A control problem for hybrid systems with discrete inputs and outputs

Systems Engineering Group
Department of Mechanical Engineering
Eindhoven University of Technology
PO Box 513
5600 MB Eindhoven
The Netherlands
http://se.wtb.tue.nl/

SE Report: Nr. 2008-13

A control problem for hybrid systems with discrete inputs and outputs

M. Petreczky P. Collins D.A. van Beek
J. H. van Schuppen J.E. Rooda

ISSN: 1872-1567

SE Report: Nr. 2008-13
Eindhoven, July 2008
SE Reports are available via http://se.wtb.tue.nl/sereports

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. In addition, we identify classes of hybrid systems for which a suitable finite-state abstraction can be computed and we sketch the algorithms for computing the finite-state abstractions. Unlike most of the existing algorithms, the algorithm of this paper is not based on discretizing the state-space. Instead, a discrete-time counterpart of the hybrid plant is constructed. The state-space of this discrete-time hybrid system consists of those state of the original system which are reachable at sampling times. In order to obtain a finite state abstraction in this way, we restrict attention to those hybrid systems, for which the set of states reachable at sampling times is finite, and the continuous dynamics and the continuous state change only under the influence of the control inputs. In addition, we present Lyapunov-like conditions for checking the former property. We also present an example of practical relevance satisfying the above restrictions.

1 Introduction

Motivated by applications in the area of high-tech systems, in particular control of printers, [22], we are interested in the following control problem. The plant is a hybrid system which is subject to discrete-valued disturbances and control inputs and which generates discrete-valued 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 on 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 rigorous formulation and solution of the control problem described above. The solution consists of the following steps

Step 1 Compute a suitable 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 time discretization of the hybrid system, but it does not involve discretization of the state-space, i.e. dividing the state-space into regions.

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 is not a supervisory control problem. It can be solved using game theory [11] or, under additional assumptions, using classical supervisory control. For more details, see [21]. Furthermore, we identify classes of hybrid systems for which the finite-state abstraction can be computed. In addition, we present a hybrid system based on an industrial use case which belongs to one of the identified classes. *We consider the identification of suitable classes of hybrid systems, for which the procedure above can be applied to and which are relevant for practice, as one of the major contributions of this paper*.

Construction of the finite-state abstraction The finite-state abstraction presented in this paper is in fact a discrete-time counterpart of the hybrid system. This discrete-time system has finitely many states, if the original hybrid system satisfies certain assumptions. If some of those assumptions are dropped, then we obtain a discrete-time hybrid system with possibly infinite state-space. The discrete-time system is obtained from the original continuous-time hybrid system in a manner which is similar to the time sampling of continuous systems. Hence, the construction of this paper represents *a generalization of time sampling for hybrid systems*.

More precisely, the state-space of the discrete-time system consists of precisely those states of the original hybrid system which can be reached at integer multiples of the sampling rate. Moreover, it is assumed that the control inputs are applied only at the sampling times. Then the challenge is to estimate the events and their effect on the system evolution between the sampling times. To this end, we put the following restrictions on the hybrid systems we consider.

- Disturbances or internal events do not influence the continuous dynamics.
- Output events do not influence the system dynamics.
- Only finitely many events are generated on any time interval.

With the assumptions above, we are able to construct a discrete-time counterpart of the original hybrid systems. The obtained discrete-time system is an abstraction of the original one, in the sense that it predicts all the possible sampled outputs and sequences of internal events which the original system generates under the influence of sampled inputs. Note that the discrete-time system may also predict outcomes which the original system never generates. But each sequence of sampled outputs and internal events generated by the original system will be a possible behavior of the discrete-time system. That is why we refer to the discrete-time system as abstraction. Here, by a sampled output we mean the collection of output events generated by the system between two sampling times. By a sampled input we mean an input signal which takes values only at sampling times.

In order to obtain a finite-state system from the discrete-time system described above, we have to assume that the set of states of the original hybrid systems which are reached at sampling times is finite. This looks like a strong assumption which is difficult to check. We present sufficient conditions for this property to hold. The conditions are formulated in terms of existence of Lyapunov-like functions. Intuitively, the existence of Lyapunov-function implies the existence of a physical quantity (potential energy, distance) which periodically decreases as the system evolves. When this quantity becomes zero, the system is set to one of the finitely many possible initial states. Distance from the end of the conveyer belt (paper path) is an example of a Lyapunov function which occurs in models of manufacturing or logistics systems or machines such as printers or copy machines.

In addition, we formulate classes of systems for which the assumptions above can be checked effectively and the finite-state abstraction can be computed. One such a class is the class of hybrid systems where the state-space is polyhedral, the reset maps are affine, the guards are defined by hyperplanes and the continuous dynamics is defined by L'ure-type systems. For this class of systems many of the assumption outlined above can be checked by an algorithm. In addition, we are able to present sufficient conditions for the finiteness of the set of states reached at sampling times. This condition is based on existence of Lyapunov-like functions and can effectively be checked. Finally, we present an example of a hybrid system of the above form which satisfies the assumptions and which is based on an industrial use-case.

Motivation The applications which motivate the presented theory differ from usual control engineering problems in the following sense. We are interested in systematic methods for designing high-level control algorithms and software for complex electro-mechanical systems. The goal is to decrease the cost of development of new generations of such systems, while increasing their reliability. In contrast to classical control, *the challenge is not so much to solve a particular control problem, but to come up with a method for systematic solution of control problems, i.e. we aim at automated "mass production" of controllers solving a class of control problems*. This calls for algorithms (and software tools) for generating controllers for a well-defined and fairly general class of plant models and requirements. The correctness of these algorithms and the ability to automatically check whether the proposed system models fits model class is of great importance. The success of this approach very much depends on our ability to reduce the role of engineering insight in the design of control software. Hence, algorithms for generating controllers which are correct by construction and which solve simple control problems (particular instances of which can be solved by hand, without using any theory), are still desirable. We believe that the class of models and control requirements considered in this paper is general enough to cover a wide range of applications while it still allows automated generation of controllers.

Related work To the best of our knowledge, the contribution of the paper is new. Some of the results described in this paper have already appeared in [20]. Control of hybrid systems using finite-state approximation is a classical topic, [10, 5, 8, 19, 17, 15]. The main difference with respect to [10, 5, 15] is the presence of partial observations, that the generation of events is not synchronized with inputs, and that the hybrid plant contains reset maps. With respect to [8, 19, 17] 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. Contrary to [26, 19, 18, 17, 25], we are not using the behavioral framework at all and we look at systems in continuous time. The results of [26, 19, 18, 17, 25] address a problem which is quite different from the one considered in this paper. In contrast to [28, 15], here we consider a hybrid plant model, as opposed to a continuous one and we allow unobservable events. In addition, for the control problem of the paper, the event generation and controller activation are not synchronized.

The approach of the paper resembles [1, 30, 7, 2]. However, the abstraction notion of this paper and the problem formulation are quite different. Note that in [1, 30] abstraction is used for hierarchical control. In contrast, here abstraction is used for computational purposes only, it has no relationship with hierarchical control.

Unobserved internal events in combination with other constraints render the control problem of this paper different from [23, 31].

In addition, the construction of the finite-state abstraction presented in this paper is different from the existing constructions described in the literature. One class of existing constructions [10, 15, 28, 1, 7, 2, 23] attempts to discretize the state-space by dividing it into regions. The state-space of the thus obtained finite-state machine is the set of regions. The state-transition map prescribes a transition from one region to another one, if there exists a trajectory of the original system which starts in one region and upon leaving the first region immediately enters the other one. In contrast, the approach of this paper does not divide the state-space into regions. In fact, the finite-state abstraction of this paper lives on a subset of the original state-space of the hybrid system. Another approach, described in [8, 5, 19, 17, 26, 19, 18, 17], appromixates the underlying system by storing the output (or state) response of the system to input sequences of finite length. In contrast, the abstraction presented in this paper lives on the same state-space as the original system. Moreover, in contrast to the two approaches above, the construction of this paper involves transition from continuous- to discrete-time. Note that the finite-state abstraction of this paper is not directly related to the finite bisimulation of [1].

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. §4.2 sketches the solution of the discrete-event control problem. 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 we illustrate the presented results by means of an example of practical relevance. We end the paper by conclusions in §7.

2 Preliminaries

The goal of this section is to present an overview of the necessary background on automata theory. In Subsection 2.1 we review the elementary notion and terminology from formal language theory. In Subsection 2.2 we recall the definition of Moore-automata and related concepts. In Subsection 2.3 we review the classical concept of monoid, automata on monoids and rational subsets of monoids. In Subsection 2.4 we will use these notions to define the concept of sequential input-output maps, quasi-sequential deterministic transducer and quasi-recognizability. The material of Subsection 2.4 can be found in [21].

2.1 General notation

Most of the time, we will use the standard notation and terminology from automata theory [6, 9]. Let Σ be a finite set, referred to as the *alphabet*. Σ^* denotes the set of finite *strings (words)* of elements of Σ, i.e. an element of Σ^* is a sequence $w = a_1a_2 \cdots a_k$, where $a_1, a_2, \ldots, a_k \in \Sigma$, and $k \geq 0$; k is *the length of* w and it is denoted by $|w|$. If $k = 0$, then w is the empty word, denoted by ϵ. The concatenation of two words v and w is denoted by vw. An *infinite (ω-) 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.

A *language* over Σ is a set of finite strings (words) over Σ. For any (in)finite word w, and for any $i \in \mathbb{N}$ (in case w is finite word, for any $i \in \mathbb{N}$ such that $i \leq |w|$), $w_{1:i}$ 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.

For any word $w \in \Sigma^* \cup \Sigma^\omega$, a finite word $p \in \Sigma^*$ is a *prefix* of w, if there exists an index $i \in \mathbb{N}$, such that $w_{1:i} = p$. If $K \subseteq \Sigma^*$, then $\lim(K) \subseteq \Sigma^\omega$ is the set of all infinite words, infinitely many prefixes of which belong to K, i.e.

$$\lim(K) = \{w \in \Sigma^\omega \mid \exists \{k_i \in \mathbb{N}\}_{i \in \mathbb{N}} : \text{ such that } \forall i \in \mathbb{N} : (k_{i+1} > k_i), \text{ and } \forall i \in \mathbb{N} : w_{1:k_i} \in K\}$$

If $L \subseteq \Sigma^* \cup \Sigma^\omega$, then the *prefix closure* of L is denoted by $\bar{L}$ and is defined by $\bar{L} = \{p \in \Sigma^* \mid \exists v \in L : p \text{ is a prefix of } v\}$; L is called *prefix closed*, if $\bar{L} = L$.

The set of non-negative reals is $\mathbb{R}_+$.

2.2 Moore-automata

Below we will review the notion of Moore-automata. Note that Moore-automata will play the role of controllers in our setting. Recall from [6, 9] that a *Moore-automaton* 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 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 $w = u_1u_2 \cdots u_k \in I^*$, $k \geq 0$ 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 $i = 1, 2, \ldots, k$. The map ϕ is *realizable* by a Moore-automaton, if there exists a Moore-automaton which is a realization of ϕ.

2.3 Monoid, automata, rational sets

The goal of this section is to recall the notions of monoid, rational and recognizable subsets of a monoid, and automata on monoids. These concepts will then be used to define the concept of sequential input-output maps and their automaton representations. The latter concepts are used to model the behavior of the discrete-event abstraction of the hybrid plant.

Recall from [3, 6] that a *monoid* M is a (not necessarily finite) semi-group with a unit element which is denoted by 1_M, or simply 1, if M is clear from the context. That is, there exists a multiplication operation, denoted by $\cdot$. The set of all finite strings Σ^* over the finite alphabet Σ forms a monoid, if we take the concatenation as multiplication and the empty word ϵ as the unit element. The monoid Σ^* is also referred to as the *free monoid*. Another example of a monoid is the cartesian product $X^* \times Y^*$, where X and Y are finite alphabets. Here, identity element is (ϵ, ϵ), and the multiplication operation defined by $(s_1, s_2)(v_1, v_2) = (s_1v_1, s_2v_2)$.

Below we will recall from [3, 6] the notion of a finite-state automaton on monoids.

Definition 1 (Automaton on monoid [3, 6]). 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 relation called the state-transition relation. We assume that E is a finite set.
- $F \subseteq Q$ is the finite set of accepting states
- $q_0 \in Q$ is the initial state

Definition 2 (Accepting run, [3, 6]). An element $m \in M$ is *accepted* by T if there exists elements $m_i \in M_i$ and states $q_i \in Q$, $i = 1, 2, \dots, k$ for some $k \geq 0$ such that $(q_i, m_{i+1}, q_{i+1}) \in E$ for $i = 0, 1, \dots, k-1$, $q_k \in F$ and $m = m_1 m_2 \cdots m_k$.

The definition of a subset of M accepted by the DFA T is completely analogous to the definition of the language accepted by an automaton.

Definition 3 (Sets recognized by DFA , [3, 6]). The set $L \subseteq M$ is *recognized by* T, and it is denoted by $L(T)$, if L consists of precisely those elements $m \in M$ which are accepted by T.

Definition 4 (Rationality). A subset $L \subseteq M$ is called rational, if there exists a finite-state automaton T on M such that L is recognized by T.

In other words, rational subsets of M are precisely those subsets which can be described by (possibly non-deterministic) finite state automata. Rational subsets of monoids have been studied since the 1960's [3, 6, 16] and the references therein.

2.4 Sequential input-output maps

The goal of this section is to define the notion of sequential input-output maps. Sequential input-output maps will be used to model the input-output behavior of non-deterministic discrete-event plants, which arise as abstractions of hybrid systems.

Definition 5 (Sequential input-output maps, [21]). Let X, Y, Σ be finite sets. A multi-valued map $R : \Sigma^* \to 2^{Y^* \times X^*}$ is called a *sequential input-output map*, if the following conditions are satisfied

1. $R(\epsilon) = \{(\epsilon, \epsilon)\}$, and for all $s \in \Sigma^*$, $R(s)$ is a non-empty set.
2. For all $s \in \Sigma^*$, if $(\underline{y}, \underline{x}) \in R(s)$, with $\underline{y} \in Y^*$ and $\underline{x} \in X^*$, the length of s and $\underline{y}$ are the same, i.e. $|s| = |\underline{y}|$.
3. R is *prefix preserving*, i.e. for each word $s \in \Sigma^*$, for each letter $a \in \Sigma$, and for each pair of words $(\underline{y}, \underline{x}) \in R(sa)$, there exist a letter $y \in Y$ and words $x \in X^*$, $\underline{\hat{y}} \in Y^*$, $\underline{\hat{x}} \in X^*$ such that $\underline{y} = \underline{\hat{y}}y$, $\underline{x} = \underline{\hat{x}}x$ and $(\underline{\hat{y}}, \underline{\hat{x}}) \in R(s)$.
4. R is *non-blocking*, i.e. for each word $s \in \Sigma^*$, for each letter $a \in \Sigma$, and for each pair of words $\underline{x} \in X^*$, $\underline{y} \in Y^*$ such that $(\underline{y}, \underline{x}) \in R(s)$, there exists a letter $y \in Y$ and a word $x \in X^*$, such that $(\underline{y}y, \underline{x}x) \in R(sa)$.

Intuitively, the set Σ corresponds to input symbols, the sets X and Y correspond to output symbols. Moreover, the map R synchronizes between Σ and Y, i.e. the length of the Y^*-valued component of R coincides with the length of the argument. However, this is not true for the X^*-valued component of R. In this paper we will mainly be interested in sequential input-output maps which are *quasi-recognizable*, i.e. sequential input-output maps whose graph is a rational subset of the monoid $M = \Sigma^* \times Y^* \times X^*$ and which can be recognized by a finite-state quasi-sequential transducer.

Definition 6 (Quasi-sequential transducer, [21]). A DFA $T = (Q, M, E, F, q_0)$ defined over the monoid $M = \Sigma^* \times Y^* \times X^*$ is called a *quasi-sequential transducer*, if

1. $F = Q$, i.e. all states are accepting,
2. the state-transition relation is a partial map $E : Q \times \Sigma \times Y \times X^* \to Q$. That is, the state-transitions are deterministic and are labeled by *letters* from Σ and Y and by *sequences* from X^*.
3. For each state $q \in Q$ and letter $a \in \Sigma$ there exist a letter $y \in Y$ and a word $\underline{x} \in X^*$ such that $E(q, u, y, \underline{x})$ is defined.

Definition 7 (Quasi-recognizable sequential input-output maps, [21]). The sequential input-output map $R : \Sigma^* \to 2^{Y^* \times X^*}$ is called *quasi-recognizable*, if the corresponding graph $\mathrm{graph}\ R$ of R, defined as

$$\mathrm{graph}\ R = \{(\underline{u}, \underline{y}, \underline{x}) \in \Sigma^* \times Y^* \times X^* \mid (\underline{y}, \underline{x}) \in R(\underline{u})\} \tag{1}$$

has the following property. If $\mathrm{graph}\ R$ is viewed as subset of the monoid $M = \Sigma^* \times Y^* \times X^*$, then $\mathrm{graph}\ R$ is recognized by a quasi-sequential deterministic transducer.

3 Control problem

Below we define the control problem we are interested in.

Plant 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 for control purposes (i.e. detectable by sensors), while internal events are not visible.

The environment and the plant generate events *asynchronously*. More precisely, the plant generates at most one output at each time instance, and at most one internal event at each time instance. However, it may happen that an output and an internal event are generated at the same time. Similarly, at most one disturbance is generated at any time, and at most one control input is generated at any time. However, it may happen that a control input and a disturbance occur simultaneously. In addition, a control or disturbance can reach the plant at the same time as the plant generates an output or internal event. Note that the plant is assumed to live in real time.

Notation 1 (Plant and events). We denote the plant by H. We denote the events of interest as follows.

- E_c is the finite set of *control inputs*,
- E_d is the finite set of *disturbances*,
- E_o is the finite set of *outputs*,
- E_i is the finite set of *internal events*.

The external behavior of the plant is formalized as an input-output map, which maps time signals of control and disturbance events to time signals of outputs and internal events. In order to formalize the input-output maps of the plant of interest, we need the notion of a time-event function. The latter is just a function obtained by interpreting a time-event sequence as a function of time.

Definition 8 (Time-event functions). Let E be a finite set and let $\perp \notin E$. Consider a finite or infinite timed sequence of elements of E.

$$s = (e_1, t_1)(e_2, t_2) \cdots (e_k, t_k) \cdots \tag{2}$$

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

$$g : \mathbb{R}_+ \ni t \mapsto E \cup \perp \ni \begin{cases} e_{i+1} \in E & \text{if } t = t_{i+1} \text{ for some } i \in \mathbb{N} \\ \quad \perp & \text{otherwise} \end{cases} \tag{3}$$

A map as in (3) induced by a sequence (2) is called a *time-event function*. The set of all time-event functions is denoted by $\mathcal{P}_E$.

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.

Notation 2. Let $g \in \mathcal{P}_E$ be a time-event function as in (3). Define *the sequence of elements of* E *induced by* g as $\mathbf{UT}(g) = e_1 e_2 \cdots e_k \cdots \in E^* \cup E^\omega$. That is, two cases are possible.

1. There exist time instances $0 \leq t_1 < t_2 < \ldots < t_k$ such that for all $s \in \mathbb{R}_+$, $g(s) \in E$ if and only if $s \in \{t_1, t_2, \ldots, t_k\}$. Then $\mathbf{UT}(g) = g(t_1)g(t_2) \cdots g(t_k) \in E^*$ and hence $\mathbf{UT}(g)$ is finite.

2. There exists an infinite sequence of time instances $0 \leq t_1 < t_2 < \ldots < t_k < \ldots$ such that for all $s \in \mathbb{R}_+$, $g(s) \in E$ if and only if $s = t_i$ for some $i = 1, 2, \ldots$. Then $\mathbf{UT}(g)$ is an infinite word, ith element of which equals $g(t_i)$.

By applying the definition of time-event functions for $E \in \{E_c, E_d, E_o, E_i\}$, we obtain spaces of functions $\mathcal{P}_{E_c}$, $\mathcal{P}_{E_d}$, $\mathcal{P}_{E_o}$, $\mathcal{P}_{E_i}$ describing the signals with values in control inputs, disturbances, outputs and internal events respectively.

The behavior of the plant H is formalized as a causal input-output map which maps time-event functions of control inputs and disturbances to time-event functions of outputs and internal events.

Definition 9 (Input-output map of the plant). The input-output map of the plant 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 for any two inputs $u_i \in \mathcal{P}_{E_c}$, and disturbance $d_i \in \mathcal{P}_{E_d}$, and for any two outputs $o_i \in \mathcal{P}_{E_o}$, and internal event signals $\hat{o}_i \in \mathcal{P}_{E_i}$ such that $(o_i, \hat{o}_i) = \upsilon_H(u_i, d_i)$, $i = 1, 2$,

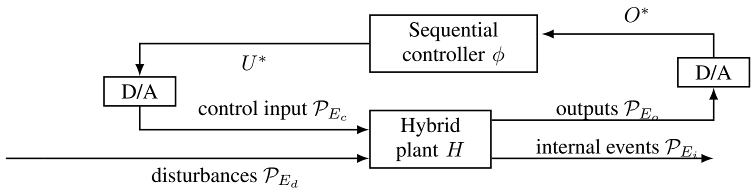
$$[\forall s \in [0, t] : d_1(s) = d_1(s) \text{ and } \forall s \in [0, t) : u_1(s) = u_2(s)] \implies o_1(t) = o_2(t), \hat{o}_1(t) = \hat{o}_2(t)$$

That is, causality means that the outputs and internal events depend only on the past inputs and on the past and present disturbances. In addition, we require that if $(o, \hat{o}) = \upsilon_H(u, g)$ for some $u \in \mathcal{P}_{E_c}$, $g \in \mathcal{P}_{E_d}$, then $o(0) = \perp \notin E_o$ and $\hat{o}(0) = \perp \notin E_i$, i.e. no output or internal event is generated at time instance 0.

Controller The controllers of interest are modeled as maps from outputs to control inputs.

Definition 10 (Controller). A *hybrid controller* is a map $\mathcal{C} : \mathcal{P}_{E_o} \to \mathcal{P}_{E_c}$.

Remark 1 (External inputs). In many application one encounters external inputs, i.e. inputs which are visible to the controller and which change the dynamics of the system, but which are generated by the environment or user. That is, external inputs cannot be influenced by the controller. External inputs can be incorporated in our framework as follows. We extend the set of disturbances and outputs by copies of external input events We model each occurrence an external input event v as the simultaneous occurrence of the disturbance event which is a copy of v and the output event which is a copy of v.



O^*
Sequential controller ϕ
U^*
D/A
D/A
control input $\mathcal{P}_{E_c}$
Hybrid plant H
outputs $\mathcal{P}_{E_o}$
internal events $\mathcal{P}_{E_i}$
disturbances $\mathcal{P}_{E_d}$

Figure 1: Control architecture

In order to define the behavior of the feedback interconnection of the plant H and controller $\mathcal{C}$, we need to define when this interconnection is mathematically well-posed.

Definition 11 (Well-posedness). The interconnection of H and $\mathcal{C}$ is well-posed if for any disturbance signal $d \in \mathcal{P}_{E_d}$, there exists a unique input signal $u \in \mathcal{P}_{E_c}$, output signal $o \in \mathcal{P}_{E_o}$, and internal event signal $\hat{o} \in \mathcal{P}_{E_i}$ such that

$$(o, \hat{o}) = \upsilon_H(u, d) \text{ and } u = \mathcal{C}(o) \tag{4}$$

Notice that the interconnection of H and $\mathcal{C}$ need not always be well-posed.

We restrict attention to controllers which have a finite-state representation and are activated on integer multiples of a fixed sampling rate $\Delta > 0$.

Notation 3. In the rest of the paper $\Delta > 0$ denotes the sampling rate.

We assume that the controller has no knowledge of the relative order or the timing of the events between sampling times. More precisely, the controller is the interconnection of a Moore-automaton with interfaces, converting time signals to discrete symbols and back. These interfaces map functions from $\mathcal{P}_{E_o}$ to sequences of *subsets* of E_o, where the ith element of the sequence is the set of outputs which took place on the time-interval $((i-1)\Delta, i\Delta]$. At each sampling time the controller generates a symbol from E_c or the symbol $\perp$. The latter encodes the case when no control input is applied. The symbols generated by the controller are converted to a time-event function $\mathcal{P}_{E_c}$ whose value at $i\Delta$ is the output of the controller at the $(i+1)$th step, and $\perp$ otherwise.

Definition 12 (Discrete input and output alphabet). Define the *set of discrete inputs* as $U = E_c \cup \{\perp\}$, and the *set of discrete outputs* as $O = 2^{E_o}$.

Remark 2 (Choice of the sampled alphabet). The choice of O made in this paper is not the only possible one. In fact, one could define a different sampling mechanism, not just simply collecting the set of output events which took place in the sampling interval. For example, often the relative order of events is known.

Definition 13 (Sequential controllers). A *sequential controller* is a map $\phi : O^* \to U$ such that ϕ is the input-output map of a Moore-automaton.

The Moore-automaton part of the desired controller will be a sequential controller. The desired hybrid controller is then defined as follows.

Definition 14 (Hybrid controller from a sequential one). For a sequential controller ϕ let the *hybrid controller* $\mathcal{C}_\phi$ *associated with* ϕ be such that for all $o \in \mathcal{P}_{E_o}$,

$$\forall t \in \mathbb{R}_+ : \mathcal{C}_\phi(o)(t) = \begin{cases} \phi(S_1 S_2 \cdots S_k) & \text{if } t = k\Delta \text{ for } k \in \mathbb{N}, k > 0 \\ \phi(\epsilon) & \text{if } t = 0 \\ \perp & \text{otherwise} \end{cases}$$

where $S_i = o(((i-1)\Delta, i\Delta]) \cap E_o$ for all $i = 1, 2, \ldots, k$.

Proposition 1. The interconnection of $\mathcal{C}_\phi$ and H is well-posed.

9 Control problem

The proof of Proposition 1 can be found in §8.

The significance of hybrid controllers associated with a sequential one is that it is precisely the type of controllers which can be implemented on computer, based on sampling.

In order to formulate the control problem we are interested in, we have to formally define the relevant aspects of the closed-loop behavior of the system. Since we are interested in the symbolic behavior of the plant, i.e. in the relative order of internal events generated by the plant, we define formally only the *closed-loop language, i.e. the set of sequences of internal events generated by the plant when interconnected with the controller*. However, in order to be able to solve the arising control problem, it is sensible to restrict the class of disturbances, by requiring that only at most a fixed number of disturbance events occurs within a sampling interval. This assumption renders the problem of controlling the plant behavior much simpler. In particular, in case of sampled-data control, the assumption allows the controller to consider only finitely many different scenarios of occurrence of disturbances within the sampling interval.

In order to keep the notation to minimum, we will define the closed-loop language only for this restricted class of disturbance signals.

Definition 15 (Bounded number of events on the sampling interval). Denote by $\Delta > 0$ the sampling rate. Let $\mu \in \mathbb{N}$ be a positive integer. The set of time-event functions g such that on any interval of the form $((i-1)\Delta, i\Delta]$, $i = 1, 2, \ldots$ the number of events of g is not greater than μ is denoted by $\mathcal{P}_{E,\mu}^{\Delta}$. That is, a time-event function $g \in \mathcal{P}_E$ belongs to $\mathcal{P}_{E,\mu}^{\Delta}$ if and only if for each $i = 1, 2, \ldots,$

$$card\{e = g(s) \in E \mid s \in ((i-1)\Delta, i\Delta]\} < \mu$$

Notation 4 (Maximal number of disturbances). In the sequel, $\mu > 0$ will denote the fixed upper bound on the number of disturbance events in a sampling interval $(0, \Delta]$. In particular, we will be interested in disturbances from $\mathcal{P}_{E_d,\mu}^{\Delta}$.

For many practical situations, this assumption is reasonable. We define the symbolic behavior the feedback interconnection of $\mathcal{C}$ and H as follows.

Definition 16 (Closed-loop). 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 time-event functions $\hat{o} \in \mathcal{P}_{E_i}$ for which there exist an input $u \in \mathcal{P}_{E_c}$ a disturbance $d \in \mathcal{P}_{E_d,\mu}^{\Delta}$ and an output $o \in \mathcal{P}_{E_o}$ such that $u, d, o, \hat{o}$ satisfy (4).

That is, $L(H/\mathcal{C})$ is just the collection of sequences of internal events generated by the feedback interconnection of the plant H and controller $\mathcal{C}$.

The control problem we are interested in can be stated as follows.

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

Notice that the closed-loop and the specification languages contain only sequences of internal events. This is done in order to simplify notation. Our results can easily be extended to include sequences of events from $E_c \cup E_d \cup E_o$ in the closed-loop and specification languages.

4 Solution of hybrid control problem

The goal of this section is to present the solution of Problem 1. The main idea is to reduce Problem 1 to a discrete-event control problem. To this end, notice that the desired controller is a sequential controller, which can only see the *symbolic sampled-data behavior* of the plant.

4.1 General idea: convert the hybrid control problem to a discrete one

We model the symbolic sampled-data behavior of the plant as a non-deterministic system R_H, which reacts to sequences of discrete inputs and disturbances and generates sequences of outputs and internal events. The inputs of the system R_H are sequences from U^*, the outputs are sequences from O^*, where the U and O are as in Notation 12. The alphabet of internal events of R_H coincides with the alphabet of internal events E_i of R. Finally, the set of disturbances of R_H is obtained by sampling the disturbance signals of R.

Definition 17 (Discrete disturbances). The set *discrete disturbances* is defined as $D = \bigcup_{k=0}^{\mu} E_d^k$. Here μ is as in Notation 4.

That is, the set of discrete disturbances D is the set of all words over E_d of length at most μ. Recall that μ is the maximal number of disturbance events which is allowed to occur in a sampling interval. An element of D is a sequence, which describes the relative order of disturbance events between two consecutive sampling times. That is $e_1 e_2 \cdots e_k \in D$ says that between the previous and the current sampling times disturbance events $e_1, e_2, \ldots, e_k$ took place, in this order. The empty sequence encodes the scenario when no disturbance event occurs between two sampling time instances.

Remark 3 (Inter-arrival time is greater than the sampling time). Notice that if $\mu = 1$, i.e. the inter-arrival time is greater than the sampling time, then $D = \{\epsilon\} \cup E_d$, i.e. D consists of the set of disturbance events and the empty sequence.

Formally, the behavior of R_H is modeled as a a multi-valued map from sequences in D^* and U^* to O^* and E_i^*, see Fig. 1. Note that due to the sampling mechanism, the relevant sequences from U^*, D^* and O^* have the same length. Formally, *by identifying the system R_H with its external behavior*, we get that R_H is a map $R_H : (U \times D)^* \to 2^{O^* \times E_i^*}$. Notice that here we have used the fact that a pair of sequences from $U^* \times D^*$ of the same length can be identified with a single sequence from $(U \times D)^*$. For the formal definition of R_H, we need the following notation.

Notation 5. Let $g \in \mathcal{P}_E$ be a time-event function as in (3). For all $t \in \mathbb{R}^+$, let $\mathbf{UT}(g, t) \in E^*$, be the sequence of events prescribed by g up to time 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 $t \in [\sum_{r=1}^{l} t_r, +\infty)$.

Alternatively, $\mathbf{UT}(g, t) = \mathbf{UT}(g^t)$, where $g^t(s) = \begin{cases} g(s) & \text{if } s \leq t \\ \perp & \text{if } s > t \end{cases}$, i.e. $\mathbf{UT}(g, t)$ is the finite sequence of events prescribed by the time-event function g^t, where the restriction of g^t to $[0, t]$ equals g, and after time t, g^t prescribes no event.

Definition 18 (Sequential input-output map of H). The *sequential input-output map R_H of H* is the map $R_H : (U \times D)^* \to 2^{O^* \times E_i^*}$ defined as follows. $R_H(\epsilon) = \{(\epsilon, \epsilon)\}$ and for each sequence of discrete input symbols $u_1, u_2, \ldots, u_k \in U$, disturbance symbols $d_1, d_2, \cdots d_k \in D$, $k \geq 0$,

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

for letters $o_1, o_2, \ldots, o_k \in O$ and words $\underline{\hat{o}} \in E_i^*$ if there exist time-event functions $g \in \mathcal{P}_{E_d,\mu}^{\Delta}$,

$o \in \mathcal{P}_{E_o}$ and $\hat{o} \in \mathcal{P}_{E_i}$ such that $(o, \hat{o}) = \upsilon_H(u, g)$ where

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

$$\underline{\hat{o}} = \mathbf{UT}(\hat{o}, k\Delta)$$

$$\forall i = 1, 2, \ldots, k : o_i = o(((i-1)\Delta, i\Delta])$$

$$\forall i = 1, 2, \ldots, k : d_i = \mathbf{UT}(g_i, \Delta) \text{ where } \forall t \in \mathbb{R}_+ : g_i(t) = \begin{cases} g(t + (i-1)\Delta) & \text{if } t > 0 \\ \perp & \text{otherwise} \end{cases}$$

Notice that $\mathbf{UT}(g, k\Delta) = d_1 d_2 \ldots d_k$.

Proposition 2. The map R_H is a sequential input-output map in the sense of Definition 19.

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

More precisely, the behavior described by R_H can be derived from the behavior of the hybrid plant as follows. Consider the sequence $s = (u_1, d_1)(u_2, d_2) \cdots (d_k, u_k) \in (U \times D)^*$. The response $R_H(s)$ is obtained as follows. We construct a time-event function $u \in \mathcal{P}_{E_c}$ which takes value u_i at time instance $(i-1)\Delta$ and $\perp$ otherwise. The input signal u corresponds to a control input generated by a sampled-data controller. We construct every possible disturbance signal $g \in \mathcal{P}_{E_d,\mu}$, such that on the interval $((i-1)\Delta, i\Delta]$ the sequence of events prescribed by g equals d_i, i.e. there exists $t_1, t_2, \ldots, t_l \in ((i-1)\Delta, i\Delta]$ such that $g(t_1)g(t_2)\cdots g(t_l) = d_i$ and $g(s) = \perp$ if $s \notin \{t_1, t_2, \ldots, t_l\}$. We feed the control input u and each such disturbance signal g into the hybrid plant H and as a result we obtain output signal $o \in \mathcal{P}_{E_o}$ and internal event signal $\hat{o} \in \mathcal{P}_{E_o}$. We then convert o into a sequence $o_1 o_2 \cdots o_k \in O^*$ by defining o_i as the set of output events which are values of o on the interval $((i-1)\Delta, i\Delta]$. Similarly, we convert $\hat{o}$ into the sequence of events $\underline{\hat{o}}$ prescribed by the time-event function $\hat{o}$. We then assign $(o_1 o_2 \cdots o_k, \underline{\hat{o}})$ as a possible response of R_H. Notice, that due to the fact that several disturbance signals g can be consistent with the sequence $d_1, d_2 \cdots d_k$, there are several possible responses $(o_1 o_2 \cdots o_k, \underline{\hat{o}})$ of R_H, i.e. R_H describes a non-deterministic discrete plant.

It turns out the in order to solve Problem 1, we can view R_H as the input-output map of a purely discrete-event plant, and solve a discrete-event control problem for R_H as a plant and K as a requirement. The solution of the latter control problem is a sequential controller, such that the corresponding hybrid controller solves Problem 1. In the subsequent subsections we present the formal definition of the discrete-event control problem and the reduction of Problem 1 to the discrete-event control problem.

4.2 Discrete control problem

The discussion above prompts us to formulate the following discrete counterpart of Problem 1. The controllers of interest are sequential controllers. The discrete-event plants of interest admit the following signals; *control inputs* from U, *disturbances* from D, *observable outputs* O, and *internal events* from E_i. We use *sequential input-output maps* to formalize the behavior of the plant.

Definition 19 (Discrete plant). A *discrete plant* is a sequential input-output map $R : (U \times D)^* \to 2^{O^* \times E_i^*}$.

The language of the closed-loop system is defined as follows. Recall that $w_{1:i}$ denotes the prefix of a (possibly infinite) word w, formed by the first i letters, and that $|w| = \infty$ if w is an infinite word.

Definition 20 (Closed-loop language). The *closed-loop language* $L(R/\phi)) \subseteq E_i^* \cup E_i^\omega$ of the interconnection of R with the sequential controller ϕ is the set of all words $\underline{\hat{o}} \in E_i^* \cup E_i^\omega$ for which there exist letters $d_i \in D$, $o_i \in O$, $u_i \in U$, $i \in \mathbb{N}$ and indices $0 = k_0 \leq k_1 \leq \cdots k_i \leq$ satisfying $\sup_{i \in \mathbb{N}} k_i = |\underline{\hat{o}}|$ such that

$$\begin{aligned} \forall i \in \mathbb{N}, i > 0 : \quad & (o_1 o_2 \cdots o_i, \underline{\hat{o}}_{1:k_i}) \in R((u_1, d_1)(u_2, d_2) \cdots (u_i, d_i)) \\ & u_i = \phi(o_1 o_2 \cdots o_{i-1}) \end{aligned}$$

The discrete counterpart of Problem 1 is the following.

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

For more details on the discrete-event control problem above, see [21]. In order to solve Problem 2 we will assume that the sequential input-output map and the specification language both have a finite-state representation. More specifically, we need to assume that R is quasi-recognizable, i.e. it is recognized by a quasi-sequential transducer. As to the specification language K, we require that its component made up of words of finite length is a regular language, and its component consisting of words of infinite length can be recognized by a Büchi automaton. If R is quasi-recognizable and K satisfies the above assumption, then in many cases Problem 2 can be reduced to finding a winning strategy of a Rabin- or parity-game [14, 11]. We defer the details to another paper. If the assumptions below hold, then a solution of Problem 2 can be obtained by using Ramadge-Wonham (**RW** for short) supervisory theory with partial observations, see [21] for more details.

Assumption 1 (Assumptions for applying RW [21]).

- R is a quasi-recognizable sequential input-output map,
- $K = K_{safe} \cup \lim(K_{safe})$ where $K_{safe} \subseteq E_i^*$ is regular and prefix-closed.

Theorem 1 ([21]). If Assumption 1 holds, then a controller solving Problem 2 can be computed using classical Ramadge-Wonham supervisory control synthesis with partial observations.

4.3 From Problem 1 to Problem 2

It turns out that any sequential controller solving Problem 2 for the sequential input-output map R_H also solves Problem 1.

Theorem 2 (Hybrid vs. discrete control). If ϕ is a sequential controller, then the closed-loop language of the interconnection of R_H with ϕ contains the closed-loop language of the interconnection of the associated hybrid controller $\mathcal{C}_\phi$ with H, i.e. $L(H/\mathcal{C}_\phi) \subseteq L(R_H/\phi)$. Hence, if ϕ is a solution of Problem 2 for R_H and $K \subseteq E_i^* \cup E_i^\omega$, then the associated hybrid controller $\mathcal{C}_\phi$ is a solution of Problem 1.

The proof of Theorem 2 can be found in §8. The only remaining problem is that R_H need not admit a finite-state representation suitable for solving Problem 2. according to §4.2. The remedy is to solve Problem 2 not for R_H but for an quasi-recognizable abstraction of R_H. The computation of a quasi-recognizable abstraction, more precisely, a finite-state quasi-sequential transducer recognizing it is discussed in §5. In fact, if K also satisfies Assumption 1 of §4.2, then Ramadge-Wonham theory can be applied to solve Problem 2, and hence Problem 1.

Informally, an abstraction of R_H is a sequential input-output map which has the property that its response to any sequence of discrete inputs and disturbances includes the response of R_H to that particular sequence. The formal definition is as follows.

Definition 21 (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 3 (Control of abstraction). 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.

The proof of Theorem 3 is presented in §8. A finite-state abstraction of R_H can be computed as described in §5. Theorem 2 and Theorem 3 yield the following procedure for solving Problem 1.

1. Use §5 to compute a finite-state abstraction R of R_H
2. Use the results of §4.2 to compute a solution to Problem 2 for R and the original control requirements specified by K.
3. Compute the hybrid controller $\mathcal{C}_\phi$ associated with ϕ.

5 Finite-state abstraction of R_H

In this section we define a quasi-sequential transducer recognizing an abstraction of R_H. To this end, we will have to restrict the class of hybrid systems under consideration. In §5.1 we define the class of hybrid systems for which a quasi-recognizable abstraction can be computed. In §5.2 we present the precise definition the above-mentioned quasi-sequential transducer, and list some system classes for which it can be computed effectively.

5.1 Hybrid systems

The definition of hybrid systems of interest is as follows.

Definition 22. A discrete i/o hybrid system H is a tuple

$$(S_H, \delta, \lambda_i, \lambda_o, E, \{f_q, R_{u,q}, \Phi_{q,e} \mid q \in Q, u \in E_c, e \in E_i \cup E_o\}, h_0) \tag{5}$$

- $E = E_c \cup E_d \cup E_o \cup E_i$ is a set of events
- E_d is the finite set of *disturbances*,
- E_c is the finite set of *control inputs*,
- E_o is the finite set of *outputs*,
- E_i is the finite set of *internal events*
- $Q = Q_c \times Q_d$ is the *discrete state-space* of H, Q_c, Q_d are finite sets.
- $\delta_c : Q \times E_c \to Q_c$ is the discrete-state transition function which determines the state-transition rules for control inputs,
- $\delta_d : Q \times (E_d \cup E_i) \to Q_d$ is the discrete-state transition function determines the state-transition rules for disturbances and internal events.
- $\mathcal{X} \subseteq \mathbb{R}^n$ is the *continuous state space*, $\mathcal{X}$ is a closed set with non-empty interior and boundary, i.e. $\operatorname{int} \mathcal{X} \neq \emptyset$, $\partial\mathcal{X} = \mathcal{X} \setminus \operatorname{int} \mathcal{X} \neq \emptyset$.
- $S_H = Q \times \mathcal{X}$ is the state-space of H.
- $f_q = f_{q^c} : \mathbb{R}^n \to \mathbb{R}^n$, $q = (q^c, q^d) \in Q$, is a continuous and globally Lipschitz map; note that f_q depends only on the Q_c-valued component q^c of q,

- $R_{u,q} : \mathcal{X} \to \mathcal{X}$ with $q \in Q$ and $u \in E_c$ is the reset map,
- $\Phi_{q,e} \subseteq \text{int } \mathcal{X}$, $q \in Q$ is a guard generating the event $e \in E_o \cup E_i$.
- $\lambda_o : Q \times E_d \to E_o$ is a partial map, responsible for generating outputs when a disturbance event occurs.
- $\lambda_i : Q \times E_d \to E_i$ is a partial map, responsible for generating internal events when a disturbance event occurs.
- $h_0 = (q_0^c, q_0^d, x_0) \in S_H$ is the initial state of the system.

The system H is simply a hybrid system [32], evolution of which follows the classical definition, but whose parameters are subject to the following restrictions. The set $E = E_c \cup E_d \cup E_o \cup E_i$ can be regarded as the set of discrete events. The disturbances from E_d are imposed by the environment. The control inputs from E_c can be used by the controller to influence the system behavior. Only disturbances from E_d and control inputs from E_c can change the continuous state of the system. An event $e \in E_o \cup E_i$ is generated by H either if the continuous state crosses the guard set, or when an event from E_d arrives. The generation of an event from E_o does not change the state of H. Generation of an event from E_i changes only the Q_d-valued component of the discrete state-space.

The discrete states of H are elements of $Q = Q_c \times Q_d$, i.e. each discrete state is a pair $q = (q^c, q^d)$ where $q_i \in Q_i$, $i = 1, 2$. 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, and the current discrete state is $q = (q^c, q^d) \in Q$, then the Q_c-valued component of the new discrete state becomes $\delta_c(q, u)$. If a disturbance event $d \in E_d$ arrives, then the Q_d-valued component of the new discrete state is $\delta_d(q, d)$. If an event $e \in E_i$ occurs, then the Q_d-valued component of the new discrete state is $\delta_d(q, e)$. For an event from E_o the discrete state does not change. The continuous dynamics in the discrete state $q = (q^c, q^d)$ is determined by the differential equation $\dot{x} = f_{q^c}(x)$. 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. Note that while the differential equations associated with a discrete state (q^c, q^d) depend only on q^c, the readout maps λ_i, λ_o, the reset maps, the discrete state-transition maps δ_c and δ_d and and the guard sets $\Phi_{q,e}$, $e \in E_o \cup E_i$ depend on both q^c and q^d.

In order to define the dynamics of H formally, we will need the following result.

Proposition 3. For any $q^c \in Q_c$, and for any initial state $z_0 \in \text{int } \mathcal{X}$, the initial value problem

$$\dot{z} = f_{q^c}(z) \text{ and } z(0) = z_0 \tag{6}$$

has a unique differentiable solution in $\mathbb{R}^n$ on the whole time axis $[0, +\infty)$. In addition, either $z(t)$ remains inside the interior $\text{int } \mathcal{X}$ of $\mathcal{X}$ forever, or it leaves $\text{int } \mathcal{X}$ through the boundary of $\mathcal{X}$ in finite time. That is, there exists $\beta = \beta(q^c, z_0) \in [0, +\infty]$ such that for all $t \in [0, \beta)$, $z(t) \in \text{int } \mathcal{X}$. In addition, $\beta < +\infty$, then $z(\beta) \in \partial X$, i.e. $z(\beta)$ belongs to the boundary of X. We refer to $[0, \beta)$ as the *maximal interval of existence of the solution of (6) inside* $\text{int } \mathcal{X}$.

Definition 23 (Flow of the vector field f_{q^c}). For any time instant $t \in [0, +\infty)$ and for any $q^c \in Q_c$ define the *flow* $f_{q^c}^t : \mathcal{X} \to \mathcal{X}$ of f_{q^c} as follows. For any $z_0 \in \text{int } \mathcal{X}$, consider the solution z of the initial value problem (6) and its maximal existence interval $[0, \beta)$ in $\mathcal{X}$. Then $f_{q^c}^t(z_0) = \begin{cases} z(t) & \text{if } t < \beta \\ z(\beta) & \text{if } \beta \leq t < +\infty \end{cases}$. For any $z_0 \in \partial X$, let $f_{q^c}^t(z_0) = z_0$.

In other words, $f_{q^c}^t(z_0)$ gives either the solution of (6) inside $\text{int } \mathcal{X}$ at time t, if it exists, or the the point of the curve z which belongs to the boundary of $\mathcal{X}$ and through which z leaves $\text{int } \mathcal{X}$, i.e. the first point of z which does not belong to the interior $\text{int } \mathcal{X}$. Notice that our definition of the flow differs from the classically accepted one. The reason for the definition above is that we are interested in the evolution of the system only in the interior $\mathrm{int}\ \mathcal{X}$ of $\mathcal{X}$.

Proposition 4 (Semigroup property). The flow defined above has the semi-group property; for each $s, t \in \mathbb{R}_+$, $f^t_{q^c}(f^s_{q^c}(x)) = f^{t+s}_{q^c}(x)$ and $f^0_{q^c}(x) = x$ for all $q^c \in Q_c$, $x \in \mathcal{X}$.

The proof of Proposition 4 is presented in §8. Using the notation above, we formulate the following additional assumptions, which will be used in the rest of the paper.

Assumption 2. For all $q = (q^c, q^d) \in Q$, $\Sigma \in \{E_o, E_i\}$,

A1. **Initial state is not on the boundary** We assume that the initial continuous state x_0 belongs to $\mathrm{int}\ \mathcal{X}$.

A2. **Disjoint guards:** $\forall e_1 \neq e_2 \in \Sigma : \Phi_{q,e_1} \cap \Phi_{q,e_2} = \emptyset$, i.e. the guard sets Φ_{q,e_1} and Φ_{q,e_2} are disjoint,

A3. **Minimum time between repeating events:** for each $e \in \Sigma$, there exists $0 < \mathbb{T} = \mathbb{T}(q,e) \in \mathbb{R}_+$ such that if $x \in \Phi_{q,e}$ then $\forall s \in (0, \mathbb{T}) : f^s_{q^c}(x) \notin \Phi_{q,e}, \forall x \in \mathcal{X}$.

A4. **Bounded number of events on bounded time interval** For each $T > 0$, there exists $\mathbb{T}(q, T, \Sigma) \in \mathbb{N}$ such that for any $x \in \mathcal{X}$, the system H generates at most $\mathbb{T}(q, T, \Sigma)$ events from Σ on the interval $[0, T]$, if started from the state (q, x). Formally, if the events $e_1, e_2, \ldots, e_k \in E_i$, and time instances $t_1 < t_2 < \ldots < t_k \in [0, T]$ are such that $f^{t_i}_{q^c}(x) \in \Phi_{q^c, q^d_{i-1}, e_i}$, where $q^d_0 = q^d$, $q^d_i \in Q_d$, $i = 1, 2, \ldots, k$ are arbitrary, then $k \leq \mathbb{T}(q, T, \Sigma)$.

A5. **Reset maps and the state-transition map δ_c depend only on Q_c.** For each $q = (q^c, q^d) \in Q$ and each $u \in E_c$, $R_{u,q}$ and $\delta_c(q, u)$ depend only on the Q_c-valued component q^c of q. I.e. if $\hat{q} = (q^c, \hat{q}^d)$, then $R_{u,q} = R_{u,\hat{q}}$ and $\delta_c(q, u) = \delta_c(\hat{q}, u)$.

A6. **Internal events generated by discrete states** The map λ_i is a complete map, i.e. 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 $\hat{q} \in Q$, $\Phi_{\hat{q},e} = \emptyset$. In other words, no internal event generated by λ_i can be generated by crossing a guard.

Remark 4 (Assumption A5 can be dropped). The definition of the state trajectory and input-output map of H which is presented below can still be used, if Assumption A5 is dropped. The computation of a finite-state abstraction which is presented in §5.2.1 can be extended to hybrid systems for which Assumption A5 does not hold. However, this extension is notationally more involved.

Proposition 5 (Assumption A3 implies Assumption A4). If Assumption A3 holds, then Assumption A4 holds and $\mathbb{T}(q^c, q^d, T)$ can be bounded from above as follows. If $\mathbb{T} = \min\{\mathbb{T}(q^c, s, e) \mid e \in E_i \cup E_o, s \in Q_d\} > 0$, then $\mathbb{T}(q^c, q^d, T) \leq \lceil |Q_q||E_i \cup E_o|(1 + T/\mathbb{T}) \rceil$.

The proof of Proposition 5 can be found in §8. The intuition behind the assumptions is the following. Assumption A2 ensures that at most one output and at most one internal event is generated at any time instance. Assumption A3 ensures that the continuous state crosses the guard set, i.e. if a continuous state hits the guard set, then it also leaves the guard set and does not return for some time. Finally, Assumption A4 ensures that only a finite number of outputs or internal events are generated on any finite time interval. In fact, it provides an upper bound on the number of events. This is needed in order to avoid accumulation of events.

Next, we define the state evolution of H, by defining the input-to-state map. The latter maps inputs from $\mathcal{P}_{E_c}$ and disturbances from $\mathcal{P}_{E_d}$ to states.

Definition 24 (State trajectory). For any state $h = (q_h, x_h)$, $q_h = (q_h^c, q_h^d) \in Q$, $x_h \in \mathcal{X}$ and for any input $u \in \mathcal{P}_{E_c}$ and disturbance $d \in \mathcal{P}_{E_d}$, the corresponding state-trajectory is the map

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

where the discrete state components $q(t) = (q^c(t), q^d(t)) \in Q$ and the continuous state component $x(t) \in \mathcal{X}$ satisfy the following.

Define $q(0^-) = q_h$ and for $t > 0$ let $q(t^-) = \lim_{s \uparrow t} q(s)$, i.e. $q(t^-)$ is the left hand-side limit of $q(s)$ at time instance t. That is, $q(t^-) = q$ if there exists $r \in (0, t)$ such that for all $s \in [t - r, t)$, $q(s) = q$. Denote by $q^c(t^-)$ and $q^d(t^-)$ the Q_c- and Q_d-valued components of $q(t^-)$, i.e. $q(t^-) = (q^c(t^-), q^d(t^-)$. Let $x(0^-) = x_h$ and if $t > 0$, then let $x(t^-) = \lim_{s \uparrow t} x(s)$, i.e. $x(t^-)$ is the left-hand side limit at t of the map $s \mapsto x(s)$. Then,

1. The value of $(q(t), x(t))$ at $t = 0$ is as follows; $q^d(0) = q_h^d$,
$$q^c(0) = \begin{cases} \delta_c(q_h, u(0)) & \text{if } u(0) \in E_c \\ q_h^c & \text{otherwise} \end{cases}$$
$$x(0) = \begin{cases} R_{u(0), q_h}(x_h) & \text{if } u(0) \in E_c \\ x_h & \text{otherwise} \end{cases}$$

2. Let $t > r > 0$ be such that for all $s \in [t - r, t)$, $u(s) = \perp$, i.e. no input event takes place between $t - r$ and t. If $u(t) = u \in E_c$, i.e. a control input arrives at time instance t, then $q^c(t) = \delta(q(t^-), u)$ and $x(t) = R_{u, q(t^-)}(x(t^-))$.

3. If $u(s) = \perp$ on the interval $(t - r, t]$ for some $t > r > 0$, then $q^c(t) = q^c(t - r) = q^c$ and $x(t) = f_{q^c}^r(x(t - r))$, where $f_{q^c}^r$ is the flow for time r as in Definition 23. In other words, we let the continuous state evolve from $x(t - r)$ according to the differential equation $\dot{z} = f_{q^c(t-r)}(z)$ for time r or until the solution z hits the boundary $\partial\mathcal{X}$, whichever happens first. In the latter case, the continuous state does not change after it has hit the boundary.

4. Let $t > r > 0$ be such that for all $s \in (t - r, t)$, $d(s) = \perp$, $u(s) = \perp$ and $x(s) \notin \bigcup_{e \in E_i} \Phi_{q(t-r), e}$, 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 $q^d(t) = \delta_d(q(t^-), e)$. Finally, if both $d(t) = \perp$ and $x(t^-) \notin \bigcup_{e \in E_i} \Phi_{q(t^-), e}$, then $q^d(t)$ is unchanged, i.e. $q^d(t) = q^d(t - r)$.

Proposition 6. The state trajectory $\xi_H(h_0, u, d)$ is well-defined.

The proof of Proposition 6 can be found in §8. Note that the proof of Proposition 6 provide an explicit construction for the state trajectory and it could be used as an alternative constructive definition. Also note that in the definition of the state trajectory the disturbances have a preference; the system first reacts to inputs, then to disturbances, and only after this the generation of events using guards takes place.

Notice that the state-trajectory $\xi_H(h, u, d)$ is well-defined, even if disturbances and control inputs happen simultaneously. Next, we define the input-output map of H induced by its initial state. This input-output map will be of the same form as in Definition 9.

Definition 25 (Input-output map of H). Define *input-output map* of the hybrid system H induced by state $h \in S_H$ as $\upsilon_{H,h} : \mathcal{P}_{E_c} \times \mathcal{P}_{E_e} \to \mathcal{P}_{E_o} \times \mathcal{P}_{E_i}$ so that for any input $u \in \mathcal{P}_{E_c}$ and disturbance $d \in \mathcal{P}_{E_d}$,

$$\upsilon_{H,h}(u, d) = (o, \hat{o})$$

where the time-event functions $o \in \mathcal{P}_{E_o}$ and $\hat{o} \in \mathcal{P}_{E_i}$ are defined as follows. For each time instance $t \in \mathbb{R}_+$ denote the current state of H by $\xi_H(h, u, d)(t) = (q(t), x(t)) \in Q \times \mathcal{X}$,

$q(t) = (q^c(t), q^d(t))$. Recall from Definition 24 the definition of the state $q(t^-)$. For each $t \in \mathbb{R}_+$, $o(t)$ and $\hat{o}(t)$ are defined then as follows.

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

$$\hat{o}(t) = \begin{cases} e \in E_i & \text{if } x(t^-) \in \Phi_{q(t^-),e} \text{ and } d(t) = \perp \text{ and } t > 0 \\ \lambda_i(q(t^-), d(t)) & \text{if } d(t) \in E_d, t > 0, \text{ and } \lambda_i(q(t^-), d(t)) \text{ is defined} \\ \perp & \text{otherwise} \end{cases} \tag{7}$$

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, the output of H is obtained from the current state (q, x) is follows. If there are no disturbances, then an output or internal event e is generated if the continuous state x belongs to the corresponding guard set $\Phi_{q,e}$. If a disturbance d arrives, then an output (resp. internal event) is generated according to the readout map λ_o (resp. λ_i). That is, the output (resp. internal event) equals $\lambda_o(q, d)$, (resp. $\lambda_i(q, d)$) whenever a disturbance d has arrived.

Remark 5 (Role of disturbances). In other words, we assume that *the disturbances do not influence the differential equations describing the continuous state evolution.*

Proposition 7 (Input-output maps are well-defined). The input-output map $\upsilon_{H,h}$ is well-defined, i.e. for any input $u \in \mathcal{P}_{E_c}$ and $d \in \mathcal{P}_{E_d}$, $(o, \hat{o}) = \upsilon_{H,h}(u, g)$ is uniquely defined and o, $\hat{o}$ are time-event functions from $\mathcal{P}_{E_o}$ and $\mathcal{P}_{E_i}$ respectively. Moreover, $\upsilon_{H,h}(u, g)$ is causal.

The proof of Proposition 7 can be found in §8.

Remark 6 (Role of Assumption A3). Notice that while (7) indeed defines o and $\hat{o}$ as functions of time with values in $E_o \cup \{\perp\}$ and $E_i \cup \{\perp\}$ respectively, Assumption A3 is needed to ensure that these maps are time-event functions.

5.2 Computation of a finite-state abstraction of R_H

Below we will present the definition of the quasi-sequential transducer, which recognizes an abstraction of the sequential input-output map R_H associated with H. Throughout the section we assume that H is the hybrid system of Definition 22, and that H satisfies Assumption A1–A6.

5.2.1 Quasi-sequential transducer recognizing the sampled input-output behavior of H

We will need a number of assumptions on H. In order to state these assumptions, we need the following definitions. We start with the definition of the *state-space* $\mathcal{R}(H)$ of the finite-state abstraction of H.

Definition 26 (State-space of the finite-state abstraction). Let $\mathcal{R}(H)$ be the set Let $\mathcal{R}(H) = \bigcup_{i=0}^{\infty} Q \times H_i$, such that

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

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

In the sequel we will use the following assumption

Assumption 3 (Finiteness of $\mathcal{R}(H)$). We assume that the set $\mathcal{R}(H)$ is finite.

The assumption above is a very strong one, and finding systems for which it is true is a non-trivial task. *We will provide sufficient conditions for the finiteness of $\mathcal{R}(H)$ in §5.2.2. In §6 we will provide an example of a system for which these sufficient conditions are true*.

Remark 7 (Finiteness of $\mathcal{R}(H)$ can be dropped). The notion of a quasi-sequential transducer can be extended to allow systems with infinite state-spaces. The concept of a sequential input-output maps recognized by a quasi-sequential transducer with an infinite state-space can be defined in the same way as for the finite-state quasi-sequential transducer. If we drop the assumption that $\mathcal{R}(H)$ is finite, then the system $H_\Delta(\mathcal{P})$ to be defined below is an infinite-state quasi-sequential transducer and all the results of this subsection hold. In particular, Proposition 8, Theorem 4 and remain true, even if $\mathcal{R}(H)$ is infinite. Hence, the construction below can be seen as a general scheme to sample a hybrid system, i.e. to convert a continuous-time hybrid system to a discrete-time one.

The main idea behind the construction of the sampled-time abstraction is that it is enough to look at states at sampling times, i.e. at elements of $\mathcal{R}(H)$. In addition, it is possible to estimate the events generated during a sampling interval by using the state at the beginning of the sampling time and applying the flow. More precisely, we will introduce the notion of *guard abstraction predicates*, i.e. predicates which are true whenever an event is generated in the sampling interval as a result of crossing a guard. The guard abstraction predicates can be thought of as an abstraction (approximation) of the guard set. The sampled-time abstraction will be parameterized by a collection of such predicates. The better these predicates approximate the guard sets, the closer the behavior is of the sampled-time abstraction to that of the original plant.

Definition 27 (Guard abstraction predicates). Consider a discrete state $q = (q^c, q^d) \in Q$ and an event $e \in E_i \cup E_o$. The relation $P_{q,e} \subseteq \mathcal{X}$ is said to be a *guard abstraction predicate* for the guard set $\Phi_{q,e}$, if either $P_{q,e} = \emptyset$ and $e = \lambda_i(q, d)$ for some $q \in Q, d \in E_d$, or

$$\forall x \in \mathcal{R}(H) : [(\exists t \in (0, \Delta] : f^t_{q^c}(x) \in \Phi_{q,e}) \implies x \in P_{q,e}] \tag{8}$$

We call a collection $\mathcal{P} = \{P_{q,e}\}_{q \in Q, e \in E_i \cup E_o}$ a *collection of guard abstraction predicates*, if for each $q^c \in Q_c$, $e \in E_i \cup E_o$, $P_{q,e}$ is a guard abstraction predicate for the guard set $\Phi_{q,e}$. The collection of guard predicates $\mathcal{P}$ is called *computable*, if for every $q \in Q$, $e \in E_i \cup E_o$ a numerical algorithm [1] exists to decide whether $x \in P_{q,e}$. The collection $\mathcal{P}$ is called *exact approximation of guards*, if for all $q = (q^c, q^d) \in Q$, $e \in E_i \cup E_o$,

$$\forall x \in \mathcal{R}(H) : [(\exists t \in (0, \Delta] : f^t_{q^c}(x) \in \Phi_{q,e}) \iff x \in P_{q,e}] \tag{9}$$

i.e. instead of the implication in (8), equivalence holds.

Intuitively, a guard abstraction predicate $P_{q,e}$ contains those continuous states, started from which the guard set corresponding to the event e is crossed within time Δ. Consequently, a computable collection of guard abstraction predicates is just a collection of computable (in a certain sense) sets $P_{q,e}$ with the above property.

We will present a general scheme for constructing a quasi-recognizable abstraction of H. The construction uses a fixed collection of guard abstraction predicates as parameters. In general, the behavior of this state abstraction will contain the original symbolic behavior of R_H. Note that finding *computable* collections of guard abstraction predicates is a non-trivial task, and represents one of the core problems in computing the abstraction. Later in this paper, we will present classes of hybrid systems, for which such computable guard abstraction predicates can be found. However, first we present the general procedure for constructing a symbolic abstraction of H.

Definition 28 (Sampled-time abstraction). Let $\mathcal{P} = \{P_{q,e}\}_{q \in Q, e \in E_i \cup E_o}$ be a collection of guard abstraction predicates for the system H. Define the quasi-sequential transducer $H_\Delta(\mathcal{P})$ as

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

[1] By a numerical algorithm we mean an algorithm which uses the usual elementary arithmetical operations on real numbers. It means that when applied to rational numbers, the algorithm becomes an algorithm in the usual sense.

- $h_0 = (q_0^c, q_0^d, x_0)$ is the initial state of H_Δ; it coincides with that of H.
- $E : \mathcal{R}(H) \times (U \times D) \times O \times E_i^* \to \mathcal{R}(H)$ is the *state-transition* relation 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 $h_i = (q_i, x_i) \in \mathcal{R}(H)$, $i = 1, 2$, 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.
  $$q_2^c = \delta_c(q_1, u) \text{ and } x_2 = f^{\Delta}_{q_2^c}(R_{u,q_1}(x_1)) \tag{10}$$
  Here $\delta_c(q_1, u)$ and $R_{u,q_1}(x_1)$ are interpreted for $u = \perp$ as identity maps, i.e.
  $$\delta_c(q_1, \perp) = q_1^c \text{ and } R_{\perp,q_1}(x_1) = x_1$$
  2. Assume that $d = e_1 e_2 \cdots e_k$, $0 \leq k \leq \mu$, $e_1, e_2, \ldots, e_k \in E_d$. Here μ is the fixed bound on the number of disturbances within the intervall $(0, \Delta]$ from Notation 4. Then the sequence $\hat{o}$ is of the form $\hat{o} = z_1 z_2 \cdots z_l$, where $k \leq l \leq \mathbb{T}((q_2^c, q_1^d), \Delta) + k$ and $z_1, z_2, \ldots, z_l \in E_i$ and the following holds. There exists indices $i_1 < i_2 < \cdots < i_k \in \{1, 2, \ldots, l\}$ and discrete states $s_i \in Q^d$, $i = 0, 1, \ldots, l$ such that $s_0 = q_1^d$, $s_l = q_d^2$ and for all $i = 1, \ldots, l$
  $$s_i = \begin{cases} \delta_d(q_2^c, s_{i-1}, z_i) & \text{if } R_{u,q_1}(x_1) \in P_{q_2^c, s_{i-1}, z_i} \text{ and } i \notin \{i_1, i_2, \ldots, i_k\} \\ \delta_d(q_2^c, s_{i-1}, e_r) & \text{if } i = i_r \text{ for some } r \in \{1, 2, 3, \ldots, k\}, \\ & \text{and } z_i = \lambda_i(q_2^c, s_{i-1}, e_r) \end{cases} \tag{11}$$
  3. The output $o \subseteq 2^{E_o}$ is an arbitrary subset of events from E_o, such that if $e \in o$, then the following condition holds;
  $$\begin{aligned} & R_{u,q_1}(x_1) \in P_{q_2^c, s_{i-1}, e} \text{ for some } i \in \{1, 2, \ldots, l\} \text{ and } i \notin \{i_1, i_2, \ldots, i_r\}, \text{ or} \\ & \lambda_o((q_2^c, s_{i_r - 1}), e_r) = e \text{ for some } r \in \{1, 2, 3, \ldots, k\} \end{aligned} \tag{12}$$
  Here $i_1, i_2, \ldots, i_k$ and $s_1, s_2, \ldots, s_l$ are the same as in (11) from the previous item.

Intuition The intuition behind the definition of $H_\Delta(\mathcal{P})$ is the following. The states of $H_\Delta(\mathcal{P})$ 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 $H_\Delta(\mathcal{P})$ 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. First, if the current state of $H_\Delta(\mathcal{P})$ is $h_1 = (q_1^c, q_1^d, x_1)$, then the new state will be $h_2 = (q_2^c, q_1^c, x_2)$, where 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 event after that,
2. H receives a disturbance signal g, such that the sequence of disturbance events corresponding to g is d
3. the sequence of internal events generated by H while moving from h_1 to h_2 equals $\hat{o}$.
4. the set of outputs generated by H while moving from state h_1 to h_2 coincides with o

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 (10).

The definition of q_2^d takes into account the fact that the evolution of the Q_d-valued state component depends on the disturbances and internal events. In order to define the value fo q_d^2, the sequence of disturbances and the sequence of internal events should be specified. The former is d, the latter is encoded in $\hat{o}$.

From the definition of H and Assumption A6 it follows that an internal event is generated either as the result the application discrete readout maps at the arrival time of a disturbances, or when crossing a guard, and for each event precisely one of the above conditions hold. The latter means that the knowledge of $\hat{o}$ and d is sufficient to determine the relative order of internal events and disturbances. This allows us to compute the sequence of Q_d-valued discrete states which the system H goes through on the interval $(0, \Delta]$ while moving from h_1 to h_2. The computation of these Q_d-valued states along with checking Condition 3 is formalized in (11). There, the first case describes the situation when an internal event is generated because of crossing a guard, and the second one describes the generation of an internal event by discrete readout map. The former is approximated by checking if x_2 belongs to the guard abstraction predicate corresponding to the guard. It is clear that if the system evolution indeed crosses the guard, then x_2 will belong to the guard abstraction predicate. The converse need not be true in general. We need guard abstraction predicates because we cannot precisely estimate the *time* and *state* in which H crosses the guard.

Finally, Condition 4 is formalized in (12). Indeed, an output event can be generated while crossing the guard, or by using the discrete readout maps. The former is stated in the first branch of (12), the latter is stated in the second branch. Notice that in (12) crossing the guard is checked by checking if x_2 belongs to the corresponding guard abstraction predicate.

Notice that the rules (12–11) allow more sequences o and $\hat{o}$ than H (more precisely, R_H) can generate. However, we will claim that *anything H can generate is also allowed by* $H_\Delta(\mathcal{P})$, i.e. $H_\Delta(\mathcal{P})$ is an abstraction of R_H.

Formally, we state the following regarding the well-posedness and computability of $H_\Delta(\mathcal{P})$.

Proposition 8 (Well-posedness and computability). The tuple $H_\Delta(\mathcal{P})$ is a quasi-sequential transducer. If $\mathcal{P}$ is computable and the reset maps and flows are numerically computable, then the state transition map E of $H_\Delta(\mathcal{P})$ is computable.

The proof of Proposition 8 is presented in §8. The most important property of $H_\Delta(\mathcal{P})$ is that it provides an abstraction of R_H.

Theorem 4 (Abstraction). The relation $R(H_\Delta(\mathcal{P}))$ recognized by $H_\Delta(\mathcal{P})$ is a sequential input-output map and it is an abstraction of R_H.

The proof of Theorem 4 is presented in §8. This and the fact that $H_\Delta(\mathcal{P})$ is a quasi-sequential transducer, implies RW theory can be used to solve Problem 4.2 for $R = R(H_\Delta(\mathcal{P}))$, if K satisfies Assumption 1, and the solution yields a solution of the original control problem for H.

5.2.2 Sufficient conditions for Assumption 3

Notice that the computation of $H_\Delta(\mathcal{P})$ relies heavily on the finiteness of $\mathcal{R}(H)$. This calls for studying conditions under $\mathcal{R}(H)$ is finite. Below we will present sufficient conditions for a finiteness of $\mathcal{R}(H)$. The conditions are based on existence of a Lyapunov-like function and are inspired by [29].

Theorem 5 (Lyapunov-like conditions for finiteness of $\mathcal{R}(H)$). . Consider the hybrid system H from Definition 22. Consider a finite set $\mathcal{X}_0 \subseteq \mathcal{X}$. If there exists a smooth map $V : \mathbb{R}^n \to \mathbb{R}$ such that

1. For all $x \in \mathcal{X}$, $V(x) \geq 0$ and $V^{-1}(0) \cap \mathcal{X} \subseteq \partial\mathcal{X}$.
2. There exists a constant $c > 0$ such that for all $q^c \in Q_c$, $\mathrm{grad} V(x) f_{q^c}(x) < -c$, $\forall x \in \mathcal{X}$,
3. For all $x \in \mathrm{int}\ \mathcal{X}$, $u \in E_c$ and $q \in Q$, $V(R_{u,q}(x)) \leq V(x)$,

4. For all $e \in E_c$, $q \in Q$, $R_{u,q}(\partial \mathcal{X}) \subseteq \mathcal{X}_0$, i.e. the boundary of $\mathcal{X}$ is mapped to the finite set $\mathcal{X}_0$ by reset maps

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

The proof of Theorem 5 is presented in §8.

Remark 8 (Possible extensions). The sufficient conditions of Theorem 5 are probably not the most general ones. In particular, on could consider non-smooth Lyapunov functions V, or Lyapunov functions which depend on the discrete state component. The investigation of all these possibilities would go beyond the scope of this paper.

Remark 9 (Feedback transformation rendering $\mathcal{R}(H)$ finite). The conditions of 5 indicate that it might be possible to transform a system by a feedback transformation to a one for which $\mathcal{R}(H)$ is finite. To this end, one could follow an approach similar to stabilization using control Lyapunov functions [27]. That is, one could try to find a control law and a function $V : \mathcal{X} \to \mathbb{R}$ such that V satisfies the conditions of Theorem 5 if the control law applied. This approach would allow to apply the construction of $H_{\triangle}(\mathcal{P})$ to systems which do not satisfy Assumption 3. In this paper we will not develop the theory of feedback transformation yielding a finite $\mathcal{R}(H)$.

Remark 10 (Robustness of computation of $H_{\triangle}(\mathcal{P})$). Assume that the conditions of Theorem 5 hold and assume that the reset maps are continuous on $\mathrm{int}\, \mathcal{X}$, and that $\mathcal{X}_0 = \{z_0\}$, i.e. is a singleton set. In addition, assume that each abstraction predicate $\mathcal{P}_{q,e}$, $q \in Q$, $e \in E_i \cup E_o$ is an open subset of $\mathrm{int}\, \mathcal{X}$. We then conjecture that if we perturb the hybrid system H by a small perturbation, and we apply Definition 28 to the perturbed hybrid system H^d, then the resulting quasi-sequential transducer $H^d_{\triangle}(\mathcal{P})$ recognizes the same sequential input-output map as $H_{\triangle}(\mathcal{P})$, i.e. $R(H^d_{\triangle}(\mathcal{P})) = R(H_{\triangle}(\mathcal{P}))$.

Finally, we would like to discuss computational methods for finding maps V satisfying the conditions of Theorem 5. Since V is a Lyapunov-like functions, the computational methods for finding V are simillar to that of for Lyapunov-functions. Assume that the reset maps are affine on $\mathrm{int}\, \mathcal{X}$, and the vector fields f_q are L'ure-type [13]. More precisely, assume the following.

Assumption 4. The reset maps of H are affine in $\mathrm{int}\, \mathcal{X}$, the vector fields of are of L'ure-type, the state-space is polyhedral, i.e.

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

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

Proposition 9. Assume that H satisfies Assumption 4. If for some $j \in \{1, 2, \dots, K\}$, $c > 0$ and for all $x \in \mathcal{X}$, $q = (q^c, q^d) \in Q$,

1. $n_j^T (A_{q^c} x + \sum_{j=1}^m (\mu_{i_j} B_{q^c,j} r^T_{q^c,j} x + \gamma_{i_j} B_{q^c,j})) > c$, for any choice of $i_1, i_2, \dots, i_m \in \{1, 2\}$.
2. If $x \in \mathrm{int}\, \mathcal{X}$, then $n_j^T (M_{u,q} x - x + b_{u,q}) \ge 0$ for any $u \in E_c$

then $V(x) = (b_j - n_j^T x)$ satisfies the conditions 1–3 of Theorem 5.

Remark 11. If $\mathcal{X}$ is a bounded closed polytope, then we can replace Condition 1 of Proposition 9 with $n_j^T(A_{q^c}x + \sum_{j=1}^{m}(\mu_{i_j}B_{q^c,j}r_{q^c,j}^T x + \gamma_{i_j}B_{q^c,j})) > 0$, for all $i_1, i_2, \ldots, i_m \in \{1,2\}$ and $\forall x \in \mathcal{X}$. Indeed, if the above condition holds, then

$$c = \min_{i_1,i_2,\ldots,i_m \in \{1,2\}, x \in \mathcal{X}} n_j^T(A_{q^c}x + \sum_{j=1}^{m}(\mu_{i_j}B_{q^c,j}r_{q^c,j}^T x + \gamma_{i_j}B_{q^c,j}))$$

exists (in fact, the minimum is taken at one of the vertices), and $c > 0$. Hence, Condition 1 of Proposition 9 holds.

Notice that the conditions of Proposition 9 can be checked numerically. In particular, by checking for each facet $n_j^T x + b_j = 0$ whether the conditions of Proposition 9 holds, one can find a Lyapunov-like map satisfying the conditions of Theorem 5. Notice the resemblance of Proposition 9 to the control-to-facet approach of [12].

5.2.3 Classes of hybrid systems for which computable collection of guard abstraction predicates exist

As it was noted before, $H_\Delta(\mathcal{P})$ is computable, if $\mathcal{P}$ is a computable collection of guard abstraction predicates. Below we present two system classes for which a computable collection $\mathcal{P}$ exists, along with the definition of $\mathcal{P}$. The first such class is the class *well-discretizable hybrid systems*. The main property of these systems is that the guard set is monotone in the continuous dynamics.

Definition 29 (Well-discretizable hybrid systems). A hybrid system H of the form (5) is called *well-discretizable* if for each $q = (q^c, q^d) \in Q = Q_c \times Q_d$ and event $e \in E_o \cup E_i$, there exist smooth maps $h_{q,e} : \mathbb{R}^n \to \mathbb{R}$, such that

$$\Phi_{q,e} \subseteq \{x \in \text{int}\,\mathcal{X} \mid h_{q,e}(x) = 0\} \tag{13}$$

$$\Phi_{q,e} \neq \emptyset \implies (\forall x \in \mathbb{R}^n : \frac{d}{dx}h_{q,e}(x)f_{q^c}(x) > 0) \tag{14}$$

That is,(13) says that an (output or internal) event is generated, if the state-trajectory passes the zero set of some smooth map, and (14) says that the image of the state-trajectory by each of this map is monotone increasing function of time. The latter means that if the state crossed a guard, then it will not come back to this guard any more, unless a discrete state change occurs. These smooth maps can originate from timing constraints or from sensor position, passing of which triggers an event. Notice that well-discretizable hybrid systems automatically satisfy Assumption A3–A4.

For well-discretizable hybrid systems, we can define the following collection of guard abstraction predicates.

Definition 30 (Guard abstraction predicates for well-discretizable systems). Assume that H is a well-discretizable hybrid system. Using the notation of Definition 29, for each $(q^c, q^d) \in Q$, $e \in E_i \cup E_o$ define the set $P_{q,e}$ as

$$P_{q,e} = \{x \in \text{int}\,\mathcal{X} \mid h_{q,e}(x) \leq 0 \text{ and } h_{q,e}(f_{q^c}^{\Delta}(x)) \geq 0\} \tag{15}$$

The collection $\mathcal{P} = \{P_{q,e}\}_{q \in Q, e \in E_i \cup E_o}$ will be called a *well-discretizable collection*.

Lemma 1. If H is a well-discretizable hybrid system, then for each $q = (q^c, q^d) \in Q$, $e \in E_i \cup E_o$, the set $P_{q,e}$ defined in (15) is an abstraction predicate of the guard $\Phi_{q,e}$, and thus the collection $\mathcal{P}$ from Definition 30 is a collection of guard abstraction predicates. In addition, $\mathcal{P}$ is computable, if $h_{q,e}(x)$ and the flow $f_{q^c}^t(x)$ can be computed for all $x \in \mathbb{R}^n$, $t \in \mathbb{R}$, $q = (q^c, q^d) \in Q = Q_c \times Q_d$, $e \in E_i \cup E_o$.

The proof of Lemma 1 is presented in §8.

23 Finite-state abstraction of R_H

Remark 12 (Exact approximation of guards). If (13) holds with equality for all $q \in Q$, $e \in E_i \cup E_o$, i.e. $\Phi_{q,e} = h_{q,e}^{-1}(0)$, then the well-discretizable collection $\mathcal{P}$ from Definition 30 is an exact approximation of guards.

Below we define the class of semi-algebraic hybrid systems for which H_Δ is computable as well. For the definition of semi-algebraic sets and maps we refer the reader to [4]. Roughly speaking, semi-algebraic sets and maps are sets and maps defined using polynomial equations and inequalities.

Definition 31 (Semi-algebraic hybrid systems). A hybrid system H of the form (5) is *semi-algebraic* if for all $q = (q^c, q^d) \in Q$,

1. The reset maps $R_{u,q}$ are semi-algebraic for all $u \in E_c$.
2. The guards sets $\Phi_{q,e}$ are semi-algebraic for all $e \in E_o \cup E_i$.
3. The flow $\mathcal{X} \times \mathbb{R}_+ \ni (x,t) \mapsto f_{q^c}^t(x)$ is a semi-algebraic map.
4. $\mathcal{X}$ is a semi-algebraic set.

The above system class is a subclass of semi-algebraic hybrid automata of [24]. For semi-algebraic hybrid systems, we can define the following collection of guard abstraction predicates.

Definition 32 (Guard abstraction predicates for semi-algebraic systems). Assume that H is a semi-algebraic hybrid system. For each $q = (q^c, q^d) \in Q$, $e \in E_i \cup E_o$, define the set $P_{q,e}$

$$P_{q,e} = \{z \in \mathbb{R}^n \mid \exists s \in (0, \Delta] : f_{q^c}^s(z) \in \Phi_{q,e}\} \tag{16}$$

We call the collection $\mathcal{P} = \{P_{q,e}\}_{q \in Q, e \in E_o \cup E_i}$ a *semi-algebraic collection of guard abstraction predicates*.

Lemma 2 (Guard abstraction predicates). Assume that H is a semi-algebraic hybrid system. For each $q = (q^c, q^d) \in Q$, $e \in E_i \cup E_o$, the set $P_{q,e}$ defined in (16) is semi-algebraic and $P_{q,e}$ is an abstraction predicate for the guard $\Phi_{q,e}$. Hence, the collection $\mathcal{P}$ from Definition 16 is a collection of guard predicates. Moreover, $\mathcal{P}$ is computable, and it is an exact approximation.

6 Illustrating example

The goal of the section is to illustrate the theory by presenting the solution of a control problem occurring in printers. A more detailed exposition of the control problem can be found in [22].

The task of desired controller is to take care of error-handling of the toner system of the printer. More explicitly, recall from [22] that the process of transferring the image onto the paper sheet takes place in a section of the printer which is known as the *fuse pinch*. This is the point where the tape containing the image meets the paper sheet, which is carried from the feeder tray of the printer to the finisher. The tape which contains the image is called the *TTF belt* and it is part of the toner subsystem. After being treated in the fuse, the paper gets transferred to the *finisher*, which is the place it can be picked up from.

The TTF belt forms a closed loop which revolves around its center. That is, each point of the TTF belt makes a full cycle. The image gets onto the TTF belt from a separate entity, which we call the *writing unit*. This unit is situated at a certain point of the loop formed by the TTF belt. The fuse is situated at another point of the loop. There is a third important item, the *cleaner*, which is the point after the fuse where the TTF belt is cleaned. The error situation we are interested in arises when there is a paper jam, i.e. the paper does not arrive to the fuse pinch at the designated time instance. In this case if nothing is done, the TTF belt touches directly the pinch of the fuse, instead of the paper. The pinch gets polluted, which leads to bad quality when printing subsequent pages. The task is to prevent the portion of the TTF belt containing the image from coming into contact with the fuse pinch, if there is no paper in the fuse. The only sensor which is available is the *X-fine sensor* which becomes activated when the sheet gets delayed in the paper path, possibly due to paper jam. When activated, the sensor generates the **Sheet too late** signal . We have the following control actions at our disposal to solve the control problem.

Lift the fuse pinch from the TTF belt. This action physically decouples the TTF belt from the fuse pinch, so that no physical contact between the two entities is possible. Note that this action requires certain amount of time.

Change the speed of the TTF belt. Note that there is a range of admissible speeds for the TTF belt.

Informally, the task is to figure out when and by how much to slow down the TTF belt so that there is time to lift the fuse pinch. The main underlying idea of the proposed plant model is the following. We model the behavior of the toner system from the point of view of the TTF belt. That is, we start looking at the system from the moment when writing unit starts writing the image on the TTF belt. Subsequently we keep track of the point of the TTF belt which was at the writing unit when the transfer of the image from the writing unit onto the TTF belt started.

6.1 Formal model of the plant

In this section we will present the formal model of the plant. When defining the model, we will use the following parameters of the toner system.

$\mathbf{Fp}$ The relative position of the fuse with respect to the writing unit along the TTF tape. More precisely, this is the length of the section of the TTF tape which is spanned between the fuse and the writing unit. The length is measured in the direction of revolution of the TTF belt.

$\mathbf{Cp}$ The relative position of the cleaner with respect to the writing unit along the TTF tape. In other words, this is the length of the section of the TTF tape which is spanned between the cleaner and the writing unit and passes through the fuse. The length is measured in the direction of revolution of the TTF belt.

$\mathbf{V}_{max}$ The maximal allowed speed of the TTF belt.

$\mathbf{V}_{min}$ The minimal allowed speed of the TTF belt.

$\mathbf{T}_{fo}$ The time which is needed to open the fuse pinch.

$\mathbf{T}_{pl,max}$ The latest time instance at which a **Sheet too late** signal can be generated.

$\mathbf{T}_{pl,min}$ The earliest time instance at which a **Sheet too late** signal can be generated.

$\mathbf{A}$ The maximal value of acceleration with which the speed of TTF can be increased.

$\mathbf{D}$ The maximal value of deceleration in with which the TTF tape can be slowed down.

In the sequel we will assume that the parameters satisfy the following conditions.

Assumption 5. $\mathbf{V}_{max} > \mathbf{V}_{min} > 0$, $\mathbf{T}_{pl,max} > \mathbf{T}_{pl,min} \geq 0$, $0 < \mathbf{Fp} < \mathbf{Cp}$, and $\mathbf{T}_{pl,max} > \frac{\mathbf{Fp}}{\mathbf{V}_{max}}$.

For the concrete parameter values occurring in practice, these conditions hold.

Formally, the plant model is a hybrid system of the form

$$H = (S_H, \delta, \lambda_i, \lambda_o, E, \{f_q, R_{u,q}, \Phi_{q,e} \mid q \in Q, u \in E_c, e \in E_i \cup E_o\}, h_0)$$

The various components of H are explained below.

The set of control actions $E_c = \{\mathbf{c}_{FU}, \mathbf{c}_{FD}, \mathbf{c}_A, \mathbf{c}_D\}$
where the elements of E_c denote the following control actions

$\mathbf{c}_{FU}$ – lifts the fuse pinch (i.e. physically decouples the fuse pinch and the TTF belt),

$\mathbf{c}_{FD}$ – puts back the fuse pinch (i.e. physically couples it with the TTF belt),

$\mathbf{c}_A$ – accelerate the TTF tape with a constant acceleration $\mathbf{A}$,

$\mathbf{c}_D$ – slow down the TTF tape with a constant acceleration $-\mathbf{D}$.

Output events $E_o = \{\mathbf{e}_{o,PL}\}$
The event $\mathbf{e}_{o,PL}$ is generated when the plant receives the **Sheet too late** signal. In a sense, it would be more logical to model the arrival of **Sheet too late** as an external input, which is visible to the controller. However, we can always model external inputs as a combination of disturbances and outputs, as described in Remark 1.

Disturbances $E_d = \{\mathbf{e}_{d,PL}\}$
The event $\mathbf{e}_{d,PL}$ models that **Sheet too late** signal was sent to the plant. The arrival of $\mathbf{e}_{d,PL}$ immediately leads to the generation of the output $\mathbf{e}_{o,PL}$ and internal event $\mathbf{e}_{i,PL}$.

Set of internal events $E_i = \{\mathbf{e}_{NPIF}, \mathbf{e}_{i,PL}, \mathbf{e}_{min,PL}, \mathbf{e}_{max,PL}, \mathbf{e}_{FUc}, \perp\}$

$\mathbf{e}_{FUc}$ The event $\mathbf{e}_{FUc}$ denotes the situation when the lifting of the fuse pinch has been completed.

$\mathbf{e}_{NPIF}$ The event $\mathbf{e}_{NPIF}$ is generated when the toner image reaches the fuse, the fuse is not decoupled from the TTF belt, but there is no sheet in the paper path at the fuse. In short, $\mathbf{e}_{NPIF}$ is generated exactly in a situation which we want to avoid.

$\perp$ is a dummy event introduced to make λ_i a complete map. It has no physical meaning.

$\mathbf{e}_{i,PL}$ The event $\mathbf{e}_{i,PL}$ is generated if **Sheet too late** signal is fed to the plant in the time interval $[\mathbf{T}_{pl,min}, \mathbf{T}_{pl,max}]$, i.e. $\mathbf{e}_{i,PL}$ is just a copy of **Sheet too late** .

$\mathbf{e}_{min,PL}$ The event $\mathbf{e}_{min,PL}$ is generated after $\mathbf{T}_{pl,min}$ time units have passed.

$\mathbf{e}_{max,PL}$ The event $\mathbf{e}_{max,PL}$ is generated after $\mathbf{T}_{pl,max}$ time units have passed.

The intuition behind the choice of internal events is the following. The internal events capture a situation which is relevant for specifying the control requirements. The role of each event will become clear after we have formulated the control requirements.

Discrete state-space Q 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 $\mathbf{Var}_d = \{\mathbf{S}_{PL}, \mathbf{S}_r, \mathbf{S}_{FUc}\}$ The interpretation of the elements of $\mathbf{Var}_d$ is as follows.

  $\mathbf{S}_{PL}$ equals True, if a **Sheet too late** has arrived.

  $\mathbf{S}_{FUc}$ equals True if the lifting of the fuse pinch has been completed.

  $\mathbf{S}_r$ equals True if the current time which has passed since the start-up of the plant is between $[\mathbf{T}_{pl,min}, \mathbf{T}_{pl,max}]$.

- Q_c is the set of all maps $\phi : \mathbf{Var} \to \{True, False\}$ where $\mathbf{Var} = \{\mathbf{S}_{FU}, \mathbf{S}_{FD}, \mathbf{S}_A, \mathbf{S}_D\}$ is the set of predicate symbols. The physical interpretation of the variables is as follows.

  $\mathbf{S}_{FU}$ equals True if an order to lift the fuse pinch has been received.

  $\mathbf{S}_{FD}$ equals True if the fuse pinch is on the TTF belt.

  $\mathbf{S}_A$ equals True if a command to accelerate the TTF belt was received.

  $\mathbf{S}_D$ equals True if a command to decrease the speed of the TTF belt was received.

That is, 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 $X \in \mathbf{Var}_c$.

Continuous state-space $X \subseteq \mathbb{R}^4$**,** $\mathcal{X} = \{x = (\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) \in \mathbb{R}^4 \mid \mathbf{P} \leq \mathbf{Cp}\}$ where $\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T} \in \mathbb{R}$ is as follows.

$\mathbf{P}$ The variable $\mathbf{P}$ denotes the current position of the point of the TTF belt which stood at the writing unit when the writing of the image began.

$\mathbf{V}$ The variable $\mathbf{V}$ denotes the current speed of the TTF tape.

$\mathbf{C}_{fu}$ The variable $\mathbf{C}_{fu}$ is active only if the pinch is being lifted. It denotes the time which has passed since the command to lift the pinch was issued.

$\mathbf{T}$ Denotes the time which has passed since the start of the TTF tape. It is an auxiliary variable which is needed to determine whether a **Sheet too late** signal arrives within the designated time interval $[\mathbf{T}_{pl,min}, \mathbf{T}_{pl,min}]$.

Vector field f_{q^c}**,** $q^c \in Q_c$ The vector field for the discrete state component q^c is defined as follows. For any $x = (\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) \in \mathbb{R}^4$, $f_{q^c}(x) = (f_{1,q^c}(x), f_{2,q^c}(x), f_{3,q^c}(x), f_{4,q^c}(x))$.

$$f_{1,q^c}(x) = \max\{\mathbf{V}_{min}, \mathbf{V}\}$$

$$f_{2,q^c}(x) = \begin{cases} \mathbf{A}\phi_{min}(x)\phi_{max}(x) & \text{if } q^c(\mathbf{S}_A) \\ -\mathbf{D}\phi_{min}(x)\phi_{max}(x) & \text{if } q^c(\mathbf{S}_D) \text{ and } q^c(\mathbf{S}_{FD}) \\ 0 & \text{otherwise} \end{cases}$$

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

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

$$f_{3,q^c}(x) = 1 \text{ and } f_{4,q^c}(x) = 1 \tag{17}$$

Here ϵ is an arbitrarily chosen small enough number. The intuition behind the definition of ϕ_{min} and ϕ_{max} and f_{2,q^c} is the following. The TTF belt is accelerated or slowed down depending on the input, but if the maximal (minimal) speed is reached, then the speed does not change anymore. The is expressed by multiplying the right-hand side with $\phi_{min}\phi_{max}$. The map ϕ_{min} is the continuous approximation of the indicator function $\begin{cases} 1 & \text{if } \mathbf{V} \geq \mathbf{V}_{min} \\ 0 & \text{if } \mathbf{V} < \mathbf{V}_{min} \end{cases}$ and ϕ_{max} is the continuous approximation of the indicator function $\begin{cases} 1 & \text{if } \mathbf{V} \leq \mathbf{V}_{max} \\ 0 & \text{if } \mathbf{V} > \mathbf{V}_{max} \end{cases}$. The number ϵ indicates the accuracy of the approximation; the smaller ϵ is, the better is the approximation. It is easy to see that with the definition above, $f_{i,q^c}(x)$, $i = 1, 2, \ldots, k$ are continuous, and globally Lipschitz.

Recall that with the definitions above, the continuous dynamic of H in state q^c is defined by

$$\begin{aligned} \dot{\mathbf{P}} &= f_{1,q^c}(\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) \\ \dot{\mathbf{V}} &= f_{2,q^c}(\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) \\ \dot{\mathbf{C}}_{fu} &= f_{3,q^c}(\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) \\ \dot{\mathbf{T}} &= f_{4,q^c}(\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T}) \end{aligned}$$

In other word, the derivative of the position $\mathbf{P}$ of the image is the variable $\mathbf{V}$ describing the speed of motion of the image (the speed of motion of the TTF belt). Moreover, we look at the motion of the image up to the point of the cleaner, i.e. if $\mathbf{P}$ is greater or equal than $\mathbf{Cp}$, the we assume that $\mathbf{P}$ does not change. This is expressed in the definition of f_{1,q^c} and $\mathcal{X}$. The speed of TTF belt either increases, decreases or stays the same depending on whether control inputs $\mathbf{c}_A$, $\mathbf{c}_D$ or neither $\mathbf{c}_A$ nor $\mathbf{c}_D$ were fed into the system in the past. Notice that $\mathbf{c}_D$ leads to slowing down the TTF belt only if the fuse pinch is still on the TTF belt, i.e when $q^c(\mathbf{S}_{FD})$ true. Moreover, the speed is not allowed to go below $\mathbf{V}_{min}$ and above $\mathbf{V}_{max}$. This is expressed in the definition of f_{2,q^c}. The component f_{3,q^c} describes the evolution of the clock variable $\mathbf{C}_{fu}$, and f_{4,q^c} describes the evolution of $\mathbf{T}$.

Reset map $R_{u,q}$, $u \in E_c$ and $q \in Q$
The value of the reset map for the continuous state $x = (\mathbf{P}, \mathbf{V}, \mathbf{C}_{fu}, \mathbf{T})$ is defined as follows.

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

That is, the reset map is the identity map except when the control input is the command to lift the fuse pinch. In the latter case, the variable $\mathbf{C}_{fu}$ is set to zero. In addition, the position $\mathbf{Cp}$ has been reached, all the states are mapped to $(\mathbf{Cp}, \mathbf{V}_{max}, \mathbf{T}_{fo}, \mathbf{T}_{pl,max})$. The latter is just for convenience, it has no physical meaning, as the evolution of the plant beyond $\mathbf{Cp}$ is of no interest.

Discrete state-transition map $\delta_c : Q \times E_c \rightarrow Q_c$**.** For each $q_1 = (q_1^c, q_1^d) \in Q$, $q_2^c \in Q^c$, and for each $u \in E_c$, $\delta_c(q_1, u) = q_2^c$ if and only if the following holds.

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

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

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

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

Discrete state-transition map $\delta_d : Q \times (E_d \cup E_i) \rightarrow Q_d$**.** For each $q = (q_1^c, q_1^d) \in Q$, for all $e \in E_d \cup E_i$, $q_2^d = \delta(q_1, e)$

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

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

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

Discrete readout map $\lambda_o : Q \times E_d \rightarrow E_o$**.** For all $q \in Q$, $e_d \in E_d$,

$$\lambda_o(q, e_d) = \begin{cases} \mathbf{e}_{o,PL} & \text{if } e_d = \mathbf{e}_{d,PL} \\ \text{undefined} & \text{otherwise} \end{cases}$$

Discrete readout-map $\lambda_i : Q \times E_d \rightarrow E_i$**.** For all $q = (q^c, q^d) \in Q$, $e_d \in E_d$,

$$\lambda_i(q, e_d) = \begin{cases} \mathbf{e}_{i,PL} & \text{if } e_d = \mathbf{e}_{d,PL} \text{ and } q^d(\mathbf{S}_r) \\ \perp & \text{otherwise} \end{cases}$$

Guard sets $\Phi_{q,e}$ For all $e \in E_o \cup E_i$, $q = (q^c, q^d) \in Q$, $\Phi_{q,\mathbf{e}_{i,PL}} = \emptyset$, and for

$$e \in \{\mathbf{e}_{NPIF}, \mathbf{e}_{FUc}, \mathbf{e}_{min,PL}, \mathbf{e}_{max,PL}\}$$

the guard set $\Phi_{q^c,e}$ is defined as follows.

$$\Phi_{q,\mathbf{e}_{NPIF}} = \begin{cases} \{(\mathbf{P},\mathbf{V},\mathbf{C}_{fu},\mathbf{T}) \in \mathbb{R}^4 \mid \mathbf{P} = \mathbf{Fp}, \mathbf{P} < \mathbf{Cp}\} & \text{if } q^c(\mathbf{S}_{FD}) \text{ or } (q^c(\mathbf{S}_{FU}) \text{ and} \\ & \neg q^d(\mathbf{S}_{FUc})) \text{ and } q^d(\mathbf{S}_{PL}) \\ \emptyset & \text{otherwise} \end{cases}$$

$$\Phi_{q,\mathbf{e}_{FUc}} = \begin{cases} \{(\mathbf{P},\mathbf{V},\mathbf{C}_{fu},\mathbf{T}) \in \mathbb{R}^4 \mid \mathbf{C}_{fu} = \mathbf{T}_{fo},, \mathbf{P} < \mathbf{Cp}\} & \text{if } q^c(\mathbf{S}_{FU}) \text{ and } \neg q^c(\mathbf{S}_{FD}), \\ & \text{and } \neg q^c\mathbf{S}_{FUc} \\ \emptyset & \text{otherwise} \end{cases}$$

$$\Phi_{q,\mathbf{e}_{min,PL}} = \begin{cases} \{(\mathbf{P},\mathbf{V},\mathbf{C}_{fu},\mathbf{T}) \in \mathbb{R}^4 \mid \mathbf{T} = \mathbf{T}_{pl,min}, \mathbf{P} < \mathbf{Cp}\} \setminus \Phi_{q,\mathbf{e}_{FUc}} & \text{if } \neg q^d(\mathbf{S}_r), \\ & \text{and } \neg q^d(\mathbf{S}_{PL}) \\ \emptyset & \text{otherwise} \end{cases}$$

$$\Phi_{q,\mathbf{e}_{max,PL}} = \begin{cases} \{(\mathbf{P},\mathbf{V},\mathbf{C}_{fu},\mathbf{T}) \in \mathbb{R}^4 \mid \mathbf{T} = \mathbf{T}_{pl,max}, \mathbf{P} < \mathbf{Cp}\} \setminus \Phi_{q,\mathbf{e}_{FUc}} & \text{if } q^d(\mathbf{S}_r), \\ & \text{and } \neg q^d(\mathbf{S}_{PL}) \\ \emptyset & \text{otherwise} \end{cases}$$

The initial state is $h_0 = (q_0^c, q_0^d, x_0)$ Here $q_0^c(X) = False$ for all $X \in \{\mathbf{S}_{FU}, \mathbf{S}_A, \mathbf{S}_D\}$, and $q_0^c(\mathbf{S}_{FD}) = True$. For $Y \in \{\mathbf{S}_{PL}, \mathbf{S}_{FUc}, \mathbf{S}_r\}$, $q^d(Y) = False$ and $x_0 = (0, \mathbf{V}_{max}, 0, 0)$. That is, in the initial state no command is issued and no **Sheet too late** has been received yet. Moreover, in the initial state the image is in position zero, the clocks $\mathbf{C}_{fu}$ and $\mathbf{T}$ are zero, the speed of the TTF belt is maximal.

It is easy to see that Assumption 1 are satisfied for H.

6.2 Formal model of the control requirements

The control requirements are formulated as a language $K = K_{safe} \cup \lim(K_{safe}) \subseteq E_i^* \cup E_i^\omega$. The language K_{safe} allows for any sequence as long as it does not contain $\mathbf{e}_{NPIF}$, i.e. $K_{safe} = (E_i \setminus \mathbf{e}_{NPIF})^*$.

6.3 Formal statement of the control problem

The control problem for the toner system can be stated in formal terms as follows. Fix a sample rate $\Delta > 0$. Assume that at most one disturbance even can occur in a sampling intervall, i.e. $\mu = 1$. That is, we assume that in any sampling interval at most one **Sheet too late** signal can arrive. The task is to find a sampled-data controller $\phi : O^* \to U$, where $O = 2^{E_o}$, and $U = E_c \cup \{\bot\}$ such that the closed-loop language satisfies $L(H/\mathcal{C}_\phi) \subseteq K$.

6.4 Solution of the control problem

We can solve the the formulated control problem using the procedure outlined in §4. That is, first we compute a finite-state approximation $H_\Delta(\mathcal{P})$ of H. Notice that the language of control requirements $K = K_{safe} \cup \lim(K_{safe})$ and $H_\Delta(\mathcal{P})$ satisfies Assumption 1. Hence, we can apply the algorithm described in [21] to solve the following control problem. Find a discrete controller $\phi : O^* \to U$ such that $L(R/\phi) \subseteq K$. Here R is the input-output map recognized by the quasi-sequential transducer $H_\Delta(\mathcal{P})$, and $L(R/\phi)$ is the closed-loop language of the interconnection of R and ϕ. By Theorem 3 and Theorem 4 it then follows that the obtained controller ϕ solves the original control problem.

Notice that $H_\Delta(\mathcal{P})$ is computable. In fact, we argue that the plant H defined above has the property that $\mathcal{R}(H)$ *is finite and H is well-discretizable*. Define the set

$$\mathcal{X}_0 = \{(\mathbf{Cp}, \mathbf{V}_{max}, \mathbf{T}_{fo}, \mathbf{T}_{pl,max})\}$$

Clearly, $\mathcal{X}_0 \subseteq \mathcal{X}$. In addition, notice that $\partial\mathcal{X} = \{(x_1, x_2, x_3, x_4) \in \mathbb{R}^4 \mid x_1 = \mathbf{Cp}\}$ and hence $R_{u,q}(\partial\mathcal{X}) \subseteq \mathcal{X}_0$ for all $u \in E_c$ and $q \in Q_c$. In addition, define the map $V : \mathcal{X} \to \mathbb{R}$ as $V(x_1, x_2, x_3, x_4) = (\mathbf{Cp} - x_1)$. We will show that V satisfies the conditions of Theorem 5. It then follows that for all $x \in \mathcal{X}$, $V(x) \geq 0$ and $V(x) = 0$ if and only if $x \in \partial\mathcal{X}$. For each $q \in Q^c$, by computing $\mathrm{grad} V f_{q^c}$ we get the following

$$\mathrm{grad} V f_{q^c}(x) = -f_{1,q^c}(x) < -\mathbf{V}_{min}$$

Finally,

$$V(R_{u,q}(x)) = \begin{cases} V(x) & \text{if } u \neq \mathbf{c}_{FU} \text{ and } x \notin \partial\mathcal{X} \\ V(x_1, x_2, 0, x_4) = V(x) & \text{if } u = \mathbf{c}_{FU} \text{ and } x \notin \partial\mathcal{X} \\ 0 & \text{if } x \in \partial\mathcal{X} \end{cases}$$

i.e. $V(R_{u,q}(x)) \leq V(x)$ for all $x \notin \partial\mathcal{X}$. That is, all the conditions of Theorem 5 hold.

It is left to show that H is well-discretizable.

To see that H is well-discretizable, notice the following. For $e \in \{\mathbf{e}_{FUc}, \mathbf{e}_{NPIF}, \mathbf{e}_{min,PL}, \mathbf{e}_{max,PL}\}$, for each $q = (q^c, q^d) \in Q$, $\Phi_{q,e} \subseteq P_{q,e} = \{x \in \mathbb{R}^4 \mid h_{q,e}(x) = 0\}$, where $h_{q,e}$ is defined as follows

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

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

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

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

It then follows that for all $q = (q^c, q^d) \in Q$, $e \in E_i$, if $\Phi_{q,e} \neq \emptyset$, then grad $h_{q,e}(x) f_{q^c}(x) = 1$ or grad $h_{q,e}(x) f_{q^c}(x) = f_{1,q^c}(x) \geq \mathbf{V}_{min} > 0$. That is, H is indeed well-discretizable and $\mathcal{P} = \{P_{q,e}\}_{q \in Q, e \in E_i}$ defines a collection of guard abstraction predicates.

Using the collection of guard abstraction predicates $\mathcal{P}$ defined above we computed the finite-state abstraction $H_\Delta(\mathcal{P})$ of H. Subsequently, we can use $H_\Delta(\mathcal{P})$ as the plant model to solve the discrete-event control problem. The controllers synthesized for various parameter values, based on an algorithm related to what was presented in this paper can be found in [22]. Note that in [22] no proof of correctness of the algorithm is provided. The current paper can be seen as a theoretical foundation of algorithms of the type [22].

7 Discussion and conclusions

We have presented a control problem for hybrid systems with discrete inputs and outputs. In addition, we have proposed a class of hybrid systems for which the control problem can be solved by solving a discrete-event control problem for a finite-state abstraction of the original hybrid system. Furthermore, we identified several system classes, for which such a finite-state abstraction can be computed.

The above system classes are not only theoretically interesting, but they can also be applied practice. In particular, we can apply our results to the error-handling problem for printers, see [22]. In [22] a preliminary version of the solution procedure of this paper was already used for the simplified problem without partial observations. We expect that the results of this paper will be relevant for control problems arising in other application domains, such as logistic and production systems.

Future research includes extension of the results to more general classes of hybrid systems and study of robustness and numerical issues. Furthermore, we would like to improve the computational complexity of the algorithms and reduce the size of the obtained finite-state controller.

Acknowledgments This work was supported by the ITEA project Twins 05004. The first named author would like to thank Aleksander Pogromsky for his valuable comments and suggestions regarding Theorem 5.

8 Proofs

The goal of this section is to present the proofs of the results stated in the main text of the paper. In §8.1 we state the proofs of the statements from §4. In §8.2 we state the proofs of the statements from §5.1. In §8.3 we state the proofs of Proposition 8, and Theorem 4. In §8.4 we present the proof of Theorem 5. In §8.5 we state the proof of Lemma 1.

8.1 Proofs of the statements from §4

Proof of Proposition 1. Consider any disturbance signal $d \in \mathcal{P}_{E_d,\mu}$. We will construct the input signal $u \in \mathcal{P}_{E_c}$, the output signal $o \in \mathcal{P}_{E_o}$ and the signal of internal events $\hat{o} \in \mathcal{P}_{E_i}$ such that (4) holds. To this end, define the input signal $u_i \in \mathcal{P}_{E_c}$, output signal $o_i \in \mathcal{P}_{E_o}$ and internal event signal $\hat{o}_i \in \mathcal{P}_{E_i}$ for all $i \in \mathbb{N}$ recursively as follows. For $i = 0$, let $u_0(s) = \begin{cases} \phi(\epsilon) & \text{if } s = 0 \\ \perp & \text{otherwise} \end{cases}$ and $(o_0, \hat{o}_o) = \upsilon_H(u_0, d)$. For $i > 0$, define u_i, o_i and $\hat{o}_i$ as follows. Let

$$u_i(s) = \begin{cases} u_{i-1}(s) & \text{if } s < i\Delta \\ \phi(S_1 S_2 \cdots S_i) & \text{if } s = i\Delta \\ \perp & \text{otherwise} \end{cases}$$

where $S_j = o_{i-1}(((j-1)\Delta, j\Delta]) \cap E_o$ for all $j = 1, 2, \ldots, i$. Moreover, let $(o_i, \hat{o}_i) = \upsilon_H(u_i, d)$. Notice that $u_i(s) = u_{i-1}(s)$ for all $s \in [0, i\Delta)$, and hence by causality of the input-output map υ_H, $o_i(s) = o_{i-1}(s)$ and $\hat{o}_i(s) = \hat{o}_{i-1}(s)$ for all $s \in [0, i\Delta)$. Define the input signal $u \in \mathcal{P}_{E_c}$ by $u(s) = u_i(s)$ for all $s \in [0, (i+1)\Delta)$, $i \in \mathbb{N}$. Consider $(o, \hat{o}) = \upsilon_H(u, d)$. By causality of υ_H we get that for all $i = 1, 2, \ldots, k$, $o(s) = o_i(s)$ and $\hat{o}(s) = \hat{o}_i(s)$ for all $s \in [0, i\Delta)$. Hence, we get that $o(((j-1)\Delta, j\Delta]) \cap E_o = o_j((j-1)\Delta, j\Delta]) \cap E_o = S_j$ for $j \in \mathbb{N}$, $j > 0$. From this, due to the definition of $\mathcal{C}_\phi$, it follows that $u = \mathcal{C}_\phi(o)$. Combining the conclusions above, we get that (4) holds. □

Proof of Proposition 2. We will show that R_H satisfies Conditions 1–4 of Definition 5. To this end, consider a sequence

$$s = (u_1, d_1)(u_2, d_2) \cdots (u_k, d_k) \in (U \times D)^* \tag{18}$$

where $k \geq 0$, $u_i \in U$, $d_i \in D$. We will construct maps $u \in \mathcal{P}_{E_c}$ and $g \in \mathcal{P}^{\Delta}_{E_d,\mu}$ such that

$$u(t) = \begin{cases} u_i & \text{if } t = (i-1)\Delta \text{ for some } i = 1, 2, \ldots, k \\ \perp & \text{otherwise} \end{cases} \quad \text{and}$$

$$\mathbf{UT}(g_i, \Delta) = d_i \text{ with } g_i(s) = \begin{cases} g(s + (i-1)\Delta) & \text{if } s > 0 \\ \perp & \text{otherwise} \end{cases} \quad \text{for all } i = 1, 2, \ldots, k \tag{19}$$

Existence of u is trivial. For any $d = e_1 e_2 \cdots e_l$, $l \leq \mu$ and $e_1, e_2, \ldots, e_l \in E_d$, define the map $g_d \in \mathcal{P}^{\Delta}_{E_d,\mu}$ as follows.

$$g_d(s) = \begin{cases} e_i & \text{if } s = i\frac{\Delta}{\mu+1} \text{ for some } i = 1, 2, \ldots, l \\ \perp & \text{otherwise} \end{cases} \tag{20}$$

It follows that $\mathbf{UT}(g_d, \Delta) = d$ and $g_d(0) = g_d(\Delta) = \perp$. The map g can then be defined as follows

$$g(s) = \begin{cases} g_{d_i}(s - (i-1)\Delta) & \text{if } s \in ((i-1)\Delta, i\Delta] \text{ for some } i = 1, 2, \ldots, k \\ \perp & \text{if } s > k\Delta \end{cases}$$

It follows that $g_i(s) = g_{d_i}(s)$ for all $s \in (0, \Delta]$ and hence $\mathbf{UT}(g_i, \Delta) = d_i$ for all $i = 1, 2, \ldots, k$.

Now we are ready to prove that Conditions 1–4 of Definition 5 hold.

Condition 1
From definition of R_H it follows that $R_H(\epsilon) = \{(\epsilon, \epsilon)\}$. Next, we show that $R_H(s)$ is a non-empty set for all $s \in (U \times D)^*$. To this end, assume that s is of the form (18) and consider maps $u \in \mathcal{P}_{E_c}$, $g \in \mathcal{P}^{\Delta}_{E_d,\mu}$ satisfying (19). Consider $(o, \hat{o}) = \upsilon_H(u, g)$ and define $\underline{\hat{o}} = \mathbf{UT}(\hat{o}, i\Delta)$, $o_i = o(((i-1)\Delta, i\Delta])$, $i = 1, 2, \ldots, k$. From the definition of R_H it follows that $(o_1 o_2 \ldots o_k, \underline{\hat{o}}) \in R_H(s)$. That is, $R_H(s)$ is a non-empty set.

Condition 2
Again, assume that s is of the form (18). Assume that $(\underline{o}, \underline{\hat{o}}) \in R_H(s)$. From the definition of $R_H(s)$ it then follows that there exist maps u and g satisfying (19) and such that $\underline{o} = o_1 o_2 \cdots o_k$ for some $o_1, o_2, \ldots, o_k \in O$. Hence, $|\underline{o}| = |s| = k$.

Condition 3
Assume that $s = \hat{s}a \in (U \times D)^*$ is of the form (18) with $k > 0$ and let $a = (u_k, d_k)$. It then follows that $\hat{s} = (u_1, d_1) \ldots (u_{k-1}, d_{k-1})$. Assume that $(\underline{o}, \underline{\hat{o}}) \in R_H(sa)$. Then there exists maps u and g which satisfy (19) and for which it holds that $(o, \hat{o}) = \upsilon_H(u, g)$, $\underline{\hat{o}} = \mathbf{UT}(\hat{o})$ and $\underline{o} = o_1 o_2 \ldots o_k$, $o_i = o((i-1)\Delta, i\Delta])$, $i = 1, 2, \ldots, k$. Let $\bar{u}(s) = u(s)$ for all $s \in [0, (k-1)\Delta)$ and $\bar{u}(s) = \perp$ otherwise. Consider $(y, x) = \upsilon_H(\hat{u}, g)$. By causality of υ_H we get that $y(s) = o(s)$ and $x(s) = \hat{o}(s)$ for all $s \in [0, (k-1)\Delta]$. Hence, $o_i = y((i-1)\Delta, i\Delta])$, $i = 1, 2, \ldots, k-1$ and $\underline{x} = \mathbf{UT}(x, (k-1)\Delta) = \mathbf{UT}(\hat{o}, (k-1)\Delta)$. Notice that $\mathbf{UT}(\hat{o}, k\Delta) = \mathbf{UT}(\hat{o}, (k-1)\Delta)\hat{x}$ for some $\hat{x} \in E_i^*$. Hence, we get that $\underline{\hat{o}} = \underline{x}\hat{x}$. Moreover, the definition of R_H implies that $(o_1 o_2 \cdots o_{k-1}, \underline{x}) \in R_H(\hat{s})$. That is, Condition 3 of Definition 5 holds.

Condition 4
Assume that s is of the form (18) and let $(\underline{o}, \underline{\hat{o}}) \in R_H(s)$. Assume that $a = (u_{k+1}, d_{k+1}) \in (U \times D)$. We will show that there exists $y_{k+1} \in O$ and $\hat{x}_{k+1} \in E_i^*$ such that $(\underline{o}y_{k+1}, \underline{\hat{o}}\hat{x}_{k+1}) \in R_H(sa)$. To this end, notice that there exist maps u and g satisfying (19) such that for $(o, \hat{o}) = \upsilon_H(u, g)$, $\underline{o} = o_1 o_2 \ldots o_k$, $\underline{\hat{o}} = \mathbf{UT}(\hat{o}, k\Delta)$ and $o_i = o((i-1)\Delta, i\Delta])$ for $i = 1, 2, \ldots, k$. Define the map $\hat{u} \in \mathcal{P}_{E_c}$ as follows; $\hat{u}(s) = u(s)$ if $s \in [0, k\Delta)$, $\hat{u}(k\Delta)) = u_{k+1}$ and $\hat{u}(s) = \perp$ otherwise. Recall the definition of the map g_d from (20) and apply it to $d = d_{k+1}$. Define the map $\hat{g}$ as follows;

$$\hat{g}(s) = \begin{cases} g(s) & \text{if } s \in [0, k\Delta] \\ g_{d_{k+1}}(s - k\Delta) & \text{if } s > k\Delta \end{cases}$$

Consider $(y,x) = \upsilon_H(\hat{u},\hat{g})$ and let $\underline{x} = \mathbf{UT}(x,(k+1)\Delta)$ and $\underline{y} = y_1y_2\cdots y_{k+1}$, $y_i = y(((i-1)\Delta, i\Delta])$, $i = 1,2,\dots,k+1$. From the definition of R_H it the follows that $(\underline{y},\underline{x}) \in R_H(sa)$. By causality of υ_H, $y(s) = o(s)$ and $x(s) = \hat{o}(s)$ for all $s \in [0,k\Delta]$. It then implies that $o_i = y(((i-1)\Delta, i\Delta])$, $i = 1,2,\dots,k$ and $\underline{x} = \mathbf{UT}(x,k\Delta) = \underline{\hat{o}}$. Moreover, from the definition of $\mathbf{UT}$ it follows that $\mathbf{UT}(x,(k+1)\Delta) = \mathbf{UT}(x,k\Delta)\hat{x}_{k+1}$. Hence, $\underline{y} = \underline{o}y_{k+1}$ and $\underline{x} = \underline{\hat{o}}\hat{x}_{k+1}$. □

Proof of Theorem 2. Consider any sequence $v \in L(H/\mathcal{C}_\phi)$ in the closed-loop language of the interconnection of H with the controller $\mathcal{C}_\phi$. We will show that v belongs to the closed-loop language $L(R_H/\phi)$ of the interconnection of the sequential input-output map R_H with the discrete controller ϕ.

To this end, notice that from $v \in L(H/\mathcal{C}_\phi)$ it follows that there exists a disturbance signal $d \in \mathcal{P}_{E_d,\mu}$, input signal $u \in \mathcal{P}_{E_c}$ and an output signal $o \in \mathcal{P}_{E_o}$, internal event signal $\hat{o} \in \mathcal{P}_{E_o}$ such that $v = \mathbf{UT}(\hat{o})$ and $(o,\hat{o}) = \upsilon_H(u,d)$ and $u = \mathcal{C}_\phi(o)$. Since $\mathcal{C}_\phi$ is a sampled-data controller, we get that $u = \mathcal{C}_\phi(o)$ if and only if there exist discrete input symbols $u_i \in U$, discrete outputs $o_i \in O$, $i = 1,2,\dots$ such that $u_i = \phi(o_1o_2\cdots o_{i-1})$ and $o_i = o(((i-1)\Delta, i\Delta]) \cap E_o$, and

$$u(t) = \begin{cases} u_j & \text{if } t = (j-1)\Delta \text{ for some } j = 1,2,\dots \\ \perp & \text{otherwise} \end{cases} \tag{21}$$

But from definition of R_H, we get that the sequences $o_1o_2\cdots o_i$ and $\mathbf{UT}(\hat{o}, i\Delta)$ are generated by the input-output relation R_H as a response to the sequence $u_1u_2\cdots u_i$ of input symbols and $d_1d_2\cdots d_i$ of disturbances, i.e.

$$(o_1o_2\cdots o_i, \hat{o}_i) \in R_H((u_1,d_1)\cdots(u_i,d_i)) \tag{22}$$

Here, $\hat{o}_i = \mathbf{UT}(\hat{o}, i\Delta)$, i.e. $\hat{o}_i$ is the sequence of internal events prescribed by the signal $\hat{o}$ on the interval $((i-1)\Delta, i\Delta]$ and $d_i = \mathbf{UT}(g_i,\Delta)$, where g_i is the shift of the disturbance signal d by $(i-1)\Delta$, i.e. $g_i(s) = d(s + (i-1)\Delta)$, for all $s \in \mathbb{R}_+$. In other words, d_i is the sequence of disturbances prescribed by d in the interval $((i-1)\Delta, i\Delta]$. Notice that $v_{1:k_i} = \hat{o}_i$, where $k_i = |\hat{o}_i|$ is the length of the word $\hat{o}_i$. That is, either $\lim k_i = +\infty$ and then v is simply the limit of the increasing sequence of words $\hat{o}_1, \hat{o}_2, \dots,$, or there exists N such that $v = \hat{o}_N = \hat{o}_j$ for all $j \geq N$. From this, (22) and $u_i = \phi(o_1o_2\cdots o_{i-1})$, $i = 1,2,\dots$ we conclude that $v \in L(R_H/\phi)$. □

Proof of Theorem 3. Consider any sequence $v \in L(R_H/\phi)$. Then there exist sequences of disturbance symbols $d_i \in D$, output symbols $o_i \in O$, input symbols $u_i \in U$, and sequences of internal events $\hat{o}_i \in E_i^*$, $i = 1,2,3,\dots$, such that $u_i = \phi(o_1o_2\cdots o_{i-1})$ and $(o_1o_2\cdots o_i, \hat{o}_i) \in R_H((u_1,d_1)(u_2,d_2)\cdots(u_i,d_i))$, and either there exists a finite $N \in \mathbb{N}$ such that $\hat{o}_i = v$ for all $i \geq N$, i.e. R_H stops producing internal events after N steps, or $v_{1:k_i} = \hat{o}_i$, where $k_i = |\hat{o}_i|$, i.e. R_H keeps on producing internal events and v is the concatenation of internal events. But by definition of abstraction we get that $(o_1o_2\cdots o_i, \hat{o}_i) \in R((u_1,d_1)(u_2,d_2)\cdots(u_i,d_i))$, i.e. $(o_1o_2\cdots o_i, \hat{o}_i)$ is also produced by the sequential input-output relation R as a response to disturbances $d_1, d_2, \dots, d_i$ and inputs $u_1, u_2, \dots, u_i$. Notice that the inputs $u_1, u_2, \dots, u_i$ represent the responses of the controller ϕ to the outputs $o_1, o_2, \dots, o_{i-1}$. Hence, v belongs to the closed-loop language of the feedback interconnection of R and ϕ, i.e. $v \in L(R/\phi)$. □

8.2 Well-posedness of H

Proof of Proposition 3. Consider the solution $z : \mathbb{R}_+ \to \mathbb{R}^n$ of the differential equation $\dot{z} = f_{q^c}(z)$ from an initial condition $z(0) = z_0$. Since f_{q^c} is continuous and globally Lipschitz, such a solution exists and it is unique. Consider the set

$$\mathbf{L} = \{t \in \mathbb{R}_+ \mid t > 0 \text{ and } \forall s \in [0,t) : z(s) \in \operatorname{int} \mathcal{X}\}$$

Since $z_0 \in \mathrm{int}\ \mathcal{X}$ and z is continuous, $z^{-1}(\mathrm{int}\ \mathcal{X})$ is open and contains 0. Hence, there exists $t > 0$ such that $[0, t) \subseteq z^{-1}(\mathrm{int}\ \mathcal{X})$, i.e. $t \in \mathbf{L}$. Hence, $\mathbf{L}$ is a non-empty set. Define $\beta(q^c, z_0) = \sup \mathbf{L}$.

If $\beta(q^c, z_0) = +\infty$, then for any $t \in \mathbb{R}_+$, there exists $t^{'} \in \mathbf{L}$ such that $t < t^{'}$ and hence $z(t) \in \mathrm{int}\ \mathcal{X}$. That is, z never leaves $\mathrm{int}\ \mathcal{X}$.

Assume now that $\beta(q^c, z_0) < +\infty$. Then $\beta(q^c, z_0) > 0$, due to the definition of $\mathbf{L}$. Assume now that $t < \beta(q^c, z_0)$. Then there exists $t^{'} \in \mathbf{L}$ such that $t^{'} > t$ and hence $z(t) \in \mathrm{int}\ \mathcal{X}$. That is, for all $t \in [0, \beta(q^c, z_0))$, $z(t) \in \mathrm{int}\ \mathcal{X}$.

Next, we show that $x(\beta(q^c, z_0)) \in \mathcal{X}$. Notice that there exists a monotonically increasing sequence $t_n \in \mathbf{L}$ such that $\lim_{n \to +\infty} t_n = \beta(q^c, z_0)$. Continuity of z implies that $z(\beta(q^c, z_0)) = \lim_{n \to +\infty} z(t_n)$. Since $z(t_n) \in \mathrm{int}\ \mathcal{X}$, we get that $z(\beta(q^c, z_0)) \in \mathcal{X}$.

We show that $z(\beta(q^c, z_0)) \notin \mathrm{int}\ \mathcal{X}$. Since $\partial\mathcal{X} = \mathcal{X} \setminus \mathrm{int}\ \mathcal{X}$, this shows that $z(\beta(q^c, z_0)) \in \partial\mathcal{X}$. Assume that $z(\beta(q^c, z_0)) \in \mathrm{int}\ \mathcal{X}$. Then $\beta(q^c, z_0) \in z^{-1}(\mathrm{int}\ \mathcal{X})$ and hence for some $\epsilon > 0$, $(\beta(q^c, z_0) - \epsilon, \beta(q^c, z_0) + \epsilon) \subseteq z^{-1}(\mathrm{int}\ \mathcal{X})$, due to the fact that $z^{-1}(\mathrm{int}\ \mathcal{X})$ is open. Hence, for all $s \in [0, \beta(q^c, z_0) + \epsilon) : z(s) \in \mathrm{int}\ \mathcal{X}$. This then implies that $\beta(q^c, z_0) < \beta(q^c, z_0) + \epsilon \in \mathbf{L}$, which is a contradiction. □

Proof of Proposition 4. Assume first that $s \geq \beta(q^c, x)$. Then $f^s_{q^c}(x) = f^{\beta(q^c,x)}_{q^c}(x) \in \partial X$ and hence $f^t_{q^c}(f^s_{q^c}(x)) = f^{\beta(q^c,x)}_{q^c}(x)$. But $t + s \geq s \geq \beta(q^c, x)$ and hence $f^{s+t}_{q^c}(x) = f^{\beta(q^c,x)}_{q_c}(x)$. That is, the required property holds.

Assume now that $s < \beta(q^c, x)$. If $s + t < \beta(q^c, x)$, then we get that $f^{s+t}_{q^c}(x)$, $f^s_{q^c}(x)$ and $f^t_{q^c}(f^s_{q^c}(x))$ are the solutions of the differential equation $\dot{z} = f_{q^c}(z)$ at time instances $s+t$, s, t and initial conditions x, x and $f^s_{q^c}(x)$ respectively. Then by the semigroup property of the solutions of differential equations we get that $f^{s+t}_{q^c}(x) = f^t_{q^c}(f^s_{q^c}(x))$. Assume that $s + t \geq \beta(q^c, x)$. Consider the solution of the differential equation $\dot{z} = f_{q^c}(z)$ from the initial state $z(0) = x$. It then follows that $z(\tau) \in \mathrm{int}\ \mathcal{X}$ for all $\tau \in [0, \beta(q^c, x))$. Hence, $f^{t+s}_{q^c}(x) = z(\beta(q^c, x))$. On the other hand, $f^s_{q^c}(x) = z(s) \in \mathrm{int}\ \mathcal{X}$. By the semi-group property of the solutions of differential equations, the solution $\dot{\hat{z}} = f_{q^c}(\hat{z})$ from the initial state $\hat{z}(0) = z(s)$ has the property that $\hat{z}(\tau) = z(s + \tau)$. In particular, $\hat{z}(\beta(q^c, x) - s) \in \partial\mathcal{X}$ and for all $\tau < \beta(q^c, x) - s$, $\hat{z}(\tau) \in \mathrm{int}\ \mathcal{X}$. Hence, by definition of $f^t_{q^c}$ we get that $f^t_{q^c}(z(s)) = \hat{z}(\beta(q^c, x) - s) = z(\beta(q^c, x))$. That is, $f^{t+s}_{q^c}(x) = f^t_{q^c}(f^s_{q^c}(x))$. □

Proof of Proposition 5. Assume that Assumption A3 holds. Consider the upper bound $N = \lceil |Q_q||E_i \cup E_o|(1 + T/\mathbb{T}) \rceil$ defined in the statement of Proposition 5. Consider any sequence $q_0, q_1, \ldots, q_k$ and $e_1, \ldots, e_k \in \Sigma$, $t_1 < t_2 < \ldots < t_k \in [0, T]$ such that $f^t_{q^c}(x) \in \Phi_{q^c, q^d_i, e_i}$ $i = 1, 2, \ldots, k$. For each discrete state $q^d \in Q_d$ and event $e \in \Sigma$, define the set

$$J_{q^d,e} = \{i \in \{1, 2, \ldots, k\} \mid q^d_i = q^d \text{ and } e_i = e\}$$

Assume that $J_{q^d,e} = \{i_1, i_k, \ldots, i_l\}$, $i_1 < i_2 < \ldots < i_l$ for some $l > 0$. From Assumption A3 it then follows that $t_{i_r} - t_{i_{r-1}} > \mathbb{T}$ for all $r = 1, 2, \ldots, l - 1$. Then

$$T \geq t_{i_l} - t_{i_1} = \sum_{r=1}^{l-1} (t_{i_r} - t_{i_{r-1}}) \geq |J_{q,e}|\mathbb{T}$$

That is, $|J_{q^d,e}| \leq T/\mathbb{T}$ holds for all $q^d \in Q_d$, $e \in \Sigma$. Consider the set

$$I = \bigcup_{q^d \in Q_d, e \in \Sigma, |J_{q^d,e}| > 1} J_{q^d,e}$$

The set I is exactly the set of those indices, for which a discrete state and event combination occurs more than once. It follows then that

$$|I| \leq \sum_{(q^d,e)\in Q_d\times\Sigma} |J_{q,e}| \leq \frac{T}{\mathbb{T}}|Q_d||\Sigma|$$

Notice that $k \leq |I| + |Q_d||E_i \cup E_o|$, since for each $i, j \in \{1, 2, \ldots, k\} \setminus I$, $(q^d_i, e_i) \neq (q^d_j, e_j)$ and there are at most $|Q_d||E_i \cup E_o|$ such elements. Taking into account that $|\Sigma| \leq |E_i \cup E_o|$, we get that

$$k \leq (T/\mathbb{T} + 1)|Q_d||E_i \cup E_o| = N$$

□

In order to prove Proposition 6 and Proposition 7 we need a number of technical results. These results state that hybrid systems which satisfy Assumption A1 – A4 generate finitely many internal and output events on each time interval.

Lemma 3 (Existence of a smallest time instances when an event is generated). For any state $q = (q^c, q^d) \in Q$, $x \in \mathcal{X}$, any $\Sigma \in \{E_i, E_o\}$ and any $T > 0$, the following holds. Either $f^{\tau}_{q^c}(x) \notin \bigcup_{e\in\Sigma} \Phi_{q,e}$ for all $\tau \in (0, T]$ or there exists a unique time instance $\tau_1 \in (0, T]$ and a unique event $e_1 \in \Sigma$ such that $f^{\tau_1}_{q^c}(x) \in \Phi_{q,e_1}$ and τ_1 is the smallest such, i.e. for all $s \in (0, \tau_1)$, $f^{s}_{q^c}(x) \notin \bigcup_{e\in\Sigma} \Phi_{q,e}$.

Proof of Lemma 3. Consider the set

$$\mathbf{SWT} = \{\tau \in (0, T] \mid f^{\tau}_{q^c}(x) \in \bigcup_{e\in\Sigma} \Phi_{q,e}\}$$

of als those time instances $\tau \in (0, T]$, for which the flow $f^{\tau}_{q^c}(x)$ crosses a guard. If $\mathbf{SWT}$ is empty, then for all $\tau \in (0, T]$, $f^{\tau}_{q^c}(x) \notin \bigcup_{e\in\Sigma} \Phi_{q,e}$, i.e. the conclusion of the lemma holds.

We will show that if $\mathbf{SWT}$ is non-empty, then the infimum $\tau_1 = \inf \mathbf{SWT}$ is in fact a minimum, i.e. $\tau_1 \in \mathbf{SWT}$ and for all $\tau_1 \neq \tau \in \mathbf{SWT}$, $\tau_1 < \tau$. This then shows that there exists $e_1 \in \Sigma$ such that $f^{\tau_1}_{q^c}(x) \in \Phi_{q,e_1}$ and for all $0 < \tau < \tau_1$, $\tau \notin \mathbf{SWT}$, i.e. $f^{\tau}_{q^c}(x) \notin \bigcup_{e\in\Sigma} \Phi_{q,e}$. Moreover, by Assumption A2 (disjointness of the guards), the choice of e_1 is unique.

We proceed to prove that $\tau_1 = \inf \mathbf{SWT} = \min \mathbf{SWT}$. Assume the contrary. Then there exists a monotonically strictly decreasing sequence $(s_n)_{n\in\mathbb{N}}$ such that $s_n \in \mathbf{SWT}$, $n \in \mathbb{N}$ and $\lim s_n = \tau_1$. Consider $\mathbb{T} = \min_{e\in\Sigma} \mathbb{T}(q, e)$, where $\mathbb{T}(q, e)$ is as defined in Assumption A3. It then follows that there exists $N_{\mathbb{T}} \in \mathbb{N}$ such that for all $n > N_{\mathbb{T}}$, $|s_n - \tau_1| < \mathbb{T}/2$, and hence $|s_k - s_l| < \mathbb{T}$ for all $k, l > N_{\mathbb{T}}$. It follows from the definition of $\mathbf{SWT}$ that for each $k \in \mathbb{N}$ there exists $e_k \in \Sigma$ such that $f^{s_k}_{q^c}(x) \in \Phi_{q,e_k}$. Notice that the choice of e_k is unique, due to disjointness of the guards according to Assumption A2. Let $k = N_{\mathbb{T}} + 1$. Then there exists $0 < l \leq |\Sigma|$ such that $e = e_k = e_{k+l}$. Since, $k, k + l > N_{\mathbb{T}}$, we get that $s_{k+l} - s_k < \mathbb{T} < \mathbb{T}(q, e)$. Notice that by definition of the sequence (s_n), $f^{s_k}_{q^c}(x) \in \Phi_{q,e}$ and $f^{s_{k+l}}_{q^c}(x) \in \Phi_{q,e}$. But by Assumption A3, $f^{s_k}_{q^c}(x) \in \Phi_{q,e}$ implies that for all $s \in \mathbb{R}_+$, $0 < s - s_k < \mathbb{T}(q, e)$, $f^{s}_{q^c}(x) \notin \Phi_{q,e}$, in particular, $f^{s_{k+l}}_{q^c}(x) \notin \Phi_{q,e}$. Hence we arrive to a contradiction. □

Lemma 4 (Finitely many events on a finite time interval). For any state $q = (q^c, q^d) \in Q$ and $x \in \mathcal{X}$ and any time duration $T > 0$, the following holds. Either $f^{\tau}_{q} \notin \bigcup_{e\in E_i} \Phi_{q,e}$ for all $\tau \in (0, T]$, or there exists a unique collection of time instances $0 = \tau_0 < \tau_1 < \tau_2 < \ldots < \tau_l \leq T$, events $e_1, e_2, \ldots, e_l \in E_i$, and discrete states $q^d = q^d_0, q^d_1, \ldots, q^d_l \in Q_d$ for some $l \in \mathbb{N}$, $l > 0$, such that for all $j = 1, 2, \ldots, l$,

$$\begin{aligned}
\delta_d((q^c, q^d_{j-1}), e_j) &= q^d_j & (23)\\
\forall s \in (\tau_{j-1}, \tau_j) &: f^s_{q^c}(x) \notin \bigcup_{e \in E_i} \Phi_{q^c, q^d_{j-1}, e} \text{ and } f^{\tau_j}_{q^c} \in \Phi_{q^c, q^d_{j-1}, e_j} & (24)\\
\forall s \in (\tau_l, T] &: f^s_{q^c}(x) \notin \bigcup_{e \in E_i} \Phi_{q^c, q^d_l, e} & (25)
\end{aligned}$$

Proof of Lemma 4. Below we present procedure for constructing a unique collection of sequences $e_1, e_2, \ldots, e_l \in E_i$, $q^d = q^d_0, q^d_1, \ldots, q^d_l \in Q_d$, $\tau_1 < \tau_2 < \ldots < \tau_l \in (0, T]$ satisfying (23–24). The procedure is recursive in l. We will show that after at most $\mathbb{T}(q, E_i, T)$ steps the procedure stops, yielding a unique collection of sequences $e_1, e_2, \ldots, e_l \in E_i$, $q^d = q^d_0, q^d_1, \ldots, q^d_l \in Q_d$, $\tau_1 < \tau_2 < \ldots < \tau_l \in (0, T]$, which satisfy (23), (24) and (25).

For $l = 0$, let $q^d_0 = q^d$. By applying Lemma 3, we get that either for all $\tau \in (0, T]$, $f^\tau_{q^c}(x) \notin \bigcup_{e \in E_i} \Phi_{q,e}$, or there exists a unique choice $\tau_1 \in (0, T]$, $e_1 \in E_i$ such that $f^{\tau_1}_{q^c}(x) \in \Phi_{q,e_1}$ and for all $0 < \tau < \tau_1$, $f^\tau_{q^c}(x) \notin \bigcup_{e \in E_i} \Phi_{q,e}$. If we set $q_1 = \delta((q^c, q^d_0), e_1)$, then we get that q^d_0, q^d_1, τ_1, e_1 satisfy (23–24) for $l = 1$.

Assume that for some l, we have constructed sequences $q^d = q^d_0, q^d_1, \ldots, q^d_l \in Q_d, e_1, e_2, \ldots, e_l \in E_i$, and $\tau_1 < \tau_2 < \ldots < \tau_l \in (0, T]$ such that (23–24) hold. By applying Lemma 3 to the discrete state $q_l = (q^c, q^d_l)$ and continuous state $x_l = f^{\tau_l}_{q^c}(x)$, and time duration $T - \tau_l$, we get that either for all $s \in (0, T - \tau_l]$, $f^s_{q^c}(x_l) \notin \bigcup_{e \in E_i} \Phi_{q_l,e}$, or there exists a unique time instance $t \in (0, T - \tau_l]$ and event $z \in E_i$ such that $f^t_{q^c}(x_l) \in \Phi_{q_l,z}$ and for all $s \in (0, t)$, $f^s_{q^c}(x_l) \notin \bigcup_{e \in E_i} \Phi_{q_l,e}$. In the former case, by using the fact that $f^s_{q^c}(x_l) = f^{\tau_l+s}_{q^c}(x)$, we get that (25) holds. In the latter case, by setting $\tau_{l+1} = t + \tau_l$, $e_{l+1} = z$, $q^d_{l+1} = \delta_d((q^c, q^d_l), z)$, we get that the time instances $\tau_1, \tau_2, \ldots, \tau_{l+1} \in (0, T]$, discrete states $q^d_0 = q^d, q^d_1, q^d_2, \ldots, q^d_{l+1}$, and events $e_1, e_2, \ldots, e_{l+1} \in E_i$ satisfy (23–24).

Thus the procedure above either stops with a collection of sequences $\tau_1 < \tau_2 < \ldots < \tau_l \in (0, T]$, $q^d = q^d_0, q^d_1, \ldots, q^d_l \in Q_q$, and $e_1, e_2, \ldots, e_l \in E_i$ which satisfy (23–25), or it produces infinite sequences $\tau_1 < \tau_2 < \ldots \in (0, T]$, $q^d = q^d_0, q^d_1, \ldots \in Q_d$, $e_1, e_2, \ldots \in E_i$, such that for each $l = 1, 2, \ldots$, (23–24) holds. In particular, we get that $f^{\tau_l}_{q^c}(x) \in \Phi_{q^c, q^d_{l-1}, e_l}$ for all $l = 1, 2, \ldots,$. But from Assumption A4 it follows that this can hold only for a *finite* sequence of time instances $\tau_1, \tau_2, \ldots, \tau_N$ of length at most $\mathbb{T}(q, E_i, T)$. Hence we arrived to a contradiction. It means that after at most $\mathbb{T}(q, E_i, T)$ steps, the procedure above stops, yielding a sequence of time instances, discrete states and internal events satisfying (23–25). This collection of sequences is unique, as the first element of each sequence is fixed, and at each step, the procedure above yields a unique extension of the sequences at hand.

□

Using the lemmas above, we are ready to present the proof of Proposition 6.

Proof Proposition 6. Define the states $\hat{q}_h = (\hat{q}^c_h, \hat{q}^d_h)$ and $\hat{x}_h$ as follows.

$$\hat{q}^d_h = q^d_h$$

$$\hat{q}^c_h = \begin{cases} \delta_c(q_h, u(0)) & \text{if } u(0) \in E_c \\ q^c_h & \text{otherwise} \end{cases}$$

$$\hat{x}_h = \begin{cases} R_{u(0), q_h}(x_h) & \text{if } u(0) \in E_c \\ x_h & \text{otherwise} \end{cases}$$

In order to prove the proposition, we need to show that the following is true.

Claim 0
There exist unique maps $q : \mathbb{R}_+ \to Q$ and $x : \mathbb{R}_+ \to \mathcal{X}$, such that $q(0) = \hat{q}_h$, $x(0) = \hat{x}_h$ and (q, x) satisfies conditions 2–4.

Indeed, assume that **Claim 0** is true. Then set $\xi_H(h, u, g)(t) = (q(t), x(t))$ for all $t \in \mathbb{R}_+$. Notice that then (q, x) also satisfies conditions 1–4 of Definition 24. In addition, if $(\hat{q}, \hat{x})$ is a pair of maps which satisfy conditions 1–4 of Definition 24, then $(\hat{q}, \hat{x})(0) = (\hat{q}_h, \hat{x}_h)$ and $(\hat{q}, \hat{x})$ satisfies conditions 2–4 of Definition 24 and hence by **Claim 0**, $(\hat{q}, \hat{x}) = (q, x)$. That is, Definition 24 defines $\xi_H(h, u, g)$ uniquely.

Proof of Claim 0
To this end, assume that t_{i+1}, $i \in \mathbb{N}$, $i < K$ are the positive switching times of u and d; i.e. $0 < t_1 < t_2 < \ldots$ such that for all $t \in \mathbb{R}_+$, $t > 0$, $u(t) \in E_c$ or $d(t) \in E_d$ if and only if $t \in \{t_{i+1} \mid i \in \mathbb{N}, i < K\}$. Here either $K = +\infty$ or $K \in \mathbb{N}$. If $K \in \mathbb{N}$, then for all $i \geq K$, let $t_i = t_{K-1} + i$. Let $\underline{K} = \{0, 1, 2 \ldots, K-1\}$ if $K < +\infty$ or $\underline{K} = \mathbb{N}$, if $K = +\infty$.

Consider the interval $\mathcal{T}_i = [t_i, t_{i+1}] \cap \mathbb{R}_+$, $i \in \underline{K}$, where $t_0 = 0$. Consider an arbitrary state $h \in S_H$. We will construct pair of maps $(q_i(h), x_i(h)) : \mathcal{T}_i \to S_H$ such that $(q_i(h)(t_i), x_i(h)(t_i)) = h$ and the following holds.

1. **Claim 1**
   Let $h_0 = (\hat{q}_h, \hat{x}_h)$ as in Definition 24 and for each $i \in \mathbb{N}$ let
   $$h_{i+1} = (q_i(h_i)(t_{i+1}), x_i(h_i)(t_{i+1}))$$
   Define the pair of maps $(q, x) : \mathbb{R}_+ \to S_H$ by $(q(t), x(t)) = (q_i(h_i)(t), x_i(h_i)(t))$ if $t \in T_i$ for some $i \in \mathbb{N}$. Then $(q(t), x(t))$ satisfies conditions 2–4 of Definition 24.

2. **Claim 2**
   If $(\hat{q}, \hat{x}) : \mathbb{R}_+ \to S_H$ is a pair of maps satisfying the conditions of Definition 24 and $\hat{q}(t_i) = q_i(h)(t_i)$, $\hat{x}(t_i) = x_i(h)(t_i)$, then the restriction of $(\hat{q}, \hat{x})$ to the interval $\mathcal{T}_i$ equals $(q_i(h), x_i(h))$.

Assume for a moment that the two statements above are true. Then by Claim 1 (q, x) satisfies conditions 2–4 of Definition 24. Moreover, if $(\hat{q}, \hat{x})$ also satisfies conditions 2–4 of Definition 24 then Claim 2 implies that $(q, x) = (\hat{q}, \hat{x})$. Indeed, by induction on i we can show that the restriction of (q, x) to $\mathcal{T}_i$ equals the restriction of $(\hat{q}, \hat{x})$ to $\mathcal{T}_i$. Since $\bigcup_{i \in \mathbb{N}} \mathcal{T}_i = \mathbb{R}_+$, this then implies that $(q, x) = (\hat{q}, \hat{x})$.

We conclude the proof by defining the functions $(q_i(h), x_i(h))$ and by showing that Claim 1 and Claim 2 hold.

Definition of $(q_i(h), x_i(h))$ for $i \in \mathbb{N}$
First, we define $(q_i(h), x_i(h))$. By applying Lemma 4 to the state $h = (q_h, x_h)$, $q_h = (q_h^c, q_h^d) \in Q$, and interval $(0, t_{i+1} - t_i]$ it follows that either $f_{q_h^c}^{t-t_i}(x_h) \notin \bigcup_{e \in E_i} \Phi_{q_h, e}$ for all $t \in (t_i, t_{i+1}]$ or there exists an integer $l(i, h) > 0$ events $e_1(i, h), e_2(i, h), \ldots, e_{l(i,h)}(i, h) \in E_i$, and time instances

$$t_i = \tau_0(i, h) < \tau_1(i, h) < \tau_2(i, h) < \ldots < \tau_{l(i,h)}(i, h) \leq \tau_{l(i,h)+1}(i, h) = t_{i+1} \qquad (26)$$

and discrete states $q_0^d(i, h) = q_h^d, q_1^d(i, h), \ldots, q_{l(i,h)}^d(i, h) \in Q_d$ such that for all $j = 1, 2, \ldots, l(i, h)$, the following holds.

$$\begin{aligned} \delta_d((q_h^c, q_{j-1}^d(i,h)), e_j(i,h)) &= q_j^d(i,h) && (27) \\ \forall s \in (\tau_{j-1}(i,h), \tau_j(i,h)) &: f_{q_h^c}^{s-t_i}(x_h) \notin \bigcup_{e \in E_i} \Phi_{q_h^c, q_{j-1}^d(i,h), e} \text{ and} && (28) \\ & f_{q_h^c}^{\tau_j(i,h)-t_i}(x_h) \in \Phi_{q_h^c, q_{j-1}^d(i,h), e_j(i,h)} && (29) \\ \forall s \in (\tau_{l(i,h)}(i,h), t_{i+1}] &: f_{q_h^c}^{s-t_i}(x_h) \notin \bigcup_{e \in E_i} \Phi_{q_h^c, q_{l(i,h)}^d(i,h), e} && (30) \end{aligned}$$

We then define the maps $q_i(h) = (q_i^c(h), q_i^d(h) : \mathcal{T}_i \to Q_c$, and $x_i(h) : \mathcal{T}_i \to \mathcal{X}$ as follows.

$$q_i^d(t) = \begin{cases} q_{j-1}^d(i,h) & \text{if } t \in [\tau_{j-1}(i,h), \tau_j(i,h)) \text{ for some } j = 1,2,\ldots,l(i,h) \\ \delta_d(q_h^c, q_{l(i,h)}^d(i,h), z) & \text{if } t = t_{i+1} \text{ and } d(t_i) = z \in E_d \\ q_{l(i,h)}^d(i,h) & \text{if } t \in [\tau_{l(i,h)}(i,h), t_{i+1}) \text{ and } \tau_{l(i,h)}(i,h) < t_{i+1} \\ q_{l(i,h)}^d(i,h) & \text{if } t = \tau_{l(i,h)}(i,h) = t_{i+1} \text{ and } d(t_{i+1}) = \perp \end{cases} \qquad (31)$$

$$q_i^c(t) = \begin{cases} q_h^c & \text{if } t < t_{i+1} \\ \delta_c(q_h^c, q_{l(i,h)}^d(i,h), u(t_{i+1})) & \text{if } t = t_{i+1} \text{ and } \tau_{l(i,h)}(i,h) < t_{i+1} \\ \delta_c(q_h^c, q_{l(i,h)-1}^d(i,h), u(t_{i+1})) & \text{if } t = t_{i+1} \text{ and } \tau_{l(i,h)}(i,h) = t_{i+1} \end{cases} \qquad (32)$$

$$x_i(t) = \begin{cases} f_{q_h^c}^{t-t_i}(x_h) & \text{if } t < t_{i+1} \\ R_{u(t_{i+1}), q_h^c, q_{l(i,h)}^d(i,h)}(f_{q_h^c}^{t_{i+1}-t_i}(x_h)) & \text{if } t = t_{i+1} \text{ and } \tau_{l(i,h)}(i,h) < t_{i+1} \\ R_{u(t_{i+1}), q_h^c, q_{l(i,h)-1}^d(i,h)}(f_{q_h^c}^{t_{i+1}-t_i}(x_h)) & \text{if } t = t_{i+1} \text{ and } \tau_{l(i,h)}(i,h) = t_{i+1} \end{cases} \qquad (33)$$

Here the notation $\delta_c(s^c, s^d, \perp) = s^c$, $R_{\perp,s}(x) = x$, and $\delta_d(s^c, s^d, \perp) = s^d$ is used for all $s = (s^c, s^d) \in Q$.

Proof of Claim 1

We will show that (q, x) satisfies Part 2 – 4 of Definition 24. To this end, consider any $t \in \mathbb{R}_+$, $t > 0$ and notice that

$$\begin{aligned} & q(t) = q_i(h_i)(t) \text{ and } x(t) = x_i(h_i)(t) \text{ for all } t \in \mathcal{T}_i \\ & h_{i+1} = (q_i(h_i)(t_{i+1}), x_i(h_i)(t_{i+1})) = (q(t_{i+1}), x(t_{i+1})) && (34) \end{aligned}$$

It then implies that if $t \in (t_i, t_{i+1}]$ for some $i \in \mathbb{N}$, then

$$\begin{aligned} & x(t^-) = f_{q^c(t_i)}^{t-t_i}(x(t_i)) \\ & x(t) = f_{q^c(t_i)}^{t-t_i}(x(t_i)) \text{ if } t < t_{i+1} \\ & q^d(t) = q_{j-1}^d(i,h_i) \text{ if } t \in [\tau_{j-1}(i,h_i), \tau_j(i,h_i)) \text{ for some } j = 1,2,\ldots,l(i,h_i) \\ & q(t^-) = (q^c(t_i), q_{j-1}^d(i,h_i)) \text{ if } t \in (\tau_{j-1}(i,h_i), \tau_j(i,h)] \text{ for some } j = 1,2,\ldots,l(i,h_i) && (35) \end{aligned}$$

We investigate three different cases, corresponding to Part 2, Part 3 and Part 4 of Definition 24 respectively. To this end notice that

1. Assume that for some $t > r > 0$, for all $s \in [t-r, t)$, $u(s) = \perp$. Without loss of generality we can assume that r is small enough so that $t-r, t \in (t_i, t_{i+1}]$ for some $i \in \mathbb{N}$. We then get that $q^c(s) = q_i^c(s) = q_i^c(h_i)(t_i) = q^c(t_i)$ for all $s \in [t-r, t)$. If $u(t) = u \in E_c$, then $t = t_{i+1}$ and from (32) we get that $q_i^c(h_i)(t_{i+1}) = \delta_c(q_i(h_i)(t_i), q_j^d(i,h_i), u(t_{i+1}))$ for some $j \in \{l(i,h), l(i,h)-1\}$. Combining this with (35) we get that $q^c(t) = \delta_c(q(t^-), u(t))$. From (33) and (35) we get that $x(t) = x_i(h_i)(t_{i+1}) = R_{u,q(t^-)}(x(t^-))$. If $u(t) = \perp$, then from definition of $q_i(h_i)$ and the discussion above it follows that $q_i^c(h_i)(t) = q_i^c(h_i)(t_i) = q^c(t-r)$. That is, (q, x) satisfies Part 2 of Definition 2.

2. Assume that for some $t > r > 0$, for all $s \in (t-r, t]$, $u(s) = \perp$. Without loss of generality we can assume that r small enough so that $t - r \in [t_i, t_{i+1})$ and $t \in (t_i, t_{i+1}])$ for some

$i \in \mathbb{N}$. From (35) it follows that $q^c(t-r) = q^c(t_i)$. The semi-group property of the flow implies that $f^{t-t_i}_{q^c(t-r)}(x(t_i)) = f^{r}_{q^c(t-r)}(f^{t-r-t_i}_{q^c(t-r)}(x(t_i))$. Hence, by (35) we get that $x(t) = f^{r}_{q^c(t-r)}(x(t-r))$. That is (q,x) satisfies Part 3 of Definition 2.

3. Assume that for some $t > r > 0$ it holds that for all $s \in (t-r,t)$, $u(s) = \perp$ and $d(s) = \perp$ and $x(s) \notin \bigcup_{e \in E_i} \Phi_{q(t-r),e}$. It then follows that $t-r, t \in \mathcal{T}_i$ for some $i \in \mathbb{N}$. Consider now the time instances $\tau_1(i,h_i), \tau_2(i,h_i), \ldots, \tau_{l(i,h)}(i,h), \tau_{l(i,h)+1}$ from (26) with $h = h_i$. For simplicity we shall denote $\tau_j(i,h_i)$ by τ_j and $l(i,h)$ by l in the sequel. We then get that there exists $j \in \{1,2,\ldots,l+1\}$ such that $t-r, t \in (\tau_{j-1}, \tau_j]$. From (35) it follows that $q^d(s) = q^d(t-r)$ for all $s \in [t-r,t)$. Assume that $d(t) = \perp$. If $x(t^-) \in \Phi_{q(t^-),e}$ for some $e \in E_i$, then from (35) and (28)–(29) it follows that $t = \tau_j$, $e = e_j(i,h_i)$ and $q^d(t) = q^d_j(i,h_i)$. Combining this with (27) and (35) we get that that $q^d(t) = \delta_d(q(t^-),e)$. If $x(t^-) \notin \bigcup_{e \in E_i} \Phi_{q(t^-),e}$, then by an argument simillar to the previous one we get that $t < \tau_j$ and $q^d(t) = q^d_{j-1}(i,h_i) = q^d(t-r)$. Finally, if $d(t) \in E_d$, then from the definition of t_i, t_{i+1} we get that $t = t_{i+1} = \tau_{l+1}$. Hence, $j \in \{l, l+1\}$ and $q_d(t) = q^d_i(h_i)(t_{i+1})$. From the definition of $q^d_i(h_i)(t_{i+1})$, using the fact that $q(t^-) = (q^c(t_i), q^d_{j-1})$, we get that $q_d(t) = \delta_d(q(t^-), d(t))$. That is, (q,x) satisfies Part 4 of Definition 24.

Proof of Claim 2
Assume now that $(\hat{q},\hat{x}) : \mathbb{R}_+ \to S_H$ satisfies the conditions of Definition 24 and $\hat{q}(t_i) = q_i(h)(t_i)$, $\hat{x}(t_i) = x_i(h)(t_i)$. The proof of Claim 2 is done in two steps.

1. **Claim 2.1** $(\hat{q},\hat{x})(s) = (q_i(h)(s), x_i(h)(s))$ for all $s \in [t_i, t_{i+1})$.
2. **Claim 2.2** $\hat{q}(t_{i+1}) = q_i(h)(t_{i+1})$ and $\hat{x}(t_{i+1}) = x_i(h)(t_{i+1})$.

Since $\mathcal{T}_i = [t_i, t_{i+1}]$, Claim 2.1 and Claim 2.2 indeed imply Claim 2.

Proof of Claim 2.1
Since on the interval (t_i, t_{i+1}) no input or disturbance event occurs, i.e. $u(t) = \perp$ and $d(t) = \perp$ for all $t \in (t_i, t_{i+1})$, by Part 2 and Part 3 of Definition 24 we get that

$$\hat{x}(t) = f^{t-t_i}_{q^c_i(t_i)}(x(t_i)) = x_i(h)(t) \text{ and } \hat{q}^c(t) = q^c_i(h)(t_i) \text{ for all } t \in [t_i, t_{i+1}) \tag{36}$$

Finally, we will argue that $q^d_i(h)(t) = \hat{q}^d(t)$ for all $t \in (t_i, t_{i+1})$ holds. To this end, consider the time instances $\tau_0(i,h), \tau_1(i,h), \ldots, \tau_{l(i,h)}(i,h)$ from (26). For the sake of simplicity we will denote $\tau_j(i,h)$ by τ_j for all $j = 0,1,\ldots l(i,h)$ and we will denote $l(i,h)$ by l. We will show by induction on j that $\hat{q}^d|_{[\tau_{j-1},\tau_j)} = q^d_i(h)|_{[\tau_{j-1},\tau_j)} = q^d_{j-1}(i,h)$ for all $j = 1,2,\ldots,l+1$.

Consider $j = 1$. By assumption we have that $\hat{q}^d(\tau_0) = q^d_0(i,h_i)$. From (36) and (29–30), it follows $\hat{x}(s) \notin \bigcup_{e \in E_i} \Phi_{\hat{q}(t_i),e}$ for all $s \in [t_i, \tau_1)$. From Definition 24, Part 4, the latter implies that $\hat{q}^d(s) = \hat{q}^d(t_i) = q^d(t_i) = q^d_0$ for all $s \in [t_i, \tau_1)$. Hence, the induction hypothesis holds for $j = 1$. Assume that $\hat{q}^d|_{[\tau_{j-1},\tau_j)} = q^d_{j-1}(i,h_i)$ for all $j = 1,2,\ldots,r$, $r < l$. From (36) and (29–30) it follows that $\hat{x}(\tau_r) \in \Phi_{\hat{q}(\tau_{r-1}),e_r}$, and for all $s \in (\tau_{r-1}, \tau_r)$, $\hat{x}(s) \notin \bigcup_{e \in E_i} \Phi_{\hat{q}(\tau_{r-1}),e}$. Hence, by Part 4 we get that that $\hat{q}^d(\tau_r) = \delta_d(\hat{q}(\tau_{r-1}), e_r)$. By induction hypothesis, $\hat{q}^d(\tau_{r-1}) = q^d_{r-1}(i,h)$, and thus from (27) we get that $\hat{q}^d(\tau_r) = q^d_r$.

Proof of Claim 2.2
Notice that Claim 2.1 implies that

$$\hat{q}(t^-_{i+1}) = \begin{cases} (q^c_i(h)(t_i), q^d_l) & \text{if } \tau_l < t_{i+1} \\ (q^c_i(h)(t_i), q^d_{l-1}) & \text{if } \tau_l = t_{i+1} \end{cases} \tag{37}$$

$$\hat{x}(t^-_{i+1}) = \lim_{s \uparrow t_{i+1}} f^{s-t_i}_{q^c_i(t_i)}(x_i(t_i)) = f^{t_{i+1}-t_i}_{q^c_i(t_i)}(x(t_i)) \tag{38}$$

First, we show that $\hat{q}^c(t_{i+1}) = q_i^c(h)(t_{i+1})$. From Part 2 of Definition 24, it follows that

$$\hat{q}^c(t_{i+1}) = \delta_c(\hat{q}(t_{i+1}^-), u(t_{i+1}))$$

From this, using (37) and the definition of $q_i^c(h)(t_{i+1})$, we get that $q_i^c(h)(t_{i+1}) = \hat{q}^c(t_{i+1})$. Next, we show that $\hat{x}(t_{i+1}) = x_i(h)(t_{i+1})$. By Part 2 of Definition 24,

$$\hat{x}(t_{i+1}) = R_{u(t_{i+1}),\hat{q}(t_{i+1}^-)}(\hat{x}(t_{i+1}^-))$$

Combining this with (38–37), and the definition of $x_i(h)(t_{i+1})$, we get that $x_i(h)(t_{i+1}) = \hat{x}(t_{i+1})$.

Finally, we show that $\hat{q}_d(t_{i+1}) = q_i^d(h)(t_{i+1})$. Three cases need to be distinguished. First, if $d(t_{i+1}) \in E_d$, then by Part 2 of Definition 24, $\hat{q}_d(t_{i+1}) = \delta_d(\hat{q}^d(t_{i+1}^-), d(t_{i+1}))$. Combining this, (37) and the definition of $q_i^d(h)(t_{i+1})$ we get that in this case $\hat{q}_d(t_{i+1}) = q_i^d(h)(t_{i+1})$. Similarly, if $d(t_{i+1}) = \perp$ and $\tau_{l(i,h)} < t_{i+1}$, then from Part 4 of Definition 24 and (38) it follows that $\hat{q}^d(t_{i+1}) = q_i^d(h)(t_{i+1})$. Finally, if $d(t_{i+1}) = \perp$ and $\tau l(i,h) = t_{i+1}$, then $\hat{x}(t_{i+1}) \in \Phi_{\hat{q}(t_{i+1}^-),e_l}$ and hence by Part 4 of Definition 24 $\hat{q}^d(t_{i+1}) = \delta_d(\hat{q}(t_{i+1}^-), e_l)$. Combining this with (37) and definition of $q_i^d(h)(t_{i+1})$, we get that $\hat{q}^d(t_{i+1}) = q_i^d(h)(t_{i+1})$.

□

The proof of Proposition 6 yields the following corollary which is interesting in its own right.

Corollary 1. Let $t_i \in \mathbb{R}_+$, $i \in \mathbb{N}$, $0 < i < K$, $K \in \mathbb{N} \cup \{+\infty\}$ be the increasing sequence of switching times of u and d, i.e. $0 = t_0 \le t_1 < t_2 < \cdots < t_k < \cdots$, and for $u(t) \in E_c$ or $d(t) \in E_d$ if and only if $t \in \{t_i \mid i \in \mathbb{N}, 0 < i < K\}$. Let $\underline{K} = \{i \in \mathbb{N} \mid i < K\}$. If $K \in \mathbb{N}$, then for all $i \ge K$, $i \in \mathbb{N}$, let $t_i = t_{K-1} + i$. Then there exists discrete state $q_i^c \in Q_c$, integers $l(i) \in \mathbb{N}$, time instances $t_i = \tau(i,0) < \tau(i,1) < \ldots < \tau(i,l(i)) \le t_{i+1}$, internal events $e(i,1), \ldots, e(i,l(i)) \in E_i$ and discrete states $q^d(i,j) \in Q_d$, $j = 0,1,\ldots,l(i)+1$ and continuous states $x_i \in \mathbb{R}_+$, $i \in \mathbb{N}$ such that

$$\xi_H(h_0,u,d)(t) = (q_i^c, q^d(i,j-1), f_{q_i^c}^{t-t_i}(x_i))$$

if $t \in [\tau(i,j-1), \tau(i,j))$ for some $i \in \mathbb{N}, j = 1,2,\ldots l(i)$. Here $h_0 = (q_0^c, q^d(0,0), x_0)$, and for all $i \in \mathbb{N}$, $j = 0,1,\ldots l(i)-1$,

$$\begin{aligned}
&q_{i+1} = \begin{cases} \delta_c(q_i^c, q^d(i,l(i)), u(t_{i+1})) & \text{if } \tau(i,l(i)) < t_{i+1} \\ \delta_c(q_i^c, q^d(i,l(i)-1), u(t_{i+1})) & \text{if } \tau(i,l(i)) = t_{i+1} \end{cases} \\
&x_{i+1} = \begin{cases} R_{u(t_{i+1}),q_i^c,q^d(i,l(i))}(f_{q_i^c}^{t_{i+1}-t_i}(x_i)) & \text{if } \tau(i,l(i)) < t_{i+1} \\ R_{u(t_{i+1}),q_i^c,q^d(i,l(i)-1)}(f_{q_i^c}^{t_{i+1}-t_i}(x_i)) & \text{if } \tau(i,l(i)) = t_{i+1} \end{cases} \\
&q^d(i,j+1) = \delta_d((q_i^c, q^d(i,j-1), e(i,j)) \text{ if } j+1 \le l(i) \\
&q^d(i+1,0) = \begin{cases} \delta_d(q_i^c, q^d(i,l(i)), d(t_{i+1})) & \text{if } d(t_{i+1}) \in E_d \text{ and } \tau(i,l(i)) < t_{i+1} \\ \delta_d(q_i^c, q^d(i,l(i)-1), d(t_{i+1})) & \text{if } d(t_{i+1}) \in E_d \text{ and } \tau(i,l(i)) = t_{i+1} \\ q^d(i,l(i)) & \text{otherwise} \end{cases} \\
&f_{q_i^c}^{t-t_i}(x_i) \notin \bigcup_{e \in E_i} \Phi_{q_i^c,q^d(i,j-1),e} \text{ if } t \in (\tau(i,j-1), \tau(i,j)) \\
&f_{q_i^c}^{\tau(i,j)-t_i}(x_i) \in \Phi_{q^c,q^d(i,j-1),e(i,j)}
\end{aligned} \tag{39}$$

Moreover, for every $t \in \mathbb{R}_+$, $t > 0$ there exists an index $i \in \mathbb{N}$ and $j = 1,2,\ldots,l(i)+1$ such that $t \in [\tau(i,j-1), \tau(i,j))$.

In other words, there exists a sequence of time instances on which either an input or disturbance event occurs ((t_i)) or an internal event is generated ($\tau(i,j)$). Moreover, this sequence of time instances has no accumulation points and it has the property that no events occurs at time instances other than those belonging to this sequence. This sequence of time instances and events induces a sequence of instantenous state transitions. Each state transition is either triggered by an input or disturbance event, or by an internal event.

Proof of Proposition 7. It is easy to see that (7) determines the maps $o : \mathbb{R}_+ \to E_o \cup \{\perp\}$ and $\hat{o} : \mathbb{R}_+ \to E_i \cup \{\perp\}$ uniquely. Causality of $\upsilon_{H,h}$ follows directly from the fact that the value of o and $\hat{o}$ at time t depends only on the state $(q(t^-), x(t^-))$ of H at time t and the values of $u(t)$ and $d(t)$. Moreover, the state $(q(t^-), x(t^-))$ of H is evidently independent of the values of u and d at time instances later than t.

In order to show that $\upsilon_{H,h}$ is of the required form, we need to show that $\hat{o}$ and o are time event functions. More precisely, we need to show that there exist monotincally increasing sequences of time instances $t_i^{int}, t_i^o \in \mathbb{R}_+$, $i \in \mathbb{N}$, such that $\hat{o}(t) \in E_i$ if and only if $t = t_i^{int}$ for some $i \in \mathbb{N}$, and $o(t) \in E_o$ if and only if $t = t_i^o$ for some $i \in \mathbb{N}$.

We begin with $\hat{o}$, i.e. with proving the existence of a sequence t_i^{int}, $i \in \mathbb{N}$ such that $\hat{o}(t) \in E_i$ if and only if $t = t_i$ for some $i \in \mathbb{N}$. To this end, consider the time instances $\tau(i,j)$, $i \in \mathbb{N}$, $j = 1, 2, \ldots, l(i)$ defined in Corollary 1. Consider the set

$$\mathbf{SW} = \{\tau(i,j) \mid i \in \mathbb{N}, j = 1, 2, \ldots, l(i)\} \cup \{t_i \mid i \in \mathbb{N}, d(t_i) \in E_d, t_i > 0\}$$

That is, **SW** consists of the elements of the sequence $\tau(i,j)$, $i \in \mathbb{N}$, $j = 1, 2, \ldots, l(i)$ and of those time instances t_i, $i \in \mathbb{N}$, for which $d(t_i) \in E_d$. Define the sequence t_i^{int} as the listing of the elements of **SW** in increasing order. Corollary 1 implies that if $d(t) \in E_d$ then $t = t_i$ for some $i \in \mathbb{N}$. In addition, Corollary 1 implies that if $\xi_H(h_0, u, d)(t) = (q(t), x(t))$, $t > 0$, $d(t) = \perp$, then $x(t^-) \in \Phi_{q(t^-),e}$ for some $e \in E_i$ if and only if $t = \tau(i,j)$ for some $i \in \mathbb{N}$, $j = 1, 2, \ldots, l(i)$. Indeed, if $t \in (\tau(i,j-1), \tau(i,j))$ for some $i \in \mathbb{N}$ and $j = 1, 2, \ldots, t_i$, then $x(t^-) = f_{q_i^c}^{t-t_i}(x(t_i))$ and $q(t^-) = (q_i^c, q^d(i,j-1))$ and hence $x(t^-) \notin \bigcup_{e \in E_i} \Phi_{q(t^-),e}$, which is a contradiction. From the definition of $\hat{o}$ it then follows that $\hat{o}(t) \in E_i$ if and only if $t \in \mathbf{SW}$, i.e. $t = t_i^{int}$ for some $i \in \mathbb{N}$.

Next, we construct a sequence t_i^o, $i \in \mathbb{N}$ such that $o(t) \in E_o$ if and only if $t = t_i^o$ for some $i \in \mathbb{N}$. To this end consider the sequences t_i, $i \in \mathbb{N}$ and $\tau(i,j)$, $i \in \mathbb{N}$, $j = 0, 1, \ldots, l(i) + 1$ defined in Corollary 1. We will show that for each $i \in \mathbb{N}$, $j = 1, 2, \ldots, l(i) + 1$, there exists an integer $l^o(i,j) \in \mathbb{N}$, time instances and output events

$$\begin{aligned} &\tau(i,j-1) = t^o(i,j,0) < t^o(i,j,1) < \ldots < t^o(i,j,l(i,j)) \leq t^o(i,j,l(i,j)+1) = \tau(i,j) \\ &e^o(i,j,1), \ldots, e^o(i,j,l^o(i,j)) \in E_o \end{aligned} \tag{40}$$

such that for all $k = 1, 2, \ldots, l^o(i,j) + 1$,

$$\begin{aligned} &\forall t \in (t^o(i,j,k-1), t^o(i,j,k)) : f_{q_i^c}^{t-t_i}(x_i) \notin \bigcup_{e \in E_o} \Phi_{q_i^c, q^d(i,j-1),e} \\ &f_{q_i^c}^{t^o(i,j,k)-t_i}(x_i) \in \Phi_{q_i^c, q^d(i,j-1), e^o(i,j,k)} \end{aligned} \tag{41}$$

Assume that sequences from (40) exist and they satisfy (41) for all $k = 1, 2, \ldots, l^o(i,j) + 1$. Consider the set

$$\begin{aligned} \mathbf{SWO} = &\{t^o(i,j,k) \mid i \in \mathbb{N}, j = 1, 2, \ldots, l(i), k = 1, 2, \ldots, l^o(i,j)\} \cup \\ &\{t_i \mid i \in \mathbb{N}, t_i > 0, d(t_i) \in E_d, \text{ and } \lambda_o(q(t_i^-), d(t_i)) \text{ is defined }\} \end{aligned}$$

It then follows that $o(t) \in E_o$ if and only if $t \in \mathbf{SWO}$. Indeed, $o(t) = \lambda_o(q(t^-), d(t))$ and $d(t) \in E_d$ if and only if $t = t_i$ for some $i \in \mathbb{N}$, $t_i > 0$ and $\lambda_o(q(t^-), d(t))$ is defined. Assume that $o(t) = e$ such that $d(t) = \perp$, $t > 0$ and $x(t^-) \in \Phi_{q(t^-),e}$. Then from Corollary 1 it follows that $t \in [\tau(i,j-1), \tau(i,j))$ for some $i \in \mathbb{N}$ and $j = 1,2,\dots,l(i)+1$ and $x(t^-) = f_{q_i^c}^{t-t_i}(x_i)$ and $q(t^-) = (q_i^c, q^d(i,j-1))$. Then from (41) it follows that $t = t^o(i,j,k)$ must hold for some $k = 1,2,\dots,l^o(i,j,k)$. Combining the results of the discussion above, we get that $o(t) \in E_o$ if and only if $t \in \mathbf{SWO}$. Let now t_i^o, $i \in \mathbb{N}$ be the listing of the elements of **SWO** in increasing order. It is easy to see that t_i^o, $i \in \mathbb{N}$ has the required properties.

In order to show the existence of a sequence of the form (40) which satisfies (41) for all $k = 1, 2\dots, l^o(i,j)+1$ we will use Lemma 3. More precisely, we will construct recursively sequences of the form (40) satisfying (41). Let $t^o(i,j,0) = \tau(i,j-1)$. Assume that time instances $t^o(i,j,1) < \dots < t^o(i,j,l)$ and and events $e(i,j,1),\dots,e(i,j,l) \in E_o$ were defined for some $l \in \mathbb{N}$ such that for $k = 1,2,\dots,l$ (41) holds. Then apply Lemma 3 to the state $((q_i^c, q_{j-1}^d), f_{q_i^c}^{t^o(i,j,l)-t_i}(x_i))$ and interval $[0, \tau(i,j) - t^o(i,j,l)]$. Then either for all $t \in (t^o(i,j,l), \tau(i,j)]$, $f_{q_i^c}^{t-t_i}(x_i) \notin \bigcup_{e \in E_o} \Phi_{q_i^c, q_{j-1}^d, e}$, or there exists $t^o(i,j,l) < t^o(i,j,l+1)$ and there exists $e(i,j,l) \in E_o$ such that $f_{q_i^c}^{t^o(i,j,l+1)-t_i}(x_i) \in \Phi_{q_i^c, q_{j-1}^d, e(i,j,l)}$ and $f_{q_i^c}^{t-t_i}(x_i) \notin \bigcup_{e \in E_o} \Phi_{q_i^c, q_{j-1}^d, e}$ for all $t \in (t^o(i,j,l), t^o(i,j,l+1))$. In the former case define $l = l(i,j)$, $t^o(i,j,l+1) = \tau(i,j)$ and we get that

$$t^o(i,j,0),\dots,t^o(i,j,l(i,j)+1), e(i,j,1),\dots,e(i,j,l(i,j))$$

are of the form (40) and for all $k = 1,2,\dots,l(i,j)+1$, (41) holds. In the latter case, consider the sequences $t^o(i,j,0),\dots,t^o(i,j,l+1)$ and $e(i,j,1),\dots,e(i,j,l+1)$. It is easy to see that then for all $k = 1,2,\dots,l+1$, (41) holds. Repeat then the step above for the sequence $t^o(i,j,0),\dots,t^o(i,j,l+1)$ and $e(i,j,1),\dots,e(i,j,l+1)$. Note that by Assumption A4, $l \le \mathbb{T}((q_i^c, q_{j-1}^d), E_o, \tau(i,j) - \tau(i,j-1))$ must hold, and hence after at most $\mathbb{T}((q_i^c, q_{j-1}^d), E_o, \tau(i,j) - \tau(i,j-1))$ iterations we get a sequence of the form (40) satisfying (41), as desired.

□

8.3 Sampled-time abstraction of H

In this section we present the proof of Proposition 8, and Theorem 4. The proofs rely on a number of lemma, which are interesting in their own right. Throughout the section we will use the notation of Definition 28.

Lemma 5. The state-transition relation E of $H_\triangle(\mathcal{P})$ is a partial map.

Proof. That is, for any $h_1 \in \mathcal{R}(H)$, and for any $u \in U$, $d \in D$, $o \in O$ and $\hat{o} \in E_i^*$, there exists at most one $h_2 \in \mathcal{R}(H)$ such that $E(h_1, u, d, o, \hat{o}) = h_2$. From (10) it follows that q_2^c and x_2 are uniquely determined by h_1, u and d. Moreover, q_2^d is independent of the output o. It is left to show that d and $\hat{o}$, h_1 uniquely determine q_2^d.

To this end, consider the decomposition $\hat{o} = z_1 z_2 \cdots z_l$ and assume that $\hat{o}$ satisfies (11) with the indices $i_1 < i_2 < \dots < i_k$. We argue that there is only one choice of indices $i_1 < i_2 < \dots < i_k$ for $\hat{o}$ such that it satisfies (11). In turn, this unique choice of indices and the decomposition of $\hat{o} = z_1 z_2 \cdots z_l$, together with d determine the states $s_i \in Q_d$, $i = 1,2,\dots,l$ uniquely. Since $q_2^d = s_l$, this already implies that $\hat{o}$ and d determine q_2^d uniquely. In order to see that the choice of $i_1, i_2, \dots, i_k$ is unique, notice that according to Assumption A6, λ_i is a complete map and no event in the range of λ_i can be generated by a guard. Hence, for each letter z_i of $\hat{o}$, if z_i is in the range of λ_i, then $P_{q_2^c, s, z_i} = \emptyset$ for all $s \in Q_d$. By (11), for each letter z_i, either $R_{u,q_1}(x_1) \in P_{q_2^c, s_{i-1}, z_i}$ or $z_i = \lambda_i(q_2^c, s_{i-1}, e_r)$. Hence, if z_i is in the range of λ_i, then $z_i = \lambda_i(s_{i-1}, e_r)$ must hold. This means that $\{i_1, i_2, \dots, i_k\}$ is precisely the set of indices i such that the letter z_i of $\hat{o}$ is in the image of λ_i. This set is unique for each $\hat{o}$. Moreover, if $\hat{o}$ satisfies (11), then the number of such letters is exactly k. □

Next, we present a relationship between state transitions of $H_{\Delta}(\mathcal{P})$ and time evolution of H. More precisely, we show that a transition step in $H_{\Delta}(\mathcal{P})$ amounts to the end-state of the evolution of H at time instance Δ.

Lemma 6. Assume that $h_1 \in \mathcal{R}(H)$, $u \in U$ and assume that $d = e_1 e_2 \cdots e_k$, $e_i \in E_d$, $k \leq \mu$. $\xi_H(h_1, u, g)(\Delta) = (q(\Delta), x(\Delta)) = h_2$. Here $u(t) = \begin{cases} u & \text{if } t = 0 \\ \perp & \text{otherwise} \end{cases}$, and $g \in \mathcal{P}^{\Delta}_{E_d,\mu}$ is such that $\mathbf{UT}(g, \Delta) = d$ and $g(0) = \perp$. Such a time-event function g exists, see for example the following map $g(t) = \begin{cases} e_i & \text{if } t = i\frac{\Delta}{\mu+1} \text{ for some } i = 1, 2, \ldots, k \\ \perp & \text{otherwise} \end{cases}$.

Then $u(t) \in \mathcal{P}_{E_c}$ and $g(t) \in \mathcal{P}^{\Delta}_{E_d,\mu}$ and hence h_2 is well-defined. Consider output signals $(\alpha, \beta) = \upsilon_{H,h_1}(u, g)$ where $\alpha \in \mathcal{P}_{E_o}$ and $\beta \in \mathcal{P}_{E_i}$. Define $o = \{e \in E_o \mid \exists s \in (0, \Delta] : \alpha(s) = e\}$, and $\hat{o} = \mathbf{UT}(\beta, \Delta)$, i.e. o is the set of output events generated by H under the input u and disturbance g on the interval $(0, \Delta]$, if started from h_1. Likewise, $\hat{o}$ is the sequence of internal events generated by H on the interval $(0, \Delta]$, under the input u and disturbance g, if started from h_1. Then $E(h_1, u, d, o, \hat{o})$ is well-defined, $E(h_1, u, d, o, \hat{o}) = h_2$ and $h_2 \in \mathcal{R}(H)$.

Proof of Lemma 6. Assume that τ_i, $i = 1, 2, \ldots, k$ are the switching times of g on $[0, \Delta]$ in increasing order, i.e. $\tau_1 < \tau_2 < \ldots < \tau_k \in [0, \Delta]$, and for all $s \in (0, \Delta]$, $g(s) \in E_d$ if and only if $s = \tau_i$ for some $i = 1, 2, \ldots, k$. Recall that $g(0) = \perp$ and hence $\tau_i \in (0, \Delta]$, $i = 1, 2, \ldots, k$. Let $h_1 = (q_1^c, q_1^d, x_1)$ and $h_2 = (q_2^c, q_2^d, x_2)$. Then from the definition of H it follows that $q_2^c = \delta_c(q_1, u)$, $x_2 = f^{\Delta}_{q_2^c}(R_{u,q_1}(x_1))$, i.e. q_2^c, x_2 satisfy (10). Here $R_{\perp,q_1}(x_1) = x_1$ and $\delta_c(q_1, \perp) = q_1^c$. In addition, it is easy to see that if $h_1 \in \mathcal{R}(H)$, then $h_2 \in \mathcal{R}(H)$. Indeed, if $h_1 \in \mathcal{R}(H)$, then with the notation of Definition 26, $x_1 \in H_i$ for some $i \in \mathbb{N}$. Since, $x_2 = f^{\Delta}_{q_2^c}(R_{u,q_1}(x_1))$, it follows then that $x_2 \in H_{i+1}$ and hence $h_2 \in Q \times H_{i+1} \subseteq \mathcal{R}(H)$.

Notice that g takes values in E_d only at time instances of the form $t = \tau_i$, and then $g(t) = e_i$. Notice that $\tau_k \leq \Delta$, i.e. all the events of $g(t)$ occur on the interval $(0, \Delta]$. Assume that $\hat{o} = z_1 z_2 \ldots z_l$ and let $t_1 < t_2 < \ldots < t_l \in [0, \Delta]$ be such that $\beta(t_i) = z_i$ and $\beta(s) = \perp$ for all $s \notin \{t_1, t_2, \ldots, t_l\}$. Notice that from the definition of υ_{H,h_1} it follows that $\beta(0) = \perp$ and $\alpha(0) = \perp$ and hence $t_1, t_2, \ldots, t_l \in (0, \Delta]$. It then follows from the definition of H, that either $t_i = t_{i_r} = \tau_r$ and $z_i = \lambda(q(t_i^-), e_r)$ for some $r = 1, 2, \ldots, k$, or $x(t_i^-) \in \Phi_{q(t_i^-),z_i}$ and $t_i \notin \{\tau_r \mid r = 1, 2, \ldots, k\}$ Hence, there exists $i_1 < i_2 < \ldots < i_k \in \{1, 2, \ldots, l\}$ such that $\{t_{i_1}, t_{i_2}, \ldots, t_{i_k}\} = \{\tau_1, \tau_2, \ldots, \tau_k\}$.

From the definition of H it then follows that for all $i = 1, 2, \ldots, l$ $q(t_i) = (q_2^c, s_i)$ and $q(t_i^-) = (q_2^c, s_{i-1})$ where $s_0 = q_1^d$, and the states $s_i \in Q_d$ satisfy the following recursion

$$s_i = \begin{cases} \delta_d(q_2^c, s_{i-1}, z_i) & \text{if } i \notin \{i_r \mid r = 1, 2, \ldots, k\} \\ \delta_d(q_2^c, s_{i-1}, e_r) & \text{if } i = i_r \text{ for some } r = 1, 2, \ldots, k \end{cases}$$

where $z_{i_r} = \lambda_i(q_2^c, s_{i_r-1}, e_r)$, for all $r = 1, 2, \ldots, k$. In addition, since $x(t_i) = f^{t_i}_{q_2^c}(R_{u,q_1}(x_1))$, we get that $x(t_i^-) \in \Phi_{q(t_i^-),z_i}$ implies that $R_{u,q_1}(x_1) \in P_{q_2^c,s_{i-1},z_i}$. Moreover, notice that $q(t) = (q_2^c, s_l)$ for all $t \in [t_l, \Delta]$. In particular, $q(\Delta) = (q_2^c, s_l)$, which implies that $q_2^d = s_l$. From Assumption A3 it follows that the number of elements in $\{1, 2, \ldots, l\} \setminus \{i_1, i_2, \ldots, i_k\}$ is at most $\mathbb{T}(q_1, \Delta)$, and hence $k \leq l \leq \mathbb{T}(q_1, \Delta) + k$. Combining all this we get that $\hat{o}$ satisfies (11).

Finally, we know that $e \in o$ if and only if either $e = \lambda_o(q(\tau_r^-), e_r)$ for some $r = 1, 2, \ldots, k$, or $x(t^-) \in \Phi_{q(t^-),e}$ for some $t \in (0, \Delta]$, $t \notin \{\tau_r \mid r = 1, 2, \ldots, k\}$. It follows from the definition of H that $q(t^-) = (q_2^c, s_{i-1})$ for all $t \in (t_{i-1}, t_i]$ for all $i \in \{1, 2, \ldots, l\}$, and $q(t^-) = (q_2^c, s_l)$ for all $t \in (t_l, \Delta]$. Hence, either $e = \lambda_o((q_2^c, s_{i-1}), e_r)$ for some $i = i_r$, $r = 1, 2, \ldots, k$, or $R_{u,q_1}(x_1) \in P_{q_2^c,s_{i-1},e}$ for some $i \in \{1, 2, \ldots, l\} \setminus \{1, 2, \ldots, l\}$. Hence, e must satisfy (12).

Combining the results above we get that $E(h_1, u, d, o, \hat{o}) = h_2$ and $E(h_1, u, d, o, \hat{o})$ is well-defined. □

Now we are ready to present the Proof of Proposition 8.

Proof of Proposition 8. We will use the notation of Definition 28. In order to show that $H_\Delta(\mathcal{P})$ is a quasi-sequential transducer, we have to show that the map E is partial map which satisfies Definition 6. But the latter follows from Lemma 5. Next, we need to show that for each u and d there exists o and $\hat{o}$ such that $E(h_1, u, d, o, \hat{o})$ is defined. But this follows from Lemma 6. Finally, notice that if $\mathcal{R}(H)$ is known, then the only component of $H_\Delta(\mathcal{P})$ which is problematic to compute is the state-transition map E. However, if the flow of the vector fields and reset maps are computable, then the state-transition map E is computable, if $R_{u,q_1}(x_1) \in P_{q,e}$ can be checked by a numerical algorithm for all $q \in Q$, $e \in E_i \cup E_o$. The latter holds if $\mathcal{P}$ is computable. □

Next, we present the proof of Theorem 4 stating that the relation recognized by $H_\Delta(\mathcal{P})$ is an abstraction of R_H. This follows essentially from Lemma 6, which informally states that $H_\Delta(\mathcal{P})$ is simulates H.

Proof of Theorem 4. Since $H_\Delta(\mathcal{P})$ is a quasi-sequential transducer according to Proposition 8, it follows that $R = R(H_\Delta(\mathcal{P}))$ is a sequential input-output relation. We will show that for all sequences $s \in (U \times D)^*$

$$(\underline{o}, \hat{\underline{o}}) \in R_H(s) \implies (\underline{o}, \hat{\underline{o}}) \in R(s)$$

To this end, assume that

$$s = (u_1, d_1)(u_2, d_k) \cdots (u_k, d_k) \text{ and } \underline{o} = o_1 o_2 \cdots o_k$$

for some discrete input symbols $u_i \in U$, disturbance symbols $d_i \in D$, output symbols $o_i \in O$. It then follows from the definition of R_H that there exists a disturbance signal $g \in \mathcal{P}^{\Delta}_{E_d,\mu}$, and a control signal $u \in \mathcal{P}_{E_c}$ such that

$$(\alpha, \beta) = \upsilon_{H,h_0}(u, g)$$

i.e. α, β are the output and internal event response of H to the input u and disturbance g, and $\hat{o} = \mathbf{UT}(\beta, k\Delta)$, $o_i = \alpha(((i-1)\Delta, i\Delta])$, $d_i = \mathbf{UT}(g_i, \Delta)$, $i = 1, 2, \ldots, k$ where

$$u(t) = \begin{cases} u_i & \text{if } t = (i-1)\Delta \text{ for some } i = 1, 2, \ldots, k \\ \perp & \text{otherwise} \end{cases}$$

$$g_i(t) = \begin{cases} g(t + (i-1)\Delta) & \text{if } t > 0 \\ \perp & \text{if } t = 0 \end{cases}$$

Define now the states $h_i \in S_H$, $i = 1, 2, \ldots, k$ of H as follows. For each $i = 1, 2, \ldots, k$, define the map $u(i) \in \mathcal{P}_{E_c}$ as follows

$$\forall s \in \mathbb{R}_+ : u(i)(s) = \begin{cases} u(s + (i-1)\Delta) & \text{if } s \in [0, \Delta) \\ \perp & \text{if } s \geq \Delta \end{cases}$$

Then define

$$h_i = \xi_H(h_{i-1}, u(i), g_i)(\Delta) \text{ for all } i = 1, 2, \ldots, k$$

Here h_0 is the initial state of H. The proof of the theorem is based on the following claims.

Claim 1
For each $i = 1, 2, \ldots, k$, consider now

$$(\alpha_i, \beta_i) = \upsilon_{H,h_{i-1}}(u(i), g_i)$$

We claim that

$$\alpha_i(s) = \alpha(s + (i-1)\Delta) \text{ and } \beta_i(s) = \beta(s + (i-1)\Delta) \text{ for all } s \in (0, \Delta]$$

Notice that by definition of the input-output maps of H, $\alpha_i(0) = \perp$ and $\beta_i(0) = \perp$ for all $i = 1, 2, \ldots, k$.

Claim 2 With the notation of **Claim 1**

$$o_i = \alpha_i((0, \Delta]) = \alpha(((i-1)\Delta, i\Delta]) \text{ for all } i = 1, 2, \ldots, k$$

Further, consider

$$\hat{o}_i = \mathbf{UT}(\beta_i, \Delta) \text{ for all } i = 1, 2, \ldots, k$$

It then follows that $\hat{o}_1\hat{o}_2 \cdots \hat{o}_k = \hat{o}$.

Assume now that **Claim 1** and **Claim 2** are true. Notice that $h_0 \in \mathcal{R}(H)$. Then by Lemma 6 and by induction on i, we get that $h_i \in \mathcal{R}(H)$ and $h_i = E(h_{i-1}, u_i, d_i, o_i, \hat{o}_i)$ for all $i = 1, 2, \ldots, k$. Hence, we get that the sequence

$$(u_1, d_1, o_1, \hat{o}_1) \cdots (u_i, d_i, o_i, \hat{o}_i) \in (U \times D \times O)^* \times E_i^*$$

is accepted by $H_\Delta(\mathcal{P})$ for all $i = 1, 2, \ldots, k$ The latter is equivalent to

$$(o_1 o_2 \cdots o_k, \hat{o}_1\hat{o}_2 \cdots \hat{o}_k) = (\underline{o}, \underline{\hat{o}}) \in R(s) = R((u_1, d_1) \cdots (u_k, d_k)),$$

since R is the relation recognized by $H_\Delta(\mathcal{P})$.

We conclude the proof by proving **Claim 1** and **Claim 2**.

Proof of Claim 1

Assume that $h_i = (q_i, x_i) \in S_H$ and $q_i = (q_i^c, q_i^d)$. By induction, we can show that for all $i = 1, 2, \ldots, k$.

$$\xi_H(h_0, u, g)(s + (i-1)\Delta) = \xi_H(h_{i-1}, u(i), g_i)(s) \text{ if } s \in (0, \Delta) \tag{42}$$

Assume that (42) is true. From the definition of $\upsilon_{H,h_0}(u, g)$ it follows that if $(\alpha, \beta) = \upsilon_{H,h_0}(u, g)$, then for all $s \in (0, \Delta]$, $\alpha(s+(i-1)\Delta)$ and $\beta(s+(i-1)\Delta)$ depend only on the left-hand side limit $\xi_H(h_0, u, g)((s+(i-1)\Delta)^-) = (q((s+(i-1)\Delta)^-), x((s+(i-1)\Delta)^-))$ and on $g(s+(i-1)\Delta)$. From (42) it then follows that

$$\xi_H(h_0, u, g)((s + (i-1)\Delta)^-) = \xi_H(h_{i-1}, u(i), g_i)(s^-) \text{ for all } s \in (0, \Delta] \tag{43}$$

If $(\alpha_i, \beta_i) = \upsilon_{H,h_{i-1}}(u(i), g_i)$, then $\alpha_i(s), \beta_i(s)$ depend only on $\xi_H(h_{i-1}, u(i), g_i)(s^-)$ and $g_i(s) = g(s + (i-1)\Delta)$ for all $s \in (0, \Delta]$. Hence, combining this with (43) and the discussion above we get the statement of **Claim 1**.

We conclude the proof of **Claim 1** by proving that (42) holds for all $i = 1, 2, \ldots, k$. Assume that $\xi_H(h_0, u, g)(t) = (q(t), x(t))$, $q(t) = (q^c(t), q^d(t))$ for all $t \in \mathbb{R}_+$. In addition, assume that $\xi_H(h_{i-1}, u(i), g_i)(s) = (\hat{q}_i(s), \hat{x}_i(s))$ for all $s \in [0, \Delta]$. Hence, $h_i = (q_i, x_i) = (\hat{q}_i(\Delta), \hat{x}_i(\Delta))$.

For $i = 1$, (42) follows from the fact that $h_{i-1} = h_0$ and $u(i)(s) = u(s)$, $g_i(s) = g(s)$ for all $s \in (0, \Delta)$, and hence $\xi_H(h_0, u, g)(s) = \xi_H(h_0, u(i), g_i)(s)$ for all $s \in (0, \Delta)$.

Suppose that (42) holds for some $i = 1, 2, \ldots, k-1$. We will show that (42) holds for $i+1$. To this end, we will argue that

$$\xi_H(h_0, u, g)(i\Delta) = (q(i\Delta), x(i\Delta)) = \xi_H(h_i, u(i+1), g_i)(0) \tag{44}$$

From this, due to the definition of state-trajectory ξ_H, it follows that (42) holds.

In order to prove (44) let us compute $\xi_H(h_0, u, g)(i\Delta) = (q(i\Delta), x(i\Delta))$. From the definition we get that $q^c(i\Delta) = \delta_c(q((i\Delta)^-), u_{i+1})$ and $x(i\Delta) = R_{u_{i+1}, q((i\Delta)^-)}(x((i\Delta)^-))$. Since $u(i)(\Delta) = \perp$, it follows that $q_i^c = \hat{q}_i^c(\Delta) = \hat{q}_i^c(\Delta^-)$. From the induction hypothesis we then get that $q_i^c = \hat{q}_i^c(\Delta^-) = q^c((i\Delta)^-)$. Since $\delta_c(q((i\Delta)^-), u_{i+1})$ and $R_{u_{i+1},q((i\Delta)^-)}$ depend only on u_{i+1} and $q^c((i\Delta)^-)$, it follows then that $q^c(i\Delta) = \delta_c(q_i, u_{i+1})$ and $x(i\Delta) = R_{u_{i+1},q_i}(x((i\Delta)^-))$. Notice that since $u(i)(\Delta) = \perp$, it follows that $\hat{x}_i(\Delta^-) = \hat{x}_i(\Delta) = x_i$. From the induction hypothesis it follows that $\hat{x}_i(\Delta^-) = x((i\Delta)^-)$ and hence $x_i = x(\Delta^-)$. That is, we get that

$$q^c(i\Delta) = \delta_c(q_i, u_{i+1}) \text{ and } x(i\Delta) = R_{u_{i+1},q_i}(x_i) \tag{45}$$

Finally, notice that either no internal event is generated on the interval $(\Delta - r, \Delta]$ for some $r < \Delta$, or an internal event is generated at Δ, when H is started from h_{i-1}. In the former case, for all $s \in (\Delta - r, \Delta]$, $g_i(s) = g(s + (i-1)\Delta) = \perp$ and $\hat{x}_i(s^-) = x((s + (i-1)\Delta)^-) \notin \bigcup_{e \in E_i} \Phi_{\hat{q}_i(\Delta - r),e} = \bigcup_{e \in E_i} \Phi_{q(i\Delta - r),e}$ From the induction hypothesis it then follows that no internal event is generated by H on the interval $(i\Delta - r, i\Delta]$, if H is started from h_0. Hence, $q^d(i\Delta) = q^d((i\Delta)^-) = \hat{q}_i^d(\Delta^-) = q_i^d$. Assume now that an internal event occurs at time Δ if H is started from h_{i-1}. i.e. if either $g_i(\Delta) = g(i\Delta) = d \in E_d$ or $\hat{x}_i(\Delta^-) = x((i\Delta)^-) \in \Phi_{\hat{q}_i(\Delta^-),e} = \Phi_{q((i\Delta)^-),e}$ for some $e \in E_i$. Then from the induction hypothesis it follows that the same internal event is generated by H at time $i\Delta$ is started from h_0, i.e. $q^d(\Delta) = \delta_d(\hat{q}_i(\Delta^-), z) = q_i^d$ where $z = e$ or $z = \lambda_i(\hat{q}_i(\Delta^-), d) = \lambda_i(q((i\Delta)^-), d)$. Combining all the cases above we get that

$$q_i^d = q^d(\Delta) \tag{46}$$

Since $u(i+1)(0) = u_{i+1}$, we then get that (45) and (46) implies (44).

Proof of Claim 2 That $o_i = \alpha_i((0, \Delta])$ follows from the fact that $o_i = \alpha(((i-1)\Delta, i\Delta])$ and **Claim 1**.

Next we show that $\hat{o} = \hat{o}_1\hat{o}_2 \cdots \hat{o}_k$ where $\hat{o}_i = \mathbf{UT}(\beta_i, \Delta)$. To this end, let $t_1 < t_2 < \ldots < t_l \in [0, k\Delta]$ be such that $\beta(s) \in E_i$ if and only if $s = t_j$ for some $j = 1, 2, \ldots, l$. From the definition of $\upsilon_{H,h_0}(u, g)$ it follows that $\beta(0) = \perp$, and hence $t_1 > 0$. For each $i = 1, 2, \ldots, k$, let J_i be the subset of indices of $j = 1, 2, \ldots, l$ such that $t_j \in ((i-1)\Delta, i\Delta]$, i.e.

$$J_i = \{j \in \{1, 2, \ldots, l\} \mid t_j \in ((i-1)\Delta, i\Delta]\}$$

From $t_1 > 0$ it then follows that $\bigcup_{i=1}^k J_i = \{1, 2, \ldots, l\}$. Moreover, it is easy to see that the sets $J_1, J_2, \ldots, J_k$ are pairwise disjoint. In addition, since the sequence $t_1, t_2, \ldots, t_l$ is increasing, we get that $J_i = \{m_{i-1} + 1, \ldots, m_i\}$ for $i = 1, 2, \ldots, k$, where $m_0 = 0$ and $m_i \in \{0, 1, \ldots, l\}$ and $m_1 \leq m_2 \leq \ldots \leq m_k$. We will show that for all $i = 1, 2, \ldots, k$

$$\hat{o}_i = \beta(t_{m_{i-1}+1})\beta(t_{m_{i-1}+2}) \cdots \beta(t_{m_i}) \tag{47}$$

Since $\hat{o} = \beta(t_1) \cdots \beta(t_l)$ and $\bigcup_{i=1}^k J_i = \{1, 2, \ldots, l\}$, it is clear that (47) implies that $\hat{o} = \hat{o}_1\hat{o}_2 \cdots \hat{o}_k$ and hence **Claim 2** is proven.

In order to show (47), notice that from **Claim 1** it follows that $\beta_i(s) = \beta(s + (i-1)\Delta)$ for all $s \in (0, \Delta]$ and $\beta_i(0) = \perp$. Hence, $\beta_i(s) \in E_i$ if and only if $s = t_j - (i-1)\Delta$ for some $t_j \in ((i-1)\Delta, i\Delta])$, i.e. if and only of $s = t_r - (i-1)\Delta$ for some $r = m_{i-1} + 1, m_{i-1} + 2, \ldots, m_i$. From this and the definition of $\hat{o}_i = \mathbf{UT}(\beta_i, \Delta)$, (47) follows. □

8.4 Proof of the Lyapunov-like conditions for finiteness of $\mathcal{R}(H)$

Proof of Theorem 5. To this end, let $K = \max\{V(x) \mid x \in \mathcal{X}_0 \cup \{x_0\}\}$. Let $N = \lceil K/c\Delta \rceil$. Consider the sets H_i, $i \in \mathbb{N}$ from Definition 26. Define the sets $\hat{H}_i \subseteq X$, $i \in \mathbb{N}$ as follows.

$$\begin{aligned} &\hat{H}_0 = \mathcal{X}_0 \cup \{x_0\} \\ &\hat{H}_{i+1} = \{f_{q^c}^{\Delta}(R_{e,s}(x)), f_{q^c}^{\Delta}(x) \mid x \in \hat{H}_i, q = (q^c, q^d), s \in Q, u, e \in E_c\} \cup \hat{H}_i \end{aligned}$$

Notice that $H_i \subseteq \hat{H}_i$ for all $i \in \mathbb{N}$. This can be show by induction on i. For $i = 0$, $H_0 = \{x_0\} \subseteq \hat{H}_0$. Assume that $H_k \subseteq \hat{H}_{k+1}$. Then if $x \in H_{k+1}$, then either $x \in H_k \subseteq \hat{H}_k \subseteq \hat{H}_{k+1}$, or $x = f^{\Delta}_{q^c}(R_{u,s}(\hat{x})$ for some $q^c \in Q$, $s \in Q$, $u \in E_c \cup \{\perp\}$, $\hat{x} \in H_k \subseteq \hat{H}_k$. Hence, $x \in \hat{H}_{k+1}$.

Notice that $\hat{H}_i$ is a finite set for each $i \in \mathbb{N}$, and $h_0 \in \hat{H}_0$. We will show that $\hat{H}_{N+1} = \hat{H}_N$. From this the finiteness of $\mathcal{R}(H)$ follows. To see this, first we show that $\hat{H}_{N+1} = \hat{H}_N$ implies that $\hat{H}_N = \hat{H}_{N+k}$ for all k. We can prove this by induction on k. For $k = 1$ the statement follows from $\hat{H}_N = \hat{H}_{N+1}$. Assume that statement is true for all $k \leq l$. In particular, $\hat{H}_N = \hat{H}_{N+l}$. Recall that $\hat{H}_{N+l+1} = \hat{H}_{N+l} \cup \{f_{q^c}(x)^{\Delta}, f^{\Delta}_{q^c}(R_{u,s}(x)) \mid q^c \in Q_c, s \in Q, u \in E_c, x \in \hat{H}_{N+l}\}$. Using $\hat{H}_N = \hat{H}_{N+l}$, we get that $\hat{H}_{N+l+1} = \hat{H}_N \cup \{f_{q^c}(x)^{\Delta}, f^{\Delta}_{q^c}(R_{u,s}(x)) \mid q^c \in Q_c, s \in Q, u \in E_c, x \in \hat{H}_N\} = \hat{H}_{N+1}$, by definition of $\hat{H}_{N+1}$. Hence, we get that $\hat{H}_N = \hat{H}_{N+k}$ for all $k \geq 0$. Since $\hat{H}_i \subseteq \hat{H}_{i+1}$, for all $i \in \mathbb{N}$, we then get that $\bigcup_{i=0}^{\infty} H_i \subseteq \bigcup_{i=0}^{\infty} \hat{H}_i = \hat{H}_N$ and hence $\mathcal{R}(H) \subseteq Q \times \hat{H}_N$. Since Q is finite, and $\hat{H}_N$ is finite, we get that $\mathcal{R}(H)$ is finite.

We conclude the proof by showing that $\hat{H}_{N+1} = \hat{H}_N$. To this end, by induction on i we will show that

$$\forall x \in \hat{H}_i \setminus \hat{H}_{i-1} : \exists z \in H_0 : V(x) \leq V(z) - ci\Delta \tag{48}$$

For $i = 0$, the statement above trivially holds. Suppose it holds up to $i > 0$ and consider $x \in \hat{H}_{i+1} \setminus \hat{H}_i$. Then, there exists $\hat{x} \in \hat{H}_i$, $e_2 \in E_c \cup \{\perp\}$ and $q_1, q_2 = (q_2^c, q_2^d) \in Q$ such that $x = f^{\Delta}_{q_2^c}(R_{e_1,q_1}(\hat{x})))$. Here, we assume that $R_{\perp,q_1}(\hat{x}) = \hat{x}$. We argue that $\hat{x} \notin \partial\mathcal{X}$ and $f^{\Delta}_{q_2^c}(R_{e_1,q_1}(\hat{x}) \notin \partial\mathcal{X}$. Indeed, assume that $\hat{x} \in \partial\mathcal{X}$. Then $R_{e_1,q_1}(\hat{x}) = x_0^{'} = R_{\perp,q_1}(x_0^{'}) \in \mathcal{X}_0$, and hence $f^{\Delta}_{q_2^c}(R_{\perp,q_1}(x_0^{'}))) \in H_1$. But $i > 0$ implies that $H_1 \subseteq H_i$, i.e. $x \in H_i$ which is a contradiction. Assume that $f^{\Delta}_{q_2^c}(R_{e_1,q_1}(\hat{x})) \in \partial\mathcal{X}$. Then $x \in \mathcal{X}_0 \subseteq H_0 \subseteq H_i$ which is a contradiction.

In order to finish the proof of (48), we need to show that for any solution of the differential equation $\dot{z}(s) = f_{\hat{q}^c}(z(s))$ with $z(0) = z_0 \in \mathcal{X}$,

$$V(z(\tau)) \leq V(z) - c\tau \tag{49}$$

In order to see (49), notice that $\frac{d}{ds}V(z(s)) = \mathrm{grad}V(z(s))f_{\hat{q}^c}(z(s)) < -c$, and hence $V(z(\tau)) = V(z) + \int_0^{\tau} \frac{d}{ds}V(z(s)) \leq V(z) - \int_0^{\tau} c = V(z) - c\tau$.

From (49) it then follows that if for all $s \in [0, \tau]$: $f^{s}_{\hat{q}^c}(z_0) \in X \setminus \partial X$, then $f^{\tau}_{\hat{q}^c}(z_0) = z(\tau) < V(z_0) - c\tau$. Hence, we get that $V(R_{q_1,e_1}(\hat{x})) \leq V(\hat{x})$, $V(f^{\Delta}_{q_2^c}(R_{q_1,e_1})(\hat{x})) \leq V(R_{q_1,e_1}(\hat{x})) - c\Delta \leq V(\hat{x}) - c\Delta$, and finally

$$V(x) = V(f^{\Delta}_{q_2}(R_{e_1,q_1})(\hat{x})) \leq V(\hat{x}) - c\Delta \tag{50}$$

Notice that $\hat{x} \in H_i \setminus H_{i-1}$; indeed, $\hat{x} \in H_{i-1}$ implies $x \in H_i$, which is a contradiction. Hence, for some $z_0 \in H_0$, we get that $V(\hat{x}) \leq V(z_0) - ci\Delta$. By combining this with (50), we immediately get that (48) holds.

Finally, we use (50) to show that $\hat{H}_{N+1} = \hat{H}_N$. By construction, $\hat{H}_N \subseteq \hat{H}_{N+1}$. Assume that $x \in \hat{H}_{N+1} \setminus \hat{H}_N$. Then there exists $x_0 \in H_0$ such that $V(x) \leq V(x_0) - c(N+1)\Delta < K - K = 0$. But $x \in \mathcal{X}$ and hence $V(x) \geq 0$, and hence a contradiction.

□

Proof of Proposition 9. We have to check that conditions 1–3 of Theorem 5 hold. Consider V of the proposition. Since for all $x \in \mathcal{X}$, $n_j^T x - b_j \leq 0$, we get that $V(x) = -(b_j - n_j^T x) \leq 0$. Moreover, $V(x) = 0$, $x \in \mathcal{X}$, if and only if $n_j^T x - b_j = 0$ and $n_i^T - b_i \leq 0$ for all $i \in$

$\{1, 2, \ldots, K\} \setminus \{j\}$. Hence, the Condition 1 of Theorem 5 holds. As to Condition 2 of Theorem 5, we need to show that for all $q^c \in Q_c$, $\text{grad}(V)(x)f_{q^c}(x) < -c$ for some $c > 0$ and for all $x \in \mathcal{X}$. Notice that $\text{grad}(V)(x) = -n_j^T$ and hence

$$\text{grad}(V)(x)f_{q^c}(x) = -n_j^T(A_{q^c}x + \sum_{l=1}^{m} B_{q^c,l}\phi_{q^c,l}(r_{q^c,l}^T x))$$

Notice that for all $l = 1, 2, \ldots, m$ $n_j^T B_{q^c,l}\phi_{q^c,l}(r_{q^c,l}^T x)$ belongs to the interval $[n_j^T B_{q^c,l}(\mu_{i_l} r_{q^c,l}^T x + \gamma_{i_l}), n_j^T B_{q^c,l}(\mu_{j_l} r_{q^c,l}^T x + \gamma_{j_l}]$ for some choice of $i_l \neq j_l \in \{1, 2\}$. Hence,

$$\begin{aligned}&n_j^T(A_{q^c}x + \sum_{l=1}^{m} B_{q^c,l}\phi_{q^c,l}(r_{q^c,l}^T x)) \in \\ &[n_j^T(A_{q^c}x + \sum_{l=1}^{m} B_{q^c,l}(\mu_{i_l} r_{q^c,l}^T x + \gamma_{i_l})), n_j^T(A_{q^c}x + \sum_{l=1}^{m} B_{q^c,l}(\mu_{j_l} r_{q^c,l}^T x + \gamma_{j_l}))]\end{aligned}$$

Condition 1 of the proposition implies that $n_j^T(A_{q^c}x + \sum_{l=1}^{m} B_{q^c,l}(\mu_{i_l} r_{q^c,l}^T x + \gamma_{i_l})) > c$ and $n_j^T(A_{q^c}x + \sum_{l=1}^{m} B_{q^c,l}(\mu_{j_l} r_{q^c,l}^T x + \gamma_{j_l})) > c$. Hence, we get that then

$$n_j^T(A_{q^c}x + \sum_{l=1}^{m} B_{q^c,l}\phi_{q^c,l}(r_{q^c,l}^T x)) > c$$

Thus $\text{grad}(V)(x)f_{q^c}(x) < -c$, i.e. Condition 2 of Theorem 5 holds.

Finally, if $x \in \text{int}\,\mathcal{X}$, then $V(R_{u,q}(x)) - V(x) = (b_j - n_j^T(R_{u,q}(x)) - (b_j - n_j^T x) = n_j^T(x - R_{u,q}(x)) = -n_j^T(R_{u,q}(x)) - x)$. But $R_{u,q}(x) - x = M_{u,q}x - x + b_{u,q}$, and hence by Condition 2. we get that $V(R_{u,q}(x)) - V(x) \leq 0$. That is, Condition 2 of Theorem 5 holds. □

8.5 Well-discretizable hybrid systems: proof of Lemma 1

Proof of Lemma 1. It is enough to show that if there exists $s \in (0, \Delta]$ such that $f_{q^c}^s(x) \in \Phi_{q^c,q^d,e}$, then $x \in P_{q^c,q^d,e}$ for each $x \in \mathcal{X}$. To this end, notice that $f_{q^c}^s(x) \in \Phi_{q^c,q^d,e}$ if and only if $h_{q^c,q^d,e}(f_{q^c}^s(x)) = 0$. In addition, in this case $\Phi_{q^c,q^d,e} \neq \emptyset$. Let z be the solution of the differential equation $\dot{z} = f_{q^c}(z)$ from $z(0) = x$. Then, it follows that $\frac{d}{dt}h_{q^c,q^d,e}(z(t)) = \text{grad}h_{q^c,q^d,e}(z(t))f_{q^c}(z(t)) > 0$. Hence, $h_{q^c,q^d,e}(z(t))$ is a monotonically non-decreasing function of t. It follows that $f_{q^c}^s(x) = z(s)$ if $s < \beta(q^c, x)$ or $f_{q^c}^s(x) = z(\beta(q^c, x))$. Since $\beta(q^c, x) \geq 0$, we get that $h_{q^c,q^d,e}(x) = h_{q^c,q^d,e}(z(0)) \leq h_{q^c,q^d,e}(z(\min\{s, \beta(q^c, x)\})) = 0$. Moreover, $f_{q^c}^\Delta(x) = z(\min\{\Delta, \beta(q^c, x)\})$. Since $s \in (0, \Delta]$, we get that $\min\{\Delta, \beta(q^c, x)\} \geq \min\{s, \beta(q^c, x)\}$, and hence

$$h_{q^c,q^d,e}(f_{q^c}^s(x)) \leq h_{q^c,q^d,e}(f_{q^c}^\Delta(x)) \leq 0$$

Hence, we get that $x \in P_{q^c,q^d,e}$. □

Bibliography

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

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

[3] J. Berstel. *Transductions and Context-Free Languages*. Teubner, Stuttgart, 1979.

[4] M. Coste, J. Bochnak, and M-F. Roy. *Real Algebraic Geometry*. Ergebnisse der Math. Springer Verlag, 1998.

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

[6] Samuel Eilenberg. *Automata, Languages and Machines*. Academic Press, New York, London, 1974.

[7] Georgios E. Fainekos, Antoine Girard, and George J. Pappas. Hierarchical synthesis of hybrid controllers from temporal logic specifications. In *HSCC*, pages 203–216, 2007.

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

[9] F. Gécseg and I Peák. *Algebraic theory of automata*. Akadémiai Kiadó, Budapest, 1972.

[10] J.M.E Gonzalez, A.E.C da Cunha, J.E.R. Cury, and B.H Krogh. Supervision of event-driven hybrid systems: Modeling and synthesis. In *Hybrid Systems: Computation and Control*, volume LNCS 2034, pages 247 – 260, 2001.

[11] E. Grädel, W. Thomas, and T. Wilke. *Automata, Logic and Infinite Games*, volume LNCS 2500. Springer, 2002.

[12] L.C.G.J.M. Habets, P.J. Collins, and J.H. van Schuppen. Reachability and control synthesis for piecewise-affine hybrid systems on simplices. *IEEE Trans. Automatic Control*, 51:938–948, 2006.

[13] H.K. Khalil. *Nonlinear Systems*. Prentice Hall, 1996.

[14] B. Khoussainov and A. Nerode. *Automata Theory and its Applications*. Birkhäuser, Berlin, 2001.

[15] X.D. Koutsoukos, P.J. Antsaklis, J.A. Stiver, and M.D. Lemmon. Supervisory control of hybrid systems. *Proceedings of the IEEE*, 2000.

[16] A. Lombardy and J. Sakarovitch. Sequential ? *Theoretical Computer Science*, 356, 2006.

[17] T. Moor, J. M. Davoren, and J. Raisch. Modular supervisory control of a class of hybrid systems in a behavioural framework. In *Proc. European Control Conference*, pages 870–875, 2001.

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

[19] Thomas Moor, Jörg Raisch, and Siu O'Young. Discrete supervisory control of hybrid systems based on l-complete approximations. *Discrete Event Dynamic Systems*, 12(1):83–107, 2002.

[20] M. Petreczky, P. Collins, D.A. van Beek, J.H. van Schuppen, and J.E. Rooda. Sampled-data control of hybrid systems with discrete inputs and outputs. In *Proceedings of 3rd IFAC Conference on Analysis and Design of Hybrid Systems (ADHS09)*, 2009.

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

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

[23] P.P.H.H. Philips, W.P.M.H. Heemels, H. Preisig, and P.P.J. van den Bosch. Control of continuous-time quantised systems. *Int. J. of Control*, 76:277–294, 2003.

[24] Carla Piazza, Marco Antoniotti, Venkatesh Mysore, Alberto Policriti, Franz Winkler, and Bud Mishra. Algorithmic algebraic model checking i: Challenges from systems biology. In *CAV*, 2005.

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

[26] Jörg Raisch and Siu O'Young. A des approach to control of hybrid dynamical systems. In *Hybrid Systems*, pages 563–574, 1995.

[27] E.D. Sontag. *Mathematical Control Theory*. Spinger-Verlag, 1990.

[28] James A. Stiver, Panos J. Antsaklis, and Michael D. Lemmon. Interface and controller design for hybrid control systems. In *Hybrid Systems II*, pages 462–492, London, UK, 1995. Springer-Verlag.

[29] R.J. Stoner. Liapunob reachability and optimization in control. *Journal of Optimization Theory and Applications*, 39(3), 1983.

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

[31] D. C. Tarraf. *A Finite State Machine Framework for Robust Analysis and Control of Hybrid Systems*. PhD thesis, Massachusetts Institute of Technology, 2006.

[32] Arjan van der Schaft and Hans Schumacher. *An Introduction to Hybrid Dybnamical Systems*. Springer-Verlag London, 2000.