

Modeling and Fault Diagnosis of Hybrid Dynamical Systems: Application to Avionics Systems

Ilhem Ben Hnaïen¹, Eric Gascard², Zineb Simeu-Abazi², Hedi Dhouibi³

¹NOCCS Laboratory, Natl. Engr. School of Sousse, University of Sousse,
Sousse, Tunisia

²Univ. Grenoble Alpes, CNRS, Grenoble INP , G-SCOP
Grenoble, France

³LARATSI Laboratory, High. Inst. of Appl. Sci. Technol. of Kairouan, University of Kairouan
Kairouan, Tunisia

*Corresponding author; Email: ilhem.benhnaïen@enis.u-sousse.tn

Article Info

Article history:

Received , 20/06/2025

Revised , 26/07/2025

Accepted , 10/08/2025

Keywords:

Fault Detection and Isolation;
Hybrid Dynamical Systems ;
Hybrid Modeling;
Hybrid Automata;
Fault Diagnosis.

ABSTRACT

Hybrid Dynamical Systems (HDS) represent a unique integration of continuous and discrete dynamics, offering robust frameworks for modeling complex physical systems with mixed behavior. This article explores the fundamentals of hybrid systems and hybrid automata, emphasizing their relevance for fault detection and isolation in various industrial fields such as robotics, avionics, and autonomous systems. Through a comprehensive hybrid automaton model, we demonstrate a diagnostic strategy that utilizes time-based intervals to detect deviations, identify failures, and ensure timely responses to system anomalies. The application of this diagnostic framework to an avionics fuel transfer system illustrates the practical utility and effectiveness of hybrid models in fault detection and system recovery, highlighting the critical role of time constraints and logical conditions in operational integrity.

I. Introduction

Hybrid Dynamical Systems (HDS) integrate both continuous and discrete dynamic behaviors [1]–[3], making them highly suitable for modeling complex systems where continuous physical processes and discrete events interact to influence the system's state. This modeling framework is increasingly utilized in various fields, including robotics, aerospace, and automotive industries, due to its ability to represent diverse operating conditions, fault behaviors, and transitions in real time. A significant aspect of HDS is their application in fault diagnosis, where hybrid automata serve as a powerful tool for detecting, isolating, and predicting faults based on system dynamics.

Recent advancements have enhanced the use of hybrid automata, such as adaptive models that adjust in real time to sensor data, significantly improving fault detection accuracy [4], [5]. These models allow engineers to simulate various fault scenarios, monitor system dynamics for anomalies, and activate corrective actions based on real-time system behavior. Furthermore, as systems grow more complex, the importance of time-based

diagnostics becomes paramount, particularly for systems that rely on tight timing constraints to distinguish between normal, degraded, and faulty modes of operation.

This paper explores the fundamentals of Hybrid Dynamical Systems, focusing on the use of hybrid automata to construct global models for fault detection. The diagnostic strategy leverages these models for real-time system monitoring, fault localization, and recovery. The study includes an application to an avionics fuel system, illustrating how hybrid automata can represent different operating modes and assist in the analysis of potential failure modes.

The paper is structured as follows: Section 2 introduces the fundamentals of Hybrid Dynamical Systems and hybrid automata, while Section 3 discusses the diagnostic strategy for fault detection and isolation. Section 4 presents the application of hybrid models to the avionics fuel system, highlighting the modeling and analysis techniques. Section 5 provides a detailed analysis of the simulation results, demonstrating the effectiveness of the proposed diagnostic approach. Finally, Section 6 concludes the paper and outlines future research directions.

II. Fundamentals Of Hybrid Dynamical Systems (HDS)

Hybrid Dynamical Systems (HDS) integrate continuous and discrete dynamic behaviors, enabling robust modeling of complex physical systems where both continuous physical processes and discrete events influence system states. These systems are widely applicable across various fields, including robotics, industrial automation, aerospace, and automotive engineering, due to their ability to represent diverse operational conditions, fault responses, and system transitions in real time. Generally, a hybrid dynamic system is composed of a continuous dynamic system, a discrete event system, and an interface that manages the interactions between the continuous and discrete dynamics (see Fig.1).[1,2]

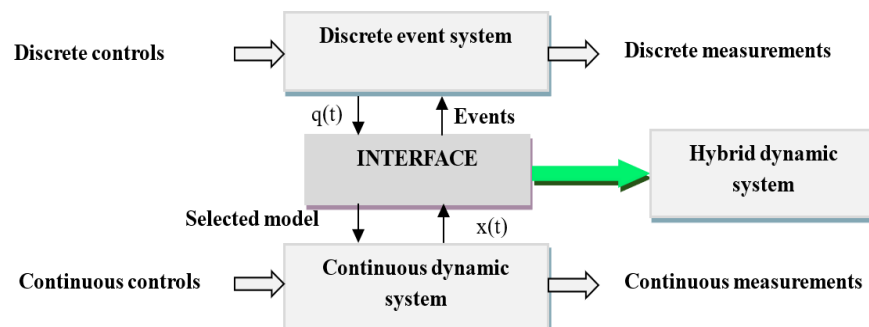


Figure 1. Hybrid dynamic system structure.

II.1. Hybrid Automata Model

Hybrid automata [6] extend finite state automata to represent systems that integrate two components: one with discrete behavior, modeled by a finite state automaton, and the other with continuous-time behavior, modeled by an algebraic-differential system. A hybrid automaton alternates between continuous steps, where state variables and time evolve continuously, and discrete steps, where instantaneous transitions may occur. A timed automaton is a specific type of hybrid automaton where all variables have derivatives with respect to time equal to one. Generally, a hybrid automaton functions as a finite state machine that manages a set of differential equations modeling the system's continuous dynamics. The state changes instantly with the occurrence of a discrete event or as time progresses to satisfy a specified logical condition on the continuous variable [7].

Example of a Hybrid Automaton

Consider the automaton in Fig. 2, which models a hybrid system. In this model, continuous evolution is represented by differential equations associated with graph nodes, and event-driven evolution by labeled arcs. The nodes "Off" and "On" represent discrete states within which continuous evolution occurs. Conditions, or guards, on the arcs (e.g., $x \in [2,4]$ and $x \in [5,10]$) specify when events (such as "turn on" and "turn off") are triggered. Invariants ($x \leq 4$ and $x \leq 10$) associated with the nodes define the conditions under which the continuous variables are allowed to remain in a given discrete state ("Off" or "On"). The system's initial state is indicated by an entry arc into the origin node, with the label $x \in [5,10] \wedge y = 0$ specifying the initial region from which the hybrid system's dynamics begin. In the "Off" and "On" states, the variables x and y evolve according to the differential equations $\dot{y}=4y$ and $\dot{x}=1$ (in "Off"), and $\dot{y}=3y+2$ and $\dot{x}=1$ (in "On"), which define the flow conditions [8]. The analysis and verification of systems using hybrid automata are central topics in hybrid dynamical systems (HDS) studies. This process involves verifying both qualitative and quantitative properties of the system through automatic methods. It begins with modeling the HDS behavior using a hybrid automaton, followed by qualitative property verification through model-checking techniques [7], and quantitative property verification through reachability analysis [6], [9].

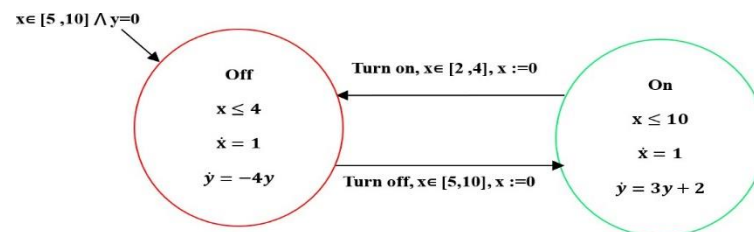


Figure 2. Example of a hybrid automaton.

II.2. Construction Of the Global Hybrid Model

Hybrid automata have become central to fault diagnosis due to their ability to simulate and predict system responses under different failure modes. Recent research has introduced advancements such as adaptive hybrid automata, which dynamically adjust guard conditions and invariants in response to real-time data, enhancing the system's ability to detect faults quickly and accurately [10]. These advancements are crucial for industries requiring high reliability and real-time adaptability, such as healthcare robotics and autonomous vehicles. In practice, hybrid automata allow engineers to design fault-tolerant systems by:

1. Simulating potential fault scenarios within each state.
2. Monitoring for deviations in continuous dynamics to detect anomalies.
3. Activating corrective transitions based on guard conditions for efficient fault isolation and system recovery.

The system's starting condition is illustrated by an incoming arc directed at the initial node. The label of this arc ($x = x_0 \wedge y = 0$) represents the continuous region of space from which the dynamics of the hybrid system starts. Hybrid dynamic systems are systems that combine discrete continuous parts. For constructing the global hybrid model. The dynamic behaviors of a hybrid system can be represented by a succession of modes, as shown in Fig. 3. Temporal information such as the different step duration of the process and the date of occurrence of the event. Procedures based on this model are only able to detect errors a priori. When information about any part of the normal operating model that is not changed in the event of errors is provided, isolation (localization) of the faulty component becomes possible.

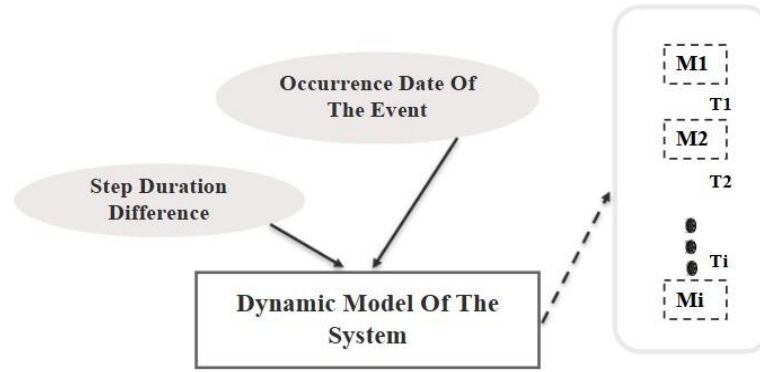


Figure 3. Dynamic Model Representation.

Modeling the system through a hybrid automaton that encompasses all system states enables the representation of its various operating modes ($M_1, M_2, \dots, M_i$). Such that, each mode M_i is characterized by a discrete state, a set of differential equation which represents the continuous evolution of each mode and a set of constraints which defines the domain of eligibility. A discrete transition T_i from one mode to another mode takes place when certain logical conditions are verified. These changes can be caused by discrete events that are generated by sensors and by the execution time of each mode. With a view to detecting, diagnosing failures and predicting faults by prognostic reasoning based on knowledge of the degradation, it is crucial to determine the instant of passage from one mode to another mode and to recognize at the active mode at all times. This information will be all the more crucial if some of the hybrid model modes correspond to degraded or faulty behaviors. Generally, the global hybrid model is an enriched model which contains all the possible knowledge on the three modes of operation of the system, namely: the normal mode when the operation of the system complies with the requirements i.e. everything is going well; the degraded operating mode when a time drift is signaled or the system partially achieves its objectives and the failure mode when a fault is declared. The modeling framework taking into account the degradation of the system is then based on the activity time of a mode associated with a specific state for a given situation.

The status of the system is monitored by the activity time of a mode. The system is in nominal mode when all system tasks are executed within a predefined interval I_m . If the duration of the task exceeds the maximum time T_{max}^{Ni} , the system switches to degraded mode for a defined interval J_m . However, if the duration of the task exceeds the critical time T_c^{Di} , the system falls into the faulty mode F_m . For each task, we define three time values: T_{min}^{Ni} , T_{max}^{Ni} , and T_c^{Di} . Based on these three values, Fig. 4. illustrates the three operating intervals such as:

$I_m = [T_{min}^{Ni}, T_{max}^{Ni}]$ (Normal mode); $J_m = [T_{max}^{Ni}, T_c^{Di}]$ (Degraded mode); $F_m = [T_c^{Di}, +\infty]$ (Faulty mode)

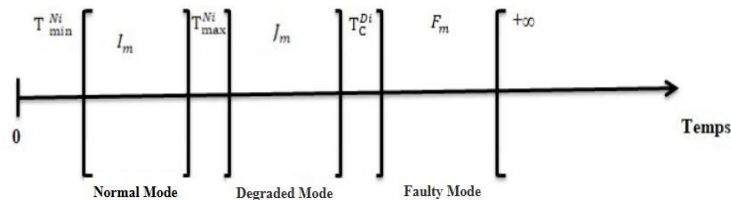


Figure 4. Activation Time and Operational Mode

The construction of the diagnostic module is based on a hybrid dynamic model which gives an appropriate modeling of the operating modes of the system. For each task of the process, we define the different normal and degraded operating intervals and the threshold at which we consider that the operation is faulty. After each task execution command, we take the time information of the start and end of the execution as this information is

received by the sensors. According to the time elapsed between these two instants with respect to the intervals specified above, the mode of operation of the system is determined: whether it is normal, degraded or faulty.

III. Diagnostic Strategy for Hybrid Models: Aims and Objectives

This study employs hybrid models to establish a diagnostic framework for fault detection, using hybrid automata as the foundational structure. Time is crucial in characterizing system evolution and is central to our diagnostic approach. Our method prioritizes time-based diagnostics to build detection and localization modules for hybrid dynamic systems, as illustrated in Fig. 5.

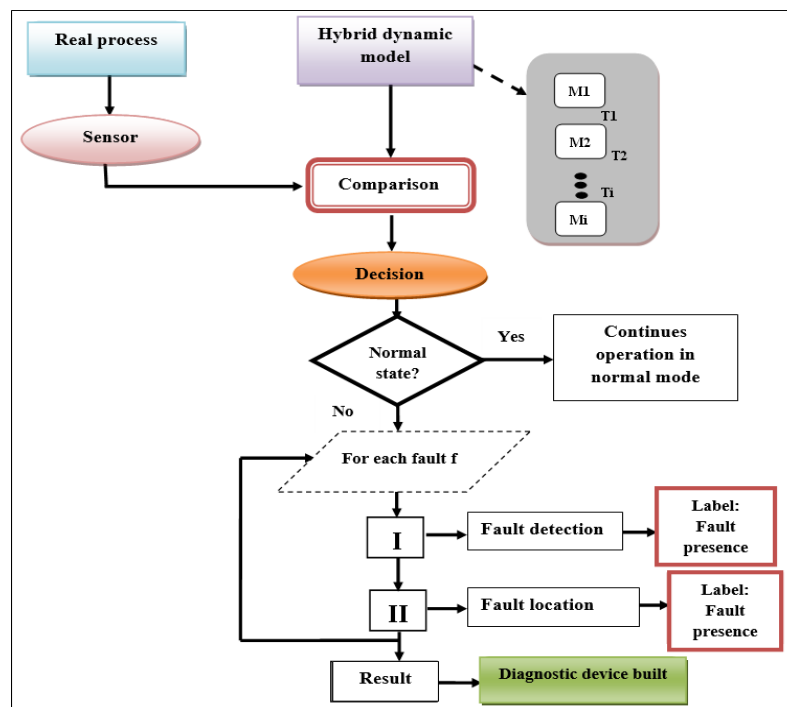


Figure 5. Diagnostic Construction Flowchart.

We begin with two key components: (1) the hybrid dynamic model describing the temporal and event-driven behavior and differential equations of the target system, allowing the hybrid automaton to simulate the process under normal conditions, and (2) sensor data from the actual system. By comparing these, we can identify deviations indicating abnormal operations. These discrepancies are detected and expressed as events or symptoms.

This step defines distinct time intervals using algebraic inequalities tied to each automaton state. Fault detection occurs when deviations exceed expected time intervals. Generally, the diagnostic framework infers unobservable faults through observable events and the intervals between them, aiming to optimize the use of time constraints in the system model to recognize possible failure modes.

IV. Application: Modeling And Analysis of An Avionics System Using Hybrid Automata

IV.1. Overview of the System

This aircraft has four main fuel tanks: two located in the left and right wings, one in the fuselage, and one for consumption. The tank in the fuselage is divided into the front fairing tank and the central tank, which are connected by an open line without a valve. The front fairing tank is positioned higher than the central tank, allowing fuel to transfer by gravity. The fuselage tank (front and central fairing), also known as the central tank, has a total capacity of 7,000 lbs of fuel, while each wing tank has a capacity of 3,000 lbs. This gives the three tanks a combined total capacity of 13,000 lbs. The fuel transfer system moves fuel from the central tank to the left and right wing tanks, and then to the consumption tank.

Functioning

First, the flow pump will open valve Q_p to fill the first tank up to level X_1 . Once this level is reached, Q_p will close, and valves V_1 and V_2 will open to fill the left and right wing tanks in parallel, until both reach levels X_2 and X_3 . The sections of the wing tanks are equal, with $S_1 = S_2 = 0.0077m^2$. Afterward, valves V_3 and V_4 will open to transfer fuel into the consumption tank.

IV.2. Modeling the System

Considering the system schematized in Fig. 6, the equations of state are derived by expressing the variation in fuel volume in the tank as the sum of the inflows minus the sum of the outflows:

$$\dot{V} = S * \dot{X} = \sum Q_{in} - \sum Q_{out} \quad (1)$$

Since the system operates in four modes, each mode must be represented by its own equation of state.

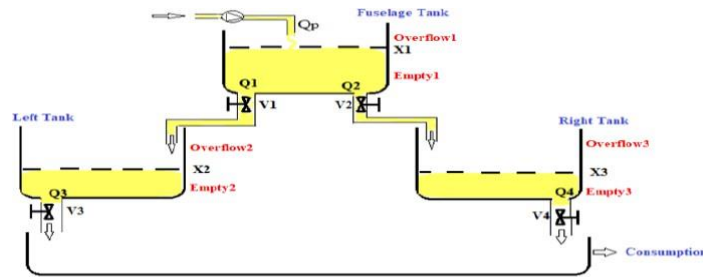


Figure 6. System Overview.

- **Mode 1:** The fuselage tank is filled until the fuel level X_1 reaches 3.1752. The state can be described by the following equation:

$$\dot{X}_1 = (Q_{ps} - Q_{1s} - Q_{2s})/S \quad (2)$$

- **Mode 2 and Mode 3:** The valves V_1 and V_2 , are opened to fill the left and right wing tanks simultaneously until both X_2 and X_3 reach 1.3608. The state equations for these modes are:

$$\dot{X}_2 = (Q_{1s} - Q_{3s})/S_1 \text{ and } \dot{X}_3 = (Q_{2s} - Q_{4s})/S_1 \quad (3)$$

- **Mode 4:** The valves V_3 and V_4 , are opened to drain the left and right wing tanks. The state can be represented as:

$$\dot{X}_4 = (Q_{3s} + Q_{4s})/S \quad (4)$$

The flow rates Q_1 , Q_2 , Q_3 and Q_4 are defined as follows:

$$1) Q_1 = A\sqrt{2g\text{sign}(X_1 - X_2)}\sqrt{|X_1 - X_2|}$$

$$2) Q_2 = A\sqrt{2g\text{sign}(X_1 - X_3)}\sqrt{|X_1 - X_3|}$$

$$3) Q3 = A\sqrt{2g\sqrt{X2}}$$

$$4) Q4 = A\sqrt{2g\sqrt{X3}}$$

With: $A = 3.6 \times 10^{-5} \text{m}^2$ constant and $g = 9.81 \text{m/s}^2$ gravitational constant.

Here, Q_1 and Q_2 represent the flow rates exiting the fuselage tank and entering the wing tanks, while Q_3 and Q_4 denote the outflows from the left and right wing tanks, respectively.

IV.3. Modeling Processes with Simulink and Stateflow

In this subsection, we delve into the modeling of processes using Simulink and Stateflow, powerful tools from the Matlab suite. These tools enable the simulation of continuous, discrete, and nonlinear systems related to working memory in Matlab (workspace).

The Simulink model, is designed to simulate the dynamic behavior of various components in the system. The model comprises several interconnected blocks, each representing a specific part of the system such as valves, sensors, and controllers.

Stateflow is employed to model the logic and decision-making processes within the system. The Stateflow chart, as shown in the Fig.7 , captures the various states and transitions that the system undergoes during operation. In this Stateflow chart, we define the states and transitions for the controller. The initial state is S0, where the valve position (V_p) is set to 1. After 1 second, the system transitions to state S1, where V_p is set to 0. The system continues to evolve through various states, as dictated by the logic embedded in the Stateflow chart.

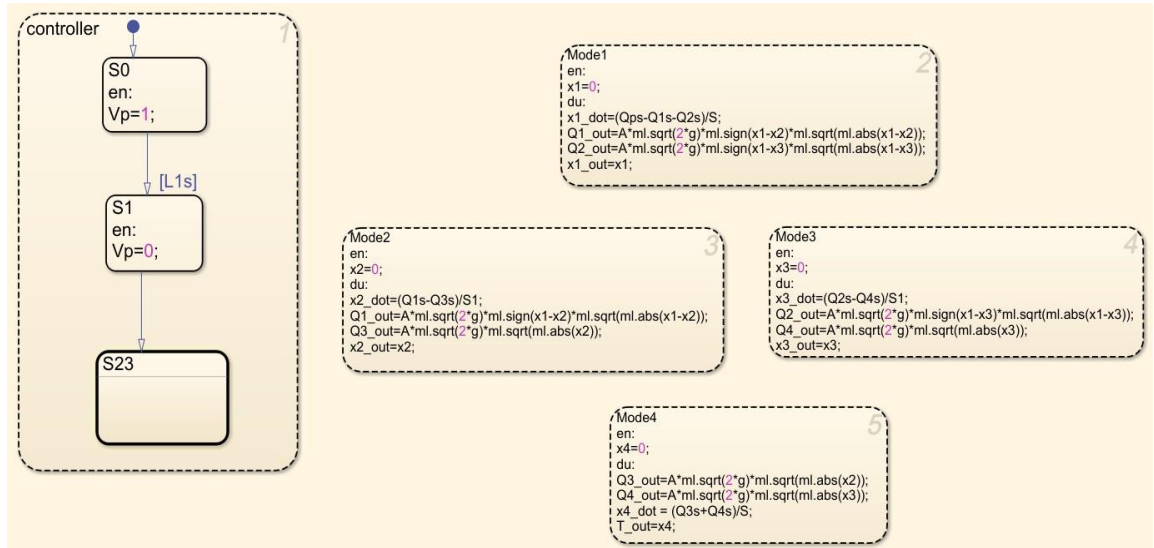


Figure 7. Stateflow Chart for System Controller

The detailed models of individual components such as valves and sensors are illustrated in the Fig.8 and Fig.9. These models are crucial for understanding the specific behaviors and interactions of each component within the system. The valve models are designed to simulate the flow control mechanisms within the system. Each valve block receives input signals and produces output signals that reflect the flow rates and pressures in the system. The sensor models are critical for monitoring the system's state and providing feedback to the controller. These models include overflow and empty sensors that detect the levels of fluids in the system.

The integration of Simulink and Stateflow provides a robust framework for modeling and simulating complex systems. By leveraging these tools, we can accurately predict system behavior, optimize performance, and ensure reliability. The detailed models of valves, sensors, and controllers form the backbone of this simulation, enabling comprehensive analysis and control of the system.

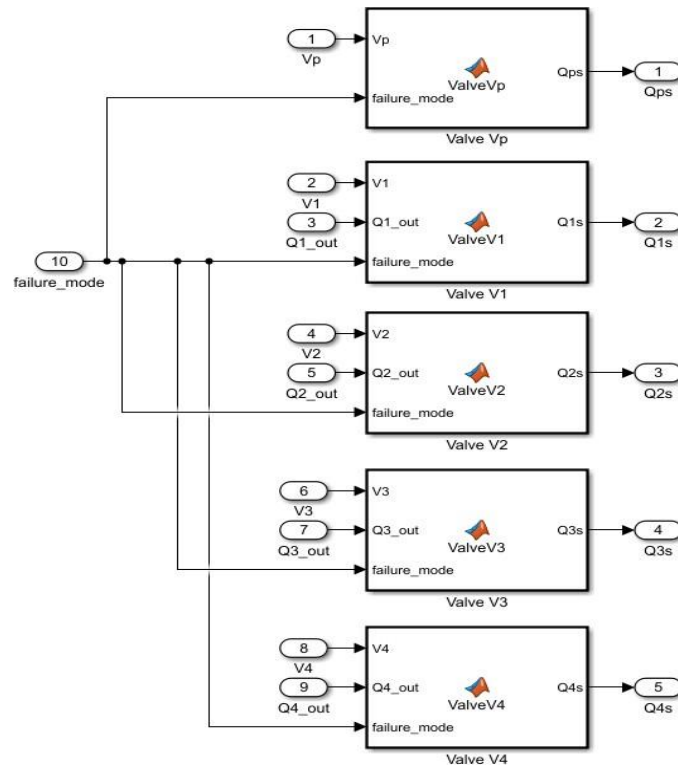


Figure 8. Valve Models in Simulink.

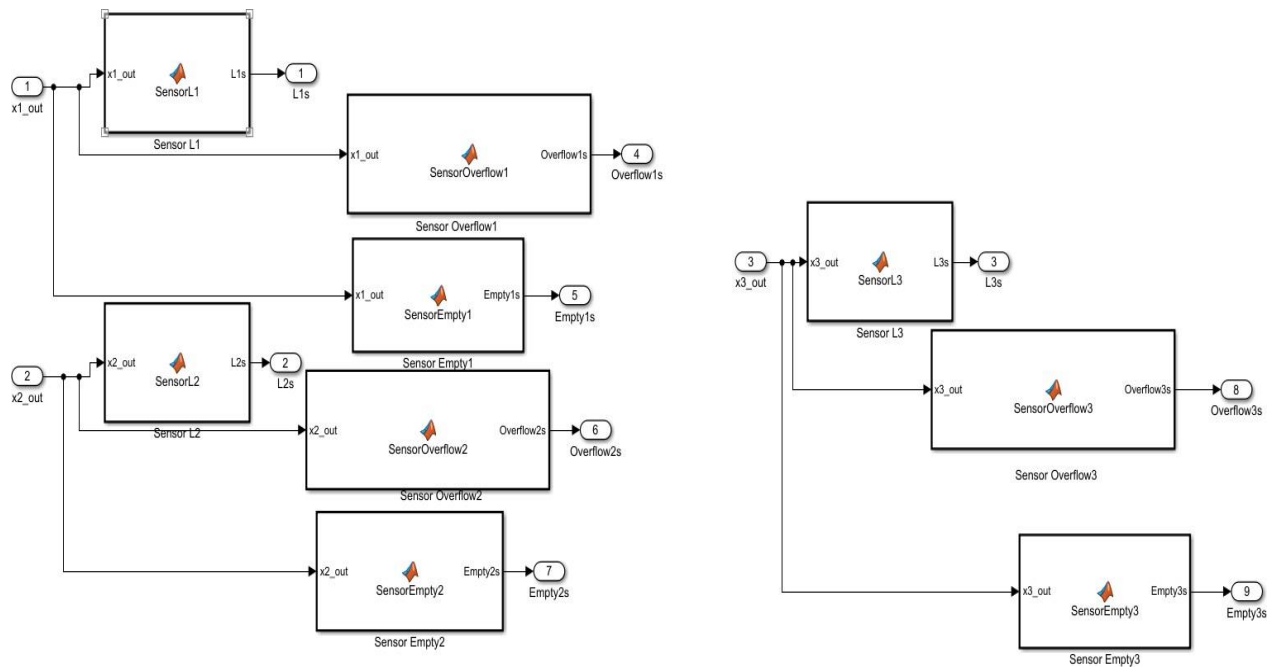


Figure 9. Sensor Models in Simulink

V. Results Analysis

The initial step in analyzing the system during normal operation is to observe its behavior without introducing any faults. The simulation was conducted over a total duration of 400 seconds (refer to Fig. 10). The development of the diagnostician relies on the temporal knowledge of the process, making it crucial to understand the duration of each operational step. After simulating normal operation, the results presented in Table 1 were obtained, highlighting key performance indicators such as flow rates, pressure levels, and system response times.

We introduced a fault V_2 -SO in the system, specifically in valve V_2 which fails to close when commanded. The goal is to detect, identify, and locate this fault

Table 1. Time intervals for each action	
Action	Time interval
Filling the main tank	Between [0 ; 50]
Filling the tanks for the left and right wing	Between [51 ; 103]
Consumption	Between [104 ; 400]

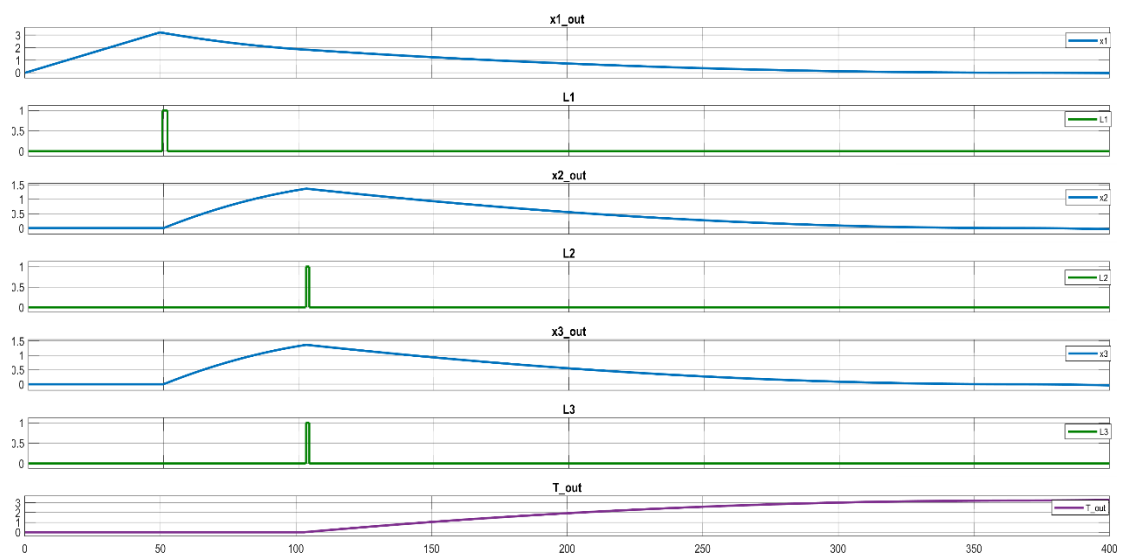


Figure 10. Results of the Simulation in Normal Mode.

Fig. 11 compares the system's normal operation (left) with the faulty state (right). In the normal state (left), the valve (blue signal, X_{2_out}) follows the expected command and closes quickly at $T = 103s$. The $L2$ signal (green) remains stable, and diagnostic signals (overflow2 and Empty2) stay at nominal values, indicating smooth operation.

In the faulty state, despite the close command at $T = 103s$, the valve stays open, with The blue signal (X_{2_out}) closing slowly and not reaching the fully closed position until $T = 180s$. The fault becomes apparent here. Around $T = 150s$, the *overflow2* signal (red) rises, signaling an issue. The delay in the system's response to the fault is noted

Finally, by $T = 280s$, the *Empty2* signal (cyan) confirms the fault location. This analysis highlights the system's ability to detect and locate the fault, ensuring safety despite the defect.

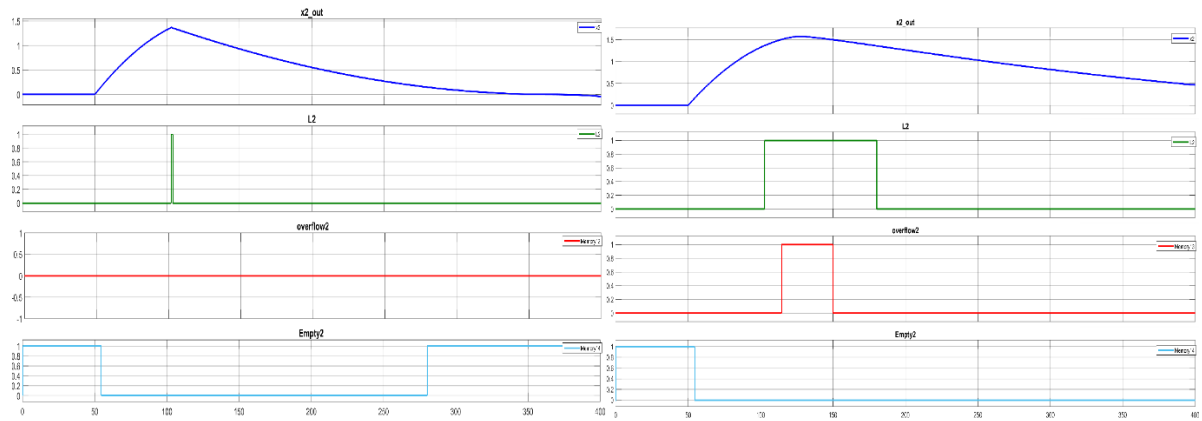


Figure 11. Detailed Faulty Operation Analysis

VI. Conclusion

Hybrid Dynamical Systems (HDS) offer a robust and unified framework for modeling and analyzing complex systems that combine continuous dynamics with discrete events. In this work, we demonstrated how hybrid automata can effectively represent different operating modes and transitions in a safety-critical application such as an avionics fuel transfer system. By integrating time-based diagnostic strategies within the hybrid model, we successfully detected and localized faults based on deviations in temporal behavior, even in the presence of delayed or hidden symptoms.

The simulation results confirmed the relevance and accuracy of our approach, particularly in capturing the system's degraded and faulty behaviors. The fault injection scenario illustrated the capacity of the diagnostic system to identify abnormal valve behavior and accurately determine the timing and location of the fault. This work not only validates the applicability of hybrid models for fault diagnosis in avionics but also lays the groundwork for future enhancements. Upcoming developments may focus on incorporating learning-based or adaptive mechanisms into hybrid automata, enabling real-time model refinement and improved fault prognosis capabilities. Such advancements are essential for supporting predictive maintenance and increasing the resilience of Cyber-Physical Systems, especially in domains such as aerospace, autonomous systems, and industrial automation.

Acknowledgements

The authors would like to express their sincere gratitude to the G-SCOP Laboratory, Université Grenoble Alpes, CNRS, Grenoble INP, France, for providing the research environment, resources, and valuable guidance that supported this work. Their collaboration and technical expertise were instrumental in advancing the modeling and diagnostic approaches presented in this study.

References

- [1] 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 computer science*, vol. 138, no. 1, pp. 3–34, 1995.
- [2] Y. Kestent, Z. Manna, and A. Pnueli, "Verification of clocked and hybrid systems," in *Lectures on Embedded Systems: European Educational Forum School on Embedded Systems Veldhoven, The Netherlands November 25–29, 1996*, pp. 4–73, Springer, 1998.

- [3] A. Taghavian, A. Safi, and E. Khanmirza, "Constrained computational hybrid controller for input affine hybrid dynamical systems," *Journal of the Franklin Institute*, vol. 361, no. 17, p. 107142, 2024.
- [4] H. Badihi, S. Jadidi, Y. Zhang, C.-Y. Su, and W.-F. Xie, "Ai-driven intelligent fault detection and diagnosis in a hybrid ac/dc microgrid," in *2019 1st International Conference on Industrial Artificial Intelligence (IAI)*, pp. 1–6, IEEE, 2019.
- [5] Y. Lu, G. Li, Y. Yue, and Z. Wang, "Fault detection and data-driven optimal adaptive fault-tolerant control for autonomous driving using learning-based smpc," *IEEE Transactions on Intelligent Vehicles*, 2024.
- [6] S. N. Krishna and A. Trivedi, "Hybrid automata for formal modeling and verification of cyber-physical systems," *arXiv preprint arXiv:1503.04928*, 2015.
- [7] H. Lin, P. J. Antsaklis, *et al.*, "Hybrid dynamical systems: An introduction to control and verification," *Foundations and Trends® in Systems and Control*, vol. 1, no. 1, pp. 1–172, 2014.
- [8] P. Santana, S. Lane, E. Timmons, B. Williams, and C. Forster, "Learning hybrid models with guarded transitions," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, 2015.
- [9] C. Le Guernic and A. Girard, "Reachability analysis of hybrid systems using support functions," in *International Conference on Computer Aided Verification*, pp. 540–554, Springer, 2009.
- [10] R. R. Bravo, E. Baratchart, J. West, R. O. Schenck, A. K. Miller, J. Gallaher, C. D. Gatenbee, D. Basanta, M. Robertson-Tessi, and A. R. Anderson, "Hybrid automata library: A flexible platform for hybrid modeling with real-time visualization," *PLoS computational biology*, vol. 16, no. 3, p. e1007635, 2020.