# Formal Non-Fragile Stability Verification of Digital Control Systems with Uncertainty

Iury Bessa, Hussama Ismail, Reinaldo Palhares, Lucas Cordeiro, and João Edgar Chaves Filho

**Abstract**—A verification methodology is described and evaluated to formally determine uncertain linear systems stability in digital controllers with considerations to the implementation aspects. In particular, this methodology is combined with the digital-system verifier (DSVerifier), which is a verification tool that employs Bounded Model Checking based on Satisfiability Modulo Theories to check the stability of digital control systems with uncertainty. DSVerifier determines the control system stability, considering all the plant interval variation set, together with the Finite Word-length (FWL) effects in the digital controller implementation; DSVerifier checks the robust non-fragile stability of a given closed-loop system. The proposed methodology and respective tool are evaluated considering non-fragile control examples from literature. Experimental results show that the approach used in this study is able to foresee fragility problems in robust controllers, which could be overlooked by other existing approaches due to underestimating of FWL effects.

**Index Terms**—Formal methods, model checking, finite word-length effects, controller fragility, robustness.

## 1 Introduction

There has been a gap between two research fields: control theory and formal methods [1], [2]. There is clearly a substantial difference between hierarchy level and specifications, which are considered by both areas. Formal methods ensure that all types of specifications for relatively (high-level) systems are represented by finite (or infinite) states transition systems, while control theory treats dynamical systems using mathematically grounded techniques. However, both areas aim at achieving reliable systems, in order to check whether implementation meets specification. The diversification and flexibility of verification tools and introduction of hybrid automata to represent hybrid systems have allowed a close convergence of both research areas.

Hybrid systems (HS) and cyber-physical systems (CPS), which are the two important parts of the actual industrial evolution trends, are usually represented by hybrid automata, for which several formal verification methods have been proposed. Alur *et al.* [3] present the earliest application of model checking for timed automata using the Timed Computation Tree Logic (TCTL) as an extension of CTL model checking for real-time systems. Those initiatives inspired the development of formal methods and model checking tools for verifying timed automata and for representing any control system by finite state machines; notable model checking tools include UPPAAL and HyTech [4].

Although formal methods provide applicability to check high-level specifications in all sorts of CPS, there is not much application of model checking for verifying different control goals, which are related to robust stability, robust performance, and non-fragility. Previous related work [5]–[8] developed symbolic execution methods for control systems to check the closed-loop performance and safety properties violations in hybrid systems. In recent work [9]–[11], formal robustness verification for CPS is proposed, considering continuous and discrete disturbance (no model uncertainty).

The main goal here is to propose a comprehensive model checking procedure, which is able to formally verify stability, fragility, and robustness of closed-loop systems without employing hybrid automata. Bounded Model Checking (BMC) based on Satisfiability Modulo Theories (SMT), showed to be suitable for investigating the fragility problem in control systems. In particular, the stability verification method does not demand the execution/simulation of the plant behavior and simultaneously considers the controllers' fragility and the closed-loop robustness to exogenous disturbances and to plant model uncertainties.

As a result, a verification methodology and respective tool implementation to deal with stability verification of closed-loop system using an SMT-based approach are proposed. The present approach considers finite-word length (FWL) effects over controllers and also parameter uncertainties in the context of verifying the (so-called) robust non-fragile stability. This approach extends previous studies in which the digital control design is verified and re-adjusted iteratively until it reaches a digital controller, which is safe w.r.t. implementation problems, such as overflows, limit cycles, round-off errors, poles and zeros sensitivity [12].

The proposed verification methodology is implemented in the digital-system verifier (DSVerifier) tool[1] that uses the Efficient SMT-Based Context-Bounded Model Checker (ESBMC) as a verification engine for checking digital system properties [13], [14]. DSVerifier builds a closed-loop model, associating the controller model with FWL effects and the plant model with non-deterministic coefficients related to model uncertainties. DSVerifier then performs a symbolic analysis to verify certain properties, *e.g.*, stability, for all the plant family defined by the uncertainties. If there is a property violation, then DSVerifier indicates a failure and presents a counterexample, a plant model belonging to the plant family, which violates the property.

This work makes two major contributions. First, a new methodology for verifying closed-loop linear time-invariant

• *I. Bessa, L. Cordeiro, and J. E. Chaves Filho are with Federal University of Amazonas, Brazil. H. Ismail is with FPF Tech (Paulo Feitoza Foundation). R. Palhares is with Federal University of Minas Gerais, Brazil. E-mails: {iurybessa, lucascordeiro, jo_edgar}@ufam.edu.br, hussama.ismail@fpf.br, and rpalhares@ufmg.br*

1. Available at http://dsverifier.org/

systems is presented considering FWL effects for non-fragile control studies. In particular, a stability verification algorithm of uncertain systems considering FWL effects to aid control system designers in validating (digital) controllers is described. Second, the efficiency and effectiveness of the verification methodology and respective tool using control system benchmarks from literature are evaluated.

## 2 PRELIMINARIES

### 2.1 Transfer Function for Discrete Systems

There are various mathematical representations for discrete-time systems, e.g., difference equations, state-space, and transfer functions (or matrices). Here, linear time-invariant (LTI) systems with single-input and single-output (SISO) are discussed and represented by transfer functions as:

$$H(z) = \frac{b_0 + b_1 z^{-1} + ... + b_M z^{-M}}{a_0 + a_1 z^{-1} + ... + a_N z^{-N}}, \quad (1)$$

where the roots of numerator are called zeros of $G(z)$ and the roots of denominator are called poles of $G(z)$.

For convenience, a vector notation for the coefficients of $H(z)$ will be used, where a vector $h$, called coefficient vector of $H(z)$, is built by the numerator coefficients followed by the denominator coefficients as described by:

$$h = [b_0 \;\; b_1 \;\; ... \;\; b_M \;\; a_0 \;\; a_1 \;\; ... \;\; a_N] \quad (2)$$

### 2.2 Stability of Discrete Systems

A discrete-time system as (1) is said to be (asymptotic) stable if every pole lies inside the unit circle, *i.e.*, a circle in $z$ complex plane with unitary radius and center in origin [15]. Additionally, a discrete-time linear system is said to be Bounded-Input and Bounded-Output (BIBO) stable if and only if every pole of its transfer function lies inside the unit circle. Another important concept about stability is the internal stability. A system is internally stable if all its internal variables are bounded in addition to the stability of the closed-loop transfer function itself [15].

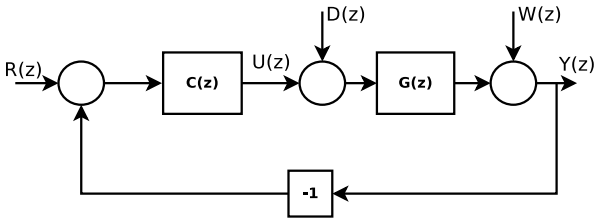



Fig. 1: Digital Control System with Disturbance and Measurement Noise.

As an example, consider the standard configuration described in Fig. 1 and choose as outputs $Y(z)$, the closed-loop system output and $U(z)$, the controller output; and as inputs $R(z)$, the reference input, $D(z)$, input disturbance, and $W(z)$, measurement noise, then

$$\begin{bmatrix} Y \\ U \end{bmatrix} = \begin{bmatrix} \frac{G(z)C(z)}{1+G(z)C(z)} & \frac{G(z)}{1+G(z)C(z)} & \frac{1}{1+G(z)C(z)} \\ \frac{C(z)}{1+G(z)C(z)} & \frac{-G(z)C(z)}{1+G(z)C(z)} & \frac{-C(z)}{1+G(z)C(z)} \end{bmatrix} \begin{bmatrix} R \\ D \\ W \end{bmatrix}. \quad (3)$$

**Definition 1.** *[15] – If all transfer functions, which relate the system inputs to the possible system outputs are BIBO stable, then the system is said to be internally stable, i.e., the system outputs $U(z)$ and $W(z)$ are still bounded (stable) for any $R(z)$, $D(z)$, and $W(z)$.*

**Theorem 1.** *[15] – The system in Fig. 1 is internally stable if and only if every closed-loop pole lies inside the open unit circle.*

A conclusion of Theorem 1 is the Lemma 1.

**Lemma 1.** *[15] – A feedback digital control system as shown in Fig. 1, with transfer functions $C(z) = N_C(z)/D_C(z)$ and $G(z) = N_G(z)/D_G(z)$, is internally stable if and only if:*

- *the roots of characteristic polynomial $S(z)$ are inside the open unit circle, where $S(z)$ is:*

$$S(z) = N_C(z)N_G(z) + D_C(z)D_G(z); \quad (4)$$

- *the direct loop product $\frac{N_C(z)}{D_C(z)} \cdot \frac{N_G(z)}{D_G(z)}$ has no pole-zero cancellation on or outside the unit circle.*

Throughout this paper, all controllers $C(z)$ are supposed to be asymptotic stable and they are not susceptible to overflow and limit cycles oscillation.

### 2.3 Digital Control System Design and Implementation

A notable issue related to digital control systems is concerned with their computational implementation, which should be considered in addition to the control performance. Issues related to digital control systems design are: sample period, quantization, computer arithmetic, word-length, memory usage, delays, controllers' realization, anti-windup action, and bumpless transfer.

#### 2.3.1 Model Uncertainty

As control systems are usually based on linear models, referred to as approximate real-world plants, then control design should comply with uncertainties (parametric variations, non-modeled dynamics, and nonlinearities) and check properties related to robustness. Among the different strategies to represent uncertainties, the usual additive uncertain representation was adopted so that the transfer function of the plant $G(z)$ in Fig. 1 can be expressed in its uncertain version

$$\hat{G}(z) = G(z) + \Delta G(z), \quad (5)$$

where $\Delta G(z)$ is a bounded additive uncertain, $\hat{G}(z)$ is the uncertain model of $G(z)$, and $\hat{g}$ is the coefficient vector of $\hat{G}(z)$.

#### 2.3.2 Digital Controllers Implementation and Fragility

Among the several issues related to digital controller implementation, FWL effects due to round-offs and quantization should be carefully considered since they might lead to (small) imprecision and even instability. A realistic model to deal with FWL effects must include the quantization of every numerical value, including each arithmetic result (sums and products), input signals, and system coefficients. A notable effect is that the error accumulation might affect the representation of the digital controller poles and zeros [16] and possibly leading to closed-loop instability or performance degradation. This system sensitivity with respect to its implementation is called fragility. Keel and Bhattacharyya show that some robust and optimum controllers might present fragility characteristics and under some conditions might destabilize the closed-loop system [17]. Several techniques to deal with non-fragility control design (or reliable control) have been described [18]–[20].

Some non-fragile techniques describe FWL effects in the digital controller implementation as a perturbation such that it can be modeled as an uncertainty. Thus, the same representation of uncertain systems given in Eq. (5) can be used with the controller transfer function represented as

$$\hat{C}(z) = C(z) + \Delta C(z), \tag{6}$$

where $\Delta C(z)$ is a bounded additive uncertain, $\hat{C}(z)$ is the FWL model of $C(z)$, and $\hat{c}$ is the coefficient vector of $\hat{C}(z)$.

In contrast to the uncertain model presented before, the controller perturbation due to FWL effects might be precisely computed if the implementation characteristics are known. Indeed, a more realistic model should neither consider FWL effects as a non-deterministic perturbation nor the plant model as a discrete model, once FWL effects may be predicted and most plants are analog.

An important contribution of this study is to consider FWL effects in digital controllers. In this study, it is assumed that the implementation aspects are well-known (*e.g.*, number of bits, realization form, and sample time), and for each implementation of $C(z)$, there exists a function $FWL[\cdot] : \mathbb{R}^n \rightarrow Q[\mathbb{R}^n]$, which applies the FWL effects to a digital-system, where $Q[\mathbb{R}]$ represents the quantized set of representable real numbers in the chosen implementation format. Therefore, $\hat{c}$ might be appropriately computed by means of

$$\hat{c} = FWL[C(z)]. \tag{7}$$

### 2.4 Stability Verification as a Decision Problem

A robust non-fragile decision problem about the stability of a hybrid control system, with a digital controller that suffers from FWL effects and a plant with an uncertain model is resolved.

**Problem description.** Given a nominal plant model $G(z)$, an additive uncertainty over this model $\Delta G(z)$, a nominal digital controller $C(z)$, and a FWL function $FWL[\cdot]$ implementation, decide about the internal stability of a closed-loop system constituted by $G(z)$ and $C(z)$.

**Proposed solution.** Assuming that $\hat{G}(z) = \frac{\hat{N_G}(z)}{\hat{N_G}(z)}$ is given by (5) and $\hat{C}(z) = \frac{\hat{N_C}(z)}{\hat{N_C}(z)}$ has the coefficient vector given by (7), and considering Lemma 1, the decidability can be summarized as a decision problem about the roots computation of

$$S(z) = FWL[N_C(z)] \cdot \hat{N}_G(z) + FWL[D_C(z)] \cdot \hat{D}_G(z), \tag{8}$$

where the closed-loop control system is stable if and only if all the roots of $S(z)$ are inside the unit circle.

The Jury's criteria represent a necessary and sufficient condition to ensure that all polynomial roots are inside the unit circle [15]; the non-fragile robust stability verification consists in verifying, by means of SMT queries, the fulfillment of that condition by checking its satisfiability. In this paper, DSVerifier [13], which is an SMT-based BMC tool used for verifying digital systems, is employed in order to check the satisfiability of the Jury criteria.

#### 2.4.1 Model Checking Digital Systems with DSVerifier

SMT-based BMC was successfully applied to verify single- and multi-threaded programs [21]. However, the application of BMC to ensure correctness of discrete-time systems considering FWL effects (*i.e.*, verifying system robustness related to implementation aspects) is somewhat recent [12], [22], [23]. The basic idea behind BMC is to check the negation of a given property at a given depth.

**Definition 2.** *[24] – Given a transition system $M$, a property $\phi$, and a bound $k$; BMC unrolls the system $k$ times and translates it into a verification condition (VC) $\psi$, which is satisfiable if and only if $\phi$ has a counterexample of depth less than or equal to $k$.*

One prominent BMC tool is ESBMC [14], which is an SMT-based context-bounded model checker for C/C++ programs. DSVerifier uses ESBMC as its verification engine to check a digital-system [13]. In ESBMC, the associated problem is formulated by constructing the logical formula:

$$\psi_k = I(S_0) \wedge \bigvee_{i=0}^{k} \bigwedge_{j=0}^{i-1} \gamma(s_j, s_{j+1}) \wedge \overline{\phi(s_i)} \tag{9}$$

where $\phi$ is a property (*e.g.*, overflow and limit cycle) and $S_0$ is a set of initial states of $M$, and $\gamma(s_j, s_{j+1})$ is the transition relation of $M$ between time steps $j$ and $j+1$. Hence, $I(S_0) \wedge \bigwedge_{j=0}^{i-1} \gamma(s_j, s_{j+1})$ represents the executions of a transition system $M$ of length $i$. The above VC, $\psi$, can be satisfied if and only if, for some $i \leq k$ there exists a reachable state at time step $i$ in which $\phi$ is violated. If the logical formula (9) is satisfiable (i.e., returns *true*), then the SMT solver provides a satisfying assignment, from which the values of the digital controller's variables can be extracted to construct a counterexample.

## 3 Stability Verification of Closed-loop Digital Control Systems with Uncertainty

The proposed methodology for verifying the stability of closed-loop digital control systems with uncertainty is described as follows. The plant model must be represented by a parametric uncertain model, *i.e.*, plant intervals. Suppose that the digital controller $C(z)$ and the plant model are given as in (10) and (11), respectively:

$$C(z) = \frac{\beta_0 + \beta_1 z^{-1} + ... + \beta_{M_C} z^{-M_C}}{\alpha_0 + \alpha_1 z^{-1} + ... + \alpha_{N_C} z^{-N_C}}, \tag{10}$$

$$G(z) = \frac{b_0 + b_1 z^{-1} + ... + b_{M_G} z^{-M_G}}{a_0 + a_1 z^{-1} + ... + a_{N_G} z^{-N_G}}, \tag{11}$$

and $c$ and $g$, respectively, are the parameter vectors of $C(z)$ and $G(z)$.

The uncertain plant $\hat{G}(z)$ expressed by (5), whose uncertain parameters vector $\hat{g}$ can be expressed as follows

$$\hat{g} = g + \Delta g = \begin{bmatrix} b_0 + \Delta b_0 \\ b_1 + \Delta b_1 \\ \vdots \\ b_{M_G} + \Delta b_{M_G} \\ a_0 + \Delta a_0 \\ a_1 + \Delta a_1 \\ \vdots \\ a_{N_G} + \Delta a_{N_G} \end{bmatrix}, \tag{12}$$

where $\Delta g$ represents plant uncertainties and $\Delta g\%$ corresponds to maximum variation coefficients percentage vector of $g$, where $0 \leq i \leq M_G$ and $0 \leq j \leq N_G$ such that

$$\|\Delta b_i\| \leq \frac{\Delta b_i\% \cdot b_i}{100}, \tag{13}$$

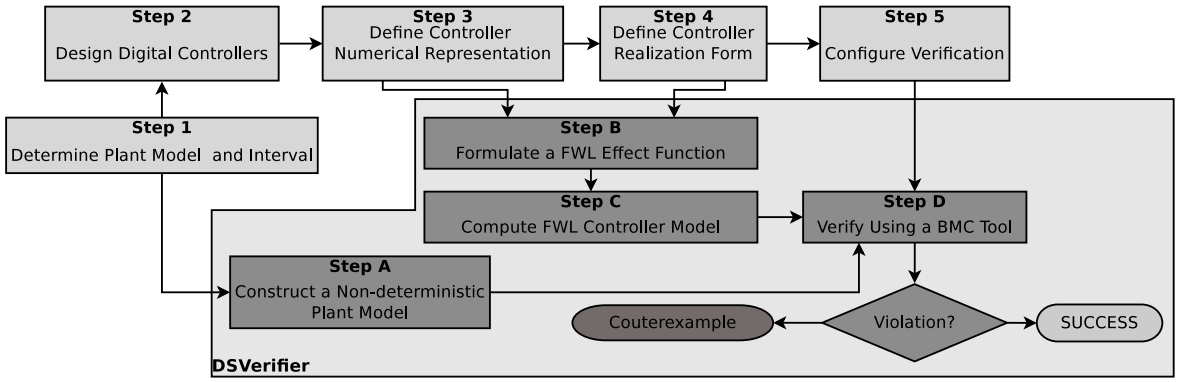



Fig. 2: DSVerifier Verification Flow.

$$\|\Delta a_j\| \leq \frac{\Delta a_j\% \cdot a_i}{100}, \tag{14}$$

$$\Delta p\% = \begin{bmatrix} \Delta b_0\% \\ \Delta b_1\% \\ \vdots \\ \Delta b_{M_G}\% \\ \Delta a_0\% \\ \Delta a_1\% \\ \vdots \\ \Delta a_{N_G}\%] \end{bmatrix}. \tag{15}$$

The possible values polynomial set of $g$ is denoted by $\mathfrak{P}$.

The DSVerifier verification process is shown in Fig. 2. Steps from 1 to 5 are performed by users and Steps A to D are automatically performed by DSVerifier, which accepts the digital controller transfer functions together with the plant model. For any digital controller, implementation details should also be incorporated into this symbolic analysis, which contain the FWL format, *e.g.*, number of bits, computational realization, and sample time.

In Step 1, the user provides inputs $g$ and $\Delta g\%$, which contain the plant model and the uncertainty interval. In Step 2, a digital controller must be designed with any preferred method, where $c$ is obtained. The controller numerical representation and realization form are chosen in Steps 3 and 4, respectively. In Step 5, the user finally configures the verification parameters, choosing verification time, properties to be verified, model checker, and SMT solver, and the verification engine is finally invoked.

DSVerifier performs an automatic verification of the desired property $\phi$. In Step A, DSVerifier builds a non-deterministic model to represent the plant family $\mathfrak{P}$ using $g$ and $\Delta g\%$, which are provided in Step 1. DSVerifier then formulates $FWL[\cdot]$ in Step B using implementation details provided from Steps 2 and 3, and then computes $FWL[C(z)]$ (that is equivalent to $\hat{c}$ in Step C). Additionally, DSVerifier symbolically checks a given property w.r.t. closed-loop systems, which are composed of $FWL[C(z)]$ and a non-deterministic $p \in \mathfrak{P}$, using a BMC tool. If any violation is found, then DSVerifier reports a counterexample, which contains system inputs or the uncertain parameter vector $\hat{g}\%$, which leads to a failure. A successful result is reported if the system is safe w.r.t. $\phi$ up to the bound $k$.

Note that the verification performed by DSVerifier can produce false alarms due to rounding effects or it can miss a bug due to the (chosen) loop unwinding bound $k$. For the robust non-fragile stability verification, there is no need for loop unwinding, so the chosen bound $k$ does not influence the verification result. Thus, the internal stability verification of closed-loop system is valid for any time-varying disturbances, and it does hold for an unbounded execution trace. However, in this paper, stability verification method is incomplete, given that false alarms can be produced due to the rounding effects, which are accumulated along the chain of arithmetic machine operations. Since DSVerifier employs a fixed-point format with 64 bits, of which 32 are precision bits (*i.e.* precision of $10^{-10}$), no false alarm is observed throughout the experimental evaluation.

### 3.1 Closed-loop Stability Verification Algorithm

To obtain a decision about the internal stability (Theorem 1), the characteristic polynomial $S(z)$ given in (4) is used. There are two different algorithms in DSVerifier that can be used for stability verification, one based on Schur's decomposition and another one based on Jury's criteria [12]; here, the Jury's method is chosen due to its efficiency.

Due to the interest in ensuring the stability for any model inside uncertain intervals and considering also the FWL effects, the basic steps of the verification algorithm are as follows: the application of FWL effects on the numerator and denominator of the controller and the use of the Jury's criteria to determine the stability of the characteristic polynomial $S(z)$ given in (4). Algorithm 1 presents the steps of the closed-loop stability verification process.

#### 3.1.1 SMT Encoding of Jury's Criteria

Jury's algorithm is used to check the stability in the $z$-domain for a given characteristic polynomial of the form

$$S(z) = a_0 z^N + a_1 z^{N-1} + ...a_{N-1} z + a_N = 0, a_0 \neq 0 \tag{16}$$

In particular, Jury stability test is already explained in the control system literature (*e.g.* [15]). This study, however, limits itself to explain the SMT encoding of Jury's criteria. For the stability test procedure, the following Jury matrix $M = [m_{ij}]_{(2N-2)\times N}$ is built from $S(z)$ coefficients:

$$M = \begin{bmatrix} V^{(0)} \\ V^{(1)} \\ \vdots \\ V^{(N-2)} \end{bmatrix}, \tag{17}$$

**Data**: $N_C(z)$, $N_G(z)$, $D_C(z)$, $D_G(z)$, implementation settings, and plant parameters intervals in percentage $\Delta g\%$.
**Result**: Stability decision: SUCCESS for stable systems and FAILED for unstable systems together with a counterexample.

```
begin
    Formulate a FWL effect function FWL[·]
    Construct the plant interval set 𝔓
    Obtain FWL[N_C(z)] and FWL[D_C(z)]
    Check ¬φ_stability for
    S(z) = FWL[N_C(z)] · N̂_G(z) + FWL[D_C(z)] · D̂_G(z)
    in 𝔓
    if ¬φ_stability is satisfiable then
        return UNSTABLE and a counterexample
    end
    else
        return STABLE
    end
end
```

**Algorithm 1:** Closed-loop stability verification

where $V^{(k)} = [v_{ij}^{(k)}]_{2\times N}$ such that

$$v_{ij}^{(0)} = \begin{cases} a_{j-1}, & \text{if } i = 1 \\ v_{(1)(N-j+1)}^{(0)}, & \text{if } i = 2 \end{cases}, \text{ and} \tag{18}$$

$$v_{ij}^{(k)} = \begin{cases} 0, & \text{if } j > n-k \\ v_{1j}^{(k-1)} - v_{2j}^{(k-1)} \cdot \frac{v_{11}^{(k-1)}}{v_{21}^{(k-1)}}, & \text{if } j \leq n-k \text{ and } i = 1 \\ v_{(1)(N-j+1)}^{(k)}, & \text{if } j \leq n-k \text{ and } i = 2 \end{cases}, \tag{19}$$

where $k \in \mathbb{Z}$, such that $0 < k < N-2$. $S(z)$ is the characteristic polynomial of a stable system if and only if the following four propositions hold:

- $R_1$: $S(1) > 0$;
- $R_2$: $(-1)^N S(-1) > 0$;
- $R_3$: $|a_0| < a_N$;
- $R_4$: $m_{11} > 0 \iff m_{31} \wedge m_{51} \wedge \cdots \wedge m_{(2N-3)(1)}$.

The stability property is then encoded by creating a constraint using the fixed size bit-vector theory, typically supported by state-of-the-art SMT solvers [25]:

$$\phi_{stability} \iff (R_1 \wedge R_2 \wedge R_3 \wedge R_4), \tag{20}$$

where the literal $\phi_{stability}$ represents the validity of the stability condition; in particular, the SMT-solver checks whether Jury criteria hold for the characteristic polynomial coefficients.

In the robust non-fragile verification presented in this study, $S(z)$ is computed by (8) and coefficients of $\hat{N}_G(z)$ and $\hat{D}_G(z)$ are non-deterministic fixed-point values within the range defined by (12),(13) and (14). If the system is unstable, *i.e.*, if $\neg\phi_{stability}$ is satisfiable for $S(z)$, then the SMT-solver provides values for each coefficients of $\hat{N}_G(z)$ and $\hat{D}_G(z)$, which make the closed-loop system into unstable.

The present closed-loop stability verification is suitable for discrete LTI systems. The plant and the digital controller models have to be linear. The verification result is unsound if nonlinearities are considered. Although most real-world plants are nonlinear, a simplified linear model is often sufficient to describe the system dynamics and behavior in its operating region. However, the digital controller might present various nonlinearities due to FWL effects, *e.g.*, saturation and wrap-around due to overflow; limit cycle oscillation (LCO) due to successive round-offs; and truncation due to underflow. The verification is considered to be sound if controllers are not susceptible to both overflow and LCO. Underflow does not affect the system stability, since it only occurs for small control actions, *i.e.*, low-level signals from the digital controller output. Round-offs are serious problems only if they affect poles and zeros position; our verification algorithms check for that effect type. Actually, DSVerifier is able to check the occurrence of overflow and LCO in digital controllers implementations; despite successfully preventing many FWL effects in digital controllers using the methodology presented in [12], DSVerifier cannot ensure the absence of overflow and LCO, unless some induction technique is used.

## 4 Experimental Evaluation

To evaluate the proposed verification methodology and the respective tool performance, classical examples previously presented in [17] and [26] regarding fragility, stability, and inter-sample response of hybrid systems are considered.

### 4.1 Example A

Consider the following description for the plant $G_1(s)$ with the controller $C_1(s)$:

$$G_1(s) = \frac{s-1}{s^2 - s - 2} \tag{21}$$

$$C_1(s) = \frac{11.44974739s + 11.242640066}{s - 7.03553383} \tag{22}$$

The DSVerifier must receive the discrete model of plant and controller. The plant sampled model with zero-order hold was employed (Step 1) and the controller was discretized using Tustin method with seven different sample times (0.5, 0.1, 0.05, 0.03, 0.01, 0.005, 0.001, 0.00001, and 0.000001 seconds) (Step 2) and implemented using three different FWL formats, 4, 8, and 12-bits (Step 3), and direct-form I realization (Step 4). The DSVerifier was configured using the ESBMC model checker with Z3 solver (Step 5)[2].

First, FWL effects are considered over stability. Alg. 1 is applied to this closed-loop system despite the plant uncertainties, *i.e.* considering $\Delta p\% = 0$ (Step 1). Hence, the stability verification is repeated for every combination of FWL format and sample time. In this experiment, DSVerifier generates the verification results presented in Table 1 for controller $C_1$; the verification time takes less than 1s .

The results confirm the conclusions in [26], which claims that this hybrid system becomes unstable for high sample times (*e.g.*, $T_s = 0.5$s) and presents low stability margins for lower sample times. As a result, it presents fragility and may easily loss the stability for numerical reasons, *e.g.*, quantization noise caused by a FWL format with small precision due to insufficient number of bits. Note that the experimental results also show instability for $T = 0.5$s and even worse results for less bits representations with low sample periods (*e.g.*, $T_s \leq 0.01$). Fig. 3 shows the FWL

2. DSVerifier is called as: `dsverifier <filename> --realization DFI --property STABILITY_CLOSED_LOOP --bmc ESBMC --solver z3`

| Sample Time (s) | FWL Format | | |
|---|---|---|---|
| | 4-bits | 8-bits | 12-bits |
| 0.5 | U | U | U |
| 0.1 | S | S | S |
| 0.05 | S | S | S |
| 0.03 | S | S | S |
| 0.01 | U | S | S |
| 0.005 | U | S | S |
| 0.001 | U | U | S |
| 0.00001 | U | U | U |
| 0.000001 | U | U | U |

TABLE 1: Closed-loop stability verification results using controller $C_1(s)$ with different sample times and FWL formats. S= Stable and U= Unstable

effect in the controller stability, the same control system, composed by the plant $G_1(s)$ and digital controller $C_1(s)$ discretized with a sample period of $0.01$s, which is stable for a digital controller implementation of a 12-bit (right frame), but it is unstable for a $4$-bit implementation (left-frame). If the controller implementation uses 8 bits, then the system is still stable but it presents a greater settling time.

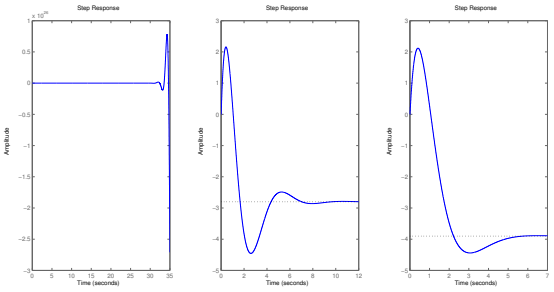



Fig. 3: Step response of $G_1(s)$ and $C_1(s)$ for sample time of $0.01s$. FWL formats: 4-bits (left), 8-bits (center), and 12-bits (right).

The verification for a different scenario can also be considered. Controller $C_1$ implemented with 12-bits of precision and sample time of $0.03$s produced a stable behavior in the previous tests, as shown in Table 1. However, if a maximum deviation of $\pm 0.25\%$ is considered in each coefficient of $G(s)$, then DSVerifier shows that this controller cannot ensure robustness, given that it presents as counterexample:

$$\Delta p\% = \begin{bmatrix} -0.18660 \\ -0.06731 \\ -0.21599 \\ 0.06569 \\ 0.06567 \end{bmatrix}. \tag{23}$$

The closed-loop step response in Fig. 4 confirms the results provided by DSVerifier; this closed-loop system is stable for nominal parameters, but it is unstable for the parametric variation given by the counterexample in (23).

### 4.2 Example B

Consider the plant and controller given by:

$$G_2(s) = \frac{s-1}{s^2 + 0.5s - 0.5} \tag{24}$$

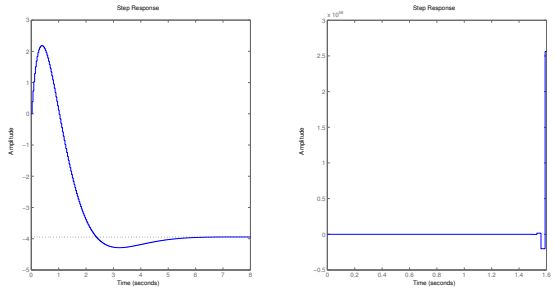



Fig. 4: Step response of $G_1(s)$ and $C_1(s)$ for sample time of $0.03s$ and 12-bits. Left: Nominal system. Right: System with uncertainty in (23).

$$C_2(s) = \frac{-124.5s^3 - 364.95s^2 - 360.45s - 120}{s^3 + 227.1s^2 + 440.7s + 220} \tag{25}$$

Table 2 presents the verification results without uncertainties and is in compliance with the conclusions in [26].

| Sample Time (s) | FWL Format | | |
|---|---|---|---|
| | 4-bits | 8-bits | 12-bits |
| 0.5 | U | S | S |
| 0.1 | U | U | S |
| 0.05 | U | U | S |
| 0.03 | U | U | U |
| 0.01 | U | U | U |
| 0.005 | U | U | U |
| 0.001 | U | U | U |
| 0.0004 | U | U | U |

TABLE 2: Closed-loop stability verification results for example B, considering different sample times and FWL formats. S= Stable and U= Unstable

Fig. 5 shows the step responses of the closed-loop system with $G_2(s)$ and $C_2(s)$ discretized with sample time of $0.5$s. The FWL implementation with higher number of bits (12-bits) is stable, but the same closed-loop system with $4$-bits implementation is unstable. The implementation with $8$-bits is also stable; however, it presents a greater overshooting if compared to the 12-bits implementation.

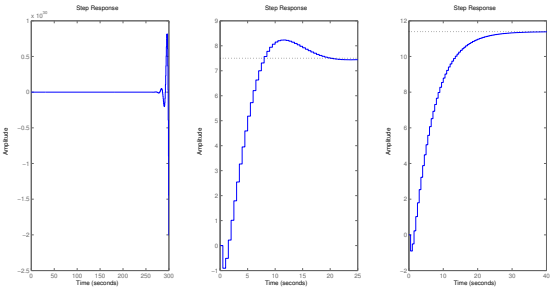



Fig. 5: Step response of $G_2(s)$ and $C_2(s)$ for sample time of $0.5s$. FWL formats: 4-bits (left), 8-bits (center), and 12-bits (right).

Note further that for the controller in the FWL format of 8-bits, assuming a sample time of $0.5$s and each coefficient for plant $G_2$ varying $\pm 1\%$, DSVerifier determines that the

system is unstable, returning the following counterexample:

$$\Delta p\% = \begin{bmatrix} 0.54336 \\ -0.12631 \\ 0.66145 \\ 0.97228 \\ 0.79776 \end{bmatrix}. \quad (26)$$

Fig. 6 compares the step response for the closed-loop system with nominal parameters and with variation given in (26). Note that a digital controller, apparently non-fragile (*i.e.*, it stabilizes the closed-loop system even in the face of FWL effects), may not work properly for uncertain systems.

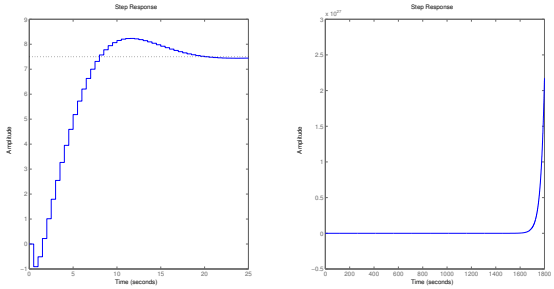



Fig. 6: Step response of $G_2(s)$ and $C_2(s)$ for sample time of $0.5s$ and 8-bits. Left: Nominal system. Right: System with uncertain in (23).

These examples show that a specific FWL implementation of a control system may not affect the stability and performance for a nominal system with well-known parameters; however, a small deviation in these parameters may fatally affect the system behavior. DSVerifier may consider simultaneously plant uncertainties and controllers fragility for providing an efficient diagnosis about the stability of closed-loop systems for a specific FWL format by means of the verification methodology presented.

## 5 Related Work

The symbolic verification of closed-loop system had an important advance in the last decades. Relevant studies (*e.g.*, [5], [7], [8], [27]) about the performance and safety verification of closed-loop systems propose verification methods based on symbolic execution of a plant model. As example, the Closed-Loop Symbolic Execution (CLSE) [8] performs a bounded-time symbolic execution of the plant dynamic, which is represented by ordinary difference equations (ODEs) combined with a concolic execution of the controller software. Additionally, a robustness analysis is also performed in [8], where the deviation on the plant states is computed due to a deviation on the sensor signals (measurement noise). The approach used in this study differs from CLSE given that its internal stability analysis does not require the system execution; it is based on Jury's Criteria, which is applied to the plant and controller model.

Costan [28] verifies the stability of closed-loop systems on an embedded C code controller, comparing the Simulink implementation of the control system with the code generated by Mathworks Fixed-Point Advisor and Real-Time Workshop [29]. One notable feature of Costan is the error calculation by means of static analysis in the controller code for bounded loops unrolling. The deviations are compared to a pre-computed error bound, which indicates the maximum admissible error for what the closed-loop system remains stable. If any violation is found, then Costan provides a concrete test input that leads the system to the failure. In contrast, DSVerifier computes the quantization effects and checks the stability in the closed-loop function for all the plant family without handling the usual stability margin concept; this makes DSVerifier slower than Costan, but DSVerifier presents an improved accuracy, which is suitable for systems that require a correct-by-design approach [30].

SAHVY simulates the systems execution, solving ODEs represented by Taylor models, for a range of initial states, and performs SMT-based BMC inside of this range, to check safety properties expressed by CTL formulas. The BMC tool is similar to the verification engine used by DSVerifier, but SAHVY is limited to hybrid systems with zero order holding sampling and not taking into account FWL effects in the controller, in contrast to Costan and DSVerifier.

Barnat *et al.* [31], [32] present a very promising approach, which uses Simulink diagrams to open up new possibilities towards verification properties beyond the standard stability tests for first-order system. This approach, however, is still under development; there are limitations mainly related to the theorem's proof (Why3). The use of Why3 can solve problems of previous studies related to the state-space explosion [31]; however, differently from model checking tools, Why3 is not fully automatic, *i.e.*, the user has to manually change parameters to produce new proofs; additionally, there is neither counterexample nor error trace generation. Any comparison to their work may not be seen as an easy task since the verification is over Simulink models, while in this study, the focus is on the controller C code.

A drawback of DSVerifier, if compared to aforementioned tools, is the limited class of systems that it can actually verify, *i.e.*, only linear systems represented by transfer functions, but that can describe a huge amount of real-world systems. DSVerifier is a unique tool since it is the only one to consider simultaneously the model uncertainties and FWL effects in the controller. In particular, DSVerifier is able to verify the robust non-fragile properties, while other existing approaches are unable to handle them. Additionally, other model checking tools are not capable of performing robust analysis and deal with FWL effects together. DSVerifier uses model checking techniques and presents the advantages of these tools, *e.g.*, higher reliability and precision, counterexamples for failures, and it is completely automated.

Recently, some studies presented SMT applications for the verification of control software and correct-by-design controller synthesis. Pajic *et al.* use input-output invariants that allow the representation of inexact controllers representation. Bessa *et al.* [12] employed DSVerifier to find FWL problems in controllers implementations in direct- and delta-form realizations. Recent studies [10], [33], [34] apply SMT-based verification to controllers synthesis.

Finally, Tabuada *et al.* [11] establish the important notion of robustness for CPS and propose a methodology for verifying the robust input-output stability for those systems. It is a first step towards the application of formal verication to ensure the CPS robustness. In contrast to the present robustness verication, Tabuada *et al.* consider only the robustness to exogenous continuous and discrete disturbances. The present methodology ensures the internal stability, which

includes the robustness to exogenous signals, considering plant model uncertainties and the fragility issue.

## 6 Conclusion

A verification methodology for checking stability of uncertain linear control systems is described and evaluated, considering FWL effects in fixed-point digital controllers. This methodology offers an alternative approach to check the fragility problem, which computes the exact effect of the FWL implementation and investigates the robustness maintenance under FWL effects. A suitable verification tool (DSVerifier) to support this methodology is also presented.

A few previous studies proposed the investigation of FWL effects in the stability of closed-loop systems; all those studies incorporate FWL noise as uncertainties or perturbations, but none of them presented a verification methodology and respective tool to verify simultaneously the systems fragility and robustness. This study, in turn, presents a verification methodology supported by a formal verification tool, which considers the FWL as deterministic effects and determines the stability maintenance for the plant interval represented by non-deterministic coefficients.

The experimental results show that the proposed methodology is efficient and effective for verifying robustness and fragility of closed-loop systems, with automatism and correctness provided by model checking techniques. Further studies include the extension of this verification approach for different classes of systems, performance requirements verification, and controller implementation synthesis.

## Acknowledgments

The authors thank the financial support of Amazonas State Research Foundation (FAPEAM), Brazil, the Brazilian National Research Council (CNPq), and the Research Foundation of the State of Minas Gerais (FAPEMIG), Brazil.

## References

[1] A. Filieri, M. Maggio, K. Angelopoulos, N. D'Ippolito, I. Gerostathopoulos, A. Hempel, H. Hoffmann, E. Jamshidi, C. Klein *et al.*, "Software engineering meets control theory," in *Proc. Int'l Symp. on Software Engineering for Adaptive and Self-Managing Systems, Florence, Italy*, 2015.

[2] B. Rawlings, J. Kim, I. Moon, and B. Ydstie, "Symbolic verification of control systems and operating procedures," *Industrial and Engineering Chemistry Research*, vol. 53, no. 13, pp. 5299–5310, 2014.

[3] R. Alur, C. Courcoubetis, and D. Dill, "Model-checking for real-time systems," in *Proc. IEEE Symp. on Logic in Computer Science*, 1990, pp. 414–425.

[4] T. Henzinger, P.-H. Ho, and H. Wong-Toi, "Hytech: the next generation," in *Proc. Real-Time Systems Symp.*, 1995, pp. 56–65.

[5] A. Zutshi, S. Sankaranarayanan, J. V. Deshmukh, and X. Jin, "Symbolic-numeric reachability analysis of closed-loop control software," in *Proc. 16th Int'l Conference on Hybrid Systems: Computation and Control*, 2016, pp. 135–144.

[6] M. Pajic, J. Park, I. Lee, G. J. Pappas, and O. Sokolsky, "Automatic verification of linear controller software," in *Proc. 12th Int'l Conference on Embedded Software*. Piscataway, NJ, USA: IEEE Press, 2015, pp. 217–226.

[7] P. S. Duggirala and M. Viswanathan, "Analyzing real time linear control systems using software verification," in *IEEE Real-Time Systems Symposium*, Dec 2015, pp. 216–226.

[8] R. Majumdar, I. Saha, K. Shashidhar, and Z. Wang, "CLSE: Closed-loop symbolic execution," *Proc. 4th Int'l Symposium on NASA Formal Methods, LNCS*, vol. 7226, pp. 356–370, 2012.

[9] M. Rungger and P. Tabuada, "Abstracting and refining robustness for cyber-physical systems," in *Proc. 17th Int'l Conference on Hybrid Systems: Computation and Control*. ACM, Apr 2014, pp. 223–232.

[10] ——, "A symbolic approach to the design of robust cyber-physical systems," in *52nd IEEE Conference on Decision and Control*, Dec 2013, pp. 3932–3937.

[11] P. Tabuada, S. Caliskan, M. Rungger, and R. Majumdar, "Towards robustness for cyber-physical systems," *IEEE Trans. on Automatic Control*, vol. 59, no. 12, pp. 3151–3163, 2014.

[12] I. Bessa, H. Ismail, L. Cordeiro, and J. Filho, "Verification of fixed-point digital controllers using direct and delta forms realizations," *Design Autom. for Emb. Sys.*, vol. 20, no. 2, pp. 95–126, 2016.

[13] H. Ismail, I. Bessa, L. Cordeiro, E. de Lima Filho, and J. Chaves Filho, "DSVerifier: A Bounded Model Checking Tool for Digital Systems," in *Model Checking Software*, ser. LNCS. Springer International Publishing, 2015, vol. 9232, pp. 126–131.

[14] L. Cordeiro, B. Fischer, and J. Marques-Silva, "SMT-Based Bounded Model Checking for Embedded ANSI-C Software," *IEEE Trans. on Software Engineering*, vol. 38, no. 4, pp. 957–974, 2012.

[15] S. Fadali and A. Visioli, *Digital Control Engineering: Analysis and Design*, ser. Electronics & Electrical. Elsevier/Academic Press, 2009, vol. 303.

[16] G. Li, "On pole and zero sensitivity of linear systems," *IEEE Trans. on Circuits and Systems–I: Fundamental Theory and Applications*, vol. 44, no. 7, pp. 583–590, 1997.

[17] L. Keel and S. Bhattacharyya, "Robust, fragile, or optimal?" *IEEE Trans. on Automatic Control*, vol. 42, no. 8, pp. 1098–1105, 1997.

[18] M. Mahmoud, *Resilient Control of Uncertain Dynamical Systems*, ser. LNCIS. Springer, 2004.

[19] G. Yang, X. Guo, W. Che, and W. Guan, *Linear Systems: Non-Fragile Control and Filtering*. Taylor & Francis, 2013.

[20] G. Yang and D. Ye, *Reliable Control and Filtering of Linear Systems with Adaptive Mechanisms*, ser. Automation and Control Engineering. CRC Press, 2010.

[21] D. Beyer, "Status report on software verification (competition summary SV-COMP 2014)," *LNCS*, vol. 8413, pp. 373–388, 2014.

[22] A. Cox, S. Sankaranarayanan, and B.-Y. E. Chang, "A bit too precise? Verification of quantized digital filters," *Int'l Journal on Soft. Tools for Tech. Transfer*, vol. 16, no. 2, pp. 175–190, 2014.

[23] R. B. Abreu, L. C. Cordeiro, and E. B. L. Filho, "Verifying Fixed-Point Digital Filters using SMT-Based Bounded Model Checking," in *Proc. Brazilian Telecommunications Symp.*, 2013.

[24] A. Biere, "Bounded model checking," in *Handbook of Satisfiability*. IOS Press, 2009, pp. 457–481.

[25] C. W. Barrett, R. Sebastiani, S. A. Seshia, and C. Tinelli, "Satisfiability modulo theories," in *Handbook of Satisfiability*. IOS Press, 2009, pp. 825–885.

[26] L. Keel and S. Bhattacharyya, "Stability margins and digital implementation of controllers," in *Proc. American Control Conference*, vol. 5, 1998, pp. 2852–2856.

[27] A. Zutshi, S. Sankaranarayanan, J. V. Deshmukh, J. Kapinski, and X. Jin, "Falsification of safety properties for closed loop control systems," in *Proc. 18th Int'l Conference on Hybrid Systems: Computation and Control*, 2015, pp. 299–300.

[28] A. Anta, R. Majumdar, I. Saha, and P. Tabuada, "Automatic Verification of Control System Implementations," in *Proc. Int'l Conf. on Embedded Software*, 2010, pp. 9–18.

[29] B. Chou and T. Erkkinen, "Converting models from floating point to fixed point for production code generation," Matlab Digest, 2008.

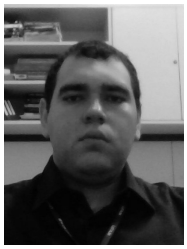
[30] K. Vorobyov and P. Krishnan, "Comparing model checking and static program analysis: A case study in error detection approaches," in *Proc. Int'l Work. on Systems Software Verification*, 2010.

[31] P. Bauch, V. Havel, and J. Barnat, "Accelerating temporal verification of simulink diagrams using satisfiability modulo theories," *Software Quality Journal*, 2014, 10.1007/s11219-014-9259-x.

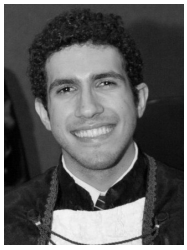
[32] J. Barnat, P. Bauch, and V. Havel, "Temporal verification of simulink diagrams," in *Proc. Int'l Symp. on High-Assurance Systems Engineering*, 2014, pp. 81–88.

[33] Z. Huang, Y. Wang, S. Mitra, G. E. Dullerud, and S. Chaudhuri, "Controller synthesis with inductive proofs for piecewise linear systems: An SMT-based algorithm," in *Proc. 54th IEEE Conference on Decision and Control*, Dec 2015, pp. 7434–7439.

[34] H. Ravanbakhsh and S. Sankaranarayanan, "Counter-example guided synthesis of control lyapunov functions for switched systems," in *Proc. 54th IEEE Conference on Decision and Control*, Dec 2015, pp. 4232–4239.



**Iury Bessa** received the BSc and the MSc degrees in electrical engineering from the Federal University of Amazonas (UFAM), Brazil, in 2013 and 2015 respectively. He is an associate professor in the Department of Electricity at Federal University of Amazonas (UFAM). He is currently working toward the PhD degree at the Federal University of Minas Gerais (UFMG). His research interests include: reconfigurable control, process supervision and control, mobile robotics, and formal verification of cyber-physical systems.



**Hussama Ibrahim Ismail** holds a B.Sc. degree in Computer Engineering from Foundation Center for Analysis, Research, and Technological Innovation (FUCAPI) in 2013 and a M.Sc. degree in Electrical Engineering from Federal University of Amazonas (UFAM) in 2015. Nowadays, he is a Systems Analyst at FPF Tech (Paulo Feitoza Foundation), who has been working with software development involving Linux, Shell Script, ANSI-C, C++, Java, and JavaScript since 2011. His current research interests are formal methods, bounded model checking, and embedded systems.

**Reinaldo Martinez Palhares** is currently a professor at the Department of Electronics Engineering, Federal University of Minas Gerais, Brazil. Palhares' main research interests include robust linear/nonlinear control theory, optimization and soft computing. Palhares has been serving as an Associate Editor for IEEE Transactions on Industrial Electronics, Guest Editor of the Journal of The Franklin Institute Special Section on Recent Advances on Control and Diagnosis via Process measurements and Guest Editor of the IEEE/ASME Transactions on Mechatronics Focused Section on Health Monitoring, Management and Control of Complex Mechatronic Systems.

**Lucas Cordeiro** received the Ph.D. degree in computer science from the University of Southampton in 2011. From 2011 to 2016, he was an adjunct professor in the Electronics and Computing Engineering Department at the Federal University of Amazonas. Since 2016, he is a researcher in verification of embedded systems in the Department of Computer Science at the University of Oxford. His current research interests include software verification, model checking, satisfiability modulo theories, and embedded systems.

**João Edgar Chaves Filho** received his Ph.D. degree in Electrical Engineering in 2001 from the Universidade Federal de Campina Grande (UFCG), Paraba, Brazil. His M.Sc. was also obtained from UFCG in 1991. Since 1983, he has been a Faculty Member with the Electrical Engineering Department at UFAM. His research interests include new industrial automation proposals, artificial intelligence, control systems, and system identification.