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 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.

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 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. 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.
- **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.

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, Gonzalez et al. (2001); Cury et al. (1998); Förstnera et al. (2002); Moor et al. (2002); Koutsoukos et al. (2000). The main difference with respect to Gonzalez et al. (2001); Cury et al. (1998); 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 cited above.

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). Let $\mathbb{N}$ be the set of positive integers including zero. Let Σ be a finite set, referred to as the *alphabet*. Σ^* denotes the set of finite *strings (words)* of elements of Σ. The empty word, 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 $0 \leq i \leq |w|$), $w_{1:i}$ 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* of A, I is the *input alphabet* of A, Y is the *output alphabet* of

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

A, $\delta : Q \times I \to Q$ is the *state-transition map* of A, $\lambda : Q \to Y$ is the *readout map* of A, and $q_0 \in Q$ is the *initial state* of A. The Moore-automaton A is a *realization* of a map $\phi : I^* \to Y$, if for all $w = u_1 u_2 \cdots u_k \in I^*$, $k \geq 0$ 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. Examples of monoids are the set of all words Σ^* and the cartesian product $X^* \times Y^*$, where X and Y are finite. Recall from Berstel (1979); Eilenberg (1974) that 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 finite set of accepting states, $q_0 \in Q$ is the initial state. An element $m \in M$ is *accepted* by T if there exists elements $m_i \in M_i$ and states $q_i \in Q$, $i = 1, 2, \ldots, k$, $k \geq 0$ such that $(q_i, m_i, q_{i+1}) \in E$ for $i = 0, 1, \ldots, k-1$, $q_k \in F$ and $m = m_1 m_2 \cdots m_k$. 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 maps*, if
(1) $R(\epsilon) = (\epsilon, \epsilon)$, and for all $s \in \Sigma^*$, $R(s)$ is a finite and non-empty set. Furthermore, R is *length-preserving* in its X-valued component, i.e. if $(\underline{o}, \underline{\hat{o}}) \in R(s)$, with $\underline{o} \in X^*$ and $\underline{\hat{o}} \in Y^*$, then the length of s and $\underline{o}$ are the same, i.e. $|s| = |\underline{o}|$,
(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 $y \in Y^*$, $\underline{\hat{x}} \in X^*$, $\underline{\hat{y}} \in Y^*$ such that $\underline{x} = \underline{\hat{x}}x$, $\underline{y} = \underline{\hat{y}}y$ and $(\underline{\hat{x}}, \underline{\hat{y}}) \in R(s)$,
(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 $E : Q \times \Sigma \times X \times Y^* \to Q$, **(2)** for each state $q \in Q$ and letter $a \in \Sigma$ there exist a letter $x \in X$ 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 $\{(\underline{u}, \underline{x}, \underline{y}) \in \Sigma^* \times X^* \times Y^* \mid (\underline{x}, \underline{y}) \in R(\underline{u})\}$.

Note that the subset of $\Sigma^* \times X^* \times Y^*$ recognized by a quasi-sequential transducer is always a sequential input-output map.

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.

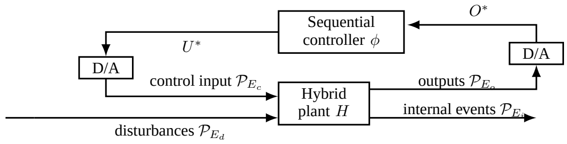

Fig. 1. Control architecture

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.

$$s = (e_1, t_1)(e_2, t_2) \cdots (e_k, t_k) \cdots \quad (1)$$

where $0 < t_1 < t_1 < t_2 < \cdots$, $e_{i+1} \in E$, $t_{i+1} \in \mathbb{R}_+$ 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

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

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 $\mathbf{UT}(g) = e_1 e_2 \cdots e_k \cdots \in E^* \cup E^\omega$.

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* map $\upsilon_H : \mathcal{P}_{E_c} \times \mathcal{P}_{E_d} \to \mathcal{P}_{E_o} \times \mathcal{P}_{E_i}$. By causality of υ_H we mean that the response of υ_H depends only on the past inputs and disturbances, i.e. for any two inputs $u_i \in \mathcal{P}_{E_c}$, disturbance $d_i \in \mathcal{P}_{E_d}$, and responses $(o_i, \hat{o}_i) = \upsilon_H(u_i, d_i)$, $i = 1, 2$, 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 $t \in \mathbb{R}_+$.

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

Next, we define when the feedback interconnection of the plant H and controller $\mathcal{C}$ is mathematically well-posed.

Definition 7. The interconnection of H and $\mathcal{C}$ is *well-posed* if for any disturbance $d \in \mathcal{P}_{E_d}$ there exists a unique input $u \in \mathcal{P}_{E_c}$, and responses $o \in \mathcal{P}_{E_o}$, $\hat{o} \in \mathcal{P}_{E_i}$ such that

$$(o, \hat{o}) = \upsilon_H(u, d) \text{ and } u = \mathcal{C}(o) \quad (3)$$

Next, we define the relevant aspects of the closed-loop behavior of the system. First, in order to avoid technical difficulties, we restrict attention to disturbances where at most a fixed number of disturbance events occurs within a sampling interval.

Definition 8. Denote by $\Delta > 0$ the sampling rate. Let $\mu \in \mathbb{N}$. The set of functions $g \in \mathcal{P}_{E_d}$ such that on any interval $(i\Delta, (i+1)\Delta]$, $i \in \mathbb{N}$ the number of events of g is not greater than μ is denoted by $\mathcal{P}^{\Delta}_{E,\mu}$. That is, $g \in \mathcal{P}^{\Delta}_{E,\mu}$, if and only if for each $i \in \mathbb{N}$, $card\{e = g(s) \in E \mid s \in ((i-1)\Delta, i\Delta)\} < \mu$.

Definition 9. If the interconnection of H and $\mathcal{C}$ is well-posed, then let *the closed-loop language* $L(H/\mathcal{C})$ be the set of words $\mathbf{UT}(\hat{o}) \in E_i^* \cup E_i^\omega$ for all internal event responses $\hat{o} \in \mathcal{P}_{E_i}$ for which there exist an input $u \in \mathcal{P}_{E_c}$ a disturbance $d \in \mathcal{P}^{\Delta}_{E_d,\mu}$, and output $o \in \mathcal{P}_{E_o}$ such that (3) holds.

I.e., $L(H/\mathcal{C})$ is the set of sequences of internal events generated by the interconnection of the plant H with the controller $\mathcal{C}$. 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.

Definition 10. Let $U = E_c \cup \{\perp\}$ be the *sampled input set*, let $O = 2^{E_o}$ be the *sampled output set*. A *sequential controller* is a map $\phi : O^* \to U$ which has a Moore-automaton realization.

Definition 11. (Sampling-based controller). For a sequential controller ϕ let the *hybrid controller* $\mathcal{C}_\phi : \mathcal{P}_{E_o} \to \mathcal{P}_{E_c}$ *associated with* ϕ be such that for all $o \in \mathcal{P}_{E_o}$, and for all $t \in \mathbb{R}_+$,

$$\mathcal{C}_\phi(o)(t) = \begin{cases} \phi(S_1 S_2 \cdots S_k) & \text{if } t = k\Delta \text{ for } , k \in \mathbb{N} \\ \perp & \text{otherwise} \end{cases}$$

where $S_{i+1} = o(((i\Delta, (i+1)\Delta]) \cap E_o$ for all $i \in \mathbb{N}$.

Notice that the interconnection of $\mathcal{C}_\phi$ and H is well-posed. The control problem of interest can be stated as follows.

Problem 1. (Sampled-data control). For a specification language $K \subseteq E_i^* \cup E_i^\omega$ and a sampling rate $\Delta > 0$, find a sequential controller ϕ such that for the associated hybrid controller $\mathcal{C}_\phi$, the closed-loop language satisfies $L(H/\mathcal{C}_\phi) \subseteq K$.

Note that the results of the paper can easily be extended so that the specification language includes events from $E_c \cup E_d \cup E_o$.

4. SOLUTION OF THE HYBRID CONTROL PROBLEM

In this section we present the solution of Problem 1. The main idea is to reduce Problem 1 to a discrete-event control problem. To this end, we model the symbolic sampled-data behavior of the plant as a discrete-event system R_H, which reacts to sampled inputs and disturbances and generates sampled outputs and internal events. The input set of R_H is U, the output set is O, the set of internal events is E_i, where the U and O are as in Definition 10.In order to define R_H, we need the following.

Definition 12. The set *sampled disturbances of* R_H is defined as $D = \bigcup_{k=0}^{\mu} E_d^k$, i.e. D is the set of all words over E_d of length at most μ, where μ is as in Definition 8.

Notation 2. Let $g \in \mathcal{P}_E$ be of the form (2). For all $t \in \mathbb{R}^+$, let $\mathbf{UT}(g,t) \in E^*$, be the sequence of events of g up to t, i.e. $\mathbf{UT}(g,t) = e_1 e_2 \cdots e_l$ if $l \in \mathbb{N}$ is such that either $l < |s|$ and $t \in (\sum_{r=1}^{l} t_r, \sum_{r=1}^{l+1} t_r]$ or $|s| = l$ and $t \in (\sum_{r=1}^{l} t_r, +\infty)$.

Definition 13. The *sequential input-output map* R_H *of* H is the map $R_H : (U \times D)^* \to 2^{O^* \times E_o^*}$ defined as follows. $R_H(\epsilon) = \{(\epsilon, \epsilon)\}$ and for each sequence of sampled inputs $u_1, u_2, \ldots, u_k \in U$ and disturbances $d_1, d_2, \cdots d_k \in D$, $k \geq 0$,

$$(o_1 o_2 \cdots o_k, \underline{\hat{o}}) \in R_H((u_1, d_1)(u_2, d_2) \cdots (u_k, d_k))$$

for some $o_1, o_2, \ldots, o_k \in O$, and $\underline{\hat{o}} \in E_i^*$, if there exist $g \in \mathcal{P}_{E_d}$, $o \in \mathcal{P}_{E_o}$, $\hat{o} \in \mathcal{P}_{E_i}$ such that $(o, \hat{o}) = \upsilon_H(u, g)$,

$$\forall t \in \mathbb{R}_+ : u(t) = \begin{cases} u_i & \text{if } t = (i-1)\Delta \text{ for } i = 1, 2, \ldots, k \\ \perp & \text{otherwise} \end{cases}$$

$\underline{\hat{o}} = \mathbf{UT}(\hat{o}, k\Delta)$, and $o_i = o(((i-1)\Delta, i\Delta])$ $d_i = \mathbf{UT}(g_i, \Delta)$, where $g_i(t) = g(t + (i-1)\Delta), \forall t \in \mathbb{R}_+$, for all $i = 1, 2, \ldots, k$.

The map R_H is a sequential input-output map of Definition 14. Intuitively, R_H is the result of composing H with the interfaces converting outputs from $\mathcal{P}_{E_o}$, internal event signals from $\mathcal{P}_{E_i}$, disturbances from $\mathcal{P}_{E_d,\mu}^{\Delta}$ to sequences in O^*, E_i^* and D^*, and with the interface which convert sequences U^* to maps $\mathcal{P}_{E_c}$.

In order to solve Problem 1, we can view R_H as a discrete-event plant, and solve a discrete-event control problem for R_H as a plant and K as a requirement. The discrete-event control problem is as follows. The controllers of interest are sequential controllers. The plants of interest are defined as follows.

Definition 14. A *discrete-event plant* is a sequential input-output map $R : (U \times D)^* \to 2^{O^* \times E_i^*}$.

Definition 15. The *closed-loop language* $L(R/\phi)) \subseteq E_i^* \cup E_i^\omega$ of the interconnection of R with the sequential controller $\phi : O^* \to U$ is the set of all words $\underline{\hat{o}} \in E_i^* \cup E_i^\omega$ for which there exist letters $d_i \in D$, $o_i \in O$, $u_i \in O$, $i \in \mathbb{N}$ and indices $k_0 \leq k_1 \leq \cdots k_i \leq$ such that $\sup_{i \in \mathbb{N}} k_i = |\underline{\hat{o}}|$, and $\forall i \in \mathbb{N}$,

$$(o_1 o_2 \cdots o_i, \underline{\hat{o}}_{1:k_i}) \in R((u_1, d_1)(u_2, d_2) \cdots (u_i, d_i))$$
$$u_i = \phi(o_1 o_2 \cdots o_{i-1})$$

Problem 2. (Discrete control problem). For the plant R, and for the *control requirements* $K \subseteq E_i^* \cup E_i^\omega$, find a sequential controller ϕ such that $L(R/\phi) \subseteq K$ holds.

For more details on the discrete-event control problem above, see Petreczky et al. (2008a). A necessary condition for effective solution of Problem 2 R is that is quasi-recognizable, i.e. it is recognized by a quasi-sequential transducer.

Theorem 1. (Hybrid vs. discrete control). If ϕ is a sequential controller, then $L(H/\mathcal{C}_\phi) \subseteq L(R_H/\phi)$. Hence, if ϕ solves Problem 2 for $R = R_H$, and $K \subseteq E_i^* \cup E_i^\omega$, then the associated hybrid controller $\mathcal{C}_\phi$ solves Problem 1 for H and K.

In fact, it can be shown that the set of prefixes of $L(R_H/\phi)$ and of $L(H/\mathcal{C}_\phi)$ coincide. Hence, if K is a limit of a prefix closed set of finite strings, i.e. K is a safety requirements, then $\mathcal{C}_\phi$ solves Problem 1 if and *only if* ϕ solves Problem 2 for $R = R_H$.

Notice that R_H needs not admit a finite-state representation suitable for solving Problem 2. The remedy is to solve Problem 2 not for R_H but for an quasi-recognizable abstraction of R_H. The construction of the latter is discussed in §5.

Definition 16. (Abstraction). The sequential input-output map R is an abstraction of the map R_H if for all $s \in (U \times D)^*$, the inclusion $R_H(s) \subseteq R(s)$ holds.

Theorem 2. Assume that R is an abstraction or R_H. Then for any sequential controller ϕ, $L(R_H/\phi) \subseteq L(R/\phi)$. Hence, if ϕ solves Problem 2 for R, then ϕ solves Problem 2 for R_H.

We get the following procedure for solving Problem 1.

1. Use §5 to compute a finite-state abstraction R of R_H
2. Compute a solution to Problem 2 for R and the original control requirements K.
3. Compute the hybrid controller $\mathcal{C}_\phi$ associated with ϕ.

5. FINITE-STATE ABSTRACTION OF R_H

In §5.1 we define the class of hybrid systems of interest. In §5.2 we present the definition the finite-state abstraction of R_H.

5.1 Hybrid systems

Definition 17. A discrete i/o hybrid system H is a tuple

$$(S_H, \delta, \lambda_i, \lambda_o, \{f_q, R_{u,q}, \Phi_{q,e}\}_{q \in Q, u \in E_c, e \in E_i \cup E_o}, h_0) \quad (4)$$

- **Events** E_d is the set of *disturbances*, E_c is the set of *control inputs*, E_o is the set of *outputs*, E_i is the set of *internal events*, and E_c, E_d, E_i, E_o are finite sets.
- **State-space** $S_H = Q \times \mathcal{X}$ is the state-space of H. Here $Q = Q_c \times Q_d$ is the *discrete state-space* of H, Q_c, Q_d are finite sets. The set $\mathcal{X} \subseteq \mathbb{R}^n$ is the *continuous state space*, $\mathcal{X}$ is a closed set with non-empty interior $\mathrm{int}\, \mathcal{X} \neq \emptyset$.
- **Discrete-state transition** is determined by the transition functions $\delta_c : Q \times E_c \to Q_c$, $\delta_d : Q \times (E_d \cup E_i) \to Q_d$.
- **Continuous dynamics** is determined by continuous, globally Lipschitz vector fields, $f_{q^c} : \mathbb{R}^n \to \mathbb{R}^n$, $q \in Q_c$, and reset maps $R_{u,q} : \mathcal{X} \to \mathcal{X}$, $q \in Q$ and $u \in E_c$.
- **Event generation** is determined by guards $\Phi_{q,e} \subseteq \mathcal{X}$, $q \in Q$, $e \in E_o \cup E_i$, and by discrete partial readout maps $\lambda_o : Q \times E_d \to E_o$, $\lambda_i : Q \times E_d \to E_i$.

- $h_0 = (q_0^c, q_0^d, x_0) \in S_H$ is the initial state of the system.

The system H is a hybrid system van der Schaft and Schumacher (2000) subject to the following restrictions. The set of discrete event is $E = E_c \cup E_d \cup E_o \cup E_i$. An event $e \in E_o \cup E_i$ is generated by H either if the continuous state crosses a guard set, or when an event from E_d arrives. Events from $E_c \cup E_d$ are generated by the controller/environment. The continuous dynamics in the discrete state (q^c, q^d) depends only on q^c. The state-transition rule for a discrete state $q = (q^c, q^d) \in Q$ is as follows. If an event u from E_c arrives, the new discrete state becomes $(\delta_c(q, u), q^d)$. If $d \in E_d$ arrives, then the discrete state changes to $(q^c, \delta_d(q, d))$. If an event $e \in E_i$ occurs, then the discrete state changes to $(q^c, \delta_d(q, e))$. For an event from E_o the discrete state does not change. The reset maps for an event $u \in E_c$ are specified by $R_{u,q}$. For all the events from $E_d \cup E_o \cup E_i$ the corresponding reset map is the identity. For the formal definition of the state evolution, we need the following.

Definition 18. (Flow of f_{q^c}). For any time $t \in \mathbb{R}_+$ and for any $q^c \in Q_c$ define the *flow* $f^t_{q^c} : \mathcal{X} \to \mathcal{X}$ of f_{q^c} as follows. For any $z_0 \in \mathcal{X}$, consider the initial value problem

$$\dot{z} = f_{q^c}(z) \text{ and } z(0) = z_0 \tag{5}$$

Since f_{q^c} is continuous globally Lipschitz, the solution is defined on on the whole time axis $\mathbb{R}_+$ and there exists $\beta = \beta(q^c, z_0) \in [0, +\infty]$ such that for all $t \in [0, \beta)$, $z(t) \in \text{int}\, \mathcal{X}$ and if $\beta < +\infty$, then $z(\beta) \in \partial X$, i.e. $z(\beta)$ belongs to the boundary of $\mathcal{X}$. Then $f^t_{q^c}(z_0) = \begin{cases} z(t) & \text{if } t < \beta \\ z(\beta) & \text{if } \beta \le t < +\infty \end{cases}$.

Notice that for any $z_0 \in \partial X$, $f^t_{q^c}(z_0) = z_0$, i.e. the continuous state evolution stops on the boundary of $\mathcal{X}$. The following assumptions will be used in the rest of the paper.

A.1. **Disjoint guards:** For any $\Sigma \in \{E_o, E_i\}$ and $q \in Q$, $\forall e_1 \neq e_2 \in \Sigma : \Phi_{q,e_1} \cap \Phi_{q,e_2} = \emptyset$. in

A.2. For each $q = (q^c, q^d) \in Q = Q_c \times Q_d$, $e \in E_o \cup E_i$ there exist smooth maps $h_{q,e} : \mathbb{R}^n \to \mathbb{R}$, such that
$\Phi_{q,e} \subseteq \{x \in \text{int}\, \mathcal{X} \mid h_{q,e}(x) = 0\}$, and
if $\Phi_{q,e} \neq \emptyset$, then $\forall x \in \mathbb{R}^n : \text{grad}(h_{q,e})(x) f_{q^c}(x) > 0$.

A.3. For any $q \in Q$, $d \in E_d$, $\lambda_i(q, d)$ is defined. Moreover, if $e = \lambda_i(q, d)$, then for any $\hat{q} \in Q$, $\Phi_{\hat{q},e} = \emptyset$.

Assumption A.1 ensures that at most one output and at most one internal event is generated at any time instance. Assumption A.2 ensures that only a finite number of outputs or internal events are generated on any finite time interval. Assumption A.3 allows to recognize whether an internal event is generated by a discrete readout map or by crossing a guard. Next, we define the state evolution and input-output behavior of H.

Definition 19. For any initial state $h = (q, x)$, input $u \in \mathcal{P}_{E_c}$ and disturbance $d \in \mathcal{P}_{E_d}$ the state-trajectory is a map

$$\xi_H(h, u, d) : \mathbb{R}_+ \ni t \mapsto (q(t), x(t)) \in S_H$$

where the state components $q(t) = (q^c(t), q^d(t)) \in Q$ $x(t) \in \mathcal{X}$ satisfy the following. If $t > 0$, then let $q(t^-) = \lim_{s \uparrow t} q(s)$, i.e. $q(t^-)$ is the left hand-side limit of $q(s)$ at time t. Let $q^c(t^-)$ and $q^d(t^-)$ be the Q_c- and Q_d-valued components of $q(t^-)$, i.e. $q(t^-) = (q^c(t^-), q^d(t^-)$. If $t > 0$, then $x(t^-) = \lim_{s \uparrow t} x(s)$, i.e. $x(t^-)$ is the left-hand side limit at t of the map $s \mapsto x(s)$. Then, $(q(0), x(0)) = h = (q, x)$ and $\forall t \in \mathbb{R}_+, t > 0$,

- if $u(t) = u \in E_c$, then $q^c(t) = \delta(q(t^-), u)$ and $x(t) = R_{u,q(t^-)}(x(t^-))$. If $u(s) = \perp$ on the interval $(t - r, t]$ for some $r > 0$, then $q^c(t) = q^c(t - r) = q^c$ and $x(t) = f^r_{q^c}(x(t-r))$, where $f^r_{q^c}$ is the flow for time $r \ge 0$ as in Definition 18
- let $r > 0$ be such that for all $s \in (t - r, t)$, $d(s) = \perp$, $u(s) = \perp$ and $x(s^-) \notin \bigcup_{e \in E_i} \Phi_{q(s^-),e}$, i.e. no disturbance, input or internal event takes place on the interval $(t - r, t)$. Then $q^d(s) = q^d(t - r)$ for all $s \in (t - r, t)$. If $d(t) = e \in E_d$, i.e. a disturbance event occurs at time t, then $q^d(t) = \delta_d(q(t^-), e)$. If $d(t) = \perp$, and $x(t^-) \in \Phi_{q(t^-),e}$ for some $e \in E_i$, then $q^d(t) = \delta_d(q(t^-), e)$. If both $d(t) = \perp$ and $x(t^-) \notin \bigcup_{e \in E_i} \Phi_{q(t^-),e}$, then $q^d(t) = q^d(t^-)$.

Definition 20. Define the *input-output map* of the hybrid system H induced by state $h \in S_H$ as $\upsilon_{H,h} : \mathcal{P}_{E_c} \times \mathcal{P}_{E_e} \to \mathcal{P}_{E_o} \times \mathcal{P}_{E_i}$ such that for any input $u \in \mathcal{P}_{E_c}$ and disturbance $d \in \mathcal{P}_{E_d}$, $\upsilon_{H,h}(u, d) = (o, \hat{o})$ if the following holds. For each $t \in \mathbb{R}_+$ consider the current state $\xi_H(h, u, d)(t) = (q(t), x(t))$. Recall from Definition 19 the definition of $q(t^-)$ and $x(t^-)$. Then

$$o(t) = \begin{cases} e \in E_o & \text{if } x(t^-) \in \Phi_{q(t^-),e} \text{ and } d(t) = \perp, \\ & \text{and } t > 0 \\ \lambda_o(q(t^-), d(t)) & \text{if } d(t) \in E_d, t > 0 \text{ and} \\ & \lambda_o(q(t^-), d(t)) \text{ is defined} \\ \perp & \text{otherwise} \end{cases}$$

$$\hat{o}(t) = \begin{cases} e \in E_i & \text{if } x(t^-) \in \Phi_{q(t^-),e} \text{ and} \\ & d(t) = \perp \text{ and } t > 0 \\ \lambda_i(q(t^-), d(t)) & \text{if } d(t) \in E_d, t > 0 \text{ and} \\ & \lambda_i(q(t^-), d(t)) \text{ is defined} \\ \perp & \text{otherwise} \end{cases}$$

We denote by υ_H the input-output map υ_{H,h_0} of H induced by the initial state h_0 of H.

Informally, if there are no disturbances, then an output or internal event is generated if the continuous state crosses a guard. If a disturbance arrives, then an output (resp. internal event) is generated according to the readout map λ_o (resp. λ_i).

5.2 Construction of a finite-state abstraction of R_H

Below we present the definition of the quasi-sequential transducer, which recognizes an abstraction of R_H. Below H denotes a hybrid system of Definition 17 satisfying Assumption A.1– A.3. In addition, we need the following.

Definition 21. Let $\mathcal{R}(H) = \bigcup_{i=0}^{\infty} Q \times H_i$, such that

$H_0 = \{x_0\}$ and $H_{i+1} = H_i \cup \{f^{\Delta}_{q^c}(x), f^{\Delta}_{q^c}(R_{u,s}(x)) \mid x \in H_i,$
$q^c \in Q_c, s \in Q, u \in E_c\}, \forall i \in \mathbb{N}$

where x_0 is the continuous component of the initial state of H.

Assumption 1. In the sequel we assume that $\mathcal{R}(H)$ is finite.

$\mathcal{R}(H)$ will be the state-space of the to be constructed abstraction. Later on we formulate conditions for finiteness of $\mathcal{R}(H)$. The main idea behind the construction of the sampled-time abstraction is that it is enough to look at states which are reached at sampling times, i.e. at a subset of elements of $\mathcal{R}(H)$. Moreover, the events generated in a sampling interval can be estimated by using the sampled state.

Definition 22. For any $q = (q^c, q^d) \in Q$ and $e \in E_i \cup E_o$, the *guard abstraction predicate* $P_{q,e} \subseteq \mathcal{X}$ is either $P_{q,e} = \emptyset$, if $e = \lambda(q, d)$ for some $d \in E_d$, or

$$P_{q,e} = \{x \in \mathcal{X} \mid h_{q,e}(x) \le 0 \text{ and } h_{q,e}(f^{\Delta}_{q^c}(x)) \ge 0\} \tag{6}$$

Informally, $P_{q,e}$ contains those continuous states, started from which the guard corresponding to e is crossed within Δ time.

Definition 23. Let $\mathcal{P} = \{P_{q,e}\}_{q \in Q, e \in E_i \cup E_o}$ the collection of sets from Definition 22. Define the *finite-state abstraction* H_{Δ} as a quasi-sequential transducer

$H_\Delta = (\mathcal{R}(H), (U \times D)^* \times O^* \times E_i^*, E, \mathcal{R}(H), h_0)$ where

Initial state $h_0 = (q_0^c, q_0^d, x_0)$ of H_Δ coincides with that of H.

State transition map $E : \mathcal{R}(H) \times (U \times D) \times O \times E_i^* \to \mathcal{R}(H)$ is defined as follows. For each $u \in U$, $d \in D$, $o \in O$ and $\hat{o} \in E_i^*$, $E(h_1, u, d, o, \hat{o})$ is defined and $E(h_1, u, d, o, \hat{o}) = h_2$ if and only if $h_i = (q_i, x_i) \in \mathcal{R}(H)$ where $q_i = (q_i^c, q_i^d) \in Q_c \times Q_d$ and $x_i \in \mathcal{X}$, $i = 1, 2$, and the following holds.

(1) The state components q_2^c and x_2 are computed as follows.

$$q_2^c = \delta_c(q_1, u) \text{ and } x_2 = f^{\Delta}_{q_2^c}(R_{u,q_1}(x_1)) \tag{7}$$

Here, for $u = \perp$, $\delta_c(q_1, u)$, $R_{u,q_1}(x_1)$ are the identity maps, i.e. $\delta_c(q_1, \perp) = q_1^c$ and $R_{\perp,q_1}(x_1) = x_1$

(2) Assume that $d = e_1 e_2 \cdots e_k$, $0 \le k \le \mu$, $e_1, e_2, \ldots, e_k \in E_d$. Then the sequence $\hat{o}$ is of the form $\hat{o} = z_1 z_2 \cdots z_l$, where $k \le l \le |Q_d||E_i| + k$ and $z_1, z_2, \ldots, z_l \in E_i \cup \{\epsilon\}$ and the following holds. There exists indices $i_1 < i_2 < \cdots < i_k \in \{1, 2, \ldots, l\}$ and discrete states $s_i \in Q$, $i = 0, 1, \ldots, l$ such that $s_0 = (q_2^c, q_1^d)$, $s_l = q_2$ and for all $i = 1, 2, \ldots, l$

$$s_i = \begin{cases} (q_2^c, \delta_d(s_{i-1}, z_i)) & \text{if } R_{u,q_1}(x_1) \in P_{s_{i-1}, z_i} \text{and } i \notin I \\ (q_2^c, \delta_d(s_{i-1}, e_r)) & \text{if } i = i_r \text{ and } z_i = \lambda_i(s_{i-1}, e_r) \\ & \quad \text{for some } r = 1, 2, \ldots, k, \end{cases} \tag{8}$$

where $I = \{i_1, i_2, \ldots, i_k\}$.

(3) The output $o \subseteq 2^{E_o}$ is the set of events $e \in E_o$ such that

$$\begin{aligned} &R_{u,q_1}(x_1) \in P_{s_i,e} \text{ for some } i \in \{1, 2, \ldots, l\} \setminus I, \text{ or} \\ &\lambda_o(s_{i_r - 1}, e_r) = e \text{ for some } r = 1, 2, \ldots, k \end{aligned} \tag{9}$$

Intuition The states of H_Δ are those states of H which can be reached from h_0 at sampling times. By assumption, this set is finite. A state transition of H_Δ associated with a discrete input u, disturbance $d \in D$, output $o \in O$ and sequence of internal events $\hat{o} \in E_i^*$ is obtained as follows. If the current state of H_Δ is h_1 then the new state h_2,is the state of H reachable from h_1 in time Δ, under the following conditions; **(1)** H receives input event u at time 0, and no input after that, **(2)** H receives a disturbance g, such that the sequence of events of g on $(0, \Delta]$ is d, **(3)** $\hat{o}$ is the sequence of internal events generated by H while moving from h_1 to h_2, **(4)** o is the set of outputs generated by H while moving from state h_1 to h_2. Condition **(1)** and the fact that the Q_c- and $\mathbb{R}^n$-valued state components depend only on the time and input events yield (7). The computation of the Q_d-valued states along with checking Condition **(2) – (3)** is formalized in (8). Finally, Condition **(4)** is formalized in (9).

Theorem 3. The tuple H_Δ is a quasi-sequential transducer, and the sequential input-output map $R(H_\Delta)$ recognized by H_Δ is an abstraction of R_H.

Finiteness of $\mathcal{R}(H)$ based on Lyapunov-like functions

Theorem 4. Consider a finite set $\mathcal{X}_0 \subseteq \operatorname{int} \mathcal{X}$ and a smooth map $V : \mathcal{X} \to \mathbb{R}$ such that for all $x \in \mathcal{X}$, $q = (q^c, q^d) \in Q$,

(1) $V(x) \ge 0$ and $V^{-1}(0) \subseteq \partial\mathcal{X}$.

(2) There exists $c > 0$ such that $\operatorname{grad}(V)(x) f_{q^c}(x) < -c$,

(3) For all $u \in E_c$, if $x \in \operatorname{int} \mathcal{X}$, then $V(R_{u,q}(x)) \le V(x)$, and if $x \in \partial\mathcal{X}$, then $V(x) \in \mathcal{X}_0$.

It then follows that $\mathcal{R}(H)$ is finite.

Computation Notice that if the reset maps, flows of the vector fields (as in Definition 18), and the functions $h_{q,e}$ defining guards are (numerically) computable then so is H_Δ. However, the computational complexity can get large as Δ decreases. *Note that in this paper by (numerical) computability we mean existence of a numerical method, not computability in a mathematically rigorous sense.* The latter is left as future work.

Assumption 2. The reset maps of H are affine in $\operatorname{int} \mathcal{X}$, the vector fields of are of L'ure-type, the state-space is a polyhedron, and the maps defining the guards are affine, i.e.

$$\begin{aligned} &\mathcal{X} = \{x \in \mathbb{R}^n \mid n_i^T x - b_i \le 0, i = 1, 2, \ldots, K\} \\ &R_{u,q}(x) = M_{u,q} x + b_{u,q}, \quad \forall x \in \operatorname{int} \mathcal{X} \\ &h_{q,e}(x) = g_{q,e}^T x + d_{q,e}, \quad \forall x \in \mathbb{R}^n \\ &f_{q^c}(x) = A_{q^c} x + \sum_{j=1}^{m} B_{q^c,j} \phi_{q^c,j}(r_{q^c,j}^T x), \quad \forall x \in \mathbb{R}^n \\ &\mu_1 \sigma + \gamma_1 \le \phi_{q^c,j}(\sigma) \le \mu_2 \sigma + \gamma_2, \quad \forall \sigma \in \mathbb{R} \end{aligned}$$

for matrices $M_{u,q}, A_{q^c} \in \mathbb{R}^{n \times n}$, vectors $b_{u,q}, r_{q^c,j}, B_{q^c,j}, g_{q,e}, n_i \in \mathbb{R}^n$, and scalars $d_{q,e}, b_i, \mu_1, \mu_2, \gamma_1, \gamma_2 \in \mathbb{R}$, $q = (q^c, q^d) \in Q$, $e \in E_i \cup E_o$, $u \in E_c$, $i = 1, 2, \ldots, K$, $j = 1, 2, \ldots, m$. The maps $\phi_{q^c,j} : \mathbb{R} \to \mathbb{R}$, $j = 1, 2, \ldots, m$ are piecewise-affine, continuous, globally Lipschitz.

If H satisfies Assumption 2, then the reset maps are and the maps $h_{q,e}$ are computable. The solution of (5) can be computed using numerical integration methods. Hence, if we can detect reaching the boundary of $\mathcal{X}$, then the flow is computable. In fact, the definition of H_Δ can be modified so that it is enough to detect if the solution of (5) has crossed the boundary the interval $(0, \Delta]$ (i.e. the precise point where the boundary was crossed is not needed). The latter is easy if the sign of each $n_i^T f_{q^c}(x)$, $i = 1, 2, \ldots, K$ is independent of x. Due to the lack of space, we omit the details. The finiteness of $\mathcal{R}(H)$ can be checked effectively using Theorem 4 and the following.

Proposition 1. Assume that H satisfies Assumption 2. If for some $j \in \{1, \ldots, K\}$, $c > 0$, for all $x \in \mathcal{X}$, $q = (q^c, q^d) \in Q$,

(1) $n_j^T (A_{q^c} x + \sum_{l=1}^{m} \mu_i (B_{q^c,l} r_{q^c,l}^T x + \gamma_i B_{q^c,l})) > c$, $i = 1, 2$.

(2) If $x \in \operatorname{int} \mathcal{X}$, then $n_j^T (M_{u,q} x - x + b_{u,q}) \ge 0$, $\forall u \in E_c$,

then $V(x) = (b_j - n_j^T x)$ satisfies Theorem 4.

Notice the resemblance of Proposition 1 to Habets et al. (2006). Note that quadratic Lyapunov-like functions satisfying Theorem 4 can also be obtained as solutions of suitable LMIs.

6. ILLUSTRATING EXAMPLE

Below we illustrate the theory by an example related to a control problem for printers from Petreczky et al. (2008b).

Formal model of the plant We will use the following parameters, meaning of which is described in Petreczky et al. (2008b): $\mathbf{Fp}$, $\mathbf{Cp}$, $\mathbf{V}_{max}$, $\mathbf{V}_{min}$, $\mathbf{T}_{fo}$, $\mathbf{T}_{pl,max}$, $\mathbf{T}_{pl,min}$, $\mathbf{A}$, $\mathbf{D}$. Formally, the plant model H is of the form (4). The components of H are explained below. The event sets are $E_c = \{\mathbf{c}_{FU}, \mathbf{c}_{FD}, \mathbf{c}_A, \mathbf{c}_D\}$, $E_o = \{\mathbf{e}_{o,PL}\}$, $E_d = \{\mathbf{e}_{d,PL}\}$, $E_i = \{\mathbf{e}_{NPIF}, \mathbf{e}_{i,PL}, \mathbf{e}_{min,PL}, \mathbf{e}_{max,PL}, \mathbf{e}_{FUc}\}$. The discrete state-space $Q = Q_c \times Q_d$ is defined as follows. Q_d is the set of maps $\phi : \mathbf{Var}_d \to \{True, False\}$, where $\mathbf{Var}_d = \{\mathbf{S}_{PL}, \mathbf{S}_r, \mathbf{S}_{FUc}\}$. Q_c is the set of all maps $\phi : \mathbf{Var} \to \{True, False\}$ where $\mathbf{Var} = \{\mathbf{S}_{FU}, \mathbf{S}_{FD}, \mathbf{S}_A, \mathbf{S}_D\}$. I.e. the elements of Q_d and Q_c are valuations of predicates from $\mathbf{Var}_d$ and $\mathbf{Var}_c$ respectively. In the sequel, we will write $\phi(X)$ instead of $\phi(X) = True$, and $\neg\phi(X)$, instead of $\phi(X) = False$ for all $\phi \in Q_d$, $X \in \mathbf{Var}_d$, or $\phi \in Q_c$ and $X \in \mathbf{Var}_c$. The continuous state-space is $\mathcal{X} = \{x = (\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) \in \mathbb{R}^4 \mid \mathbf{P} \le \mathbf{Cp}\}$ where $\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T} \in \mathbb{R}$ are state variables. The vector fields f_{q^c}, $q^c \in Q_c$ and the reset maps $R_{u,q}$, $q \in Q$, $u \in E_c$ are as follows. For any $x = (\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) \in \mathcal{X}$,

$$f_{q^c}(x) = \begin{bmatrix} \max\{\mathbf{V}_{min}, \mathbf{V}\} & f_{2,q^c}(x) & 1 & 1 \end{bmatrix}^T$$

$$f_{2,q^c}(x) = \begin{cases} \mathbf{A}\phi_{min}(x)\phi_{max}(x) & \text{if } q^c(\mathbf{S}_A) \\ -\mathbf{D}\phi_{min}(x)\phi_{max}(x) & \text{if } q^c(\mathbf{S}_D) \text{ and } q^c(\mathbf{S}_{FD}) \end{cases}$$

$$\phi_{min}(x) = \begin{cases} 1 & \text{if } \mathbf{V} \in (\mathbf{V}_{min} + \epsilon, +\infty) \\ \dfrac{(\mathbf{V} - \mathbf{V}_{min})}{\epsilon} & \text{if } \mathbf{V} \in (\mathbf{V}_{min}, \mathbf{V}_{min} + \epsilon] \\ 0 & \text{if } \mathbf{V} \in (-\infty, \mathbf{V}_{min}] \end{cases}$$

$$\phi_{max}(x) = \begin{cases} 1 & \text{if } \mathbf{V} \in (-\infty, \mathbf{V}_{max} - \epsilon) \\ \dfrac{(\mathbf{V}_{max} - \mathbf{V})}{\epsilon} & \text{if } \mathbf{V} \in [\mathbf{V}_{max} - \epsilon, \mathbf{V}_{max}) \\ 0 & \text{if } \mathbf{V} \in [\mathbf{V}_{max}, +\infty) \end{cases}$$

$$R_{u,q}(x) = \begin{cases} (\mathbf{P}, \mathbf{V}, 0, \mathbf{T}) & \text{if } u = \mathbf{c}_{FD} \text{ and } \mathbf{P} < \mathbf{Cp} \\ (\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) & \text{if } u \neq \mathbf{c}_{FD} \text{ and } \mathbf{P} < \mathbf{Cp} \\ (\mathbf{Cp}, \mathbf{V}_{max}, \mathbf{T}_{fo}, \mathbf{T}_{pl,max}) & \text{if } \mathbf{P} = \mathbf{Cp} \end{cases}$$

The state-transition maps δ_c and δ_d are such that for each $q_1 = (q_1^c, q_1^d) \in Q$, $u \in E_c$, $e \in E_i \cup E_d$, $\delta_c(q_1, u) = q_2^c$ and $\delta_d(q_1, e) = q_2^d$ if and only if the following holds.

$$(q_2^c(\mathbf{S}_{FD}), q_2^c(\mathbf{S}_{FU})) = \begin{cases} (True, False) & \text{if } u = \mathbf{c}_{FD} \\ (False, True) & \text{if } u = \mathbf{c}_{FU} \\ (q_1^c(\mathbf{S}_{FD}), q_1^c(\mathbf{S}_{FU})) & \text{otherwise} \end{cases}$$

$$(q_2^c(\mathbf{S}_A), q_2^c(\mathbf{S}_D) = \begin{cases} (False, True) & \text{if } u = \mathbf{c}_D \\ (True, False) & \text{if } u = \mathbf{c}_A \\ (q_1^c(\mathbf{S}_A), q_1^c(\mathbf{S}_D)) & \text{otherwise} \end{cases}$$

$$q_2^d(\mathbf{S}_{PL}) = \begin{cases} True & \text{if } e = \mathbf{e}_{d,PL} \text{ and } q_1^d(\mathbf{S}_r) \\ q_1^d(\mathbf{S}_{PL}) & \text{otherwise} \end{cases}$$

$$q_2^d(\mathbf{S}_r) = \begin{cases} True & \text{if } e = \mathbf{e}_{min,PL} \text{ and } \neg q_1^d(\mathbf{S}_r) \\ False & \text{if } e = \mathbf{e}_{max,PL} \text{ and } q_1^d(\mathbf{S}_r) \\ q_1^d(\mathbf{S}_r) & \text{otherwise} \end{cases}$$

$$q_2^d(\mathbf{S}_{FUc}) = \begin{cases} True & \text{if } e = \mathbf{e}_{FUc} \\ q_1^d(\mathbf{S}_{FUc}) & \text{otherwise} \end{cases}$$

The readout maps λ_o and λ_i are defined as follows; $\lambda_i(q, e_d) = \mathbf{e}_{i,PL}$ and $\lambda_o(q, e_d) = \mathbf{e}_{o,PL}$. The guard are defined as follows.

$$\Phi_{q,e} \subseteq \{x \in \text{int } \mathcal{X} \mid h_{q,e}(x) = 0\}, \forall e \in (E_i \cup E_o) \setminus \{\mathbf{e}_{o,PL}, \mathbf{e}_{i,PL}\},$$
$$\Phi_{q,\mathbf{e}_{o,PL}} = \Phi_{q,\mathbf{e}_{i,PL}} = \emptyset \text{ and } \Phi_{q,e_1} \cap \Phi_{q,e_2} = \emptyset, \forall e_1 \neq e_2 \in E_i$$

$$h_{q,\mathbf{e}_{FUc}}(x) = \begin{cases} (x_3 - \mathbf{T}_{fo}\} & \text{if } q^c(\mathbf{c}_{FU}) \\ 1 & \text{otherwise} \end{cases}$$

$$h_{q,\mathbf{e}_{min,PL}}(x) = \begin{cases} (x_4 - \mathbf{T}_{pl,min}) & \text{if } \neg q^d(\mathbf{S}_r) \text{ and } \neg q^d(\mathbf{S}_{PL}) \\ 1 & \text{otherwise} \end{cases}$$

$$h_{q,\mathbf{e}_{max,PL}}(x) = \begin{cases} (x_4 - \mathbf{T}_{pl,max}) & \text{if } \neg q^d(\mathbf{S}_{PL}) \text{ and } q^d(\mathbf{S}_r) \\ 1 & \text{otherwise} \end{cases}$$

$$h_{q,\mathbf{e}_{NPIF}}(x) = \begin{cases} x_1 - \mathbf{Fp} & \text{if } q^d(\mathbf{S}_{PL}) \text{ and} \\ & ((q^c(\mathbf{S}_{FD}) \text{ or } (q^c(\mathbf{S}_{FU}) \text{ and } \neg q^d(\mathbf{S}_{FUc})) \\ 1 & \text{otherwise} \end{cases}$$

The initial state $h_0 = (q_0^c, q_0^d, x_0)$ is of the following form.

$$q_0^c(X) = False, X \in \mathbf{Var}_c \setminus \{\mathbf{S}_{FD}) \text{ and } q_0^c(\mathbf{S}_{FD}) = True$$
$$q^d(Y) = False, \forall Y \in \mathbf{Var}_d \text{ and } x_0 = (0, \mathbf{V}_{max}, 0, 0)$$

Control requirements $K = (E_i \setminus \mathbf{e}_{NPIF})^* \cup (E_i \setminus \mathbf{e}_{NPIF})^\omega$.
Solution It is easy to see that Assumption A.1– A.3 and Assumption 2 are satisfied for H. We can solve Problem 1 for H and K above using the procedure outlined in §4. Notice that H_Δ is computable, and $\mathcal{R}(H)$ is finite. For the latter, define $\mathcal{X}_0 = \{(\mathbf{Cp}, \mathbf{V}_{max}, \mathbf{T}_{fo}, \mathbf{T}_{pl,max})\}$, and define the map $V : \mathcal{X} \to \mathbb{R}$ as $V(x_1, x_2, x_3, x_4) = (\mathbf{Cp} - x_1)$. It follows from Proposition 1 that V and $\mathcal{X}_0$ satisfy Theorem 4. In Petreczky et al. (2008b) controllers were synthesized based on an algorithm and a model related to the one presented above.

7. DISCUSSION AND CONCLUSIONS

We have presented a control problem for a class of hybrid systems and we have proposed a solution based on computing finite-state discrete-event abstraction of hybrid systems. We believe that the results are relevant for practice. In fact, the paper can be viewed as a theoretical foundation of Petreczky et al. (2008b), where control problems arising in error-handling of printers were investigated.

Future research includes extension of the results to other classes of systems and the study of robustness and computational issues such as rigorous decidability and computational complexity.

REFERENCES

Alur, R., Henzinger, T., Lafferriere, G., and Pappas, G.J. (2000). Discrete abstractions of hybrid systems. *Proccedings of the IEEE*, 88(2), 971–984.

Belta, C., Isler, V., and Pappas, G. (2005). Discrete abstractions for robot motion planning and control in polygonal environments. *IEEE Transactions on Robotics*, 21(5), 864– 874.

Berstel, J. (1979). *Transductions and Context-Free Languages*. Teubner, Stuttgart.

Cury, J., Krogh, B., and Niinomi, T. (1998). Synthesis of supervisory controllers for hybrid systems based on approximating automata. *IEEE Trans. Automatic Control*, 43(4).

Eilenberg, S. (1974). *Automata, Languages and Machines*. Academic Press, New York, London.

Fainekos, G.E., Girard, A., and Pappas, G.J. (2007). Hierarchical synthesis of hybrid controllers from temporal logic specifications. In *HSCC*, 203–216.

Förstnera, D., Jung, M., and Lunze, J. (2002). A discrete-event model of asynchronous quantised systems. *Automatica*, 38, 1277 – 1286.

Gonzalez, J., da Cunha, A., Cury, J., and Krogh, B. (2001). Supervision of event-driven hybrid systems: Modeling and synthesis. In *Hybrid Systems: Computation and Control*, volume LNCS 2034, 247 – 260.

Grädel, E., Thomas, W., and Wilke, T. (2002). *Automata, Logic and Infinite Games*, volume LNCS 2500. Springer.

Habets, L., Collins, P., and van Schuppen, J. (2006). Reachability and control synthesis for piecewise-affine hybrid systems on simplices. *IEEE Trans. Automatic Control*, 51, 938–948.

Koutsoukos, X., Antsaklis, P., Stiver, J., and Lemmon, M. (2000). Supervisory control of hybrid systems. *Proceedings of the IEEE*.

Moor, T. and Raisch, J. (1999). Supervisory control of hybrid systems within a behavioural framework. *Systems and Control Letters*, 38, 157 – 166.

Moor, T., Raisch, J., and O'Young, S. (2002). Discrete supervisory control of hybrid systems based on l-complete approximations. *Discrete Event Dynamic Systems*, 12(1), 83–107.

Petreczky, M., Theunissen, R., Su, R., van Beek, D., van Schuppen J.H., and Rooda, J. (2008a). Control of input-output discrete-event systems. Technical Report 2008-12, Eindhoven University of Technology, Systems Engineering.

Petreczky, M., van Beek, D.A., and Rooda, J.E. (2008b). Supervisor for toner error-handling. Technical Report 2008-11, Eindhoven University of Technology, Systems Engineering.

Philips, P., Heemels, W., Preisig, H., and van den Bosch, P. (2003). Control of continuous-time quantised systems. *Int. J. of Control*, 76, 277–294.

Raisch, J. (2000). Discrete abstractions of continuous systems - an input/output point of view. *Mathematical and Computer Modelling of Dynamical Systems*, 6(1), 6–29.

Raisch, J. and O'Young, S. (1995). A des approach to control of hybrid dynamical systems. In *Hybrid Systems*, 563–574.

Tabuada, P. and Pappas, G.J. (2005). Hierarchical trajectory generation for a class of nonlinear systems. *Automatica*, 41(4), 701–708.

van der Schaft, A. and Schumacher, H. (2000). *An Introduction to Hybrid Dynamical Systems*. Springer-Verlag London.