Lazy Symbolic Controller for Continuous-Time Systems Based on Safe Set Boundary Exploration

Elena Ivanova, Antoine Girard

Lazy Symbolic Controller for Continuous-Time Systems Based on Safe Set Boundary Exploration ⋆

Elena Ivanova * **Antoine Girard** *

* *Laboratoire des Signaux et Systèmes (L2S)*
CNRS, CentraleSupélec, Université Paris-Sud, Université Paris-Saclay
3, rue Joliot-Curie, 91192 Gif-sur-Yvette, cedex, France
(e-mail: {elena.ivanova,antoine.girard}@l2s.centralesupelec.fr).

Abstract: In this paper, we present an abstraction-based approach to robust safety controller synthesis for continuous-time nonlinear systems. To reduce the computational complexity associated with symbolic control approaches, we develop a lazy controller synthesis algorithm, which iteratively explores states on the boundary of controllable domain while avoiding exploration of internal states, supposing that they are safely controllable a priory. A closed-loop safety controller for the original problem is then defined as follows: we use the abstract controller to push the system from a boundary state back towards the interior, while for inner states, any admissible input is valid. We then compare the proposed approach with the classical safety synthesis algorithm and illustrate the advantages, in terms of run-time and memory efficiency, on an adaptive cruise control problem.

Keywords: Safety specifications; Lazy controller synthesis; Symbolic control.

1. INTRODUCTION

Abstraction-based synthesis approaches attracted a lot of attention from the research community in the last decade. These methods consist in creating a finite-state abstraction (or a symbolic model) for a continuous or a hybrid system and refining the controller synthesized for the abstraction to a controller for the original system (Tabuada (2009); Belta et al. (2017)). The replacement of a dynamical system by its abstraction allows us to leverage discrete controller synthesis techniques (Cormen et al. (2001)) to deal with non-linear effects, state-input constraints, and a broad class of specifications, given, for instance, by automata or a temporal logic formula. In this paper, we focus on a safety specification, aiming to keep the system's trajectories within a given safe set. This specification often appears in real-world problems, e.g. temperature regulation in smart-buildings (Meyer et al. (2018), Thavlov and Bindner (2015)), blood glucose rate control for diabetic patients (Kushner et al. (2019)), adaptive cruise control (Darbha (1997)), Nilsson et al. (2016)), satellite station keeping (Weiss et al. (2018)).

The symbolic model is usually represented as a finite transition system with a set of states obtained due to a finite partitioning of the original state space. As an abstract set of inputs, one commonly chose a finite number of admissible for the original plant control actions. Then, if a robust reachable tube computed for initial state A and a control action C intersect at the moment τ (a time-sampling parameter) with an abstract state B, a state A links with a state B by a transition associated to an input C. If the exact reachable set cannot be found it is replaced by an over-approximation (Girard (2005), Kurzhanski and Varaiya (2014), Reissig et al. (2017), Moor and Raisch (2002), Meyer et al. (2019)).

Intuitively, the more accurate abstractions with finer discretization parameters better mimic the original system's behavior, increasing the chance of successful controller synthesis. However, their construction is a more demanding process. Moreover, the computational complexity of the discrete controller synthesis algorithms typically depends on symbolic models' size. Finally, while all these computations are typically handled off-line, a controller obtained via abstraction-based techniques using symbolic models with a large number of states would require a considerable amount of memory for its real-time implementation.

To mitigate these intensive computational requirements, several lazy controller synthesis algorithms were recently introduced. In such approaches, the transitions are explored as needed and computed on the fly during a controller synthesis. In (Gol et al. (2013); Nilson et al. (2017); Girard et al. (2016); Hsu et al. (2018); Ivanova and Girard (2020)) the authors proposed to start with rough abstractions and then iteratively refine them if necessary. In (Girard et al. (2016); Ivanova and Girard (2020)) the controller is synthesized only for reachable from the initial set states. The other authors use structural properties of the dynamics such as monotonicity (Coogan and Arcak (2015); Saoud et al. (2019)) or incremental stability (Girard et al. (2016)) to ease these computations.

⋆ This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 725144).

In this paper, we first introduce a novel lazy synthesis algorithm for a finite transition system with a safety specification, which avoids exploring a priori controllable states. Then, relying on ideas of adaptive time-sampling techniques (Ivanova and Girard (2020)), we construct an abstraction where only transitions between neighboring states are allowed. We then iteratively explore only boundary states of the controlled area since if all boundary states are controllable, then all internal states are also controllable. We also extend this idea towards a more general case, since no matter which time-sampling approach was used, as soon as a safely controllable boundary is found, the abstract controller can be refined to a safe controller for the original continuous-time system: for the internal states, any admissible input is applicable, while for boundary states we use the abstract controller to push the system back towards the interior. In spirit, this idea is close to Nagumo theorem result (Blanchini (1999)) and extremal aiming principle (Subbotin (1995)), but we benefit from abstraction based approaches to tackle the complex dynamic of the original system.

This paper is organized as follows. In Section 2, we present a lazy safety controller synthesis algorithm for finite transition systems. Section 3 provides an abstraction-based approach for a safety controller synthesis for a continuous-time system with bounded disturbance and input-state constraints. The practical implementation of the algorithm and controller refinement are discussed. In Section 4, we consider an illustrative example to show the benefits of the approach.

2. TRANSITION SYSTEM

2.1 Safety Controllers for Finite Transition Systems

Definition 1. A finite *transition system* is a tuple $\Sigma = (Q, U, F)$, consisting of a finite set of states Q, a finite set of inputs U, and a transition relation $F \subseteq Q \times U \times Q$.

For every transition $(q, u, q') \in F$ the state q is named *u-predecessor* of q' and similarly the state q' is named *u-successor* of q. Let $F(q, u)$ denotes the set of all u-successors of a state q. If there is $q \in Q,\ u \in U$ such that $|\, F(q, u)\,| > 1$, then the transition system is called *non-deterministic*, otherwise it is *deterministic*. Since $F(q, u)$ may be empty let us introduce a set $\mathrm{Enab}_F(q) = \{u \in U \mid F(q, u) \neq \emptyset\}$ of all enabled inputs at a state $q \in Q$. We also introduce $F(q, U) = \cup_{u \in U} F(q, u)$.

Definition 2. A *controller* for a transition system $\Sigma = (Q, U, F)$ is a map $C\colon Q \to 2^U$, such that $C(q) \subseteq \mathrm{Enab}_F(q)$ for every $q \in Q$.

We also use notation $\mathrm{Dom}(C) = \{q \in Q \mid C(q) \neq \emptyset\}$ for a domain of controller C. If $\mathrm{Dom}(C) = \emptyset$ the controller is called *trivial*, otherwise *non-trivial*.

Definition 3. A *safety controller* for a transition system $\Sigma = (Q, U, F)$ and a safe set $Q_S \subseteq Q$ is a controller C such that the following two properties hold

(1) $\mathrm{Dom}(C) \subseteq Q_S$;
(2) for all $q \in \mathrm{Dom}(C)$ and for all $u \in C(q)$ the inclusion $F(q, u) \subseteq \mathrm{Dom}(C)$ is satisfied.

Lemma 4. For a given transition system $\Sigma = (Q, U, F)$ and a safety specification $Q_S \subseteq Q$, there exists a unique maximal safety controller $\bar{C}$ such that for any safety controller C the following hold

(1) $\mathrm{Dom}(C) \subseteq \mathrm{Dom}(\bar{C})$;
(2) for all $q \in \mathrm{Dom}(C)$ the inclusion $C(q) \subseteq \bar{C}(q)$ is satisfied.

The maximal safety controller $\bar{C}$ is the best possible safety controller in the sense that any other controller solving the same safety problem would be more restrictive.

Algorithm 1: Classical Safety Controller Synthesis

Input: $\Sigma = (Q, U, F)$ and a safe set Q_S
Output: Maximal Safety Controller C

```
begin
    for q ∈ Q do
        C(q) = {u ∈ Enab_F(q) | F(q,u) ⊂ Q_S};
    repeat
        Q_C := Dom(C);
        for q ∈ Q_C do
            C(q) := {u ∈ C(q) | F(q,u) ⊆ Q_C};
    until fixed point for map C is reached;
    return C;
```

Algorithm 2: Lazy Controller Synthesis

Input: $\Sigma = (Q, U, F)$, a safe set Q_S, and a function $I\colon Q \times 2^Q \to \{True, False\}$
Output: A controller C

```
begin
    for q ∈ Q do
        C(q) = {u ∈ Enab_F(q) | F(q,u) ⊂ Q_S};
    repeat
        Q_C := Dom(C);
        for {q ∈ Q_C | I(q, Q_C) is False} do
            C(q) := {u ∈ C(q) | F(q,u) ⊆ Q_C};
    until fixed point on map C is reached;
    return C;
```

2.2 Controller Synthesis

There is a well-known fixed-point algorithm converging to the maximal safety controller $\bar{C}$ for a given Q_S (Tabuada (2009)). However, the classical synthesis procedure (see Algorithm 1) is a very computationally-demanding process. So, to tackle real-world problems, where the huge transition systems should be explored, we have to develop more efficient controller synthesis techniques.

One way to reduce the computational burden is to use lazy synthesis algorithms (Hussien and Tabuada (2018), Hsu et al. (2018) Saoud et al. (2019), Ivanova and Girard (2020)), aiming to provide safety controllers, while avoiding non-essential computations. In this paper, we introduce a lazy Algorithm 2, which, in general case, does not return a safety controller. However if the information function $I : Q \times 2^Q \to \{True, False\}$ distinguish a priori controllable on the current iteration states from those which should be explored, then Algorithm 2 synthesise the maximal safety controller.

Theorem 5. Let C_1 and C_2 be controllers computed by the Algorithm 1 and Algorithm 2 correspondingly. Then

(1) in general case for all $q \in Q,\ C_1(q) \subseteq C_2(q)$.
(2) if at every iteration of the loop 4-8 of the Algorithm 2, for all $q \in Q_C$ such that $I(q, Q_C)$ is $True$ the inclusion $F(q,u) \subseteq Q_C$ is satisfied for all $u \in U$, then C_1 coincides with C_2.

The proof of this result can be found in the Appendix.

3. ABSTRACTION BASED CONTROLLER SYNTHESIS

3.1 Problem Statement

A control system $\Sigma = (T, \mathbb{R}^{n_x}, U, W, f)$ consists of a time domain $T = [0, +\infty)$, a state space $\mathbb{R}^{n_x}$, a compact set $U \subset \mathbb{R}^{n_u}$, a compact set $W \subset \mathbb{R}^{n_w}$, and a non-linear function $f\colon \mathbb{R}^{n_x} \times U \times W \to \mathbb{R}^{n_x}$, such that for any control $u(\cdot) \in \mathcal{L}^\infty(T, U)$, any disturbance $w(\cdot) \in \mathcal{L}^\infty(T, W)$ and any initial condition $x(0) \in \mathbb{R}^{n_x}$ there exists a unique solution $x_f(t \mid x(0), u(\cdot), w(\cdot)),\ t \in T$ of the following differential equation

$$\dot{x}(t) = f(x(t), u(t), w(t)) \tag{1}$$

in the sense of Caratheodory. The notation $\mathcal{L}^\infty(T, S)$ is used for the space of functions $s(\cdot)$, measurable on T, such that $s(t) \in S,\ t \in T$ almost everywhere.

In this paper, we look for admissible safety controllers, which keep all trajectories of the closed-loop system inside a safety set Y. Here the controller is said to be admissible if it is robust against any measurable bounded disturbance $w(\cdot) \in \mathcal{L}^\infty(T, W)$ and a solution of the closed-loop system exists.

To synthesize a desirable controller, we use an abstraction-based approach. For the original system, we create a finite-state abstraction (also known as a symbolic model) in such a manner that a controller synthesized for the abstraction can be refined to a controller for $\Sigma = (T, \mathbb{R}^{n_x}, U, W, f)$.

3.2 Symbolic Model with Adaptive Time Sampling

This section we construct a finite transition system $\Sigma_A = (Q_A, U_A, F_A)$, which mimics the dynamic of the original plant $\Sigma = (T, \mathbb{R}^{n_x}, U, W, f)$.

First, we introduce a finite partitioning $Q_S = \{q_1, \dots, q_n\}$ on an internal approximation S of the safety set Y such that $S = \cup_{i=1}^{n} q_i,\ q_i \cap q_j = \emptyset$ for all $i \neq j$ and define Q_A, as follows $Q_A = Q_S \cup \{q_{us}\}$, where $q_{us} = \mathbb{R}^{n_x} \setminus S$ is an unsafe state. Let us remark that, depending on the context, a symbol q represents an abstract state or a subset of the space $\mathbb{R}^{n_x}$, corresponding to this abstract state. We will use this duality in notation through the paper for the union of discrete states as well.

Let us formally define a neighborhood of a state $q \in Q_A$, as follows,

$N_A(q) = \{q' \in Q_A \setminus \{q\} \mid \mathrm{cl}(q) \cap q' \neq \emptyset \text{ or } q \cap \mathrm{cl}(q') \neq \emptyset\}$.

Here $\mathrm{cl}(q)$ is a closure of a set $q \subset \mathbb{R}^{n_x}$.

Then for every $q \in Q_S$ we consider a finite set $U_S(q) = \{u_1, \dots, u_m\} \subseteq U$ and define for every $u \in U_S(q)$ a set of all reachable set

$$\begin{aligned}\mathrm{Reach}(t \mid q, u(\cdot)) = \{x \in \mathbb{R}^{n_x} \mid \exists\, x(0) \in q \\ \text{and } \exists\, w(\cdot) \in \mathcal{L}^\infty([0,t], W) \\ \text{such that } x_f(t \mid x(0), u(\cdot), w(\cdot)) = x\}\end{aligned}$$

corresponding to an initial set q, a constant control function $u^* : [0,t] \to u$ and all admissible disturbances $w(\cdot)$. Since exact computation of reachable set is rarely possible its over-approximation $\overline{\mathrm{Reach}}(t \mid q, u)$ is usually used to build symbolic models. Several methods exist for computing such over-approximations (Girard (2005), Kurzhanski and Varaiya (2014), Reissig et al. (2017), Moor and Raisch (2002), Meyer et al. (2019)), and it is always a question of compromise between their precision and a simplicity of implementation.

Choosing $U_A = \cup_{q \in Q_S} U_S(q)$ as an input set for our abstraction we then say that for every $q \in Q_A, u \in U_A$ transition $(q, u, q') \in F_A$ if and only if $q \in Q_S$, $u \in U_S(q)$, $q' \in Q_A$, the intersection $q' \cap \overline{\mathrm{Reach}}(\tau^A_{q,u} \mid q, u) \neq \emptyset$, and for all $t \in [0, \tau^A_{q,u}]$ the condition of a collision avoidance $\overline{\mathrm{Reach}}(t \mid q, u) \cap (\mathbb{R}^{n_x} \setminus Y) = \emptyset$ is satisfied. Here, we use the adaptive time-sampling and a transition duration defined as $\tau^A_{q,u} = \min(\tau, \tau_{q,u} - \varepsilon)$, where τ is a given parameter which determine the maximal evolution time, while $\tau_{q,u}$ is a moment of time

$$\tau_{q,u} = \inf_{t \in [0, +\infty)} \left\{ \overline{\mathrm{Reach}}(t \mid q, u) \not\subseteq N_A(q) \right\}$$

when the over-approximation of reachable set leaves the neighborhood of $N_A(q)$. We chose $\varepsilon < \tau_{q,u}$ arbitrary small to stop evolution just before leaving, while τ should be big enough since it serves only to manage situations when a solution stuck within the box. Let us remark that relation $N_A \subseteq Q_A \times Q_A$ is symmetric, i.e. for any $q, q' \in Q_A$ we have $(q, q') \in N_A$ if and only if $(q', q) \in N_A$.

Theorem 6. Let for all $q \in Q_C$ the information function

$$I_A(q, Q_C) = \begin{cases} True & \text{if } N_A(q) \subseteq Q_C \\ False & \text{if } N_A(q) \not\subseteq Q_C \end{cases} \tag{2}$$

Then the controller C_2 given by the Algorithm 2 is the maximal safety controller for transition system $\Sigma_A = (Q_A, U_A, F_A)$ and a safe set $Q_S = Q_A \setminus \{q_{us}\}$.

Proof. Indeed, $F_A(q, U_A) \subseteq N_A(q)$ for any $q \in Q_C$. Consequently, at every iteration of the loop 4-8 of the Algorithm 2, for all $q \in Q_C$ such that $I_A(q, Q_C)$ is $True$ the inclusion $F_A(q, U_A) \subseteq Q_C$ is satisfied. Hence, from Theorem 5, C_2 is the maximal safety controller. □

Let us remark that while using lazy synthesis algorithm it reasonable compute the abstraction on the fly. Indeed, at every iteration of Algorithm 2 only boundary states are explored, so there is no need to pre-compute the symbolic model for the internal states.

3.3 Arbitrary Time Sampling

In the last section, we have shown that a symbolic model for the original problem can be constructed in a particular way allowing us to compute the maximal safety controller with efficient Algorithm 2 instead of the classical approach. However, with the transition relation defined in section 3.2, we lost flexibility in scaling time sampling parameters independently from space sampling parameters. In this section, we consider a more general case.

Let us define a transition relation $F_A^* \subseteq Q_A \times U_A \times Q_A$ such that for every $q \in Q_A, u \in U_A$ a transition $(q, u, q') \in F_A^*$ if and only if $q \in Q_S$, $u \in U_S(q)$, $q' \in Q_A$, the intersection $q' \cap \overline{\text{Reach}}(\tau^*_{q,u} \mid q, u) \neq \emptyset$, and for all $t \in [0, \tau^*_{q,u})$ the following $\overline{\text{Reach}}(t \mid q, u) \cap (X \setminus S) = \emptyset$ is satisfied. Here we assume only that sampling parameter $\tau^*_{q,u} > 0$ and it is determined for any given $q \in Q_A, u \in U_A$. Such a definition doesn't put any requirements on a choice of time-sampling parameters and allows us to handle fixed (Nilson et al. (2017)), multi-scale (Hsu et al. (2018); Girard et al. (2016)) or adaptive time-samplings (Ivanova and Girard (2020)) in a unified way. The sampling procedure described in previous section is also obviously incorporated.

It is clear that if we synthesize a controller C_2 for the abstraction $\Sigma^*_A(Q_A, U_A, F^*_A)$, the safety set Q_S and the information function $I_A(q, Q_C)$ with the Algorithm 2 there is no guarantee that C_2 is a safety controller. However, in the next section, we show that C_2 still can be refined towards a continuous time controller, which solves the original safety problem.

3.4 Controller Refinement

Let C_2 be a controller given by the Algorithm 2 for transition system $\Sigma^*_A = (Q_A, U_A, F^*_A)$ and a safe set $Q_S = Q_A \setminus \{q_{us}\}$, while the $I_A(q, Q_C)$ is defined as in (2).

For every $q \in Q_A \setminus \text{Dom}(C_2)$, we define $C_2^{dur}(q) = \emptyset$, while if $q \in \text{Dom}(C_2)$ and $u \in C_2(q)$, then the pair $(u, \tau^*_{u,q}) \in C_2^{dur}(q)$. Hence, the controller C_2^{dur} store not only safe inputs, but a real duration of safe transitions.

Let us now define a set of border points

$$Q_B = \text{cl}\left(\{x \in \mathbb{R}^{n_x} \mid \exists\, q \in \text{Dom}(C_2) \text{ such that } x \in q \text{ and } N_A(q) \not\subseteq \text{Dom}(C_2)\}\right)$$

and a set of all internal points $Q_I = \text{Dom}(C_2) \setminus Q_B$ correspondingly. We then define a quantizer, associating every border point $x \in Q_B$ with a unique state of transition system $Q^x(x) = \{q \in \text{Dom}(C_2) \mid x \in \text{cl}(q)\}$.

Now we are ready to introduce a controller refinement procedure for $\Sigma = (T, \mathbb{R}^{n_x}, U, W, f)$. Let us consider the control input given for all $t \in [t_k, t_{k+1})$ by

$$\begin{cases} u(t) \in U & \text{if } m_k = 0 \\ u(t) = u_k, & \text{if } m_k = 1 \end{cases} \tag{3}$$

where

$$\begin{cases} m_k = 0, & \text{if } x(t_k) \in Q_I \\ m_k = 1, & \text{if } x(t_k) \in Q_B \end{cases} \tag{4}$$

$$(u_k, \tau_k) \in C_2^{dur}(Q^x(x(t_k))), \text{ if } m_k = 1 \tag{5}$$

and the sequence of instants (t_k) is given by $t_0 = 0$ and

$$\begin{cases} t_{k+1} = \inf\{t > t_k | Q^x(x(t)) \in Q_B\} & \text{if } m_k = 0 \\ t_{k+1} = t_k + \tau_k & \text{if } m_k = 1 \end{cases} \tag{6}$$

Theorem 7. All trajectories of closed-loop system (1)-(6) starting from a $\text{Dom}(C_2)$ at $t = 0$ stays within a safe set S for all $t \in T$.

Proof. For any initial condition $x(0) \in \text{Dom}(C_2)$ and any disturbance $w(\cdot) \in \mathcal{L}^\infty(T, W)$ the closed-loop trajectory $\dot{x}(t) = f(x(t), u(t), w(t))$, $t \in T$ can not leave the $\text{Dom}(C_2)$ without passing through the set Q_B, and for every state in Q_B there is a controller which brings us back

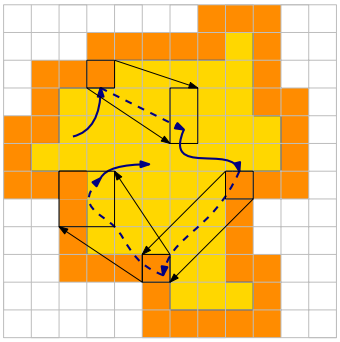

Fig. 1. A piece of a closed-loop trajectory. Mode 0: a normal line; mode 1: a dashed line.

to the $\text{Dom}(C_2) \subseteq S \subseteq Y$ (see line 7 of the Algorithm 2). The Zeno behaviour is impossible since being in mode 0 at $t = s \in T$ we can either stay in mode 0 for all $t \in [s, +\infty)$ or switch to mode 1. If we have switched to mode 1 we spent there at least τ_k seconds before switching back to mode 0. □

Let us remark that while we stay inside the set Q_I in the mode 0 we can apply any admissible closed-loop control: e.g. $u(t) = v(t, x)$ We say that a closed-loop control $v(t, x)$ is admissible if the solution of $\dot{x}(t) = f(x(t), v(t, x), w(t))$, $t \in T$ exists and $v(t, x) \in U$ for all $t \in T$ and $x \in \mathbb{R}^{n_x}$.

4. EXAMPLE

As an illustrative example, let us consider the adaptive cruise control problem for two vehicles moving along a straight line (Darbha (1997), Nilsson et al. (2016)). Each vehicle is modeled as a point mass m with velocity changing according to the law

$$\dot{v}_i = \alpha(F_i, v_i) = (F_i - (f_0 + f_1 v_i + f_2 v_i^2))/m, \quad i = 1, 2.$$

In equation above, F_i represents a net action of braking and engine torque applied to the wheels, while the second term $(f_1 + f_2 v_i + f_3 v_i^2)$ describes aerodynamic and rolling resistance effects. The net force F_i is viewed as a control input for the following vehicle and as a disturbance for the lead one. It is assumed that $F_i \in [a, b]$, where $a = -0.3mg$, $b = 0.2mg$, g is a gravitational constant. Such bounds are consistent with non-emergency braking and acceleration.

First, we do a feedback linearization of the model by introducing $F_{i,lin} = \alpha(F_i, v_i)$. Let us assume that the first car doesn't violate speed restrictions $v_1 \in [0, v^{max}]$. Since for any $v_2 \in [0, v^{max}]$, zero belongs to $[\alpha(a, v_2), \alpha(b, v_2)]$ we can always choose a suitable control law to do the same for the second car.

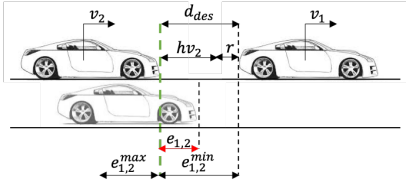

Fig. 2. An illustration for an adaptive cruise control problem.

Table 1. Vehicle and safety parameters

Parameter	Value	Unit	Parameter	Value	Unit
m	1370	Kg	v^{max}	25	m/s
f_0	51.0709	N	h	2	s
f_1	0.3494	Ns/m	r	2.5	m
f_2	0.4161	Ns^2/m^2	$e_{1,2}^{max}$	55	m

Then, we chose a feedback stabilizer as a control law $F_{2,lin} = u_1 + u_2(v_1 - v_2) + u_3 e_{1,2}$. Here $u_1 \in \mathbb{R}, u_2, u_3 \geq 0$. The deviation $e_{1,2}$ from the desirable distance between the cars changes according to the following equation $\dot{e}_{1,2} = (v_1 - v_2) - \dot{d}_{des}$, where $d_{des} = hv_2 + r$, $h > 0$. Hence, the constant time headway spacing policy is considered (see Fig.2 for an illustration).

Finally, the system dynamic is described as follows

$$\begin{aligned} \dot{v}_1 &= \beta(F_{1,lin}, v_1) \\ \dot{v}_2 &= \beta(u_1 + u_2(v_1 - v_2) + u_3 e_{1,2}, v_2) \\ \dot{e}_{12} &= v_1 - v_2 - h\dot{v}_2 \end{aligned} \tag{7}$$

where

$$\beta(z, v) = \begin{cases} z, \text{ if } v \in (0, v^{max}) \\ \max(z, 0), \text{ if } v = 0 \\ \min(z, 0), \text{ if } v = v^{max} \end{cases}$$

Varying the new control parameters u_1, u_2, u_3 we want to keep the error $e_{1,2}$ in a range $[-hv_2 - r, e_{1,2}^{max}]$, while ensuring that $u_1 + u_2(v_1 - v_2) + u_3 e_{1,2} \in [\alpha(a, v_2), \alpha(b, v_2)]$. To construct a symbolic model we introduce on a set $X = [0, v^{max}] \times [0, v^{max}] \times [-hv^{max} - r, e_{1,2}^{max}]$ a uniform Cartesian partition. Number of intervals in every direction is described by a given parameter n_x^p. Intervals on the border are flat, while internal intervals whether semi-closed from the right side or open if they are next to the right border. For example, in first direction we have $\{0\}, (0, \eta], (\eta, 2\eta], \ldots, (v^{max} - \eta, v^{max}), \{v^{max}\}$, where $\eta = v^{max}/n_x^p(1)$. For every state belonging to a safe set Y, we chose n_u^p different admissible inputs $u = [u_1, u_2, u_3]$, ensuring that $F_{2,lin} \in [\alpha(a, v_2), \alpha(b, v_2)]$, and apply them $\tau, \tau/2, \tau/4$ or $\tau/8$ seconds. While computing the symbolic model we use interval over-approximations instead reachable sets. Since the system (7) is a mixed-monotone every over-approximation can be easily obtained by solving 6 differential equations (Meyer et al. (2019)).

Setting $n_x^p = [41, 41, 41]$, $n_u^p = 18$, $\tau = 1$ we use the Algorithm 1 and Algorithm 2 to compute a controllers C_1 and C_2 for the abstraction. In our particular example, we got that the controllable domains $\mathrm{Dom}(C_1)$ and $\mathrm{Dom}(C_2)$ coincide. So, both of them are represented on Fig.3(right), while Fig.3(left) illustrate the safe specification in the abstract domain. However, our approach, with run-time equals to 70.3 min, is 2.58 times faster than the classical one since it explores 19574 less states. In the Fig.4, a closed-loop trajectory simulated $500s$ for a given disturbance is also shown.

5. CONCLUSION

In this paper, we introduce a novel lazy synthesis algorithm for a finite transition system with a safety specification. The main idea is to iteratively explore only the boundary states of the controllable domain, supposing that internal states are safely controllable a priory. The worst-case complexity of the proposed algorithm coincides with the complexity of the classical brute-force exploration (Tabuada (2009)), but, our approach is more efficient in practice. Indeed, if a controllable domain for the abstract controller is non-empty then we have a computational gain since we don't explore internal states. Moreover, real-time implementation of a closed-loop controller for the original continuous system is more memory efficient since the information for the internal states is not stored.

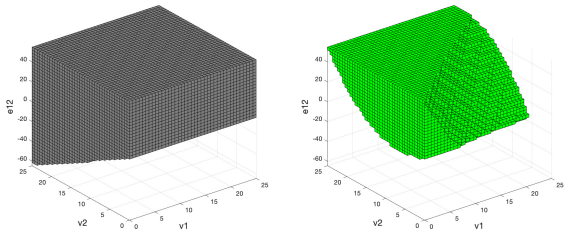

Fig. 3. Left: a safe set. Right: a controllable set.

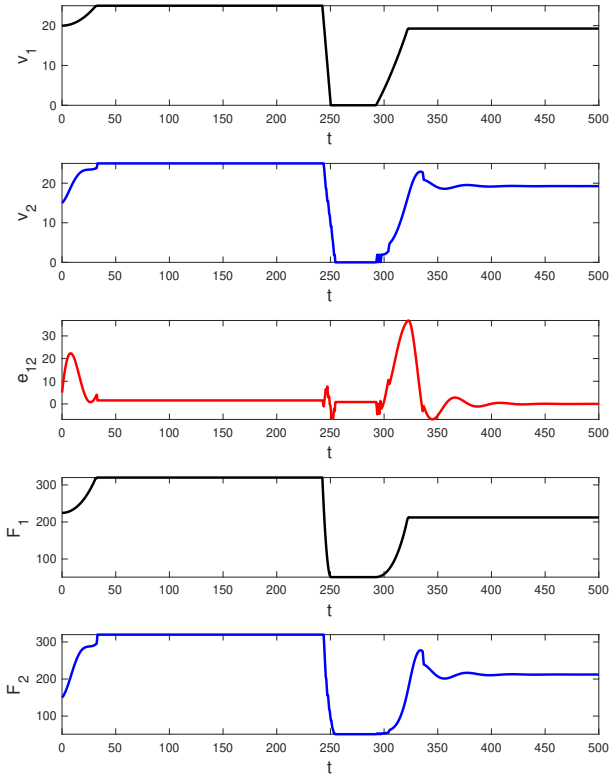

Fig. 4. A closed-loop trajectory. Initial point $[20, 15, 5]$

APPENDIX

Proof of Theorem 5

Proof. 1) The statement can be proven by induction. Let C_1^i, C_2^i be controllers before i iteration of the loops 4-8 of the Algorithm 1 and Algorithm 2 correspondingly. For all $q \in Q$ from lines 2-3 we have $C_1^0(q) = \mathrm{Enab}_F(q) = C_2^0(q)$. Let us show that if $C_1^i(q) \subseteq C_2^i(q),\ q \in Q$ then $C_1^{i+1}(q) \subseteq C_2^{i+1}(q),\ q \in Q$. For any $q \in Q$, the following two cases are possible

- $C_1^i(q) = \emptyset$. Then, $q \notin Q_{C_1}^{i+1} = \mathrm{Dom}(C_1^i)$ and $C_1^{i+1}(q) = C_1^i(q) = \emptyset \subseteq C_2^{i+1}(q)$.

- $C_1^i(q) \neq \emptyset$. Since $C_1^i(q) \subseteq C_2^i(q)$ we have $C_2^i(q) \neq \emptyset$. Then $q \in Q_{C_1}^{i+1} = \mathrm{Dom}(C_1^i)$ and from line 7 it follows

$$C_1^{i+1}(q) := \{u \in C_1^i(q) \mid F(q,u) \subseteq Q_{C_1}^{i+1}\}. \quad (8)$$

and, as a consequence, $C_1^i(q) \supseteq C_1^{i+1}(q)$. Moreover, if $I(q, Q_{C_2}^{i+1})$ is $True$, then $C_2^{i+1}(q) = C_2^i(q)$ and we get $C_2^{i+1}(q) \supseteq C_1^{i+1}(q)$ since $C_2^i(q) \supseteq C_1^i(q)$. If $I(q, Q_{C_2}^{i+1})$ is $False$ then from line 7 of the Algorithm 2 it follows

$$C_2^{i+1}(q) := \{u \in C_2^i(q) \mid F(q,u) \subseteq Q_{C_2}^{i+1}\}. \quad (9)$$

Combining (8) and (9) with the fact that $C_1^i(q) \subseteq C_2^i(q)$ and $Q_{C_1}^{i+1} = \mathrm{Dom}(C_1^i) \subseteq \mathrm{Dom}(C_2^i) = Q_{C_2}^{i+1}$ we finally obtain $C_1^{i+1}(q) \subseteq C_2^{i+1}(q)$.

2) Again $C_1^0(q) = \mathrm{Enab}_F(q) = C_2^0(q)$. Supposing that $C_1^i(q) = C_2^i(q)$ for all $q \in Q$ let us show that $C_1^{i+1}(q) = C_2^{i+1}(q)$ for all $q \in Q$. Since from the previous item $C_1^{i+1}(q) \subseteq C_2^{i+1}(q)$, it is enough to prove that $C_1^{i+1}(q) \supseteq C_2^{i+1}(q)$. Indeed, $Q_{C_1}^{i+1} = \mathrm{Dom}(C_1^i) = \mathrm{Dom}(C_2^i) = Q_{C_1}^{i+1}$. Let us, for simplicity, introduce a notation $Q_C^{i+1} = Q_{C_1}^{i+1} = Q_{C_2}^{i+1}$. For all $q \in Q$ three cases are possible

- if $q \notin Q_C^{i+1}$ then $C_1^{i+1}(q) = C_1^i(q) = C_2^i(q) = C_2^{i+1}(q)$.
- if $q \in Q_C^{i+1}$ and $I(q, Q_C^{i+1})$ is $True$ then $C_2^{i+1}(q) = C_2^i(q)$ and $C_1^{i+1}(q) := \{u \in C_1^i(q) \mid F(q,u) \subseteq Q_C^{i+1}\}$. For all $q \in Q_C^{i+1}$ such that $I(q, Q_C^{i+1})$ is $True$ the inclusion $F(q,u) \subseteq Q_C^{i+1}$ is satisfied, consequently, $C^{i+1}(q) = C^i(q)$. Hence, $C_2^{i+1}(q) = C_1^{i+1}(q)$, since $C_2^i(q) = C_1^i(q)$.
- if $q \in Q_C^{i+1}$ and $I(q, Q_C^{i+1})$ is $False$ then $C_2^{i+1}(q) := \{u \in C_2^i(q) \mid F(q,u) \subseteq Q_C^{i+1}\}$ and $C_1^{i+1}(q) := \{u \in C_1^i(q) \mid F(q,u) \subseteq Q_C^{i+1}\}$. Remembering that $C_2^i(q) = C_1^i(q)$ we get $C_2^{i+1}(q) = C_1^{i+1}(q)$. □

REFERENCES

Belta, C., Yordanov, B., and Göl, E.A. (2017). *Formal Methods for Discrete-Time Dynamical Systems*. Springer.

Blanchini, F. (1999). Set invariance in control. *Automatica*, 35(11), 1747–1767.

Coogan, S. and Arcak, M. (2015). Efficient finite abstraction of mixed monotone systems. *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, 58–67.

Cormen, T.H., Leiserson, C.E., Rivest, R.L., and Stein, C. (2001). *Introduction to Algorithms, 2nd ed.* MIT Press.

Darbha, S. (1997). String stability of interconnected systems: An application to platooning in automated highway systems. *Transportation Research Part A: Policy and Practice*, 31(1), 65.

Girard, A. (2005). Reachability of uncertain linear systems using zonotopes. *Hybrid Systems: Computation and Control. HSCC 2005. Lecture Notes in Computer Science*, 3414, 291–305.

Girard, A., Gössler, G., and Moueli, S. (2016). Safety controller synthesis for incrementally stable switched systems using multiscale symbolic models. *IEEE Transactions on Automatic Control*, 61(6), 1537–1549.

Gol, E.A., Lazar, M., and Belta, C. (2013). Language-guided controller synthesis for linear systems. *IEEE Transactions on Automatic Control*, 59(5), 1163–1176.

Hsu, K., Majumdar, R., Mallik, K., and Schmuck, A.K. (2018). Lazy abstraction-based controller synthesis. *IEEE Conference on Decision and Control (CDC)*, 4902–4907.

Hussien, O. and Tabuada, P. (2018). Lazy controller synthesis using three-valued abstractions for safety and reachability specifications. In *2018 IEEE Conference on Decision and Control (CDC)*, 3567–3572. IEEE.

Ivanova, E. and Girard, A. (2020). Lazy safety controller synthesis with multi-scale adaptive-sampling abstractions of nonlinear systems. In *21st IFAC World Congress*.

Kurzhanski, A.B. and Varaiya, P. (2014). *Dynamics and Control of Trajectory Tubes. Theory and Computation*. Birkhäuser.

Kushner, T., Bequette, B.W., Cameron, F., Forlenza, G., Maahs, D., and Sankaranarayanan, S. (2019). Models, devices, properties, and verification of artificial pancreas systems. *Automated Reasoning for Systems Biology and Medicine*, 825–833.

Meyer, P.J., Devonport, A., and Arcak, M. (2019). TIRA: Toolbox for interval reachability analysis. *HSCC 2019 - Proceedings of the 2019 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 224–229.

Meyer, P.J., Girard, A., and Witrant, E. (2018). Compositional abstraction and safety synthesis using overlapping symbolic models. *IEEE Transactions on Automatic Control*, 63(6), 1835–1841.

Moor, T. and Raisch, J. (2002). Abstraction based supervisory controller synthesis for high order monotone continuous systems. *Modelling, Analysis, and Design of Hybrid Systems, Springer*, 247–265.

Nilson, P., Ozay, N., and Liu, J. (2017). Augmented finite transition systems as abstractions for control synthesis. *Discrete Event Dynamic Systems*, 27(3), 301–340.

Nilsson, P., Hussien, O., Balkan, A., Chen, Y., Ames, A.D., Grizzle, J.W., Ozay, N., Peng, H., and Tabuada, P. (2016). Correct-by-Construction Adaptive Cruise Control: Two Approaches. *IEEE Transactions on Control Systems Technology*, 24(4), 1294–1307.

Reissig, G., Weber, A., and Rungger, M. (2017). Feedback refinement relations for the synthesis of symbolic controllers. *IEEE Transactions on Automatic Control*, 62(4), 1781–1796.

Saoud, A., Ivanova, E., and Girard, A. (2019). Efficient synthesis for monotone transition systems and directed safety specifications. *IEEE Conference on Decision and Control*.

Subbotin, A.I. (1995). *Generalized Solutions of First Order PDEs: The Dynamical Optimization Perspective*. Birkhäuser Basel.

Tabuada, P. (2009). *Verification and control of hybrid systems: a symbolic approach*. Springer.

Thavlov, A. and Bindner, H.W. (2015). A heat dynamic model for intelligent heating of buildings. *International Journal of Green Energy*, 12(3), 240–247.

Weiss, A., Kalabić, U.V., and Cairano, S.D. (2018). Station keeping and momentum management of low-thrust satellites using mpc. *Aerospace Science and Technology*, 76, 229–241.