $\mathcal{N}_{LF_1 \to LF_2}(v_0)$ as $\{N_0 \in \mathbb{Z}_+ : v(t_k) = v_0, c_y(LF_1) + \sum_{k=0}^{N} v(t_k) \sin(\bar{\boldsymbol{\theta}}(t_k)) = c_y(LF_2)\}$. This is the set of integers N_0 that make the state enter $\mathcal{G}^*(LF_2)$. We evaluate $\mathcal{N}_{LF_1 \to LF_2}(v_0)$ by using numerical methods. For $N_0 \in \mathcal{N}_{LF_1 \to LF_2}(v_0)$, we have $N_{\text{leave}} \in [0, N - 2N_0]$ and $N_{\text{reach}} = N_{\text{leave}} + 2N_0$. The set $\Theta(LF_1 \to LF_2)$ contains the heading angle profiles corresponding to different N_0 and N_{leave}.

By defining different motion primitives, corresponding to behaviors such as parking or U-turns, we can consider scenarios with different decisions.

V. Problem solution

We define a *safe state sequence* as a state sequence that satisfies the two conditions in Problem 1 for a finite horizon.

Definition 2: A state sequence $\{x(t_k)\}_{k=0}^{N}$ is a *safe state sequence*, denoted by $\mathbf{x}_{\text{safe}}(q_{i+1}, x_0)$, if for some positive integer $N_{\text{reach}} \in \{0, \ldots, N\}$,

1) $x(t_0) = x_0$;
2) $x(t_{k+1})$ and $x(t_k)$ satisfy the dynamical model (1) for some $u(t_k) \in U$ for $k \in \{0, \ldots, N-1\}$;
3) $x(t_k) - x^f(t_k) \notin B$, $x^l(t_k) - x(t_k) \notin B$, and $x(t_k) \notin \mathcal{I}$ if $x^{\text{cross}}(t_k) \in \mathcal{I}$ for $k \in \{0, 1, \ldots, N\}$;
4) $x^l(t_k) - x(t_k) \notin \bigcup_{k=0}^{\infty} \mathcal{C}^k(B)$ for $k \in \{N_{\text{reach}}, \ldots N\}$;
5) If $q_{i+1} = LF$, $x(t_{N_{\text{reach}}}) \in \mathcal{G}^*(q_{i+1})$;
6) If $q_{i+1} = S$, $x(t_k) \in \mathcal{G}(S)$ if $x^l(t_k) \notin \mathcal{G}(S)$ for all $k \in \{N_{\text{reach}}, \ldots, N\}$.

In Definition 2, 1) and 2) indicate that the sequence is a solution of the dynamical equation (1). Conditions 3) and 4) guarantee the safety condition, and 5) and 6) guarantee the liveness condition during the finite horizon. In 6), $x(t_k) \in \mathcal{G}(S)$ is not attainable when the lead vehicle occupies the stop goal. In this case, 4) ensures that the ego vehicle can fully stop after the stationary lead vehicle. We set a finite horizon N such that without presence of obstacles, the vehicle can change lanes, cross intersections, or stop within the finite horizon.

If there is a safe state sequence $\mathbf{x}_{\text{safe}}(q_{i+1}, x_0)$, the discrete input controller $\pi_\varepsilon(q_i, x_0)$ can change the mode to q_{i+1} from q_i. Otherwise, it maintains the current mode or changes the mode to another mode by issuing ε_i', which exists only when ε_i is for changing lanes (see Fig. 2). The discrete input controller returns ε_i' after waiting until the continuous state exits the goal $\mathcal{G}^*(q_i \to q_{i+1})$, that is, until it is impossible to reach the goal of q_{i+1}. To sum up, the controller is designed as follows:

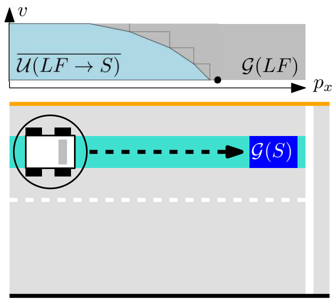
$$\pi_\varepsilon(q_i, x_0) = \begin{cases} \varepsilon_i & \text{if } x_0 \in \mathcal{G}^*(q_i \to q_{i+1}) \text{ and } \exists \mathbf{x}_{\text{safe}}(q_{i+1}, x_0), \\ \varepsilon_i' & \text{if } x_0 \in \mathcal{G}^*(q_i \to q_{i+1}), \varepsilon_i' \in \mathcal{E}, \\ & \text{and } \text{Reach}^*(x_0) \cap \mathcal{G}^*(q_i \to q_{i+1}) = \emptyset, \\ \varepsilon_0 & \text{otherwise.} \end{cases} \tag{4}$$

Here, $\text{Reach}^*(x_0) = \{x \in \text{Reach}(x_0) : x^l(t_1) - x \notin \bigcup_{k=0}^{\infty} \mathcal{C}^k(B)\}$. Then, $\pi_\varepsilon(q_i, x_0)$ is the solution to Problem 1.

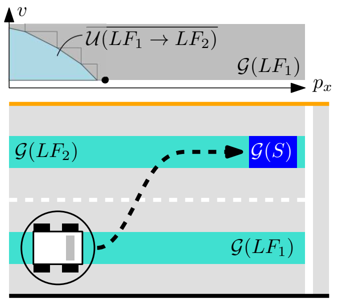
In the following, we present an algorithm implementing (4) based on a computationally feasible method of determining each condition in (4).

A. Membership in the Goal and in the Capture Set

Given a set of points (p_x^i, v^i) for $i \in \{0, 1, \ldots\}$ with ordering $p_x^{i+1} \le p_x^i$ and $v^i \le v^{i+1}$, let $\mathcal{P}$ be the set of states $x \in K \subseteq X$ such that $v \le v^i$ and $p_x \le p_x^i$ for some i. To convexify the set, we define $\overline{\mathcal{P}}$ as a polyhedron that is tightly



(a)



(b)

Fig. 4: Projections of the goals. $\overline{\mathcal{U}(LF \to S)}$ and $\overline{\mathcal{U}(LF_1 \to LF_2)}$ are convex under-approximations of the goals $\mathcal{G}^*(LF \to S)$ and $\mathcal{G}^*(LF_1 \to LF_2)$, respectively.

contained in the set $\mathcal{P}$ such that $\overline{\mathcal{P}} = \{x \in K : a^i p_x + b^i v \leq 1, \forall i \in \{1, 2, \ldots\}\}$, where a^i and b^i satisfy $a^i p_x^i + b^i v^{i-1} = 1$ and $a^i p_x^{i+1} + b^i v^i = 1$.

1) Determination of $x_0 \in \mathcal{G}^*(q_i \to q_{i+1})$: An under-approximation of $\mathcal{G}^*(q_i \to q_{i+1})$, denoted by $\mathcal{U}(q_i \to q_{i+1})$, is given as follows. We have $\mathcal{U}(S \to LF) = \mathcal{G}(S)$ because the set $\mathcal{G}^*(S) = \mathcal{G}(S)$ is known.

For the mode transition from LF to S, we first let $x(t_0) = (c_x(S), c_y(S), 0, 0) \in \mathcal{G}(S)$, and search for the smallest integer N_0 such that $x(t_{-N_0}) \in \mathcal{G}(LF)$, where the states $x(t_{-k})$ for all $k \in \{1, \ldots, N_0\}$ satisfy the unicycle model (1). The integer N_0 is computed by applying $u_v(t_{-k}) = u_{v,\min}$ and $u_\theta(t_{-k}) = 0$ for all $k \in \{1, \ldots, N_0\}$ and checking when $v(t_{-k}) \geq v_{\min}$. We have $v(t_{-N_0}) = v_{\min}$ by applying an appropriate input at time t_{-N_0}. We denote the state $x(t_{-N_0})$ by x_S^*. Then, an under-approximation of $\mathcal{G}^*(LF \to S)$ is

$$\mathcal{U}(LF \to S) := \{x \in \mathcal{G}(LF) : \exists i \in \mathbb{Z}_{\geq 0} : p_x \leq p_x^i, v \leq v^i, \text{ where } x^0 = x_S^*, v^{i+1} = v^i - T_s u_{v,\min}, p_x^{i+1} = p_x^i - T_s v^{i+1}\}.$$

This is the set of continuous states from which stop can be achieved before the stop line by applying $u_v(t_k) = u_{v,\min}$. Its convex subset $\overline{\mathcal{U}(LF \to S)}$ is depicted in Fig. 4a.

For the mode transition from LF_1 to LF_2, we let $x(t_0) = x_{S_2}^*$ and find $x(t_{-2N_0}) \in \mathcal{G}(LF_1)$ where $N_0 \in \mathcal{N}_{LF_1 \to LF_2}(v_{\min})$ by applying the inputs $u_v(t_{-k}) = 0$ and

$$u_\theta(t_{-k}) = \begin{cases} u_{\theta,\min} & \text{if } k \leq N_0, \\ u_{\theta,\max} & \text{if } N_0 < k \leq 2N_0 \end{cases}$$

to the unicycle model (1). Since the maximal steering input is considered, the states $x(t_{-k})$ for $k \in \{1, 2, \ldots, 2N_0\}$ exhibit the sharpest lane changing curve on the $p_x - p_y$ plane. We denote the state $x(t_{-2N_0})$ by x_{LF}^*. Then, an under-approximation of $\mathcal{G}^*(LF_1 \to LF_2)$ is

$$\mathcal{U}(LF_1 \to LF_2) := \{x \in \mathcal{G}(LF_1) : \exists i \in \mathbb{Z}_{\geq 0} : p_x \leq p_x^i, v \leq v^i, \text{ where } x^0 = x_{LF}^*, v^{i+1} = v^i - T_s u_{v,\min}, p_x^{i+1} = p_x^k - T_s v^{i+1}\},$$

which is the set of continuous states that can slow down to $v_{\min}$ and change lanes by applying maximal steering inputs. Its convex subset is depicted in Fig. 4b.

We use the convex set $\overline{\mathcal{U}(q_i \to q_{i+1})} \subset \mathcal{U}(q_i \to q_{i+1})$ to determine $x_0 \in \mathcal{G}^*(q_i \to q_{i+1})$ because $\mathcal{U}(q_i \to q_{i+1})$ is not convex due to time discretization.

2) Determination of $e_0 \notin \bigcup_{k=0}^{\infty} \mathcal{C}^k(B)$: We compute an under-approximation of the complement set of the capture set. Recall that the bad set is the set of error states such that $e_x < d_{\min}$. An under-approximation $\mathcal{S}$ is

$$\mathcal{S} := \{e \in E : \exists i \in \mathbb{Z}_{\geq 0} : e_x \geq e_x^i, e_v \geq e_v^i \text{ where } e_x^0 = d_{\min}, e_v^0 = 0, e_v^{i+1} = e_v^i - T_s(u_{v,\min}^l - u_{v,\min}), e_x^{i+1} = e_x^i - T_s e_v^{i+1}\}.$$

This is the set of error states where the ego vehicle can avoid rear-end collisions with the lead vehicle by applying full braking, even in the worst case when the lead vehicle also fully decelerates. The convex set $\overline{\mathcal{S}} \subset \mathcal{S}$ is defined as $\overline{\mathcal{S}} := \{e \in E : e_x \geq d_{\min}, a^i e_x + b^i e_v \geq 1, \forall i \in \{1, 2, \ldots\}\}$, where $a^i e_x^i + b^i e_v^{i-1} = 1$ and $a^i e_x^{i+1} + b^i e_v^i = 1$. We can prove that $\overline{\mathcal{S}}$ enables the determination of $e_0 \notin \bigcup_k \mathcal{C}^k(B)$.

3) Determination of $\text{Reach}^*(x_0) \cap \mathcal{G}^*(q_i \to q_{i+1}) = \emptyset$: This condition is considered only when the mode changes from LF_1 to LF_2, and $x_0 \in \underline{\mathcal{G}^*(q_i \to q_{i+1})}$. Thus, we check $f(x_0, (u_{v,\min}, 0)) \in \overline{\mathcal{U}(LF_1 \to LF_2)}$ to determine if $\text{Reach}^*(x_0) \cap \overline{\mathcal{U}(LF_1 \to LF_2)} = \emptyset$. This is because if $f(x_0, (u_{v,\min}, 0)) \notin \overline{\mathcal{U}(LF_1 \to LF_2)}$, then for any $u_v > u_{v,\min}$, $f(x_0, (u_v, 0)) \notin \overline{\mathcal{U}(LF_1 \to LF_2)}$ by the definition of $\mathcal{U}(LF_1 \to LF_2)$.

B. Safe state trajectory

We can find a safe state trajectory by solving a linear programming problem for a given $\bar{\boldsymbol{\theta}} \in \Theta(q_i \to q_{i+1})$. Due to the fixed heading angle based on motion primitives and the convex under-approximations, Conditions 1), 2), 4), 5), and 6) in Definition 2 are linear with respect to a set of decision variables $\{p_x(t_k), p_y(t_k), v(t_k), u_v(t_k)\}_{k=0}^N$. In Condition 3), the ego vehicle sometimes has to choose the position to which it changes lanes. In other words, the ego vehicle chooses which vehicles will be future lead or following vehicles. In this case, we check whether there is a choice that yields a feasible safe state sequence. It can be shown that Condition 3) can also be written as a (mixed integer) linear inequality.

C. Discrete Input Controller (Algorithm)

By employing the results in the previous sections, we design the following algorithm, which returns a discrete input and a reference trajectory.

Algorithm 1 Discrete input controller $\pi_\varepsilon(q_i, x_0)$

```
1: if $x_0 \notin \overline{\mathcal{U}(q_i \to q_{i+1})}$ then
2:     $\pi_\varepsilon(q_i, x_0) \leftarrow \varepsilon_0$, $\mathbf{x}_{\text{ref}} \leftarrow \mathbf{x}_{\text{safe}}(q_i, x_0)$
3: else
4:     for all $\bar{\theta} \in \Theta(q_i \to q_{i+1})$ do
5:         if $\exists \mathbf{x}_{\text{safe}}(q_{i+1}, x_0)$ then
6:             $\pi_\varepsilon(q_i, x_0) \leftarrow \varepsilon_i$, $\mathbf{x}_{\text{ref}} \leftarrow \mathbf{x}_{\text{safe}}(q_{i+1}, x_0)$
7:             return $(\pi_\varepsilon(q_i, x_0), \mathbf{x}_{\text{ref}})$
8:     if $\varepsilon_i' \in \mathcal{E}$ then
9:         if $f(x_0, (u_{v,\min}, 0)) \notin \overline{\mathcal{U}(q_i \to q_{i+1})}$, then
```

```
10:             $\pi_\varepsilon(q_i, x_0) \leftarrow \varepsilon_i'$, $\mathbf{x}_{\text{ref}} \leftarrow \mathbf{x}_{\text{safe}}(q_{i+1}', x_0)$
11:         else
12:             $\pi_\varepsilon(q_i, x_0) \leftarrow \underline{\varepsilon_0, \mathbf{x}_{\text{ref}} \leftarrow \mathbf{x}_{\text{safe}}(q_i, x_0)}$ with a
    constraint that $\mathbf{x}_{\text{ref}}(t_1) \in \overline{\mathcal{U}}(q_i \to q_{i+1})$
13:     else $\pi_\varepsilon(q_i, x_0) \leftarrow \varepsilon_0$,   $\mathbf{x}_{\text{ref}} \leftarrow \mathbf{x}_{\text{safe}}(q_i, x_0)$
14: return $(\pi_\varepsilon(q_i, x_0), \mathbf{x}_{\text{ref}})$
```

The algorithm implements the discrete input controller (4) with $\overline{\mathcal{U}}(q_i \to q_{i+1})$ in place of $\mathcal{G}^*(q_i \to q_{i+1})$. If there is a safe state sequence (line 5), then the discrete input controller triggers the mode change. Otherwise, it returns a void input (line 13) or returns another discrete input ε_i', if it is available (line 8) and if the continuous state at the next step is outside of the backward reachable set of the goal of q_{i+1} (line 9). It can be proved that if the continuous state tracks $\mathbf{x}_{\text{ref}}$ exactly, Algorithm 1 solves Problem 1.

VI. Simulation

We implemented the controller (4) as a stand-alone system where the vehicle tracks $\mathbf{x}_{\text{ref}}$ exactly.[1]

The simulation results are shown in Fig. 5. Each panel is associated with a mode on which the ego vehicle operates. For example, in (b), the mode changes to following the next lane, and thus, the vehicle starts changing lanes. Note that the vehicle sequentially operates on discrete modes q_1, q_2, q_3, q_4 illustrated in Fig. 2. The goal of each mode is achieved, thereby satisfying the liveness condition of Problem 1, and the vehicle does not collide with other vehicles, thereby satisfying the safety condition of Problem 1.

In the simulated scenario, Algorithm 1 takes less than 0.07 s per iteration at sampling period of 0.1 s. The computation time can be reduced by decreasing the number of heading angle profiles, and obviously by implementing the algorithm in C.

VII. Conclusion

We have presented the design of a discrete input controller for city driving scenarios, which enables a mode transition when there exists a continuous input sequence that makes the continuous state reach the goal of the next mode while avoiding collisions with other vehicles. The controller guarantees safety and liveness, which can be validated using computer simulations. The controller exhibits a promising computation time for real-time implementation because of the computationally feasible method of computing reachable sets based on approximations and motion primitives.

The initial result of the decision making system will be extended to more complex vehicle dynamics by combining our discrete input controller with motion planners, and by introducing disturbances around nominal trajectories that account for the mismatch between the simplified and actual vehicle behaviors, unmodeled dynamics, and road uncertainties. We also plan to perform computation studies and validate the algorithm on a Hamster robot platform [15].

[1]Using MATLAB on a personal computer consisting of Intel Core i7-3770S with 8 GB RAM.

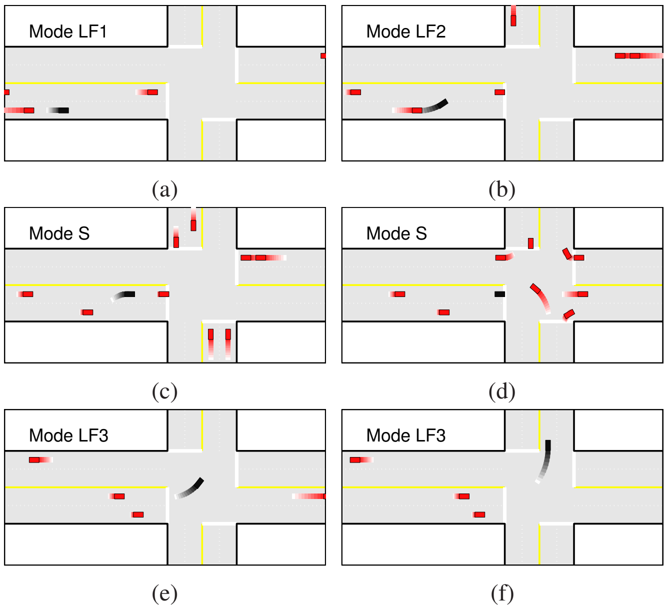


Fig. 5: The ego vehicle (black boxes) initially follows a lane in (a), changes lanes in (b), starts braking in (c), waits until the intersection is unoccupied in (d), crosses the intersection in (e), and follows a new lane in (f).

References

[1] J. Hardy and M. Campbell, "Contingency planning over probabilistic obstacle predictions for autonomous road vehicles," *IEEE Trans. Robot.*, vol. 29, no. 4, pp. 913–929, Aug 2013.
[2] S. Söntges and M. Althoff, "Computing possible driving corridors for automated vehicles," in *IEEE Intell. Veh. Symposium*, June 2017, pp. 160–166.
[3] A. Gray, M. Ali, Y. Gao, J. K. Hedrick, and F. Borrelli, "A unified approach to threat assessment and control for automotive active safety," *IEEE Trans. Intell. Transport. Syst.*, vol. 14, no. 3, pp. 1490–1499, 2013.
[4] G. S. Aoude, B. D. Luders, K. K. H. Lee, D. S. Levine, and J. P. How, "Threat assessment design for driver assistance system at intersections," in *IEEE Int. Conf. Intell. Transport. Syst.*, Sep 2010, pp. 1855–1862.
[5] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using Monte Carlo sampling," *IEEE Trans. Intell. Transport. Syst.*, vol. 9, no. 1, pp. 137–147, 2008.
[6] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.
[7] D. Hoehener, G. Huang, and D. Del Vecchio, "Design of a lane departure driver-assist system under safety specifications," in *IEEE Conf. Decision and Control*, Dec 2016, pp. 2468–2474.
[8] H. Ahn and D. Del Vecchio, "Safety verification and control for collision avoidance at road intersections," *IEEE Trans. Automat. Contr.*, 2017 (Early Access).
[9] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, 2016.
[10] K. Berntorp and S. Di Cairano, "Particle filtering for online motion planning with task specifications," in *Amer. Control Conf.*, Jul 2016.
[11] K. Berntorp, A. Weiss, C. Danielson, I. Kolamovsky, and S. Di Cairano, "Automated driving: Safe motion planning using positively invariant sets," in *IEEE Conf. Intell. Transport. Syst.*, (To appear).
[12] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA urban challenge: autonomous vehicles in city traffic*. springer, 2009, vol. 56.
[13] K. Okamoto, K. Berntorp, and S. Di Cairano, "Similarity-based vehicle-motion prediction," in *Amer. Control Conf.*, May 2017, pp. 303–308.
[14] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.
[15] The Hamster. [Online]. Available: http://cogniteam.com/hamster4.html