Sampled-data control of hybrid systems with discrete inputs and outputs

M. Petreczky[†] **P. Collins**[‡] **D.A. van Beek**[†] **J. H. van Schuppen**[‡] **J.E. Rooda**[†]

[†]*Eindhoven University of Technology, The Netherlands*
{*M.Petreczky, D.A.v.Beek, J.E.Rooda*}*@tue.nl*
[‡]*Centrum voor Wiskunde en Informatica (CWI), The Netherlands,*
{*Pieter.Collins,J.H.van.Schuppen*}*@cwi.nl*

Abstract: We address the control synthesis of hybrid systems with discrete inputs, disturbances and outputs. The control objective is to ensure that the events of the closed-loop system belong to the language of the control requirements. The controller is sampling-based and it is representable by a finite-state machine. We formalize the control problem and provide a theoretically sound solution. The solution is based on solving a discrete-event control problem for a finite-state abstraction of the plant. We propose a specific construction for the finite-state abstraction. This construction is not based on discretizing the state-space, but rather on converting the continuous-time hybrid system to a discrete-time one based on sampling. The construction works only for a specific class of hybrid systems. We describe this class of systems and we provide an example of such a system, inspired by an industrial use-case.

Keywords: hybrid systems, discrete-event systems, symbolic control

1. INTRODUCTION

Motivated by applications in the area of high-tech systems, in particular control of printers, Petreczky et al. (2008b), we are interested in the following control problem. The plant is a continuous-time hybrid system which is subject to discrete disturbances and control inputs and which generates discrete outputs and internal events. The disturbances are imposed by the environment and the control inputs can be used to influence the system behavior. The desired *controller* can read the outputs and it generates control inputs. Furthermore, the controller should be realizable by a finite-state machine, and it is activated at equidistant sampling times with sampling rate Δ. The control objective is to ensure that the sequences of internal events generated by the plant satisfy the *control requirements*.

Contribution We present a mathematical formulation of the control problem above. We also propose the following solution.

Step 1 Compute an abstraction (over-approximation) of the symbolic (event) behavior of the plant, such that the abstraction has a finite-state representation. This abstraction is based on transforming the original system to a discrete-time one. The states of the abstraction are those states of the hybrid system which can be reached at sampling times. Under suitable assumptions, the thus obtained state-space is finite.

Step 2 Solve the related discrete-event control problem for the finite-state abstraction. The solution is a discrete-event controller representable by a Moore-automaton. Interpret the solution as a controller for the original plant.

We prove that the procedure above is theoretically sound. The discrete-event control problem of Step 2 can be solved using game theory, see Grädel et al. (2002) or, under additional assumptions, using classical supervisory control, see Petreczky et al. (2008a). We also present a procedure for constructing a finite-state abstraction. The procedure can be made effective, but it may be computationaly expensive. The finite-state abstraction can be computed only for a specific class of hybrid systems which satisfies the following properties; (**1**) disturbances or internal events do not influence the continuous dynamics, (**2**) output events do not influence the system dynamics, (**3**) only finitely many events are generated on any time interval, (**4**) the set of states reachable at sampling times is finite. For the last property we present sufficient conditions in terms of existence of a Lyapunov-like function. While these assumptions are strong, there are hybrid systems of practical relevance (see Petreczky et al. (2008b) and the example of this paper) for which they hold.

Related work To the best of our knowledge, the contribution of the paper is new. Control of hybrid systems using finite-state approximation is a classical topic, Alur et al. (2000); Gonzalez et al. (2001); Chutinan and Krogh (2003); Förstnera et al. (2002); Moor et al. (2002); Koutsoukos et al. (2000). The main difference with respect to Gonzalez et al. (2001); Chutinan and Krogh (2003); Koutsoukos et al. (2000) is the presence of partial observations, that the generation of events is not synchronous with inputs, and that the hybrid plant contains reset maps. With respect to Förstnera et al. (2002); Moor et al. (2002) the main differences are that we consider hybrid systems as opposed to continuous ones, and we address partial observations. In addition, we do not propose a general purpose finite-state abstraction, rather the proposed abstraction is intended as a vehicle for solving the specific control problem. The results of Raisch and O'Young (1995); Moor and Raisch (1999); Raisch (2000) address a problem which is quite different from the one considered in this paper. The approach of the paper resembles Alur et al. (2000); Tabuada and Pappas (2005); Fainekos et al. (2007); Belta et al. (2005). However, the abstraction notion of this paper and the problem formulation are different. The control problem of this paper is different from Philips et al. (2003). In addition, the computation of the finite-state abstraction proposed in this paper is quite different from that of the papers

[1] This work was partially supported by the ITEA project Twins 05004.

cited above. In Chutinan and Krogh (2003); Koutsoukos et al. (2000); Alur et al. (2000); Fainekos et al. (2007); Belta et al. (2005); Philips et al. (2003) the finite-state abstraction is computed by dividing the state-space of the system into regions. In Förstnera et al. (2002); Moor et al. (2002); Raisch and O'Young (1995); Moor et al. (2002); Moor and Raisch (1999); Raisch (2000), the abstraction of the system is constructed by storing the output (or state) response of the system to input sequences of finite length. In contrast, here the abstraction is obtained by sampling the hybrid system in time, not by discretizing it in space. In particular, the abstraction lives on the same state-space as the original system.

Outline of the paper In §3 we state the control problem we want to solve. The reduction of the hybrid problem to a discrete-event one is discussed in §4. In §5 the class of hybrid systems of interest is defined and the computation of a finite-state abstraction of the hybrid plant is discussed. In §6, as an illustration, we present an example.

2. PRELIMINARIES

General notation We use the standard notation and terminology from automata theory Eilenberg (1974). $\mathbb{N}$ is the set of natural numbers including zero. If Σ is a finite *alphabet*, then Σ^* denotes the set of finite *strings (words)* on Σ. The empty word is denoted by ϵ. An *infinite word* over Σ is an infinite sequence $w = a_1a_2 \cdots a_k \cdots$ with $a_i \in \Sigma$, $i \in \mathbb{N}$. The set of infinite words is denoted by Σ^ω. The length of a (in)finite word is denoted by $|w|$; if w is an infinite word, then $|w| = +\infty$. For any (in)finite word w, and for any $i \in \mathbb{N}$ (in case w is finite word, for any $0 \le i \le |w|$), $w_{1:i}$ denotes the finite word formed by the first i letters of w, i.e. $w_{1:i} = a_1a_2 \cdots a_i$. If $i = 0$, then $w_{1:i}$ is the empty word ϵ. The set of non-negative reals is $\mathbb{R}_+$.

Moore-automata A *Moore-automaton* (Eilenberg (1974)) is a tuple $A = (Q, I, Y, \delta, \lambda, q_0)$ where Q is the finite *state-space*, I is the *input alphabet*, Y is the *output alphabet*, $\delta : Q \times I \to Q$ is the *state-transition map*, $\lambda : Q \to Y$ is the *readout map*, and $q_0 \in Q$ is the *initial state*. The Moore-automaton A is a *realization* of a map $\phi : I^* \to Y$, if for all $w = u_1u_2 \cdots u_k \in I^*$, $k \ge 0$ and $u_1, u_2, \ldots, u_k \in I$, $\phi(w) = \lambda(q_k)$ where $q_i = \delta(q_{i-1}, u_i)$ for all $i = 1, 2, \ldots, k$.

Monoid automata Recall from Berstel (1979); Eilenberg (1974) that a *monoid* M is a semi-group with a unit element. A *finite-state automaton on a monoid* M, abbreviated as DFA, is a tuple $T = (Q, M, E, F, q_0)$ where Q is a finite set of states, M is the monoid of inputs, $E \subseteq Q \times M \times Q$ is a state-transition relation, where E is a finite set, $F \subseteq Q$ is the set of accepting states, $q_0 \in Q$ is the initial state. An element $m \in M$ is *accepted* by T if there exists $m_i \in M_i$ and $q_i \in Q$, $i = 1, 2, \ldots, k$, $k \ge 0$ such that $(q_i, m_{i+1}, q_{i+1}) \in E$ for $i = 0, 1, \ldots, k-1$, $q_k \in F$ and $m = m_1m_2 \cdots m_k$. The set $L \subseteq M$ is *recognized by* T, denoted by $L(T)$, if L consists of precisely those elements $m \in M$ which are accepted by T.

Sequential input-output maps will be used to model the discrete-event abstractions of hybrid systems. The concepts below are discussed in more detail in Petreczky et al. (2008a).

Definition 1. A multi-valued map $R : \Sigma^* \to 2^{X^* \times Y^*}$ is called a *sequential input-output map*, if

(1) $R(\epsilon) = (\epsilon, \epsilon)$, and for all $s \in \Sigma^*$, $R(s)$ is a non-empty set. Furthermore, R is *length-preserving* in its X-valued component, i.e. if $(\underline{x}, \underline{y}) \in R(s)$, with $\underline{x} \in X^*$ and $\underline{y} \in Y^*$, then the length of s and $\underline{x}$ are the same, i.e. $|s| = |\underline{x}|$,

(2) R is *prefix preserving*, i.e. for each word $s \in \Sigma^*$ and letter $a \in \Sigma$, if $(\underline{x}, \underline{y}) \in R(sa)$, then there exist $x \in X$ and $y \in Y^*$, $\hat{\underline{x}} \in X^*$, $\hat{\underline{y}} \in Y^*$ such that $\underline{x} = \hat{\underline{x}}x$, $\underline{y} = \hat{\underline{y}}y$ and $(\hat{\underline{x}}, \hat{\underline{y}}) \in R(s)$,

(3) R is *non-blocking*, i.e. for each $s \in \Sigma^*$, $a \in \Sigma$, $(\underline{x}, \underline{y}) \in R(s)$, $(\underline{x}x, \underline{y}y) \in R(sa)$ for some $x \in X$, $y \in Y^*$.

Definition 2. A DFA $T = (Q, M, E, F, q_0)$ defined over the monoid $M = \Sigma^* \times X^* \times Y^*$ is called a *quasi-sequential transducer*, if **(1)** $F = Q$, i.e. all states are accepting, **(2)** the state-transition relation E is a partial map $E : Q \times \Sigma \times X \times Y^* \to Q$, **(3)** for each state $q \in Q$ and letter $a \in \Sigma$ there exist $x \in X$ and $\underline{y} \in Y^*$ such that $E(q, u, x, \underline{y})$ is defined.

Definition 3. The sequential input-output map $R : \Sigma^* \to 2^{X^* \times Y^*}$ is *quasi-recognizable*, if there exists a quasi-sequential transducer which recognizes the graph of R, i.e. which recognizes the set $\{(\underline{u}, \underline{x}, \underline{y}) \in \Sigma^* \times X^* \times Y^* \mid (\underline{x}, \underline{y}) \in R(\underline{u})\}$.

3. CONTROL PROBLEM

The plant of interest is a hybrid system which reacts to discrete-valued control inputs and disturbances, and generates discrete-valued outputs and internal events. We view the inputs and outputs as discrete events. Thus, the control inputs are events generated by a potential controller, the disturbances are events generated by the environment. The outputs and internal events are events generated by the plant. The only difference between outputs and internal events is that outputs are visible (i.e. detectable by sensors), while internal events are not.

Notation 1. (Plant and events). We denote the plant by H. We denote by E_c the set of *control inputs*, E_d the set of *disturbances*, E_o the set of *outputs*, E_i the set of *internal events*. We assume that E_c, E_d, E_o, E_i are finite sets.

In order to define the input-output behavior of the plant formally, we need the following notion.

Definition 4. Let E be a finite set and let $\perp \notin E$. Consider a (in)finite timed sequence of elements of E.

$$s = (e_1, t_1)(e_2, t_2) \cdots (e_k, t_k) \cdots \tag{1}$$

where $0 \le t_1 < t_1 < t_2 < \cdots$, $e_{i+1} \in E$, $t_{i+1} \in \mathbb{R}_+$ for $i \in \mathbb{N}$, $i < |s|$. Here $|s|$ is the length of s, and $|s| = +\infty$ if s is an infinite sequence. If $|s| = +\infty$, then we assume that $\sup_{i \in \mathbb{N}} t_{i+1} = +\infty$. We can identify s with a map

$$g : \mathbb{R}_+ \ni t \mapsto \begin{cases} e_{i+1} \in E & \text{if } t = t_{i+1} \text{ for some } i \in \mathbb{N} \\ \perp & \text{otherwise} \end{cases} \tag{2}$$

The map g above, is called a *time-event map. The set of all such maps is denoted by* $\mathcal{P}_E$. Denote *the sequence of elements of* E *induced by* g by $\mathbf{UT}(g) = e_1e_2 \cdots e_k \cdots \in E^* \cup E^\omega$.

I.e., the timed-event function g takes values in the event set E at isolated time instances, and the value $\perp$ encodes the absence of events at a certain time instance. By applying the above definition to $E \in \{E_c, E_d, E_o, E_i\}$, we obtain the sets $\mathcal{P}_{E_c}, \mathcal{P}_{E_d}, \mathcal{P}_{E_o}, \mathcal{P}_{E_i}$ describing the time signals with values in inputs, disturbances, outputs and internal events respectively.

Definition 5. (Input-output map of the plant). The input-output map of H is a *causal* [2] map $\upsilon_H : \mathcal{P}_{E_c} \times \mathcal{P}_{E_d} \to \mathcal{P}_{E_o} \times \mathcal{P}_{E_i}$.

Definition 6. A *hybrid controller* is a map $\mathcal{C} : \mathcal{P}_{E_o} \to \mathcal{P}_{E_c}$.

We study controllers which have a finite-state representation and are activated at fixed sampling rate $\Delta > 0$. The controller can only detect the set of outputs which occurred in a sampling interval. The formal definition is as follows.

[2] By causality of υ_H we mean that the response of υ_H depends only on the past inputs and on the past and present disturbances, i.e. for any $u_i \in \mathcal{P}_{E_c}$ $d_i \in \mathcal{P}_{E_d}$, $(o_i, \hat{o}_i) = \upsilon_H(u_i, d_i)$, $i = 1, 2$, if $d_1|_{[0,t]} = d_2|_{[0,t]}$, $u_1|_{[0,t)} = u_2|_{[0,t)}$ then $o_1(t) = o_2(t)$ and $\hat{o}_1(t) = \hat{o}_2(t)$, for all $t \in \mathbb{R}_+$.