2015 IEEE 54th Annual Conference on Decision and Control (CDC)
December 15-18, 2015. Osaka, Japan

Trajectory-Based Observer for Hybrid Automata Fault Diagnosis

Yi Deng, Alessandro D'Innocenzo and A. Agung Julius

Abstract— **A method for constructing state observers for hybrid automata is proposed. The approach is trajectory-based. So it does not require the system to be fully observable. The discrete and continuous states of the system can be estimated constantly. We present implementation of the observer, and its application to fault diagnosis problem of hybrid automata.**

I. INTRODUCTION

Hybrid systems can effectively model interactive continuous and discrete dynamics [1], [2], [3], where the system state performs continuous flow and discrete jump (called an *event*) alternately. Their observability problem is in general more complex than that of classical dynamical systems, because of stability issue in the state estimation error [4] and involvement of the discrete states (also called *locations* or modes) estimation [5]. Intuitively, knowing the current location where the system can be is of critical importance for the estimation of continuous states. Measurement of continuous states in turn helps to improve the observability of locations. By triggering an event, the system state reaches a location from another. The event itself can output a signal, making the current event and even the current location immediately observable; but this is not always the case. When these intrinsic event-output signals are not enough to determine the current location, one can increase the event and location observability by measuring continuous states, for example, by designing residual generators [6], [7], or directly observing the output continuous states and their derivatives [8]. These methods are without doubt facilitative, but still the state estimation problem is not always extricable from unobservable events and uncertain current location for free.

In the literature, researchers have developed various methods on designing continuous state observer for linear hybrid systems [4], [6], [8]; but the continuous state observability in each location is required. In the present work, we study the state estimation problem from the perspective of bisimulation theory [9]. The basic idea is, given a trajectory simulated from an initial state, and the elapsed time of system operation, the current location and continuous state of the system can be estimated for any trajectory initiated from a neighborhood around the simulated initial state [10]. Thus, it has the following differences from classical observer design approaches: (i) The measured state variable that drives the observer is time, instead of the system output continuous states (of course, time can also be considered as a continuous system state). (ii) The estimate is not designed to be asymptotically converging to the continuous state; instead, it always stays inside a neighborhood of the continuous state, and the size of the neighborhood (error bound) can be chosen according to need.

With this approach, the observer is able to work for systems that do not have full observability. This does not mean that we can recover the current state (or initial state) of the system with arbitrarily high precision. Given the limited event observability, the approach can only estimate the current state (or initial state) as being in a set of neighborhoods, which will become more apparent in later sections.

We apply the observer to fault diagnosis of hybrid autonomous systems. System fault diagnosis is concerned with detection and isolation of faults [11]. For clarity we define faults as faulty events in this paper; but the results are also applicable to diagnosis of faulty discrete and continuous states, since the observer will estimate the state of the system. Even if we only want to diagnose faulty events, for continuous-time systems we can use more than just the sequence of the observed events (for classical discrete event system diagnosis, see [12]). For example, we can use the timing information of the events. As in our previous research [13], temporal properties of trajectories, which are closely related to continuous dynamics, can be combined with discrete dynamics to improve diagnosability. In the present work, we make more direct use of time as a state, and extract the idea of bisimulation-based state estimation. So the approach presented in this paper is different from [13] in the following aspects: (i) The observer (constructed as a finite automaton driven by exogeneous events and an external timer) actually estimate both discrete and continuous states for every time instant. (ii) Infinite time horizon is considered, given that the over-approximated reachable set is compact. (iii) At some states, we assume that the faults can happen at any time non-deterministically.

In Section II, we introduce the hybrid automata model and fault diagnosability notion. In Section III, we briefly review the robust neighborhood approach [10]. Based on that, a timed abstraction of the hybrid system is constructed. The abstraction is halfway between grid-based approaches and the approach in [14]. It is computationally feasible, and arbitrarily more precise than [14], which is very fast but very

YD and AAJ would like to acknowledge the support of NSF CAREER grant CNS-0953976.

The research leading to these results has received funding from the Italian Government under Cipe resolution n.135 (Dec. 21, 2012), project *INnovating City Planning through Information and Communication Technologies* (INCIPICT).

YD and AAJ is with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute. AD is with the Department of Engineering and Information Sciences and Mathematics Center of Excellence DEWS University of LAquila, Italy.

978-1-4799-7886-1/15/$31.00 ©2015 IEEE

conservative. In Section IV, an observer of the hybrid system state is constructed based on the timed abstraction, which can be used as a fault diagnoser. The paper [15] also studies diagnoser construction from timed automata. In our case, the timed abstraction serves the purpose of hybrid system state estimation, and belongs to a special class of timed automata (guards are intervals, and the clock is always reset to zero, as opposed to general timed automata considered in [15]). Thus, compared with [15], the diagnoser constructed in this paper maintains a finite state space, and has explicitly expressed transition functions, and is able to update state immediately when a location change of the hybrid system possibly occurs ([15] updates the observer state based on observable event interrupt and time-out interrupt). In Section V, we implement the approach with a numerical example.

II. HYBRID AUTOMATA FAULT DIAGNOSABILITY

A. System Formulation

A hybrid autonomous system is defined to be a 5-tuple $H = (L \times X, L^0 \times X^0, D, E, Inv)$ [16]:

- $L \times X$ is a set of hybrid states (ℓ, x), where $\ell \in L$ is discrete state (location), and $x \in X$ is continuous state.
- $L^0 \times X^0 \subset L \times X$ is a set of initial states.
- D associates with each location $\ell \in L$ the autonomous continuous time-invariant dynamics, $D_\ell : \dot{x} = D_\ell(x)$, which is assumed to admit a unique global solution $\xi_\ell(t, x_\ell^0)$, where ξ_ℓ satisfies $\frac{\partial \xi_\ell(t,x_\ell^0)}{\partial t} = D_\ell(\xi_\ell(t, x_\ell^0))$, and $\xi_\ell(0, x_\ell^0) = x_\ell^0$ is the initial condition in ℓ.
- $Inv : L \to X$ associates an invariant set $Inv(\ell) \subset X$ with each location. Only if the continuous state satisfies $x \in Inv(\ell)$, can the discrete state be at the location ℓ.
- E is a set of events. In each location ℓ, the system state evolves continuously according to D_ℓ until an event $e := (\ell, \ell', g, r), e \in E$ occurs. The event is guarded by $g \in Inv(\ell)$. Namely, a necessary condition for the occurrence of e is $x \in g$. After the event, the state is reset from (ℓ, x) to $(\ell', r(x))$.

When a hybrid system runs, the system state alternately flows continuously and triggers events in E. For convenience, we also define an initialization event $e^0 \notin E$. Then a trajectory of the system can be defined as a sequence:

Definition 1 (Trajectory):

$$\rho = \{(e^i, \ell^i, x^i, \tau^i)\}_{i=0}^N,$$

where

- $\forall i \geq 0$, $(\ell^i, x^i) \in L \times X$, and $(\ell^0, x^0) \in L^0 \times X^0$;
- $\forall i \geq 0$, $\tau^i \in \mathbb{R}_{\geq 0}$ (nonnegative real), and $\forall t \in [0, \tau^i]$, $\xi_{\ell^i}(t, x^i) \in Inv(\ell^i)$;
- $\forall i \geq 1$, $e^i = (\ell^{i-1}, \ell^i, g^i, r^i)$, $\xi_{\ell^{i-1}}(\tau^{i-1}, x^{i-1}) \in g^i$, $x^i = r^i(\xi_{\ell^{i-1}}(\tau^{i-1}, x^{i-1}))$, i.e., (ℓ^i, x^i) is the reset state for $(\ell^{i-1}, \xi_{\ell^{i-1}}(\tau^{i-1}, x^{i-1}))$.

Suppose there is a trajectory $\rho' = \{(e'^i, \ell'^i, x'^i, \tau'^i)\}_{i=0}^{N'}$ such that $N' \leq N$, and $\forall i \in [0, N'-1]$, $(e'^i, \ell'^i, x'^i, \tau'^i) = (e^i, \ell^i, x^i, \tau^i)$, and also $(e'^{N'}, \ell'^{N'}, x'^{N'}) = (e^{N'}, \ell^{N'}, x^{N'})$, $\tau'^{N'} \leq \tau^{N'}$, then we call ρ' a *sub-trajectory* of ρ.

B. Fault Diagnosability

Let $E^f \subset E \cup \{e^0\}$ be the set of events that model a fault. We call E^f the faulty events. Assume in some of the locations one faulty event can occur. That is, for any $\ell \in L$, the following set is either empty or a singleton:

$$Feas^f(\ell) := \{e^f \in E^f | e^f = (\ell, \ell', g, r)\}. \qquad (1)$$

It is also assumed that given $Feas^f(\ell) \neq \emptyset$, its guard is $Inv(\ell)$, i.e., the fault can happen anywhere in $Inv(\ell)$.

We start from defining trajectories that trigger a faulty event and then keep flowing for enough long time [13]:

Definition 2: A trajectory $\rho = \{(e^i, \ell^i, x^i, \tau^i)\}_{i=0}^N$ is δ-faulty if and only if there exists a finite index $i^f \in [0, N]$ such that $e^i \notin E^f$ for all $i < i^f$, $e^{i^f} \in E^f$, and $\sum_{i=i^f}^{N} \tau^i \geq \delta$.

Each event $e \in E$ has a (possibly unobservable) output symbol ψ. If the output is observable, we write $\psi \in \Psi_v$. Otherwise it is called an unobservable output symbol $\psi = \epsilon$. The initialization event $e^0 \notin E$ has the special output symbol ι (starting signal), signaling the start of fault diagnosis. Note that different events may have identical output symbols, and some events are not observable at all. Hence, we introduce the definition of *projected timed output symbol sequences*:

Definition 3: Given a trajectory $\rho = (e^i, \ell^i, x^i, \tau^i)_{i=0}^N$, the sequence of timed output symbols produced by ρ is

$$\Omega(\rho) = \omega = \{(\Delta^i, \psi^i)\}_{i=0}^{N+1},$$

where $(\Delta^0, \psi^0) = (0, \iota)$, $\Delta^i = \tau^{i-1}$, $\psi^i \in \Psi_v \cup \{\epsilon\}$ is the output symbol associated with $e^i \in E$ for all $i \in [1, N]$, and $(\Delta^{N+1}, \psi^{N+1}) = (\tau^N, \epsilon)$. In words, ρ produces a sequence of alternating time intervals and output symbols; a symbol ψ^i appears Δ^i time units later than the preceding symbol.

The observable output symbol sequence of ω is

$$\{\psi^{i_n}\}_{n=0}^{N'} \subset \Psi_v \cup \{\iota\}, N' \leq N,$$

where ψ^{i_n} is the n^{th} observable output symbol after the starting signal $\psi^{i_0} = \psi^0 = \iota$, and i_n is its index in ω.

We define the projected output symbol sequence of ω:

$$\Pi(\omega) = \pi = \{(\Delta'^n, \psi'^n)\}_{n=0}^{N'+1},$$

where $(\Delta'^0, \psi'^0) = (0, \iota)$, $\Delta'^n = \sum_{i=i_{n-1}+1}^{i_n} \Delta^i$, $\psi'^n = \psi^{i_n}$ for all $n \in [1, N']$, $(\Delta'^{N'+1}, \psi'^{N'+1}) = (\sum_{i=i_{N'}+1}^{N} \Delta^i, \epsilon)$. In words, Π absorbs all the timed output symbols with the unobservable output symbol ϵ into the first observable one that follows, while leaves the rest unchanged. If a trajectory has consecutive unobservable output symbols at its end, then the corresponding dwell time is summed, and ϵ is assigned to the end. For instance, $(\Delta^0, \psi^0), (\Delta^1, \epsilon), (\Delta^2, \psi^2)$ is projected to $(\Delta^0, \psi^0), (\Delta^1 + \Delta^2, \psi^2)$, $\psi^2 \in \Psi_v \cup \{\epsilon\}$.

Definition 4 (δ_d-Diagnosability): H is δ_d-diagnosable if it does not have a δ_d-faulty trajectory ρ and a normal trajectory ρ' such that $\Pi(\Omega(\rho)) = \Pi(\Omega(\rho'))$.

In this definition δ_d is the delay parameter, characterizing the maximum time delay allowed to diagnose a fault. Similar definitions can be found in [14], [15].

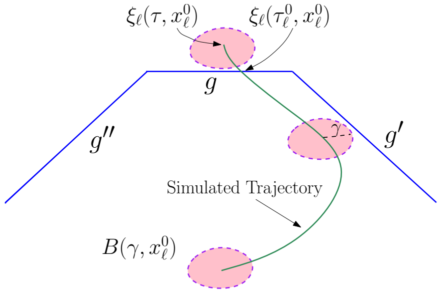

Fig. 1. Any trajectory initiated from $B(\gamma, x^0_\ell)$ will not reach g'.

III. SYSTEM ABSTRACTION

A. Robust Neighborhood Approach

In this section, we briefly review the robust neighborhood approach proposed in [10], which is based on the bisimulation theory [9].

Definition 5: [9] Let $\phi_\ell(x_1, x_2) : X \times X \rightarrow \mathbb{R}$ be a pseudo-metric on the state space of the dynamical system

$$\dot{x} = D_\ell(x), x \in X.$$

Let $\xi_\ell(t, x^0_\ell)$ denote the solution of D_ℓ under the initial condition x^0_ℓ. If for any initial states x^0_ℓ and $\tilde{x}^0_\ell$, the function $\phi_\ell(\xi_\ell(t, x^0_\ell), \xi_\ell(t, \tilde{x}^0_\ell))$ is non-increasing with respect to time t, then ϕ_ℓ is a bisimulation function between the system and itself.

For any $\gamma > 0, t > 0$, if $\phi_\ell(x^0_\ell, \tilde{x}^0_\ell) < \gamma$, then

$$\begin{aligned} \phi_\ell(\xi_\ell(t, x^0_\ell), \xi_\ell(t, \tilde{x}^0_\ell)) &\leq \phi_\ell(\xi_\ell(0, x^0_\ell), \xi_\ell(0, \tilde{x}^0_\ell)) \\ &= \phi_\ell(x^0_\ell, \tilde{x}^0_\ell) \\ &< \gamma. \end{aligned}$$

Thus, ϕ_ℓ can be used to bound continuous state divergence of trajectories. The ball of ϕ_ℓ is denoted as

$$B_\ell(\gamma, x^0_\ell) := \{x | \phi_\ell(x, x^0_\ell) < \gamma\}. \quad (2)$$

Let $e = (\ell, \ell', g, r)$ be an event triggered by a trajectory simulated from x^0_ℓ. If we want all trajectories initiated from within $B_\ell(\gamma, x^0_\ell)$ to avoid reaching a location ℓ'' through $e' = (\ell, \ell'', g', r')$, then we can let

$$\gamma = \inf_{y \in g'} \inf_{t \in [0, \tau]} \phi_\ell(\xi_\ell(t, x^0_\ell), y),$$

where τ is sum of the time for the simulated trajectory to transition out of ℓ and the allowed lag for other trajectories initiated from $B_\ell(\gamma, x^0_\ell)$ to transition out of ℓ compared with the simulated one. Then by the preceding argument, for any $\tilde{x}^0_\ell \in B_\ell(\gamma, x^0_\ell), t \in [0, \tau]$, $\xi_\ell(t, \tilde{x}^0_\ell)$ cannot reach g' that guards e'. See Fig. 1 for illustration.

With the basic idea reviewed above, the robust neighborhood approach is to compute a neighborhood around a simulated initial state, such that any trajectory initiated from the neighborhood will reach the same location sequence as the simulated one, and the continuous state always stays inside a neighborhood around the continuous state of the simulated trajectory. The specific algorithm is presented in [10], which computes robust neighborhoods $B_{\ell^i}(\gamma^i, x^i)$ around the (reset) initial continuous states x^i of a simulated trajectory $\rho = \{(e^i, \ell^i, x^i, \tau^i)\}^N_{i=0}$. Formally, the following property holds [10]:

Proposition 1: For any covered initial state $(\tilde{\ell}^0, \tilde{x}^0) \in \{\ell^0\} \times B_{\ell^0}(\gamma^0, x^0)$, for any trajectory $\tilde{\rho}'$ initiated from $(\tilde{\ell}^0, \tilde{x}^0)$, there is a trajectory $\tilde{\rho} = \{(\tilde{e}^i, \tilde{\ell}^i, \tilde{x}^i, \tilde{\tau}^i)\}^{\tilde{N}}_{i=0}$ such that $\tilde{\rho}$ is a sub-trajectory of $\tilde{\rho}'$ or $\tilde{\rho}'$ is a sub-trajectory of $\tilde{\rho}$, and $\rho, \tilde{\rho}$ satisfy

- $\tilde{N} = N$; for all $i \in [0, N]$, $\tilde{e}^i = e^i$, $\tilde{\ell}^i = \ell^i$, $\tilde{x}^i \in B_{\ell^i}(\gamma^i, x^i)$, $\tilde{\tau}^i \in [\tau^i - lead^i, \tau^i + lag^i]$, and $\phi_{\ell_i}(\xi_{\ell^i}(t, x^i), \xi_{\tilde{\ell}^i}(t, \tilde{x}^i)) \leq \gamma^i$ for all $t \in [0, \tilde{\tau}^i]$.

Following the above idea, we simulate normal trajectories $\rho_k = \{(\ell^n_k, x^n_k, e^n_k, \tau^n_k)\}^{N_k}_{n=0}$, $k \in [1, K]$ from the initial set $L^0 \times X^0$, and compute their robust neighborhoods. Faulty events are not considered in the simulation and neighborhood computation. The robust neighborhood around the (reset) initial state x^n_k in location ℓ^n_k is denoted as

$$\begin{aligned} Ball(k, n) &= B_{\ell^n_k}(\gamma^n_k, x^n_k) \\ &= \{x | \phi_{\ell^n_k}(x^n_k, x) < \gamma^n_k\}, \end{aligned} \quad (3)$$

where $\phi_{\ell^n_k}$ is the bisimulation function in location ℓ^n_k, and γ^n_k is the radius of the computed robust neighborhood. It is assumed the robust neighborhoods around the simulated initial states $(\ell^0_k, x^0_k), k \in [1, K]$ fully cover $L^0 \times X^0$, i.e.,

$$\bigcup_{1 \leq k \leq K} \{\ell^0_k\} \times Ball(k, 0) \supset L^0 \times X^0. \quad (4)$$

Let $[\tau_1, \tau_2]$ be a time interval. We define the robust tube around the trajectory segment indexed by (k, n) for the interval $[\tau_1, \tau_2]$ as

$$\begin{aligned} & Tube(k, n, [\tau_1, \tau_2]) \\ := & \bigcup_{t \in [\tau_1, \tau_2]} B_{\ell^n_k}(\gamma^n_k, \xi_{\ell^n_k}(t, x^n_k)) \cap Inv(\ell^n_k). \end{aligned} \quad (5)$$

We assume that for all $\tilde{k} \in [1, K]$, there exists

$$Cover(\tilde{k}) \subset \{(k, n) | 1 \leq k \leq K, 0 \leq n \leq N_k\}$$

such that

$$Tube(\tilde{k}, N^{\tilde{k}}, [\tau^{N_{\tilde{k}}}_{\tilde{k}}, \tau^{N_{\tilde{k}}}_{\tilde{k}}]) \subset \bigcup_{(k,n) \in Cover(\tilde{k})} Ball(k, n). \quad (6)$$

Clearly, $Tube(\tilde{k}, N^{\tilde{k}}, [\tau^{N_{\tilde{k}}}_{\tilde{k}}, \tau^{N_{\tilde{k}}}_{\tilde{k}}])$ represents the end of the robust tube around $\rho_{\tilde{k}}$. So Assumption Eq. (6) means that the ends of all the tubes around the normal trajectories are covered by some robust neighborhoods. According to Proposition 1 and Assumption Eq. (6), all the trajectories will stay inside the tubes around the simulated normal trajectories for infinitely long horizon, as long as no faulty event has been triggered. In other words, we are assuming that the over-approximated reachable set (robust tubes) of the system for infinite time horizon is compact. See Fig. 2 for illustration.

Remark 1: The assumption above on infinite-horizon compactness is made for normal trajectories of the system. When a fault occurs, we need to diagnose it within maximum time delay. So faulty trajectories only need finite-horizon

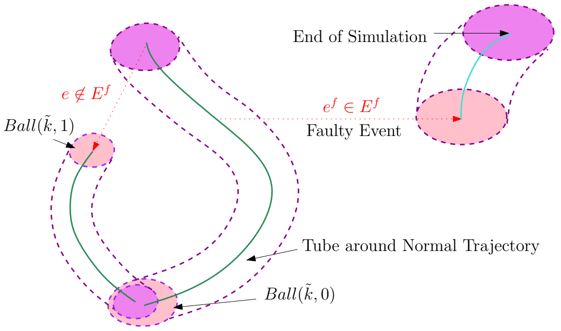

Fig. 2. The end of the robust tube around the normal trajectory $\rho_{\tilde{k}}$ (left) is covered by the robust neighborhood $Ball(\tilde{k},0)$, i.e., $Cover(\tilde{k}) = \{(\tilde{k},0)\}$. Any trajectory initiated from $Ball(\tilde{k},0)$ will stay inside the tube around $\rho_{\tilde{k}}$ until it triggers the faulty event.

analysis. If normal trajectories are studied also for finite time horizon, then the assumption can be removed.

Next, we simulate (faulty part of) the faulty trajectories corresponding to the locations $\ell_k^n, 1 \leq k \leq K, 0 \leq n \leq N_k$ such that $Feas^f(\ell_k^n) \neq \emptyset$. Namely, ℓ_k^n are the source locations of the faulty events.

For any ℓ_k^n, we simulate a set of faulty trajectories indexed by $Ind^f(k,n)$ such that the inverse image of the robust neighborhoods around their initial states cover the entire $Tube(k,n,[0,\tau_k^n + lag_k^n])$. That is,

$$\begin{aligned} & Tube(k,n,[0,\tau_k^n + lag_k^n]) \\ \subset & \bigcup_{\hat{k} \in Ind^f(k,n)} r^{-1}(Ball(\hat{k},0)), \end{aligned} \tag{7}$$

where $r^{-1}(\cdot)$ denotes the inverse image through the reset map of the faulty event.

The set of all the (faulty part of) faulty trajectories simulated for all ℓ_k^n is denoted as $\{\rho_k\}_{k=K+1}^{\hat{K}}$. Namely, we have simulated $\{\rho_k\}_{k=1}^{\hat{K}}$, where $k \in [1,K]$ are normal, and $k \in [K+1,\hat{K}]$ are faulty.

B. Timed Abstraction

Based on the simulated trajectories $\{\rho_k\}_{k=1}^{\hat{K}}$, we construct a timed automaton [17] that is an abstraction of H.

Definition 6: Define $T = (Q, Q^0, C, \tilde{E}, \tilde{Inv})$:

- The state space is $Q := \{(k,n) | k \in \{1,\ldots,\hat{K}\}, n \in \{0,\ldots,N_k\}\} \cup \{EoS\}$ (EoS means end of simulation).
- The initial set is $Q^0 := \{1,\ldots,K\} \times \{0\}$.
- The set of clock C is a singleton $\{c\}$.
- The events $\tilde{e} \in \tilde{E}$ are defined as $\tilde{e} = (q,q',\tilde{g},\tilde{r})$ such that $\tilde{r}(c) = 0$, i.e., the only clock is reset after any event, and one of the following cases should be satisfied:
  1) $q = (k,n)$, where $n < N_k$; $q' = (k,n+1)$; and $\tilde{g} = [\tau_k^n - lead_k^n, \tau_k^n + lag_k^n]$; $\tilde{e}$ is associated with the output symbol of e_k^{n+1};
  2) $q = (k,N_k)$, where $k \in [1,K]$; $q' = (k',n')$, where $(k',n') \in Cover(k)$; and $\tilde{g} = [\tau_k^{N_k}, \tau_k^{N_k}]$; $\tilde{e}$ is associated with the unobservable ϵ;
  3) $q = (k,N_k)$, where $k \in [K+1,\hat{K}]$; $q' = EoS$ (end of simulation); and $\tilde{g} = [\tau_k^{N_k}, \tau_k^{N_k}]$; $\tilde{e}$ is associated with the unobservable ϵ;
  4) $q = (k,n)$; $q' = (k',0)$, where $k' \in Ind^f(k,n)$; and $\tilde{g} = [0, \tau_k^n + lag_k^n]$; $\tilde{e}$ is associated with the output symbol of the faulty event $Feas^f(\ell_k^n)$.
- The invariant set is $\tilde{Inv}(q) := [0, \tau_k^n + lag_k^n]$ if $n < N_k$, $\tilde{Inv}(q) := [0, \tau_k^n]$ if $n = N_k$, where $q = (k,n) \in Q$.

We assume that all the numbers that appear in the clock constraints are rational. This is realizable since the lead/lag times are flexible. We can relax the lead/lag constraints by arbitrarily small amount to make them rational. From this point on we assume the clock constraints only involve integers, since any rational timed automaton has an integer counterpart whose runs are isomorphic to the original runs [17].

Since timed automata can be considered as a subclass of hybrid automata, trajectories and projected timed output symbol sequences can be defined the same way as before. By construction, for any normal trajectory ρ of H, there is a trajectory $\tilde{\rho}$ of T such that $\Pi(\Omega(\rho)) = \Pi(\Omega(\tilde{\rho}))$. For faulty trajectories, a similar property holds for finite horizon.

IV. OBSERVER

Based on the timed abstraction T, we construct for H an observer O. By using the history of system output, i.e., a projected timed output symbol sequence, O over-approximates the current state reached by H. Each state s of O can be represented by a subset of the set

$$\begin{aligned} \{(k,n)[\bar{a},\bar{b}] \quad | \quad & k,n,\bar{a},\bar{b} \text{ are integers,} \\ & 1 \leq k \leq \hat{K}, 0 \leq n \leq N_k, \\ & [\bar{a},\bar{b}] \subset \tilde{Inv}((k,n))\}. \end{aligned}$$

The state of the observer being just updated to s means that the system H must be at some state within $\bigcup_{(k,n)[\bar{a},\bar{b}] \in s} Tube(k,n,[\bar{a},\bar{b}] \cap \mathbb{R}_{\geq 0})$.

Given $T = (Q, Q^0, C, \tilde{E}, \tilde{Inv})$, for each $(k,n) \in Q$, let $Feas : Q \to 2^{\tilde{E}}$ be the feasible event function:

$$Feas((k,n)) := \{\tilde{e} \in \tilde{E} | \tilde{e} = ((k,n),(k',n'),\tilde{g},\tilde{r})\}. \tag{8}$$

For $(k,n)[\bar{a},0]$, we define the $\epsilon[0]$-successors:

$$\begin{aligned} & Succ^{\epsilon[0]}((k,n)[\bar{a},0])) \\ := & \{(k',n')[\bar{a}-\tilde{b},0] | \exists \tilde{e} = ((k,n),(k',n'),[0,\tilde{b}],\tilde{r}) \\ & \in Feas((k,n)), \tilde{e} \text{ outputs } \epsilon\}. \end{aligned} \tag{9}$$

The $\epsilon[0]$-closure of $(k,n)[\bar{a},0]$, denoted as $Cl^{\epsilon[0]}(q[\bar{a},0])$, is defined to be union of $(k,n)[\bar{a},0]$, the $\epsilon[0]$-successors of $(k,n)[\bar{a},0]$, and the $\epsilon[0]$-successors of all the $\epsilon[0]$-successors of $(k,n)[\bar{a},0]$.

Given s as a set of $(k,n)[\bar{a},\bar{b}]$, the $\epsilon[0]$-closure of s is then

$$Cl^{\epsilon[0]}(s) := s \cup \{Cl^{\epsilon[0]}((k,n)[\bar{a},0]) | (k,n)[\bar{a},0] \in s\}. \tag{10}$$

Definition 7 (Observer): We construct $O = (S, s^0, \bar{\Sigma}, f)$ by the following steps, where $S, S^0, \bar{\Sigma}, f$ are the state space, initial state, transition labels and transition function:

1) Define $s^0 := Cl^{\epsilon[0]}(\{(1,0)[0,0], \ldots, (K,0)[0,0])\})$. Set $S = \{s^0\}$.
2) For each new state $s \in S$, for each $\bar{q} = (k,n)[\bar{a}, \bar{b}] \in s$, compute the following, where $\tilde{e} \in Feas((k,n))$ has the guard $[\tilde{a}, \tilde{b}]$ and outputs the symbol ψ:

$$\begin{aligned} Blank(\bar{q}, \tilde{e}) &:= \begin{cases} \tilde{a} - \bar{b}, \text{ if } \psi = \epsilon, \bar{b} < \tilde{a}, \\ \tilde{b} - \bar{a}, \text{ if } \psi \in \Psi_v, \bar{a} < \tilde{b}, \\ \infty, \text{ otherwise.} \end{cases} \\ Blank^{\tilde{Inv}}(\bar{q}) &:= \tilde{b} - \bar{a}. \\ Blank_{min}(s) &:= \min\{Blank^{\tilde{Inv}}(\bar{q}), \\ &\qquad \min_{\bar{q} \in s, \tilde{e} \in Feas((k,n))} Blank(\bar{q}, \tilde{e})\}. \end{aligned}$$

Add the label $\epsilon[\Delta t], \Delta t = Blank_{min}(s)$ to $\bar{\Sigma}$.

$$\begin{aligned} & f'(\bar{q}, \epsilon[\Delta t]) \\ := \; & Cl^{\epsilon[0]}(\{(k',n')[\bar{a} + \Delta t - \tilde{b}, 0] | \exists \tilde{e} = \\ & ((k,n),(k',n'),[\tilde{a}, \tilde{b}], \tilde{r}) \in Feas((k,n)), \\ & \tilde{e} \text{ outputs } \epsilon, \bar{b} + \Delta t = \tilde{a}\}) \cup \\ & \{(k,n)[\bar{a} + \Delta t, \bar{b} + \Delta t] \cap \tilde{Inv}((k,n)) | \\ & (\bar{a} + \Delta t, \bar{b} + \Delta t] \cap \tilde{Inv}((k,n)) \neq \emptyset\}. \\ & f(s, \epsilon[\Delta t]) \\ := \; & \{f'(\bar{q}, \epsilon[\Delta t]) | \bar{q} \in s\}. \end{aligned}$$

3) For each new state $s \in S$, check if there exist $\bar{q} = (k,n)[\bar{a}, \bar{b}] \in s$ and $\tilde{e} \in Feas((k,n))$ that has the guard $[\tilde{a}, \tilde{b}]$ and outputs $\psi \in \Psi_v$ such that

$$([\max\{0, \tilde{a} - \bar{b}\}, \tilde{b} - \bar{a}] \setminus \{0\}) \cap (0, Blank_{min}(s)] \neq \emptyset.$$

If so, define $a := \max\{0, \tilde{a} - \bar{b}\}, b := Blank_{min}(s)$, $\bar{\sigma}' := \psi\langle a, b]$, where $\langle a, b]$ stands for $(a, b]$ if $a = 0$, $[a, b]$ if $a > 0$. Classify the obtained labels $\bar{\sigma}'$ into the sets $[\bar{\sigma}']_\psi$ according to distinct ψ.
For each $[\bar{\sigma}']_\psi = \{\psi\langle a_1, b], \psi\langle a_2, b], \ldots\}$, order the distinct a_i values increasingly and let the result be $a_{(1)} < \ldots < a_{(m)}$. Then add to $\bar{\Sigma}$ the transition labels $\{\psi\langle a_{(1)}, a_{(2)}), \ldots, \psi\langle a_{(m-1)}, a_{(m)}), \psi\langle a_{(m)}, b]\}$.
For $\bar{\sigma} = \psi\langle a, b]$ or $\psi\langle a, b)$, $\psi \in \Psi_v$, define

$$\begin{aligned} & f'(\bar{q}, \bar{\sigma}) \\ := \; & Cl^{\epsilon[0]}(\{(k',n')[0,0] | \exists \tilde{e} = \\ & ((k,n),(k',n'),[\tilde{a}, \tilde{b}], \tilde{r}) \in Feas((k,n)), \\ & \tilde{e} \text{ outputs } \psi, \tilde{b} - \bar{a} \geq b, \tilde{a} - \bar{b} \leq a\}). \\ & f(s, \bar{\sigma}) \\ := \; & \{f'(\bar{q}, \bar{\sigma}) | \bar{q} \in s\}. \end{aligned}$$

4) If $s' := f(s, \bar{\sigma}) \notin S$, add the new state s' to S.
5) Repeat Steps 2-4 until no new states are created.

The observer is constructed as a deterministic finite automaton driven by an external timer and output symbols observed from H. Whenever the observer reaches a new state, the timer reading t is immediately reset to 0.

Proposition 2: Given that the current state of O is s, and the state of the timer is t, then the state of H is in $\{(\ell_k^n, x) | (k,n)[\bar{a}, \bar{b}] \in s, x \in Tube(k, n, [\bar{a}+t, \bar{b}+t] \cap \mathbb{R}_{\geq 0})\}$.

Proof: Directly follow from construction of O. ■

By checking the reachable states of O, we can analyze the δ_d-diagnosability of H as follows:

Let $\bar{Q}^0$ denote the set

$$\{\bar{q}^0 = (k,0)[\bar{a}, 0] | k \in [K+1, \hat{K}], \bar{q}^0 \in s^0 \in S\}.$$

Given $\bar{q}^0 \in \bar{Q}^0$, and sequences $\{\bar{\sigma}^i\}_{i=1}^m, \{\bar{q}^i\}_{i=1}^m$ such that for all $i \in [1, m]$, $\bar{q}^i \in f'(\bar{q}^{i-1}, \bar{\sigma}^i)$, we define $Delay(\bar{q}^0, \{\bar{\sigma}^i\}_{i=1}^m)$ for the following cases:

- If there exists $m' \in [1, m]$ such that $k' \in [K+1, \hat{K}]$ for all $(k', n')[\bar{a}', \bar{b}'] \in s^{m'}$, where $s^{m'} \ni \bar{q}^{m'}$, then

$$Delay(\bar{q}^0, \{\bar{\sigma}^i\}_{i=1}^m) := \sum_{i=1}^{m'} time(\bar{\sigma}^i), \text{ where}$$

$$time(\bar{\sigma}) := \begin{cases} b, \text{ if } \bar{\sigma} = \psi\langle a, b) \text{ or } \psi\langle a, b], \\ \Delta t, \text{ if } \bar{\sigma} = \epsilon[\Delta t]. \end{cases}$$

- If the case above is not satisfied, while $s^m \ni \bar{q}^m$ has no outgoing transitions, then $Delay(\bar{q}^0, \{\bar{\sigma}^i\}_{i=1}^m) := \infty$. Essentially, this case means the simulation horizon of the faulty trajectories is not long enough to discriminate faulty trajectories from normal trajectories.

The hybrid automaton H is δ_d-diagnosable, where

$$\delta_d = \max_{\bar{q}^0 \in \bar{Q}^0} \max_{\{\bar{\sigma}^i\}_{i=1}^m} Delay(\bar{q}^0, \{\bar{\sigma}^i\}_{i=1}^m). \tag{11}$$

V. IMPLEMENTATION EXAMPLE

We implement the approach to a hybrid system that has 4 locations. The continuous dynamics in location ℓ_i is given by $\dot{x} = A_i x + b_i, i \in \{1, 2, 3, 4\}$, where $x = [x_1, x_2, x_3, x_4]^T$, A_i are diagonal matrices with a_i on the diagonals,

$$a_1 = \begin{pmatrix} -1 \\ -2 \\ -3 \\ -4 \end{pmatrix}, a_2 = a_3 = a_4 = \begin{pmatrix} -4 \\ 2 \\ 3 \\ -1 \end{pmatrix},$$

$$b_1 = \begin{pmatrix} 100 \\ 200 \\ 300 \\ 400 \end{pmatrix}, b_2 = b_3 = b_4 = \begin{pmatrix} 300 \\ 200 \\ 100 \\ 400 \end{pmatrix}.$$

The invariant sets are $Inv(\ell_1) = \{x | x_2 + 2x_3 \leq 25\}, Inv(\ell_2) = \{x | 2x_1 + x_2 \leq 15\}, Inv(\ell_3) = \{x | 2x_1 + x_2 \leq 25\}, Inv(\ell_4) = \mathbb{R}^4$. On the boundaries of the invariant sets, three events are modeled:
$e_{11} = (\ell_1, \ell_1, g_{11}, r_{11})$, $g_{11} = \{x | x_2 + 2x_3 = 25\}$, $r_{11}(x) = 0.01x + [1, 0, 0, 2]^T$; $e_{23} = (\ell_2, \ell_3, g_{23}, r_{23})$, $g_{23} = \{x | 2x_1 + x_2 = 15\}$, $r_{23}(x) = x$; $e_{34} = (\ell_3, \ell_4, g_{34}, r_{34})$, $g_{34} = \{x | 2x_1 + x_2 = 25\}$, $r_{34}(x) = x$.

Assume that a state in ℓ_1 can incur a fault and transition to ℓ_2 from anywhere in $Inv(\ell_1)$. Based on that we define a faulty event $e_{12} = (\ell_1, \ell_2, g_{12}, r_{12})$, where $g_{12} = Inv(\ell_1)$, $r_{12}(x) = 0.01x + [0, 0, 1, 1]^T$ is the reset map.

Let $L^0 \times X^0 = \{\ell_1\} \times \{x | 1 \leq x_1 \leq 1.1, 0 \leq x_2 \leq 0.1, 0 \leq x_3 \leq 0.1, 2 \leq x_4 \leq 2.2\}$ be the initial set. By using the MATLAB Toolbox STRONG [18], we can verify that $L^0 \times X^0$ is covered by a robust neighborhood

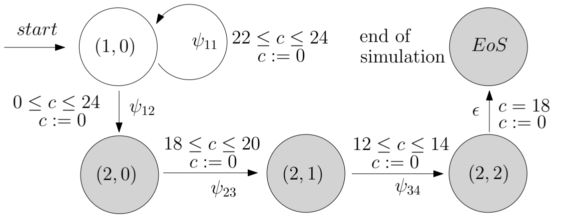

Fig. 3. Timed abstraction T of the hybrid automaton H.

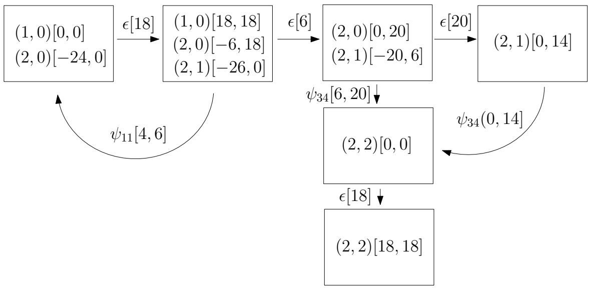

Fig. 4. State observer O of the hybrid automaton H. H is 24-diagnosable.

$B_{\ell_1}(0.14, x_1^0) = \{x|\phi_{\ell_1}(x, x_1^0) < 0.14\}$ around the initial state of the trajectory

$$\rho_1 = (e^0, \ell_1, x_1^0, 0.023), x_1^0 = [1.039, 0.046, 0.068, 2.108]^T,$$

where ϕ_{ℓ_i} is a bisimulation function computed with YALMIP Toolbox [19] for ℓ_i. The last state of ρ_1, $\xi_{\ell_1}(0.023, x_1^0)$, triggers the event e_{11}. Also we can verify that $B_{\ell_1}(0.14, \xi_{\ell_1}(0.023, x_1^0))$ is covered by $r_{11}^{-1}(B_{\ell_1}(0.14, x_1^0))$. Thus, any trajectory initiated from $L^0 \times X^0$ will trigger a sequence of e_{11}, and the dwell time in ℓ_1 will satisfy $\tau \in [0.023 - lead, 0.023 + lag]$, where $lead = lag = 0.001$ are transition time lead and lag in the robust neighborhood computation.

Next ,we use STRONG Toolbox to cover the robust tube $Tube(1, 0, [0, 0.024])$ around the initial segment (also the only segment) of ρ_1. For $lead = lag = 0.001$, the inverse image through r_{12} of the robust neighborhood computed around the initial state of the following faulty part of trajectory will cover the entire tube.

$$\begin{aligned}
\rho_2 &= (e_{12}, \ell_2, x_2^0, 0.019), (e_{23}, \ell_3, x_2^1, 0.013),\\
&\quad (e_{34}, \ell_4, x_2^2, 0.018),\\
x_2^0 &= [0.023, 0.026, 1.039, 1.070]^T,\\
x_2^1 &= [5.539, 3.922, 3.067, 8.619]^T,\\
x_2^2 &= [9.138, 6.723, 4.548, 13.791]^T.
\end{aligned}$$

Converting rationals to integer time units, we abstract H as a timed automaton in Fig. 3.

The output symbol of an event e_{ij} is denoted by ψ_{ij}. Suppose $\psi_{12} = \psi_{23} = \epsilon$, $\psi_{11} = \psi_{34} \in \Psi_v$. The observer is shown in Fig. 4.

VI. CONCLUSIONS

We presented an approach to trajectory-based observer construction for hybrid automata. The approach relies on bisimulation theory and robust neighborhood approach [9], [10] to abstract the original system by simulating finitely many trajectories. Based on the abstraction, an observer is constructed as a finite automaton, which provides estimates for the discrete and continuous states of the hybrid automaton constantly. We applied the observer to fault diagnosis as well as fault diagnosability analysis for the hybrid automaton.

REFERENCES

[1] R. Alur, C. Belta, F. Ivancic, V. Kumar, M. Mintz, G. J. Pappas, H. Rubin, and J. Schug, "Hybrid modeling and simulation of biomolecular networks," in *Proc. Hybrid Syst.: Comput. and Control*, Rome, Italy, 2001, pp. 19–32.

[2] J. P. Hespanha, S. Bohacek, K. Obraczka, and J. Lee, "Hybrid modeling of tcp congestion control," in *Proc. Hybrid Syst.: Comput. and Control*, Rome, Italy, 2001, pp. 291–304.

[3] H. D. Jong, J. L. Goué, C. Hernandez, M. Page, T. Sari, and J. Geiselmann, "Hybrid modeling and simulation of genetic regulatory networks: A qualitative approach," in *Proc. Hybrid Syst.: Comput. and Control*, Prague, Czech Republic, 2003, pp. 267–282.

[4] A. Alessandri and P. Coletta, "Design of luenberger observers for a class of hybrid linear systems," in *Proc. Hybrid Syst.: Comput. and Control*, Rome, Italy, 2001, pp. 7–18.

[5] M. Babaali and G. J. Pappas, "Observability of switched linear systems in continuous time," in *Proc. Hybrid Syst.: Comput. and Control*, ser. Lecture Notes in Comput. Sci. Springer Berlin Heidelberg, 2005, vol. 3414, pp. 103–117.

[6] A. Balluchi, L. Benvenuti, M. D. Di Benedetto, and A. L. Sangiovanni-Vincentelli, "Design of observers for hybrid systems," in *Proc. Hybrid Syst.: Comput. and Control*, Stanford, CA, 2002, pp. 76–89.

[7] M. Bayoudh, L. Travé-Massuyes, X. Olive, and T. A. Space, "Hybrid systems diagnosis by coupling continuous and discrete event techniques," in *Proc. IFAC World Congr.*, Seoul, Korea, 2008, pp. 7265–7270.

[8] P. Collins and J. H. van Schuppen, "Observability of piecewise-affine hybrid systems," in *Proc. Hybrid Syst.: Comput. and Control*, Philadelphia, PA, 2004, pp. 265–279.

[9] A. Girard and G. J. Pappas, "Approximation metrics for discrete and continuous systems," *IEEE Trans. Autom. Control*, vol. 52, no. 5, pp. 782–798, May 2007.

[10] A. A. Julius, G. E. Fainekos, M. Anand, I. Lee, and G. J. Pappas, "Robust test generation and coverage for hybrid systems," in *Proc. Hybrid Syst.: Comput. and Control*, Pisa, Italy, 2007, pp. 329–342.

[11] J. J. Gertler, "Survey of model-based failure detection and isolation in complex plants," *IEEE Control Syst. Mag.*, vol. 8, no. 6, pp. 3–11, Dec. 1988.

[12] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Diagnosability of discrete-event systems," *IEEE Trans. Autom. Control*, vol. 40, no. 9, pp. 1555–1575, Sept. 1995.

[13] Y. Deng, A. D'Innocenzo, M. D. Di Benedetto, S. Di Gennaro, and A. A. Julius, "Verification of hybrid automata diagnosability with measurement uncertainty," *IEEE Trans. Autom. Control*, to be published.

[14] M. D. Di Benedetto, S. Di Gennaro, and A. D'Innocenzo, "Verification of hybrid automata diagnosability by abstraction," *IEEE Trans. Autom. Control*, vol. 56, no. 9, pp. 2050–2061, Sept. 2011.

[15] S. Tripakis, "Fault diagnosis for timed automata," in *Formal Techniques in Real-Time and Fault-Tolerant Syst.*, ser. Lecture Notes in Comput. Sci. Berlin, Germany: Springer, 2002, vol. 2469, pp. 205–221.

[16] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P. H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," *Theoretical Comput. Sci.*, vol. 138, no. 1, pp. 3–34, Feb. 1995.

[17] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical Comput. Sci.*, vol. 126, no. 2, pp. 183–235, Apr. 1994.

[18] Y. Deng, A. Rajhans, and A. A. Julius, "Strong: A trajectory-based verification toolbox for hybrid systems," in *Proc. Quantitative Evaluation of Syst.*, Buenos Aires, Argentina, 2013, pp. 165–168.

[19] J. Lofberg, "Yalmip : a toolbox for modeling and optimization in matlab," in *Proc. 13th IEEE Int. Symp. Comput. Aided Control Syst. Design*, Taipei, Taiwan, 2004, pp. 284–289.