Proceedings of the 8th IEEE Mediterranean Conference
on Control and Automation (MED 2000)
Rio, Patras, GREECE 17-19 July, 2000

CONTRIBUTION TO THE FAULT DETECTION FOR HYBRID SYSTEMS†

G. K. FOURLAS, K. J. KYRIAKOPOULOS, N. J. KRIKELIS

Control Systems Laboratory, Mechanical Eng. Dept. National Technical University of Athens, NTUA, Athens 10682, Greece e-mail: {gfourlas,kkyria,nkrik}@central.ntua.gr

Abstract. In this work we propose an approach to the problem of failure diagnosis for Hybrid Systems (HS). This approach is applicable to a wide rage of systems since hybrid systems involve both continuous and discrete dynamics. The states of the HS model reflect the normal and the failed status of the system components. The faults in our setting are modeled as either discrete or continuous (detrimental) state changes.

Key Words. Fault diagnosis, failure detection, hybrid systems.

1. INTRODUCTION

A number of large-scale dynamic systems can be viewed as Hybrid Systems (HS). By definition HS are systems for which the state space may change [1]. This is useful in modeling component failures. Plant sensors offer measurements of the state space variables. In a fault detection process we have to answer whether a transition from the normal to a faulted state has occurred.

The continuous evolution of the system is disturbed by a discrete change in the plant caused by a fault. The discrete changes appear from the low level of the system (plant). In most cases the signal fault is not measurable so the decision will have to be based on the measurable inputs and outputs of the system. Faults in most cases take place instantaneously and cause a qualitative change in the dynamics of the plant in the sense that the dynamics before and after the transition are qualitatively different. Therefore fault diagnosis is inherently a hybrid process since a discrete transition has to be inferred from the continuous input-output measurements.

The first problem, which is imposed, is how fault detection can be done via input-output measurements. So a system model definition is required which can be used for the description of the plant and can detect the change in dynamics.

In this work we propose an approach to the problem of failure diagnosis for Hybrid Systems. This approach is applicable to a wide rage of systems since hybrid systems involve both continuous and discrete dynamics. The states of the HS model reflect the normal and the failed status of the system components. The faults in our setting are treated at two levels: first as a discrete state change and second as a continuous state change.

In this paper we develop a framework for our approach. The behavior of the system is modeled by a HIOA (Hybrid Input/Output Automaton) [4] since this is capable of describing both the continuous and the discrete behavior.

The system is assumed to consist of several distinct components (i.e. actuators, main structure and sensors) and a controller.

- We first built a HIOA for each component, to capture both normal and failed behavior.
- Next we compose these individual models using the same composition procedure as in [4]. The overall model will be the composition of a number of automata. (So the plant will be a hybrid automaton containing the dynamics of all components).
- The faults can be modeled:
  - by discrete transitions from the normal to faulted state, or

† This work was partially supported by the ARCHIMEDES Basic Research Initiative of the Institute of Communication and Computer Systems at NTUA.

- as deviation of trajectories describing the continuous evolution from the predefined set point.

- We construct the "Diagnoser" which is a HIOA that detects the occurrence of a fault and generates a signal for the fault occurrence.

We are only interested in abrupt faults, which occur in the components of the plant, especially faults occurring to actuators. When a fault occurs it is due to the actuators (main structure, controller and sensors are fail-safe).

2. THE SYSTEM MODEL

Traditionally, in fault diagnosis, a plant to be automated can be considered to consist of three major types of subsystems: actuators, main structure and sensors. A fault-monitoring scheme is usually designed especially to detect and correct faults in only one of those three subsystems [6]. The design of this fault diagnosis scheme has a different aspect depending on what kinds of models are used for the system and for the fault mode descriptions. Before presenting our framework a few definitions will be provided.

A number of pre-determined state-variables characterize the dynamic behavior of the system.

Definition 1: *A process is said to be in a normal state of operation if its observed state-variables are in the neighborhood of a predefined set point*.

The *state of fault* or *failure* is observed by an output value of the pre-determined variables either if the operating point lies outside of the neighborhood of the predefined set point or certain functional criteria are violated.

Definition 2: *Faults (or failures) are malfunctions disturbing the normal operation of the system, causing an unacceptable decay of its performance and are modeled as transitions from a normal state to a failure state which correspond either to discrete state change or to continuous to continuous state change*.

The faults may occur at any of the components of the main structure, the actuators, or the sensors of the plant. The effects that can cause true or false alarms are due to [2]:

- Faults of the components (any of, the main structure, actuators, or sensors).
- Modeling errors between the actual system and its mathematical model and
- System or measurement noise.

The faults according to the mode, which may occur, are classified as:

- Abrupt faults that cause significant changes in the behavior of the system and play role in safety-relevant systems.
- Incipient faults that are small and are relevant in maintenance problems.

In the present paper we consider a Hybrid System (HS) including both continuous and discrete dynamics of each components of the system, since the components contain switching behavior. If limited to linear hybrid systems, the continuous dynamics are described by ordinary differential equations (ODE's). To model the discrete behavior we follow the standard practice and use automata [3] due to the fact that they provide useful tools to handle logical operations.

2.1 Overview of the HIOA Model

The whole system is model by HIOA, which capture both continuous and discrete behavior. Based on [4], we consider a hybrid automaton A, for the description of systems, which include both continuous and discrete behavior. This automaton is a dynamical system that describes the evolution of a finite collection of variables, V, and allows shared variables as well as shared actions. Within this model it is allowed to describe the continuous behavior of hybrid systems separately from the discrete behavior.

Variables are typed, where for each $\upsilon \in V$, let *type(υ)* denote the type of υ. For each $Z \subseteq V$, a valuation of Z is a function that to each $\upsilon \in Z$ assigns a value in type(υ). Let $\boldsymbol{Z}$ denote the set of valuations of Z. Often, valuations will be referred to as states. We refer to $s \in \boldsymbol{V}$ as a system state. The evolution of variables involves both continuous and discrete dynamics.

The continuous time evolution of the valuations of the variables in V is described by a trajectory ω over V, that is a function that maps interval of $T^{\geq 0} = \{t \in \Re \mid t \geq 0\}$ to V. The *first state* of a trajectory ω is denoted by $\omega.fstate$, and the *last state* is denoted by $\omega.lstate$.

Discrete *dynamics* are encoded by *actions*. Upon the occurrence of an action the system state instantaneously "jumps" to a new value. The set of actions that affect the evolution of A is denoted by Σ.
A hybrid I/O automaton
$A = (U, X, Y, \Sigma^{in}, \Sigma^{\text{int}}, \Sigma^{out}, \Theta, D, W)$ consist of:
– Three disjoint sets U, X and Y of variables, called *input*, *internal* and *output variables*, respectively. We set $V = U \cup X \cup Y$.
– Three disjoint sets Σ^{in}, Σ^{int} and Σ^{out} of actions called *input*, *internal* and *output actions*, respectively. We set $\Sigma = \Sigma^{in} \cup \Sigma^{\text{int}} \cup \Sigma^{out}$.
– A non-empty set $\Theta \subseteq \boldsymbol{V}$ of *initial states*.
– A set $D \subseteq \boldsymbol{V} \times \Sigma \times \boldsymbol{V}$ of *discrete transitions*.
– A set W of *trajectories* over V.

A ***hybrid execution α***, of A is an alternating infinite or finite sequence of trajectories and actions $\alpha = \omega_0\alpha_1\omega_1\alpha_2\omega_2\ldots$, and the first state of α is an element of Θ. If α is a finite sequence then it ends with a trajectory and if ω_i is not the last trajectory its domain is right-closed and the discrete transition $(\omega_i.lstate, \alpha_{i+1}, \omega_{i+1}.fstate) \in D$. A state s is defined to be *reachable* if there exists a finite hybrid execution and s is the last state.

Two HIOA A_1 and A_2 are ***compatible*** if

$$X_1 \cap V_2 = X_2 \cap V_1 = Y_2 \cap Y_1 = \Sigma_1^{\text{int}} \cap \Sigma_2 =$$
$$= \Sigma_2^{\text{int}} \cap \Sigma_1 = \Sigma_1^{out} \cap \Sigma_2^{out} = 0$$

which means they have no output actions or output variables in common and no internal variable of either is a variable of the other. If A_1 and A_2 are compatible then they can be composed and so it is possible to model complex hybrid systems. Their composition $A_1 \times A_2$ is defined to be a new HIOA

$$\mathrm{A} = \left(U, X, Y, \Sigma^{in}, \Sigma^{\text{int}}, \Sigma^{out}, \Theta, D, W\right)$$

given by

$$U = \left(U_1 \cup U_2\right) - \left(Y_1 \cup Y_2\right), X = X_1 \cup X_2, Y = Y_1 \cup Y_2,$$

$$\Sigma^{in} = \left(\Sigma_1^{in} \cup \Sigma_2^{in}\right) - \left(\Sigma_1^{out} \cup \Sigma_2^{out}\right), \Sigma^{\text{int}} = \Sigma_1^{\text{int}} \cup \Sigma_2^{\text{int}},$$
$$\Sigma^{out} = \Sigma_1^{out} \cup \Sigma_2^{out}$$

Θ, D are W, are such that the executions of $A_1 \times A_2$ are also executions of each automaton when restricted to the corresponding variables and actions.
The ***hybrid trace*** of an hybrid execution α of A, denoted by *htrace(α)*, records the visible behavior of the execution and is the sequence obtaining by projecting α onto the external variables of A and subsequently removing all inert internal and environment actions. The set of all hybrid traces of A, denoted by *h-traces(A)* is the set of hybrid traces that arise from all the finite and admissible hybrid executions of A and describes the external behavior of a HIOA.

3. MODEL CONSTRUCTION FOR DIAGNOSIS

The system to be diagnosed consists of, the plant (decomposed as: actuators, main structure, sensors) and a controller. The subsystem of actuators is a set $A_i, i = 1,\ldots,n_A$ and the subsystem of sensors is a set $S_j, j = 1,\ldots n_S$.

For each element of the plant as well for the controller we construct a HIOA. The overall model will be the composition of a number of automata. The model discussed above can be structured according to the block diagram representation displayed in "Fig. 1".

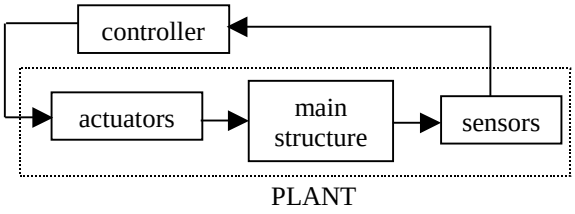

Fig. 1. Control System representation

In this work we are only interested in abrupt faults, which occur in the components of the plant, especially faults occurring to actuators, and to simplify our framework we make the following assumptions.

Assumption 1: *When the system starts functioning all its subsystems are in normal mode.*

Assumption 2: *When a fault occurs the system will remain in that failure state.*

When a fault occurs it is due to the actuators (main structure, controller and sensors are fail-safe).

Assumption 3: *The sensor and controller automata are simple input/output maps.*

A sensor automaton S_j reads the values of the main structure output variable as inputs and produces real valued output variables. A controller automaton C reads the corresponding sensor output variables and uses them to generate the input action of an actuator. An actuator A_i reads the corresponding controller output variables to generate the input action of the main structure.

***Main Structure*:** As mentioned above the main structure is modeled by an automaton P that is:

$$P = \left(U_P, X_P, Y_P, \Sigma_P^{in}, \Sigma_P^{\text{int}}, \Sigma_P^{out}, \Theta_P, D_P, W_P\right)$$

The main structure automaton P has no internal and output actions, hence $\Sigma_P^{\text{int}} = \Sigma_P^{out} = 0$, and there are only input actions. Therefore the automaton P will take the form

$$P = \left(U_P, X_P, Y_P, \Sigma_P^{in}, \Theta_P, D_P, W_P\right)$$

The input action set Σ_P^{in} is partitioned into subsets $\Sigma_{P_i}^{in} \quad i = 1,\ldots,n_A$ one for each actuator. The main structure automaton P communicates with the automaton of each subsystem through the set of input actions and the set of output variables. These input actions might be characterized as either *normal* or

faulty according to the effects, which affect to the plant behavior. The continuous system evolution is interrupted by the input actions.

***Actuators*:** An actuator is modeled as an automaton A_i that has internal actions, so we have:

$$A_i = \left(U_i, X_i, Y_i, \Sigma_i^{in}, \Sigma_i^{\text{int}}, \Sigma_i^{out}, \Theta_i, D_i, W_i\right)$$

Using this hybrid automaton we can model the effects of faults captured from both the discrete transitions and the trajectories. Consider a fault and assume that the same automaton models both the normal and the faulty behavior. We consider that the faults do not affect the system input, output and state space, i.e. $U_{iN} = U_{iF}, X_{iN} = X_{iF}, Y_{iN} = Y_{iF}$ where the subscripts *N* and *F* indicate whether the system is normal or faulty.

When a fault occurs there is some kind of internal action. This means that $\Sigma_i^{\text{int}} = 0$ if the actuator operates in normal mode and $\Sigma_i^{\text{int}} \neq 0$ if the actuator malfunctions.

According to the definition of HIOA the states may change either continuously or discreetly. Thus the variables will evolve either continuously as functions of time or be subject to instantaneous "jumps". The continuous state evolution is modeled by trajectories while the discrete state evolution is represent by the actions.

Consider $s \in V_{A_i}$ a state of an actuator. This state can keep evolving continuously, as long as:

$$\forall s_t \in V_{Ai},\ s_t \in \omega_i \text{ then } s_{t+\Delta t} \in \omega_i$$

where s_t is the state of actuator the moment *t* and Δt is the time interval at which the state evolves continuously at the trajectory ω_i.

Whenever an input action occurs to an actuator its state will either jump to another state or remain to its current state and evolve continuously. The second case will take place whenever the actuator's output variables coincide with the desired ones. In our approach the information about the occurrence of fault will be given at two levels.

First from the set D_i, which determines the discrete evolution of the state. From all news states after the jumping only a certain number of them correspond to the commands and so they represent a normal behavior of the actuator. Therefore the set D_i of discrete transitions is partition into two subsets D_N and D_F respectively for the transitions, which correspond to the normally operation and faulty operation. Then

$$D_i = D_{iN} \cup D_{iF}$$

The two aforementioned sets are define as follow:

$$D_{iN} = \bigcup \left\{ (s, \alpha, s') \mid (s, s') \in V_A, \alpha \in \Sigma_i^{in} \right\} \subset D_i$$

is the set of transitions for which the actuator transit from normal to normal operation, while

$$D_{iF} = \bigcup \left\{ (s, \alpha, s'') \mid (s, s'') \in V_A, \alpha \in \Sigma_i^{in} \right\} \subset D_i$$

is the set of transitions for which the actuator transit from normal to fault operation.

Second from the set W_i that describes the continuous behavior of the HIOA. This approach will be based on the standard technique of analytical redundancy and so we do not extended to that technique.

***Plant*:** The plant is modeled as an automaton *H* that has no output actions

$$H = \left(U_H, X_H, Y_H, \Sigma_H^{in}, \Sigma_H^{\text{int}}, \Theta_H, D_H, W_H\right)$$

Based on assumption 3, sensors and controllers are modeled as automata that are simple input/output maps.

***System*:** The system is modeled as an automaton *S* that has no input output actions and input output variables, so we have

$$S = \left(X_S, \Sigma_S^{\text{int}}, \Theta_S, D_S, W_S\right)$$

4. DIAGNOSER

The diagnoser *G* is a hybrid automaton that generates a signal whenever a fault occurs. Its role is to observe and check the behavior of the automaton *S* (and to compare its evolution with the predefined acceptable behavior). Moreover whenever it detects a fault it should generate a signal, indicating the malfunctioning component.

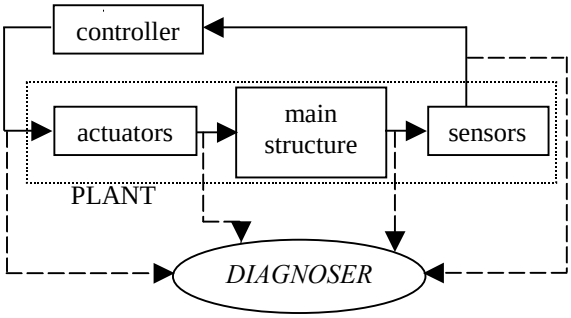

Fig. 2. Control and Detection systems

The automaton that modeling the diagnoser "Fig. 2" is defined as

$$G = \left(S, R, M, NF, Y, d\right) \text{ where:}$$

– The S automaton representing the whole system (composed of the actuator, main structure, controller and the sensors).
– The set R refers to the set of states satisfying the required properties of the system.
– The set M is the set of states, which correspond to faulty operation of the system.
– The set NF of discrete input, $NF \in \{0,1\}$, where $NF=1$ if a fault occurs and $NF = 0$ otherwise.
– The set Y of output variables.
– The variable d denoting the time required by the diagnoser G to performs diagnosis, i.e. the diagnosis loop time.

5. APPLICATION TO WATER - LEVEL MONITOR SYSTEM

In our example we have a water – level monitor system consisting of a pump, a controller and three sensors, two water level sensors and a pressure sensor on the pump "Fig. 3". The hybrid behavior of this system is due to the on/off position change of the pump switch controlled by the switching controller, the switching of sensors between on/off and the starting and stopping of water through the pipe. We are interested to capture only abrupt faults occurring to pump.

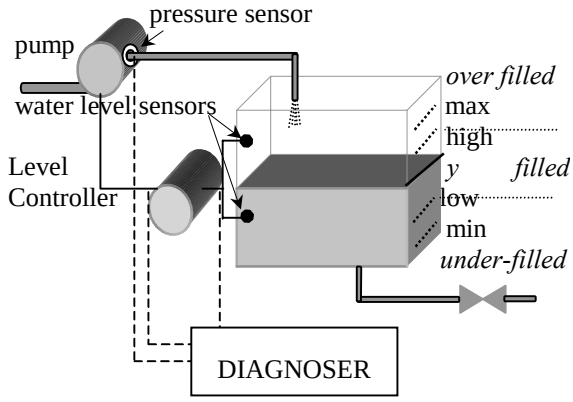

Fig. 3. Water–level monitor system

The pump is assumed to be an electromechanical system "Fig. 4" controlling the flow inlet of the water tank. The input voltage V_e controls its behavior.

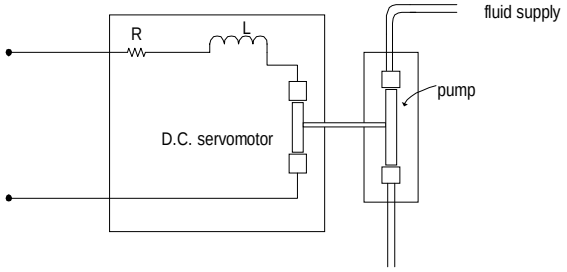

Fig. 4. The electromechanical system of gear pump

The continuous behavior of the system is described by the following equation

$$\dot{\underline{x}} = A \cdot \underline{x} + B \cdot \underline{u}$$

We consider the water level y as the state variable of the water supply plant. The initial state of the water tank is designated as *INIT.* Additional state variables of the system are the angular velocity ω_J of the servomotor and the current I_L.

The level y is controlled by a controller, which observes the level via level sensors. The controller automaton appears in "Fig. 5".

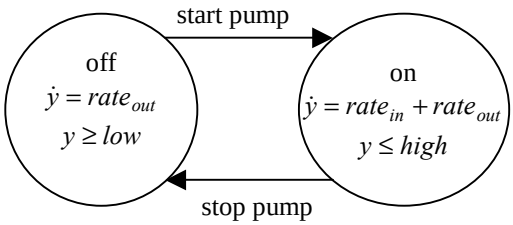

Fig. 5. Controller automaton

The sensors can be in two states *ON* or *OFF*. We assumed that the water level sensors go on when are wet and the pressure sensor go on when the pump pumping (table 1). Their automata appear in "Fig. 6 & 7".

Table 1. Sensor States

	HIGH	LOW	PRESSURE
Over-filled	on	on	off
Filled	off	on	off
Under-filled	off	off	on

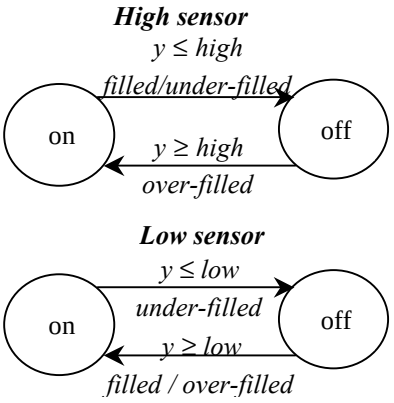

Fig. 6. Level sensors automata

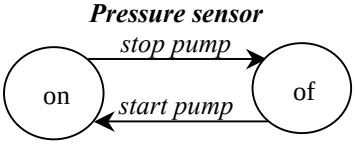

Fig. 7. Pressure sensor automaton

The pump has three failure events: *F1* – pump failed off (stuck closed), *F2* – pump failed on (turn off), *F3* pump failed on (stuck open). The pump has four overall states: *Pon* represent the normally open behavior, *Poff* represent the normally off, while *FPon* represent the failed on and *FPoff* represent the failed off status of the pump. The automaton model of the pump is presented in "Fig. 8".

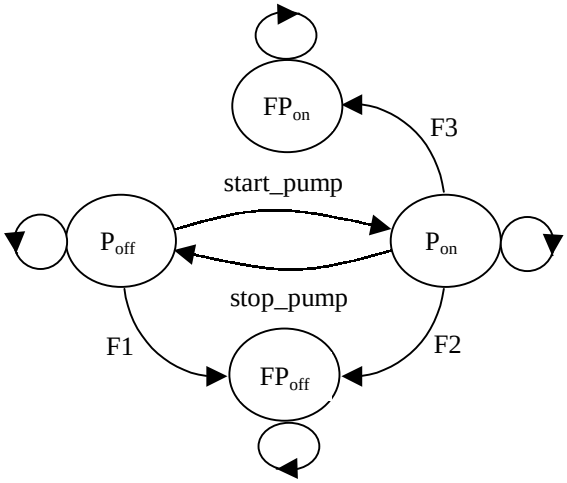

Fig. 8. Pump automaton

The overall model is a composition of a number of automata and is appear in "Fig. 9". In this figure both normal and failure behavior are shown. Solid lines indicate the normal behavior while dotted lines indicate faulty. The *SPon, SPoff, SLon, SLoff, SHon, SHoff* represent the sensors states, pressure, low level, high level respectively and *Con*, *Coff* the controller states.

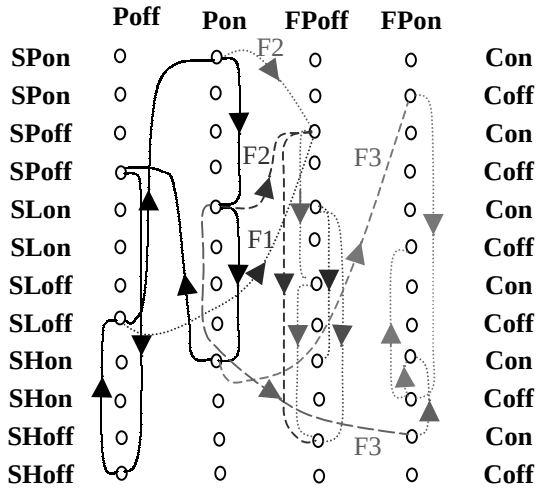

Fig. 9. Composition of overall model

Based on the previous figure we obtain the fact that there is a number of states where the system behave normally, as well as other states which indicate the pump malfunctioning.

According its definition the construction of diagnoser require the creation of the set *M*. These can be done by the composition of the overall model. From these automaton we observe that for each state that characterizes pump malfunction correspond a combination of components states and actions of controller. So the above procedure enable us to construct the set M of detector, which correspond to faulty operation of the system.

When an appropriate combination of component's states is occurred a fault is detected thus generating a signal indicating the pump malfunction.

6. CONCLUSION

We have introduced the underlying concepts for our approach to the problem of failure diagnosis for Hybrid Systems. The class of systems studied is the class of linear hybrid systems and the discussion in the paper is mainly focused at the study of their discrete behavior. This approach was illustrated via a simple application to a water level system.

The next step is to study faults occurring at other system components. The theory presented must be modified appropriately in order to take into account the collaboration of discrete and continuous modeling, as well as to provide us the ability to isolate the malfunctioning component.

7. REFERENCES

[1] Branicky M.S. Studies in Hybrid Systems: Modeling, Analysis and Control, PhD thesis, Massachusetts Institute of Technology, Dept. of Electrical Eng. and Computer Science, June 1995.

[2] Frank P.M. Fault Diagnosis in dynamic Systems Using Analytical and Knowledge-based Redundancy, A Survey and Some New Results, Automatica, vol. 26, no. 3, 1990, pp. 459-474.

[3] Henzinger T. The theory of hybrid automata, Proc. of 11th annual IEEE symposium on Logic in Computer Science, LICS 1996, pp. 278-292.

[4] Lynch N., Segala R., Vaandrager F., Weinberg H. Hybrid I/O automata, Hybrid Systems III, no. 1066 in LNCS, 1996, pp. 496-510, Springer Verlag.

[5] Manna Z., Sipma H. Deductive verification of hybrid systems using SteP, Hybrid Systems: Computation and Control, no 1386 in LNCS, 1998, pp. 305-318, Springer Verlag.

[6] Patton R., Frank P., Clark R. Fault Diagnosis in Dynamic Systems – Theory and Application, 1989, Prentice Hall.

[7] Sampath M., Sengupta R., Lafortune S., Sinnamohideen K., Teneketzis D.C. Failure Diagnosis using discrete-event models, Tran. On Cont. Sys. Tech. vol. 4, no. 2 pp. 105-124, 1996.