Monotonicity-Based Symbolic Control for Safety in Real Driving Scenarios

Stanley Smith
Adnane Saoud
Murat Arcak

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2021-5
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2021/EECS-2021-5.html

March 4, 2021

Monotonicity-Based Symbolic Control for Safety in Real Driving Scenarios

Stanley W. Smith, Adnane Saoud, and Murat Arcak

***Abstract*— In this letter we use a monotonicity-based approach to design a safety controller in two realistic driving situations: a vehicle-following scenario and an unprotected left turn scenario. For each scenario we construct a symbolic abstraction of the system and efficiently synthesize a safety controller by exploiting the monotonicity property of the dynamics. In the vehicle-following scenario, we show how monotonicity-based reasoning can be used to handle model uncertainty; in the unprotected left turn scenario, we show how such reasoning can be applied to efficiently deal with the complex scenario of two oncoming vehicles.**

I. Introduction

A commonly studied traffic situation is the vehicle-following scenario. For this scenario, one typically designs a controller for the ego vehicle with the goal of meeting a particular specification, e.g. ensuring a safety constraint on the distance between the ego and lead vehicle is maintained at all times [1]. In some cases, it is beneficial to relax this safety specification slightly - for example, in *vehicle platooning*, where the goal is to have a group of vehicles drive closely together in a tight formation (see [2] and [3]). To this end, in [4] the authors allow a soft impact with bounded relative velocity to occur in a worst-case driving scenario. By doing so, the time required to safely execute a platoon join maneuver, for example, is reduced.

Another common maneuver that a driver must execute is an unprotected left turn. Recent works have studied this scenario due to its complexity; see *e.g.* [5]. In [6] the authors consider an analogous scenario of highway merging for connected vehicles, where the ego vehicle can merge either ahead of or behind the other vehicle. In each case, the state space is separated into conflict, nonconflict, and uncertain regions, where the boundaries of these regions are dependent on the acceleration capabilities of each vehicle. Similarly, in [7] the authors compute a *capture set*, i.e. the set of states that lead to conflict regardless of input choice. In particular, this computation can be done efficiently if the system has an order preserving property. The authors propose a control map ensuring the capture set is avoided, and the approach is demonstrated on an example where two connected vehicles approach an intersection, and also on an autonomous roundabout scenario in [8].

In this letter, we apply symbolic control techniques from [9] and [10] to both driving scenarios mentioned above. Symbolic control techniques have multiple advantages - for example, they can handle complex specifications, and can be applied directly to nonlinear systems. In contrast, [4], [6], and [8] ignore nonlinearities in the vehicle dynamics, and in [7] the authors use feedback linearization. Similarly, in [11] the authors only consider nonlinear vehicle dynamics on a restricted input space, otherwise using a linear approximation. Furthermore, we note that by exploiting the monotonicity property of the system dynamics, we can reduce the computational complexity of the controller synthesis and implementation [10]. In the vehicle-following scenario we also use monotonicity-based reasoning to consider model uncertainty, and in the left turn scenario we apply it to efficiently handle two oncoming vehicles.

The contribution of this work is two-fold. First, we showcase the flexibility of symbolic control techniques by applying them in two realistic driving situations: a vehicle-following scenario in Section III, and an unprotected left turn scenario in Section IV, each of which is of independent interest. Second, to deal with the specification in each scenario we construct a non-standard abstraction, in which we introduce a sink state to transform the considered specifications into *lower closed safety specifications* [10]. We also introduce a new construction of the transition relation which ensures monotonicity of the abstraction.

II. Monotonicity Concepts

In this section, we overview the monotonicity and symbolic control concepts we will use throughout the letter.

A. Partial orders

A binary relation $\leq_{\mathcal{L}} \subseteq \mathcal{L} \times \mathcal{L}$ is a partial order if and only if for all $l_1, l_2, l_3 \in \mathcal{L}$ we have: (i) $l_1 \leq_{\mathcal{L}} l_1$, (ii) if $l_1 \leq_{\mathcal{L}} l_2$ and $l_2 \leq_{\mathcal{L}} l_1$ then $l_1 =_{\mathcal{L}} l_2$ and, (iii) if $l_1 \leq_{\mathcal{L}} l_2$ and $l_2 \leq_{\mathcal{L}} l_3$ then $l_1 \leq_{\mathcal{L}} l_3$. Given a partially ordered set $\mathcal{L}$, for $a \in \mathcal{L}$ the lower closure of the element $a \in \mathcal{L}$ is denoted $\downarrow a$ and defined as $\downarrow a := \{x \in \mathcal{L} : x \leq_{\mathcal{L}} a\}$. Similarly the upper closure of $a \in \mathcal{L}$ is defined as $\uparrow a := \{x \in \mathcal{L} : a \leq_{\mathcal{L}} x\}$. The lower closure (respectively upper closure) of a set $A \subseteq \mathcal{L}$ is $\downarrow A := \bigcup_{a \in A} \downarrow a$ (respectively $\uparrow A = \bigcup_{a \in A} \uparrow a$). A subset $A \subseteq \mathcal{L}$ is said to be lower-closed (respectively upper-closed) if $\downarrow A = A$ (respectively $\uparrow A = A$).

B. Monotone Transition Systems

Below we recall the notion of a transition system [12] and define transition systems that preserve order on input and state spaces.

Definition 1: A transition system is a tuple $T = (X, X_0, U, \Delta)$, where X is a set of states, $X_0 \subseteq X$ is a

This work was supported by NSF grants ECCS-1906164, CNS-1545116, AFOSR grant FA9550-18-1-0253, and an NDSEG Graduate Fellowship.

The authors are with the EECS Dept., University of California, Berkeley {swsmth,asaoud,arcak}@berkeley.edu.

set of initial states, U is a set of inputs and $\Delta : X \times U \to X$ is a deterministic transition relation. A transition system is said to be finite if X and U are finite.

Definition 2: A transition system $T = (X, X_0, U, \Delta)$ is said to be input-state monotone if X and U are equipped with partial orders $\leq_X$, $\leq_U$, respectively, and for all $x_1, x_2 \in X$, for all $u_1, u_2 \in U$, with $x_1 \leq_X x_2$ and $u_1 \leq_U u_2$, it follows that $\Delta(x_1, u_1) \leq_X \Delta(x_2, u_2)$.

C. Controller Synthesis for Safety Specifications

1) Maximal safety controller: Given a transition system $T = (X, X_0, U, \Delta)$, a controller for T is a set-valued map $\mathcal{C} : X \rightrightarrows U$ and its domain is defined as $dom(\mathcal{C}) = \{x \in X : \mathcal{C}(x) \neq \emptyset\}$. A safety controller is then defined as:

Definition 3: A safety controller $\mathcal{C}$ for the transition system $T = (X, X_0, U, \Delta)$ and the safe set $X^S \subseteq X$ satisfies:

- $dom(\mathcal{C}) \subseteq X^S$;
- $\forall x \in dom(\mathcal{C})$ and $\forall u \in \mathcal{C}(x)$, $\Delta(x, u) \subseteq dom(\mathcal{C})$.

A suitable solution to the safety problem is a controller that enables as many actions as possible. This controller $\mathcal{C}^*$ is said to be a maximal safety controller, in the sense that for any other safety controller and for all $x \in X$, we have $\mathcal{C}(x) \subseteq \mathcal{C}^*(x)$.

2) Lazy controller synthesis for safety specifications: Consider an input-state monotone transition system $T = (X, X_0, U, \Delta)$ and a safety specification $X^S \subseteq X$. The safety specification X^S is said to be lower closed (respectively, upper closed) if X^S is a lower closed (respectively, upper closed) subset of X. While classical approaches to deal with general safety specifications rely on the use of the fixed-point algorithm [12], efficient symbolic abstractions and lazy synthesis approaches have been proposed recently in [9] and [10], respectively, making it possible to deal with upper and lower safety specifications. These approaches allow us to compute the maximal safety controller while reducing the computational cost required for the synthesis and implementation of the maximal safety controller.

III. Vehicle-Following Scenario

In this section, we consider a vehicle-following scenario. We first introduce the vehicle dynamics model that we use and present the control objective. We then use the monotonicity properties of the model to construct a symbolic abstraction and to synthesize a controller.

A. Monotone Vehicle Dynamics

The vehicle-following model is:

$$\begin{aligned} \dot{h} &= v_L - v, \\ \dot{v} &= \alpha(T, v, \theta), \\ \dot{v}_L &= \alpha(T_L, v_L, \theta_L), \end{aligned} \tag{1}$$

where $h \in \mathbb{R}$ is the headway between the vehicles, $v,\ T \in \mathbb{R}$ are the velocity and wheel torque for the ego vehicle, $v_L,\ T_L \in \mathbb{R}$ are the velocity and wheel torque for the lead vehicle, and $\theta,\ \theta_L \in \mathbb{R}^5$ contain modelling parameters. The individual vehicle dynamics evolve according to

$$\alpha(T, v) := \begin{cases} \frac{1}{M}\left(\frac{T}{R_w} - \frac{1}{2}\rho v^2 C_D A\right), & v > 0, \\ \max(T, 0), & v = 0, \end{cases} \tag{2}$$

which ensures that the vehicles never reverse, that is $v(t) \geq 0$ and $v_L(t) \geq 0$ for $t \geq 0$. Furthermore, (2) contains the following modelling parameters: $M > 0$ is the vehicle mass, $R_w > 0$ is the wheel radius, $\rho > 0$ is air density, $C_D > 0$ is the vehicle drag coefficient, and $A > 0$ is the vehicle reference area. We collect all modelling parameters in $\theta := [M;\ R_w;\ \rho;\ C_D;\ A]$ for the ego vehicle and, similarly, in θ_L for the lead vehicle (which may have different modelling parameters). We assume for the sake of simplicity the values of $M = 2044$kg, $R_w = 0.3074$m, and $A = 2.6292\text{m}^2$ are known and identical for each vehicle. Furthermore, air density $\rho = 1.206\text{kg/m}^3$, but the air drag coefficient for each vehicle is unknown and belongs to the set $C_D,\ C_D^L \in [C_D^{\min}, C_D^{\max}]$, where $C_D^{\min} = 0.20$ and $C_D^{\max} = 0.40$. Since these parameters are unknown, we aim to set C_D and C_D^L to their worst-case values during controller synthesis.

Next, we define the state of (1) as $x(t) := [h(t);\ v(t);\ v_L(t)]$, the input $u(t) := T(t)$, and the disturbance $w(t) := T_L(t)$, each of which are assumed to lie within a corresponding constraint set at all times

$$\begin{aligned} X &:= \big\{x : 0 \leq v \text{ and } 0 \leq v_L \leq 20\big\}, \\ U &:= \big\{u : -1800 \leq T \leq 1200\big\}, \\ W &:= \big\{w : -2500 \leq T_L \leq 1200\big\}. \end{aligned} \tag{3}$$

The solution of the vehicle model (1) at time $t > 0$, from an initial condition $x_0 \in X$, under a control input $u : [0, t] \to U$, a disturbance input $w : [0, t] \to W$ and a vector of unknown parameters $[\theta; \theta_L]$ is denoted $\Phi(t; x_0, u, w, [\theta; \theta_L])$. Hence, under the same conditions, the reachable set over the time interval $[0, t]$ reads $\Phi([0, t]; x_0, u, w, [\theta; \theta_L])$. Several methods exist for the computation of overapproximations of such reachable sets for the class of monotone systems considered in this letter [13].

Finally, we equip the state, input and disturbance spaces of the model in (1) with the partial orders

$$\begin{aligned} &(x_1 \leq_X x_2) \iff \big[(h_1 \geq h_2) \wedge (v_1 \leq v_2) \wedge (v_{L,1} \geq v_{L,2})\big] \\ &(u_1 \leq_U u_2) \iff (T_1 \leq T_2), \\ &(w_1 \leq_W w_2) \iff (T_{L,1} \geq T_{L,2}) \end{aligned} \tag{4}$$

where $\leq$ is the usual partial order on $\mathbb{R}$. With the partial order defined above, it is easy to verify that the dynamics in (1) are monotone, which can be done via the Kamke-Muller conditions [14]. This property states that for $x_1 \leq_X x_2$, $u_1 \leq_U u_2$, and $w_1 \leq_W w_2$, we have:

$$\Phi(t; x_1, u_1, w_1, [\theta; \theta_L]) \leq \Phi(t; x_2, u_2, w_2, [\theta; \theta_L]), \quad \forall t \geq 0. \tag{5}$$

B. Control Objective

We now discuss the control objective we want the ego vehicle to satisfy. Typically, one would require

$$x(t) \in X \cap \mathcal{H}, \quad \forall t \geq 0 \tag{6}$$

where

$$\mathcal{H} := \big\{x : \ 0 < h \text{ and } v \leq 20\big\}. \tag{7}$$

From the definition of the set of constraints X in (3), the condition $x(t) \in X$, for all $t \geq 0$ is already satisfied. The objective here is to synthesize a controller for the ego vehicle ensuring that $x(t) \in \mathcal{H}$, for all $t \geq 0$, which, as discussed in Section II-C, is a lower closed safety specification with respect to the partial order (4). In words, (6) means the ego vehicle must ensure it never collides with the lead vehicle. Moreover, the ego vehicle velocity must be bounded by the maximum velocity of 20m/s, while assuming the lead vehicle velocity is also bounded by 20m/s.

Next, we define the set of states for which a *soft impact* has occurred [4]:

$$S := \big\{x : h \leq 0 \text{ and } v - v_L \leq v_{\text{allow}}\big\}. \tag{8}$$

For our modified safety specification, we allow a soft impact to occur in a worst-case driving scenario, but never an unsafe impact - that is, one that violates (8). This is beneficial since it relaxes the restrictive constraint (6) on the ego vehicle, allowing it to follow the lead vehicle more closely, for example. We now formally state the control objective considered in this section:

Problem 1: Given the model of the vehicle-following scenario in (1), synthesize a sampled-data controller $\mathcal{C} : X \rightrightarrows U$ such that *either* (6) holds *or* the following holds:

$$\exists t_0 \geq 0 \text{ s.t. } x(t_0) \in S \text{ and } x(t) \in X \cap \mathcal{H} \text{ for } t \in [0, t_0). \tag{9}$$

The control objective described above is in the same spirit of a reach-avoid specification, in the sense that the system state must either remain in the set $X \cap \mathcal{H}$ for all time (avoiding an unsafe impact), or eventually reach the set S. We emphasize that the set S will only be reached in a worst-case scenario - for example, if the ego vehicle fails to satisfy (6) because the lead vehicle applied harsh brakes.

C. Synthesis using the symbolic approach

In this section, we design a control law $\mathcal{C} : X \rightrightarrows U$ which is a solution to Problem 1 using the symbolic control approach [12] that relies on the use of symbolic models, which are discrete abstractions of continuous dynamics.

1) Symbolic abstraction: An abstraction Σ^a for the vehicle model in (1) is a transition system $\Sigma^a := (X^a, X_0^a, U^a, \Delta^a)$, where X^a, X_0^a and U^a are finite (symbolic) sets of states and control inputs respectively, while $\Delta : X^a \times U^a \to X^a$ is a transition relation. For constructing the symbolic sets and in view of the control objective defined in Problem 1, the set X of constraints on the state-space defined in (3) is decomposed into three regions: an impact-free region, represented by the set $\mathcal{H}$ in (7), a region of soft impact, represented by the set S in (8), and a remaining unsafe region given by $X \setminus (\mathcal{H} \cup S)$. Each of these regions is represented in symbolic form as follows:

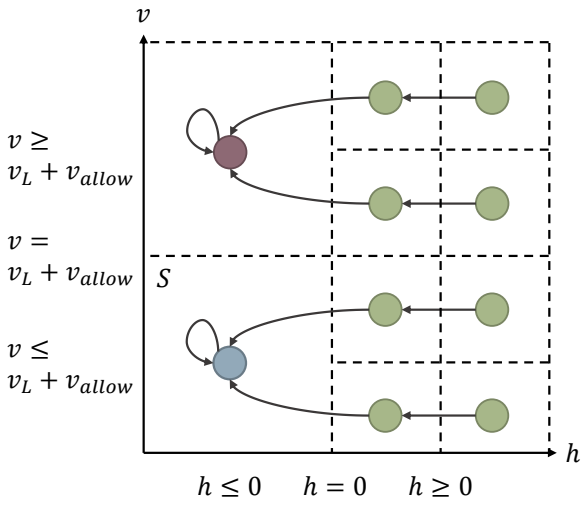


Fig. 1: The state space is divided into three areas: the area corresponding to set S (bottom left cell), the area corresponding to unsafe impacts (top left cell), and the area where no impact has occurred (right cells).

- We discretize the impact-free region $\mathcal{H}$ into $N \geq 1$ half-open intervals $q_i = (\underline{q}_i; \overline{q}_i]$ using a finite partition. Since $\mathcal{H}$ is unbounded, we follow the approach in [15, Section V-B-3] which uses bounded and unbounded intervals to construct the partition. The states of these regions are represented by the green states in Figure 1.
- We use a unique state q_{sink} to model the safe-impact region S, represented by the blue state in Figure 1.
- We use a unique state q_{unsafe} to represent the unsafe region $X \setminus (\mathcal{H} \cup S)$; see the red state in Figure 1.

The symbolic set X^a consists then of $N + 2$ states:

$$X^a := \big\{q_i : i = 1, \ldots, N\big\} \cup \{q_{sink}, q_{unsafe}\}.$$

The set of initial conditions corresponds to $X_0^a = \big\{q_i : i = 1, \ldots, N\big\}$. Moreover, we discretize the set of inputs U into $M \geq 2$ values, with the discrete input set given by

$$U^a := \big\{u_j : j = 1, \ldots, M\big\}.$$

Assuming the controller to be designed is implemented by a microprocessor with a sampling time $\tau > 0$, the transition relation $\Delta : X^a \times U^a \rightrightarrows X^a$ can be defined as follows. For any $q, q' \in X^a, u \in U^a$, $q' = \Delta(q, u)$ if and only if one of the following scenarios holds:

(i) For $q, q' \in X_0^a$ and $u \in U^a$, $q' = \Delta(q, u)$ if and only if $\Phi([0, \tau]; \overline{q}, u, T_{\max,\text{brake}}, [\theta_{\min}; \theta_{L,\max}]) \subseteq \mathcal{H}$ and $\Phi(\tau; \overline{q}, u, T_{\max,\text{brake}}, [\theta_{\min}; \theta_{L,\max}]) \in q'$;

(ii) For $q \in X_0^a \cup \{q_{sink}\}$ and $u \in U^a$, $q_{sink} = \Delta(q, u)$ if and only if $q = q_{sink}$ or there exists $s \in [0, \tau]$ such that $\Phi(s; \overline{q}, u, T_{\max,brake}, [\theta_{\min}; \theta_{L,\max}]) \in S$ and $\Phi([0, \tau]; \overline{q}, u, T_{\max,\text{brake}}, [\theta_{\min}; \theta_{L,\max}]) \subseteq \mathcal{H} \cup S$;

(iii) For $q \in X_0^a \cup \{q_{unsafe}\}$ and $u \in U^a$, $q_{unsafe} = \Delta(q,u)$ if and only if $q = q_{unsafe}$ or $\Phi([0,\tau];\overline{q},u,T_{\max,\text{brake}},[\theta_{\min};\theta_{L,\max}]) \cap (X \setminus (\mathcal{H} \cup S)) \neq \emptyset$.

In each scenario, the vector of unknown parameters $[\theta_{\min};\theta_{L,\max}]$ used in the construction of the transition relation above is given by $\theta_{\min} = [M;\ R_w;\ \rho;\ C_D^{\min};\ A]$ and $\theta_{L,\max} = [M;\ R_w;\ \rho;\ C_D^{\max};\ A]$, with the parameter values mentioned previously. This represents the worst-case values for the modelling parameters in (1). Furthermore, $T_{\max,\text{brake}} = -2500$Nm is the maximum braking torque for the lead vehicle. Intuitively, we want to underestimate how much air drag will help the ego vehicle avoid a collision, and overestimate how much it will help the lead vehicle cause one. For the construction of the transition relation, the first scenario is used to represent the impact-free case where the trajectory of the vehicles remains in the set $X \cap \mathcal{H}$. The second scenario represents the case of soft impact. Moreover, in this second scenario we added a self-loop to the sink state q_{sink} to transform the reach-avoid specification in Problem 1 to a safety problem. Finally, the last scenario is used to represent the fact that the trajectory of the vehicle is unsafe, in the sense that an unsafe impact violating (8) occurs.

Remark 1: In view of (9), a transition to q_{sink} should be created from $q \in X^a$ and $u \in U^a$ if and only if $q = q_{sink}$ or there exists $s \in [0,\tau]$ such that $\Phi(s;\overline{q},u,T_{\max,\text{brake}},[\theta_{\min};\theta_{L,\max}]) \in S$ and also $\Phi([0,s];\overline{q},u,T_{\max,\text{brake}},[\theta_{\min};\theta_{L,\max}]) \subseteq \mathcal{H} \cup S$. The latter condition is replaced in (ii) by $\Phi([0,\tau];\overline{q},u,T_{\max,\text{brake}},[\theta_{\min};\theta_{L,\max}]) \subseteq \mathcal{H} \cup S$ in order to preserve the monotonicity property of the transition system, at the cost of a small additional conservatism.

Remark 2: The proposed construction of the symbolic abstraction Σ^a makes it possible to deal with the inter-sampling behaviour. Indeed, our construction is based on continuous-time reachability, ensuring safety at continuous-time and not only at sampling instants.

2) Abstract control objective: Using such construction of the symbolic abstraction Σ^a, the concrete control objective in Problem 1 can be transformed to the following abstract control objective:

Problem 2: Given the abstraction Σ^a of the vehicle-following model in (1), synthesize the maximal discrete safety controller $\mathcal{D} : X^a \rightrightarrows U^a$ keeping the trajectories of the transition system Σ^a in the set $X_0^a \cup \{q_{sink}\}$.

To synthesize the controller $\mathcal{D}$, we rely on the use of the monotonicity concepts introduced in Section II. We first have the following result, characterizing the structural properties of the abstraction Σ^a and the considered specification.

Proposition 1: The transition system $\Sigma^a := (X^a, X_0^a, U^a, \Delta^a)$ defined above is an input-state monotone transition system and the safety specification $X_0^a \cap \{q_{sink}\}$ is lower closed.

Proof: We first show that Σ^a is an input-state upper monotone transition system. We start by defining the partial order for the discrete state and input spaces. We define a partial order $\leq X^a$ over the set of discrete states X^a as follows: for $q_1, q_2 \in X^a$, $q_1 \leq_{X_0^a} q_2$ if and only if $\overline{q}_1 \leq_X \overline{q}_2$. For the special states q_{unsafe} and q_{sink} we have the following: for all $q \in X_0^a$, $q \leq_{X^a} q_{sink} \leq_{X^a} q_{unsafe}$. Moreover, since $U^a \subseteq U$, the partial order $\leq_{U^a}$ on the discrete input space is inherited from $\leq_U$.

Consider $q_1, q_2 \in X^a$, $u_1, u_2 \in U^a$ with $q_1 \leq_{X^a} q_2$ and $u_1 \leq_{U^a} u_2$. We will show that $\Delta(q_1,u_1) \leq \Delta(q_2,u_2)$. From the definition of the monotonicity property in (5), we have that $\Phi(\tau;\overline{q}_1,u_1,T_{\max,brake},[C_D^{\max};C_D^{\max}]) \leq \Phi(\tau;\overline{q}_2,u_2,T_{\max,brake},[C_D^{\max};C_D^{\max}])$. To complete the proof, we distinguish three cases:

- $\Delta(q_1,u_1) \in X_0^a$. In this case, we have two options. If $\Delta(q_2,u_2) \in X_0^a$, in then we get directly from (5) that $\Delta(q_1,u_1) \leq \Delta(q_2,u_2)$. Otherwise, we have that $\Delta(q_2,u_2) = q_{sink}$ or $\Delta(q_2,u_2) = q_{unsafe}$, which implies from the construction of the partial order $\leq_{X^a}$ above that $\Delta(q_1,u_1) \leq \Delta(q_2,u_2)$.
- $\Delta(q_1,u_1) = q_{sink}$. In this case, we have from (5) that $\Delta(q_2,u_2) = q_{sink}$ or $\Delta(q_2,u_2) = q_{unsafe}$, which implies from the construction of the partial order $\leq_{X^a}$ above that $\Delta(q_1,u_1) \leq \Delta(q_2,u_2)$.
- $\Delta(q_1,u_1) = q_{unsafe}$. In this case we have either $q_1 \in X_0^a$ or $q_1 = q_{unsafe}$. If $q_1 \in X_0^a$, we have from the construction of the transition relation Δ and using (5) that $\Delta(q_2,u_2) = q_{unsafe}$, which implies that $\Delta(q_1,u_1) \leq \Delta(q_2,u_2)$. Similarly, if $q_1 = q_{unsafe}$ and $q_1 \leq q_2$ then $q_2 = q_{unsafe}$ and $\Delta(q_1,u_1) = q_{unsafe} \leq \Delta(q_2,u_2) = q_{unsafe}$.

Finally, the fact that the set $X_0^a \cap q_{sink}$ is lower closed follows immediately from the definition of the partial order $\leq_{X^a}$. ■

We now have all the ingredients to provide a solution to Problem 1. First, using the lazy controller synthesis approach for input-state upper monotone transition systems and directed safety specification (see Section II-C.2) we can construct the maximal abstract safety controller $\mathcal{D} : X^a \rightrightarrows U^a$ for the transition system Σ^a and lower closed safety specification $X_0^a \cup \{q_{sink}\}$, which is indeed a solution to Problem 2. Second, using the construction of the abstraction Σ^a, one can show, similarly to [9], that the abstraction Σ^a is related to the original system in (1) by an upper alternating simulation relation. This relation is useful for controller refinement for our lower closed safety specification $X_0^a \cup \{q_{sink}\}$. Based on this relationship, we can refine the abstract controller $\mathcal{D} : X^a \rightrightarrows U^a$ into a concrete controller $\mathcal{C} : X \rightrightarrows U$, providing a solution to Problem 1. In this case, the concrete controller $\mathcal{C}$ can be defined for $x \in X$ as follows: $\mathcal{C}(x) = \mathcal{D}(Q(x))$, where Q is the *quantizer* associated to the abstraction Σ^a and relating the continuous state-space X to the discrete state-space X^a as follows: $Q : X \to X^a$, with $Q(x) = q$ if and only if $x \in q$.

Using the lazy controller synthesis approach, we compute a safe set (that is, the set $Z = dom(\mathcal{C}) \subset X$ where we can enforce the given specification) with respect to both the strict specification (6) and the relaxed specification given in Problem 1. For constructing our abstraction, we use the discretization parameters given in Table I. The resulting

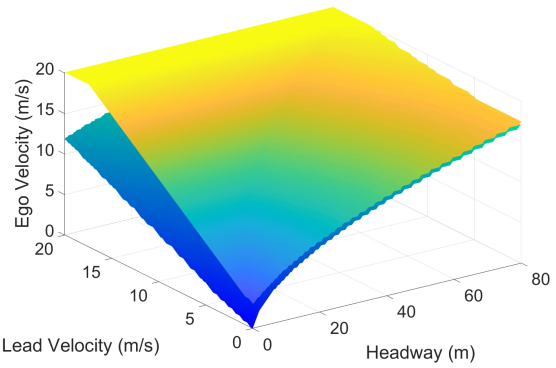


Fig. 2: Boundary of safe set $Z \subset X$ for the strict (top surface) and relaxed (bottom surface) vehicle-following specification.

TABLE I: Vehicle-Following Abstraction Parameters

h_{res}	headway resolution	m	0.4
v_{res}	velocity resolution	m/s	0.2
T_{res}	wheel torque resolution	Nm	50
τ	sampling time	s	0.2

safe sets are shown in Figure 2. As expected, relaxing the safety specification with $v_{allow} = 3$m/s expands the safe set. This allows the vehicles to drive more closely together and potentially improve traffic efficiency - for example, in vehicle platooning [3].

IV. Unprotected Left Turn Scenario

In this section, we compute a safety controller for an unprotected left turn scenario using the approach established in Section III. Indeed, the vehicle dynamics in this scenario are monotone, and collision avoidance only requires the ego vehicle to adjust its velocity along its current path [11].

A. Monotone Vehicle Dynamics and Control Objective

We model the vehicle dynamics in the unprotected left turn scenario as follows

$$\begin{aligned} \dot{s} &= v, \\ \dot{v} &= \beta(a, v), \\ \dot{r} &= v_0, \end{aligned} \tag{10}$$

where $s,\ v,\ a \in \mathbb{R}$ are the position, velocity, and acceleration of the ego vehicle along its (curved) path, and $r \in \mathbb{R}$ is the position of the oncoming vehicle along its path, see Figure 3. Furthermore, $v_0 = 12$m/s is the (constant) velocity of the oncoming vehicle, and the ego vehicle dynamics are

$$\beta(a, v) := \begin{cases} a, & v \in (0, v_{\max}), \\ \max(a, 0), & v = 0, \\ \min(a, 0), & v = v_{\max}. \end{cases} \tag{11}$$

Here, we use a simplified vehicle dynamics model - typically, the task of computing the velocity trajectory in this scenario would be handled by a *planner* system, which uses such a model to reduce complexity [16]. Importantly, we also note that the positions s and r increase in the direction of travel, and at the point $s = r = 0$ the vehicle paths cross.

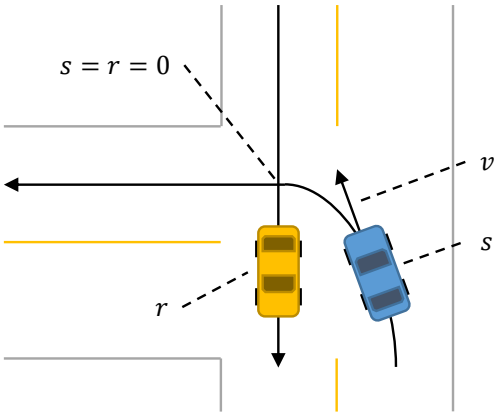


Fig. 3: Depiction of the states for the ego (blue) and oncoming (yellow) vehicle in the unprotected left turn scenario.

To address the possibility of a collision between the ego and oncoming vehicles, we define a *conflict zone* [17] around this crossing point, and require the two vehicles to never occupy the conflict zone simultaneously. Formally, we define the following set of conflicting states

$$C := \{x : |s| \leq \ell \text{ and } |r| \leq \ell\}, \tag{12}$$

where $\ell > 0$ is an adjustable parameter - here, we take $\ell = 10$m. To avoid the unsafe set (12) at all times, the ego vehicle can either go first and complete its turn before the oncoming vehicle enters the intersection, or wait for the the oncoming vehicle to pass through the intersection first, and then start its turn. For each case, we define a respective goal set

$$G^{\text{wait}} := \{x : r > \ell\}, \quad G^{\text{go}} := \{x : s > \ell\}, \tag{13}$$

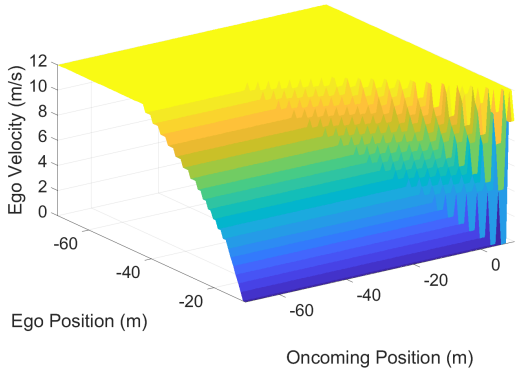
which represents the opposite side of the intersection for each vehicle. Next, we define the following constraint sets for the state and input, respectively

$$\begin{aligned} X^{\text{wait}} &:= \left\{ x : \begin{array}{l} -70 \leq s \leq -10,\ 0 \leq v \leq 12, \\ -70 \leq r \leq 10 \end{array} \right\}, \\ X^{\text{go}} &:= \left\{ x : \begin{array}{l} -70 \leq s \leq 10,\ 0 \leq v \leq 12, \\ -70 \leq r \leq -10 \end{array} \right\}, \\ U &:= \{a :\ -3 \leq a \leq 2\}, \end{aligned} \tag{14}$$

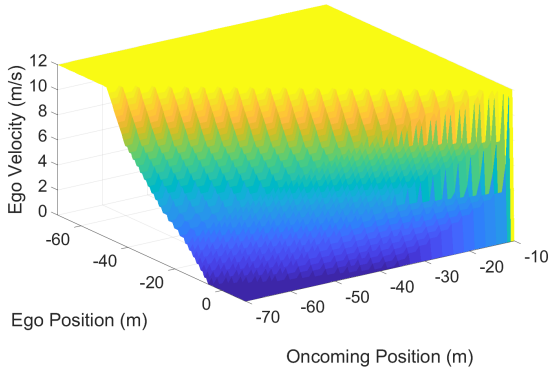
where the bounds on each of the state variables depend on the ego vehicle's strategy for executing the turn. For example, the set X^{go} excludes states where the oncoming vehicle occupies the intersection, since we want the ego vehicle to go first in this case. With (12) - (14), we state our control objective.

Problem 3: Our control objective is to ensure the conflict set is avoided at all times

$$x(t) \notin C,\ \forall t \geq 0, \tag{15}$$



(a) Safe set $Z^{\text{wait}} \subset X^{\text{wait}}$.



(b) Safe set $Z^{\text{go}} \subset X^{\text{go}}$.

Fig. 4: Boundary of each safe set for the unprotected left turn scenario.

and a goal set is eventually reached:

$$\exists t_0 \text{ s.t. } x(t) \in G^i \text{ for } t > t_0 \text{ and } x(t) \in X^i \text{ for } t \in [0, t_0] \tag{16}$$

where $i \in \{wait,\ go\}$, depending on the ego vehicle's strategy for executing the turn.

We again wish to accurately characterize the set of states $Z^{\text{wait}} \subset X^{\text{wait}}$ and $Z^{\text{go}} \subset X^{\text{go}}$ from which it is possible for the ego vehicle to safely execute its left turn, by either waiting for the oncoming vehicle or going first, respectively. Since the system dynamics are monotone, and since we are again considering a (directed) reach-avoid type specification, we are able to compute safe sets Z^{wait} and Z^{go} using the same symbolic control approach outlined in Section III-C. The resulting safe sets are shown in Figure 4.

B. Two Oncoming Vehicles

We now apply safe sets Z^{wait} and Z^{go} in an unprotected left turn scenario with two oncoming vehicles. Our goal is to design a controller for the ego vehicle such that it safely cuts in-between the two oncoming vehicles to execute its turn. Since we represent the 'wait' and 'go' strategies for executing the turn as upper and lower-closed safety specifications, we can do this by performing an incremental synthesis procedure for the intersection of an upper and lower-closed

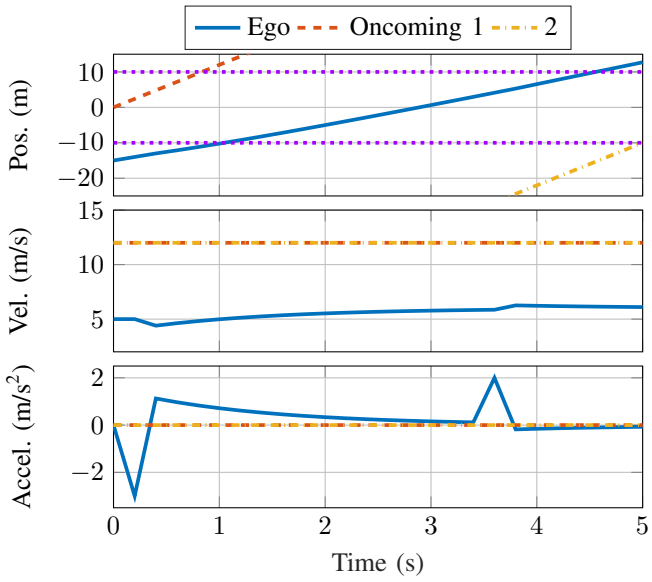


Fig. 5: Simulation results for the unprotected left turn scenario. We note two vehicles never occupy the intersection (bounded by the dotted purple lines) simultaneously.

safety specification. For more details on this approach, we refer to [18, Section 6]. The idea is to synthesize a controller which keeps the state in $Z^{\text{wait}} \cap Z^{\text{go}}$ at all times. Since the oncoming vehicles travel at constant velocity and thus are separated by constant distance, we can map points in Z^{wait} to corresponding points in Z^{go}, and vice versa.

The resulting controller is tested in simulation with the results shown in Figure 5. At each time step during simulation, we obtain a feasible range of inputs via the synthesized controller. As long as a control input in this range is selected, the ego vehicle will not conflict with either oncoming vehicle. A simple model-predictive controller is used to choose the optimal control input in this feasible range, with the objective of maintaining a velocity of 6m/s. We note that at the beginning of the simulation and at 3.4s the control input changes rapidly to avoid a conflict.

V. Conclusion

We used a monotonicity-based approach to design vehicle controllers for two realistic driving scenarios: a vehicle-following scenario and an unprotected left turn scenario. For each scenario we considered a reach-avoid type control specification and showed that we can apply a controller synthesis procedure for safety specifications by augmenting our symbolic abstraction of the system dynamics with two unique states. Since the vehicle dynamics in each scenario are monotone, the controller synthesis and implementation computations are be performed efficiently. Lastly, in the vehicle-following scenario we showed how monotonicity-type reasoning can be applied to handle model uncertainty, and in the unprotected left turn scenario we showed how it enables us to consider two oncoming vehicles.

REFERENCES

[1] P. Nilsson, O. Hussien, A. Balkan, Y. Chen, A. D. Ames, J. W. Grizzle, N. Ozay, H. Peng, and P. Tabuada, "Correct-by-construction adaptive cruise control: Two approaches," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 4, pp. 1294–1307, 2015.

[2] S. W. Smith, Y. Kim, J. Guanetti, A. A. Kurzhanskiy, M. Arcak, and F. Borrelli, "Balancing Safety and Traffic Throughput in Cooperative Vehicle Platooning," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 2197–2202.

[3] S. W. Smith, Y. Kim, J. Guanetti, R. Li, R. Firoozi, B. Wootton, A. A. Kurzhanskiy, F. Borrelli, R. Horowitz, and M. Arcak, "Improving Urban Traffic Throughput With Vehicle Platooning: Theory and Experiments," *IEEE Access*, vol. 8, pp. 141 208–141 223, 2020.

[4] P. Li, L. Alvarez, and R. Horowitz, "AHS safe control laws for platoon leaders," *IEEE Transactions on Control Systems Technology*, vol. 5, no. 6, pp. 614–628, 1997.

[5] S. Oh, L. Zhang, E. Tseng, W. Williams, H. Kourous, and G. Orosz, "Safe Decision and Control of Connected Automated Vehicles for an Unprotected Left Turn," in *ASME 2020 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers Digital Collection, 2020.

[6] H. M. Wang, T. G. Molnár, S. S. Avedisov, A. H. Sakr, O. Altintas, and G. Orosz, "Conflict Analysis for Cooperative Merging Using V2X Communication," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 1538–1543.

[7] D. Del Vecchio, M. Malisoff, and R. Verma, "A separation principle for a class of hybrid automata on a partial order," in *2009 American Control Conference*. IEEE, 2009, pp. 3638–3643.

[8] V. Desaraju, H. C. Ro, M. Yang, E. Tay, S. Roth, and D. Del Vecchio, "Partial order techniques for vehicle collision avoidance: Application to an autonomous roundabout test-bed," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 82–87.

[9] E. S. Kim, M. Arcak, and S. A. Seshia, "Symbolic control design for monotone systems with directed specifications," *Automatica*, vol. 83, pp. 10 – 19, 2017.

[10] A. Saoud, E. Ivanova, and A. Girard, "Efficient synthesis for monotone transition systems and directed safety specifications," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 6255–6260.

[11] H. Ahn and D. Del Vecchio, "Safety verification and control for collision avoidance at road intersections," *IEEE Transactions on Automatic Control*, vol. 63, no. 3, pp. 630–642, 2017.

[12] P. Tabuada, *Verification and control of hybrid systems: a symbolic approach*. Springer Science & Business Media, 2009.

[13] N. Ramdani, N. Meslem, and Y. Candau, "Computing reachable sets for uncertain nonlinear monotone systems," *Nonlinear Analysis: Hybrid Systems*, vol. 4, no. 2, pp. 263–278, 2010.

[14] D. Angeli and E. D. Sontag, "Monotone control systems," *IEEE Transactions on automatic control*, vol. 48, no. 10, pp. 1684–1698, 2003.

[15] A. Saoud, A. Girard, and L. Fribourg, "Contract-based Design of Symbolic Controllers for Safety in Distributed Multiperiodic Sampled-Data Systems," *IEEE Transactions on Automatic Control*, 2020.

[16] S. W. Smith, H. Yin, and M. Arcak, "Continuous abstraction of nonlinear systems using sum-of-squares programming," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 8093–8098.

[17] O. Grembek, A. Kurzhanskiy, A. Medury, P. Varaiya, and M. Yu, "Making intersections safer with I2V communication," *Transportation Research Part C: Emerging Technologies*, vol. 102, pp. 396–410, 2019.

[18] E. Ivanova, A. Saoud, and A. Girard, "Lazy Controller Synthesis for Monotone Transition Systems and Directed Safety Specifications," 2020.