Modeling and Simulation of Event Systems

Yuri Shornikov *, Dmitry Dostovalov *, Viktor Astapchuk, Natalie Ganelina, Konstantin A Timofeev

Posted Date: 3 July 2024

doi: 10.20944/preprints202407.0261.v1

Keywords: hybrid systems; event detection; numerical integration; algorithm for correct event detection; Harel diagrams; statechart; event-continuous systems; event-discrete systems; state machine behavior; event function

Article

Modeling and Simulation of Event Systems

Yury Shornikov *, Dmitry Dostovalov, Viktor Astapchuk, Natalie Ganelina and Konstantin Timofeev

Novosibirsk State Technical University, Novosibirsk, Russia; d.dostovalov@corp.nstu.ru; astapchuk@corp.nstu.ru; ganelina@corp.nstu.ru; timofeevka@gmail.com

* Correspondence: shornikov@corp.nstu.ru

Abstract: The advanced industrial systems are characterized by complex event-driven processes. The specification and computer analysis are illustrated by the common problem of a servo drive with the PWM controller. A mathematical model from the class of Cauchy problems with non-liner control logic is presented to resolve the task discussed. The designed model is performed in two stages: the mathematical model specification (modeling) and implementation (simulation). Two approaches to make a model specification are considered, the first one is the traditional structural technique, and the advanced method is based on the finite state machine paradigm. Structural and finite state software models are built in progressive simulation tools. Computer implementation of the software models of hybrid systems is performed using platform built-in libraries with numerical methods and algorithms for correct event detection. The results of computational experiments for transient processes of phase variables are qualitatively identical, that proves the adequacy of the developed models. The algorithm for asymptotic event detection is considered in detail. This technique demonstrated the best results by the one-sidedness criterion, which is explained by using the author's algorithm for the integration step determining and the event function dynamics monitoring in the vicinity of the event limits.

Keywords: hybrid systems; event detection; numerical integration; algorithm for correct event detection; Harel diagrams; statechart; event-continuous systems; event-discrete systems; state machine behavior; event function

1. Introduction

The PWM controllers are widely used in advanced industrial systems for efficient engine controlling [1,2]. Objective designing and adequate stable planning the functioning modes of controlling systems for servo drives within the Industry 4.0 technologies is impossible without applying computer modeling and simulation. The article considers modeling the dynamics of driving gears with the pulse-width modulation (PWM) controllers using Russian and foreign software tools. Traditionally, the presented illustrative example of the servo drive with the PWM controller is defined as a continuous system with a software model in the form of the schematic diagram in Matlab/Simulink [3] and SimInTech [4] platforms. On the other hand, the same control object is considered as an event-driven continuous or hybrid system, defined by the advanced finite state machine formalism as statecharts or the Harel diagrams [5]. Such a specification is performed using ISMA [6] and Ptolemy II [7] tools. The conclusion about software models with the specification applied to be appropriate in terms of the domain-driven adequacy is made. Efficient simulating and computer analysis of the hybrid systems requires not only specific integration techniques, considering discontinuities in the right side of the differential equation, as well as modes stiffness [8,9], but also original event detection methods, which are responsible for correct mode switching [10]. The developed algorithm for asymptotic detection in ISMA is presented.

Applying Matlab, SimInTech, Ptolemy II and ISMA libraries with numerical schemes and algorithms for event detection in numerical analysis of the illustrative example model of the servo drive with the PWM controller is discussed. The general computational algorithm for the finite state machine approach to hybrid systems studying is presented.

 doi:10.20944/preprints202407.0261.v1

2. Event-Driven Systems and the Harel Diagrams

An event-driven system is a multi-mode dynamic system with continuous (*C*) or discrete (*D*) behavior, and modes of the system are determined by certain events (*E*).

Event-driven discrete or event-discrete systems (*ED*) are defined by the state vector $d \in W^r$, $W = R \cup \Im \cup B$, $\Im$ is a set of non-negative integer numbers; $B = \{false, true\}$ – a set of Boolean values; R – a set of real numbers. The discrete behavior of the event system is determined by the mapping $D: R \times W^r \to W^r$, $r \in \Im$.

Event-continuous systems (*EC*), defined in a time interval $[t_0, t_k]$, are specified by the state vector $y(t) \in R^N$ and determined by the mapping $C: R^m \to R^m$, m – the number of system modes.

The modes of the EC system are defined by the state vector $x(t) \in R^n$ with local behaviors c_j, $1 \le j \le m$, which range of permissible values is defined on a set of continuously differentiable event functions [8,10] $g(t,x): R \times R^n \to R^s$, $s \le n$, and these values exist in the half-interval $\left[t_j^0, t_j^*\right) \subseteq [t_0, t_k]$ in a local region g_j, $g_j \in G$, which is the range of permissible values for the local states vector $x(t) \in R^n$.

An event mode function $g_j(t,x)$ [10] works in such a way, that the corresponding predicate of the mode behavior pr_j: $g_j(t,x) < 0$, $pr_j \in B$, is true ($pr_j = true$) in the entire half-interval of the mode solution $\left[t_j^0, t_j^*\right) \subseteq [t_0, t_k]$. Therefore, the moment $t = t_j^*$, when the next event $E_j, 1 \le j \le m$, occurs and the mode behavior changes, is determined by the moment of time, when the predicate $pr_j(t,x) \in B$ changes its value from *false* to *true*.

Let's change phase variables $x(t) = y_j(t)$ and constrain the mapping C by the class of ordinary differential equations systems in the Cauchy form. Then local modes are presented as

$$y_j' = f_j(y_j, t), \quad y_j(t_0) = y_{j0}, \quad t \in \left[t_j^0, t_j^*\right), \tag{1}$$

where $y_j(t) \in R^n$; $f_j: R \times R^n \to R^n$ is a non-linear vector function, satisfying the Lipschitz conditions in a local mode; $y_{j0} \in R^n$ – the initial conditions vector for the local mode.

The sequence of continuous system modes generates a cause-and-effect chain of events

$$s_0 = (t_0, y_0) \to E_1 = \left(t_1^*, y_{10}\right) \to \ldots \to E_m = \left(t_m^*, y_{m0}\right),$$

that are defined on a solution of (1) with initial conditions $s_0 = (t_0, y_0)$. The set of discrete behaviors is determined by the non-empty set of event functions $g = \{g_j\}$, $1 \le j \le m$. In moments t_j^*, when the next event E_j occurs, the right side of (1) meets discontinuity points. The Lipschitz conditions are broke at discontinuity points. In this case, taking into account discontinuities of phase variables, obtaining the only solution requires the Caratheodory conditions to be satisfied [8]. As a result, the global behavior of the entire event-continuous system is determined by solving the Cauchy problem with constraints of the event function

$$y' = f(y,t), y(t_0) = y_0, g(y,t) < 0, \quad t \in [t_0, t_k], \tag{2}$$

 doi:10.20944/preprints202407.0261.v1

where $y(t) \in R^N$; $f: R \times R^N \to R^N$ is a piecewise continuous vector function, satisfying the Caratheodory conditions [8]; $g(y,t): R \times R^N \to R^S$, $S \le N$ – a continuously differentiable event vector function (event function).

Designing multi-mode event-continuous systems (2) is based on the advanced finite state specification of the model (2) in the form of the Harel diagrams or statecharts [5,11]. The event-continuous system behavior changing from c_i, $1 \le i \le m$, to a new one c_j, $1 \le j \le m$, $j \ne i$, taking into account equations (1) – (2), will be determined by a new predicate

$$p_k = \neg pr_i \wedge pr_j, \quad 1 \le k \le m^2, \tag{3}$$

On the analogy of graph theory [12], let a pair of behaviors (c_i, c_j) be *adjacent* behaviors of an event system, if the following condition is met: $p_k = true$, $1 \le k \le m^2$. If a non-empty set of behaviors C is presented as vertices or nodes, a non-empty set of predicates $P = \{p_k, k = 1,2,...\}$ is presented as a set of directed edges, connecting adjacent nodes (c_i, c_j), then a directed graph $G(s_0, C, P)$ is a *chart of discrete behavior*. Finally, considering the fact, that both continuous c_i and discrete d_i behaviors take places in corresponding states $s_i \in S$, $1 \le i \le m$, the directed graph $G(s_0, S, P)$ can be called a *statechart* or a *Harel diagram*.

In the modern traditional terminology, event-continuous systems are usually called hybrid systems (HS) [8,13,14]. The study of hybrid systems is based on numerical analysis of the equations (2) by the advanced tools with libraries for numerical integration and event detection algorithms to identify discrete moments of mode switching [8,10,15].

3. Problem Statement

The model of a basic servo drive with symmetrical pulse width modulation can be described by the following system of equations [16]

$$\begin{cases} \dfrac{d\omega}{dt} = k_\omega f(u - k\omega - \varphi, t), \\ \dfrac{d\varphi}{dt} = \omega, \\ \omega(0) = \omega_0, \\ \varphi(0) = \varphi_0, \end{cases} \tag{4}$$

where ω is the angular velocity with the initial value ω_0; k_ω is the speed reinforcing coefficient; u is the control action or setpoint; k is the damping ratio; φ is the angle of rotation with the initial value φ_0; $f: R^2 \times R \to R$ is the PWM function.

Let the PWM modulating signal be defined as

$$x = u - k\omega - \varphi, \tag{5}$$

then

$$f(x,t) = sign(x) \cdot z(x,t), \tag{6}$$

where

$$sign(x) = \begin{cases} -1, if\ x < 0, \\ 0, if\ x = 0, \\ 1, if\ x > 0. \end{cases} \tag{7}$$

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 July 2024 doi:10.20944/preprints202407.0261.v1

The function $z(x,t)$ from (6) is given as

$$z(x,t)=\begin{cases}1, if\ k_p|x|\geq saw(t),\\ 0, if\ k_p|x|<saw(t),\end{cases} \tag{8}$$

where $k_p=T_p^{-1}$ is the PWM coefficient, $saw(t)$ is the reference sawtooth signal.

It's clear from (6) – (8), that $f(x,t)$ can be equal to three values. Accordingly, the right side of the first equation in (4) changes depending on conditions (7), (8). The mathematical model (4) – (8) can be presented as an event-continuous or hybrid system, identified by the directed graph with three nodes (states) $s_i\in S, i=\overline{1,3}$, and directed edges (transitions from a state to another one), in this case, can be determined by the predicates $p_i\in P, i=\overline{1,3}$, in accordance to equations (7), (8). Such a directed graph $G(s_0,S,P)$ is called a Harel diagram or a statechart [5]. A hybrid system specification in the form of a statechart is a widely used advanced universal method to describe an event-continuous system [13]. The specification by the finite state machine with predicates, defining transitions, is implemented and developed in many advanced frameworks and platforms [7,14]. In Figure 1 the Harel diagram for a hybrid system, defined by (4) – (8) with the following predicates $p_i\in P, i=\overline{1,3}$ and $s_0=init$, is presented:

$$\begin{aligned}&p_1:(x\geq 0)\wedge(saw(t)-k_p|x|\leq 0);\\ &p_2:(x<0)\wedge(saw(t)-k_p|x|\leq 0);\\ &p_3:k_p|x|-saw(t)\leq 0.\end{aligned} \tag{9}$$

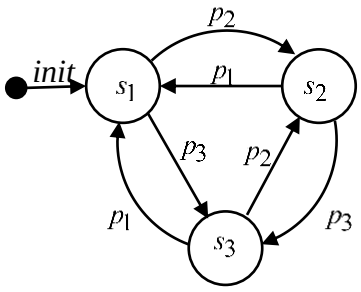


Figure 1. A statechart.

4. Model Specification in ISMA

The source of the reference sawtooth signal in ISMA software [6] is implemented as an integrator with a reset in the following form

$$saw(t)=\begin{cases}u(0)+\int_0^t u(\tau)d\tau, if\ saw(t)\leq saw^*,\\ 0, otherwise.\end{cases} \tag{10}$$

where saw^* is the value of the integrator reset to 0.

A software model for a servo drive with a PWM controller (4) – (8) by the LISMA language [8,17] in the form of structural and text specification (STS) is presented in Figure 2.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 July 2024 doi:10.20944/preprints202407.0261.v1

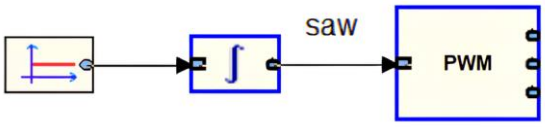


Figure 2. The structural part of software model (4)-(8) in LISMA.

The textual part of the specification (the "PWM" unit in Figure 2) in LISMA is illustrated in Figure 3 and, up to the operation signs (lines 1 - 4), replicates the mathematical notation of the Cauchy problem for the system of differential equations in the form (4). The software model is given the following parameters: $k_\omega = 100$; $k = 0.1$; $k_p = 0.1$. Zero initial conditions for the ordinary differential equations system (4) are set by default.

```
1  f ~= 1.0;
2  omega' = 100.0 * f;
3  phi' = omega;
4  x ~= 1.0 - 0.1 * omega - phi;
5
6  s1 [x >= 0 and (0.1 * abs(x) >= saw)] is f ~= 1.0; from init, s2, s3;
7  s2 [x < 0 and (0.1 * abs(x) >= saw)] is f ~= -1.0; from s1, s3;
8  s3 [(0.1 * abs(x)) < saw] is f ~= 0.0; from s1, s2;
```

Figure 3. The textual part of the software model (4)-(8) in LISMA.

Lines 1 – 4 define the initial state *init*, which determines the initial function $f : R^2 \times R \to R$ and the system of algebraic-differential equations (4). Lines 6 – 8 present the chart with three states $s_i \in S, i = \overline{1,3}$, with corresponding transition predicates (8), indicated in square brackets, and actions (*is*) or behaviors, that were obtained by the resolving the system of differential equations (4), which was delivered to each state $s_i \in S, i = \overline{1,3}$ from the initial state *init* with the determined value f .

5. Model Specification in Simulink

The model implementation by schematic diagrams in MATLAB/Simulink [3] is presented in Figures 4 and 5. The main structural scheme in Figure 4 is adjusted to equations (4) – (5) and the parameters values. In addition to built-in units from the platform libraries, the model includes a PWM macro-structure with the input $x(t)$, defined in (5), and the output modulating signal $out(t)$, defined by the structure, associated to expressions (6) – (9).

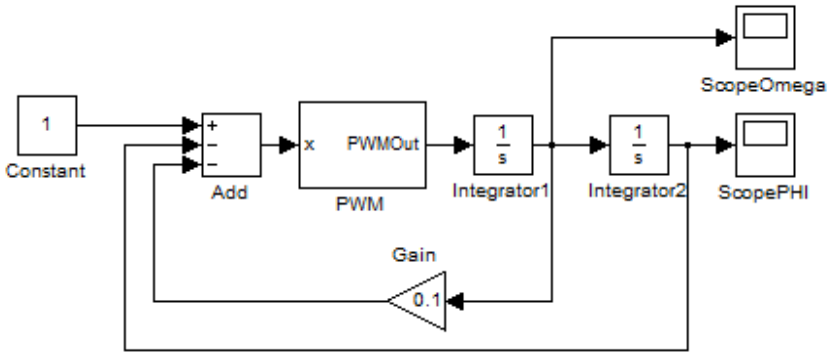


Figure 4. The schematic diagram in MATLAB/Simulink.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 July 2024 doi:10.20944/preprints202407.0261.v1

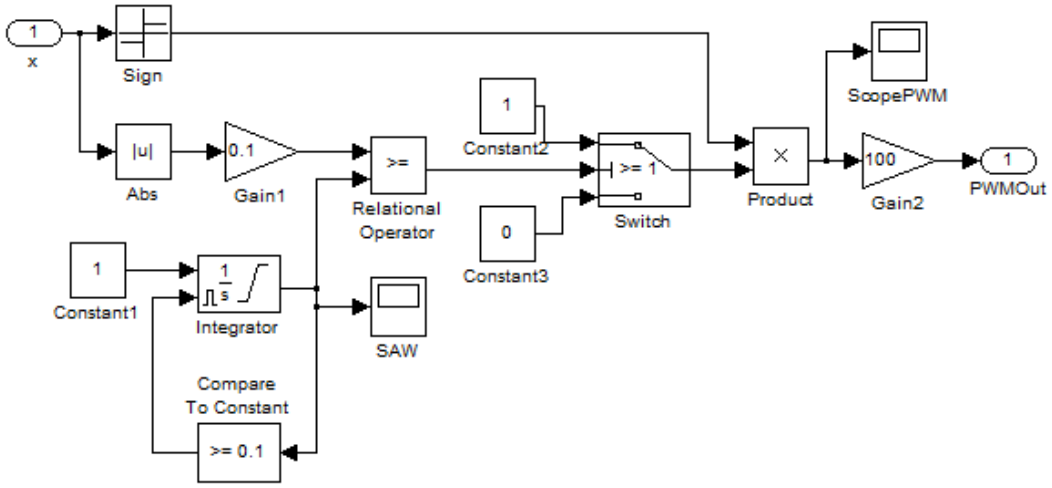
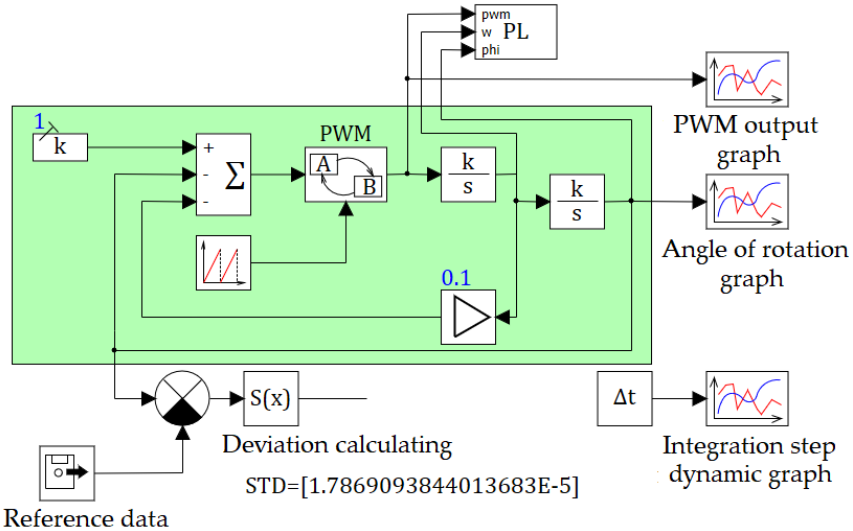


Figure 5. The PWM macro-unit structure diagram.

The presented approach to built schematic diagrams using the 'input-output' paradigm, unlike the advanced state machine specification, doesn't allow applying relevant description and, consequently, a new technique for hybrid systems analysis, based on the paradigm with states, predicates and event functions. As a result, designing the PWM submodel becomes more complicated, since the specification of logical and arithmetical expressions (6) – (9) traditionally can be represented much easier not by the diagram, but by the expression, a symbol string including operands and logical and/or arithmetical operations. Applying expressions to hybrid systems definition and specification is used not only in the LISMA language, as was shown earlier, but also is accepted in the advanced software platforms for hybrid computer modeling and simulation [7,14,18,19]. It should be noticed, that even if the advanced platform implies designing continuous processes as in (4), the expressions, in case there are some of them, are presented via text units with symbol form for logical and arithmetical formulas as in (6) – (9), rather than by structural submodels [7,20].

6. Model Specification in SimInTech

A software model for a servo drive with PWM, applying statecharts of the finite state machine, is presented in Figure 6. The PWM model was built as a compound representation of the finite state machine statecharts with instantaneous determining the following state and calculating the state function values in the form of conditionally executing submodels.



Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 July 2024 doi:10.20944/preprints202407.0261.v1

Figure 6. The block diagram of a PWM servo drive using statecharts.

The schematic diagram of the PWM submodel is presented in Figure 7. It includes the following structures: calculating transition predicates P1, P2, P3 (the left part in Figure 7); the statechart with 3 states designed as controlled submodels (the right part in Figure 7). The behaviors in states are determined by the numerical solution of the corresponding schematic diagram. The transition from the current state to another is defined by the specific transition unit (denoted at each state submodel on the right side).The transition unit generates the output of the state submodel, which can be connected to the input of another state submodel. The transition unit input gets a logical value, which is obtained by the calculating the transition predicate. The calculating can be conducted both inside and outside the state submodel. The data, obtained by the block diagram in each state, are sent to the unit for data selecting from the active state (DATA SELECTOR).

STATE1 is a submodel for $f = 0$. The schematic diagram of this state is presented in Figure 8. Submodels for other states look similar.

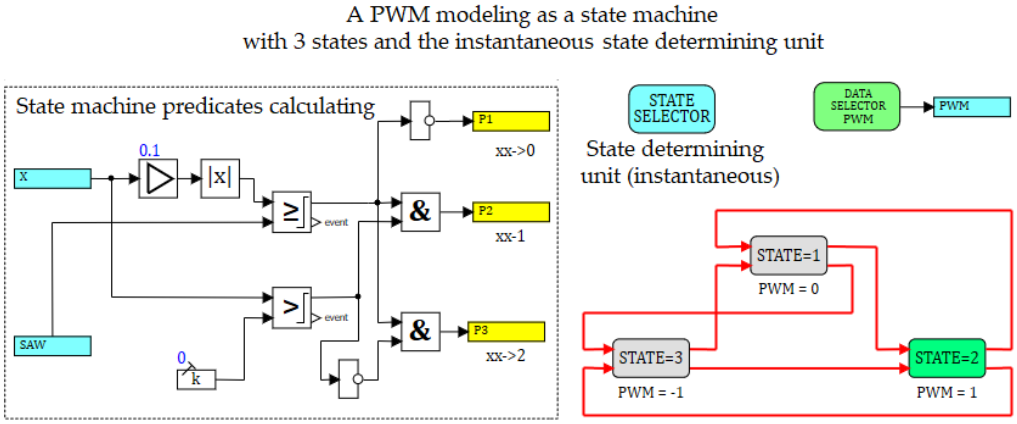


Figure 7. The structure of the PWM submodel using statecharts.

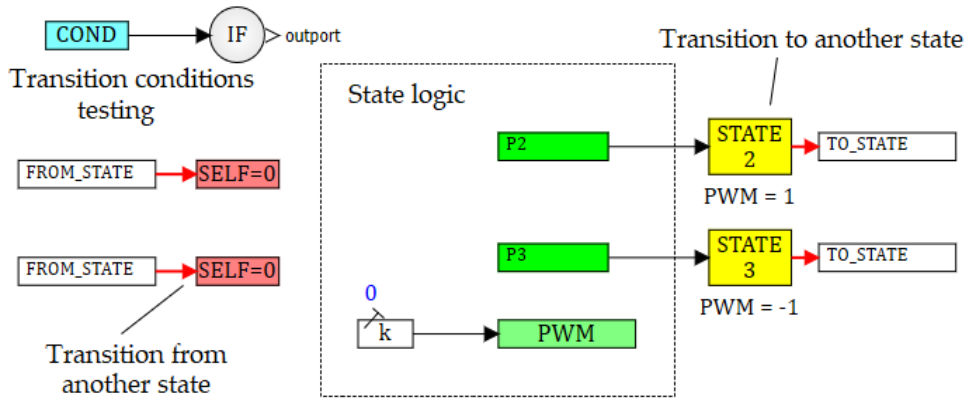


Figure 8. The block diagram of the STATE1 state.

This specification, combining two approaches, state machine and structural paradigms, is the most unsuccessful. It includes both all the disadvantages and contradictions of the structural approach applied in MATLAB/Simulink, that were discussed earlier, and confusion, associated with the finite state machine model. The SimInTech developers, aiming at applying advanced trends in hybrid systems definition, i.e. finite state machine paradigm, include the PWM finite state unit into the top level structure, corresponding to the system of equations (4). But this quite inefficient option is the only implementation of the declared finite state approach. The following process of the PWM submodel designing is carried out using structural techniques and the 'input-output' paradigm instead of the directed graph approach. The developers unsuccessfully try to construct the state of

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 July 2024 doi:10.20944/preprints202407.0261.v1

the directed graph using schematic diagram, but inputs and ouptuts of the structural diagram, when the output has functional dependency on the input, are not equivalent to the edges with predicates of the digraph. As a result, instead of a compact and transparent digraph, i.e. the Harel diagram (Figure 1), the model in SimInTech is presented as a lengthy tedious structure, including inexplicit embedded submodels (6 units), which hardly can be allied to the finite state paradigm. Therefore, the structural approach is not applied to hybrid systems definition. And that explains the fact, that for hybrid systems describing and analysis in Matlab, the Stateflow tool is used [21].

7. Model Specification in Ptolemy II

The specification for the servo drive in Ptolemy II [7] also combining two approaches. Top-level structural model, shown in Figure 9, include two integrators for first and second equations form (4). Sawtooth signal formed by *Generator* block and the PWM modulating signal (5) are sent to the input of *Statechart* block, which generates a control action.

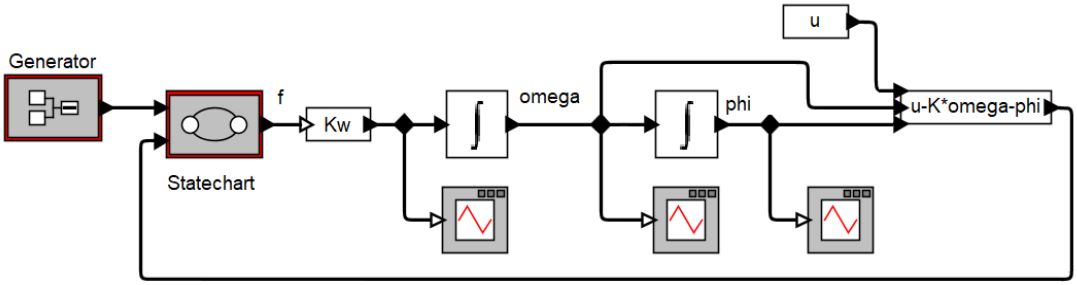


Figure 9. The continuous model (4)-(8) with the PWM controller in the form of a finite state machine.

The *Statechart* block is shown in Figure 10. It replicates the Harel diagram according to Figure 1.

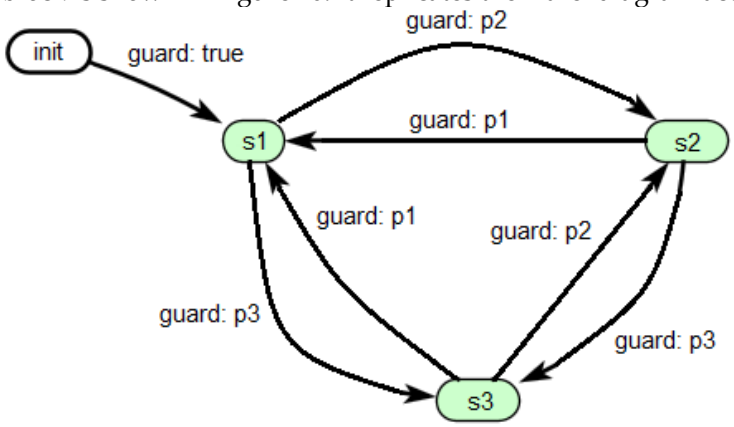


Figure 10. The Harel diagram in Ptolemy II.

The transition predicates (8) named *guard* and defined according to (9). The blocks for three states $s_i \in S, i = \overline{1,3}$, describe the rules for generating the output signal *f*, as shown in Figure 11.

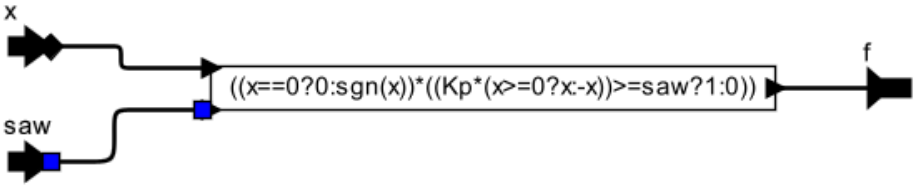


Figure 11. An example of a state description (for the state st3).

The output *f* is calculated according to formula (6). The first part of expression in Figure 11 defines the function *sign* in accordance with (7). Hereinafter the ternary operator (like as C programming language) is used. The second part of expression corresponds to function $z(x,t)$ from

 doi:10.20944/preprints202407.0261.v1

(8). The modulus of x is obtained using a ternary operation *(x>=0?x:-x)*. The result is multiplied by Kp and compared with the sawtooth signal.

8. Event Detection Algorithms in ISMA

Let's consider the algorithms for obtaining the event mode function $g(y,t)$ as in (2), implemented in the ISMA software tool. The LISMA grammar [17] is presented in the extended Backus–Naur form for the top-dawn parsers generator ANTLR4 [22]. The fragment of the grammar for conditional expressions in the software model is the following:

```
conditionalExpression: conditionalOrExpression;
conditionalOrExpression: conditionalAndExpression ( '||' conditionalAndExpression )*;
conditionalAndExpression: conditionalNotExpression: ( '&&' conditionalNotExpression: )*;
conditionalNotExpression: relationalExpression | '!' relationalExpression
relationalExpression: inequalityExpression | '(' conditionalExpression ')'
```
(11)

In accordance to the rules in (11), the software model includes conditions for the transition to a state $s_i \in S$. For example, the condition expression for the predicate p_1 from the system (9) is written down in square brackets in the line 6 (Figure 3). Taking into account the condition (3), let's find the predicate pr_i for the state s_i in the following form:

$$pr_i = \bigwedge_{j \in J} \left(\neg p_j \right), \tag{12}$$

where J is the adjacent states for the state s_i.

Based on the syntactic tree, which was built while analyzing the software model, and its semantic attributes, the syntactic trees for the predicates pr_i in every state of the system are constructed in the ISMA. An n-ary tree is built with the root corresponding to the conjunction operation. The root has n descendant nodes, where n is the number of adjacent states for the current mode of the hybrid system. All outgoing branches contain a vertex, corresponding to the logical negation (inversion). This vertex is the root of the syntactic tree, built for the predicate pr_i. Next, the syntactic tree is modified in a way that inversion operations are applied only to inequalities (11). De Morgan's rules for transforming logical expressions are applied here. The syntactic tree is top-down traversed, and for every node the transformations are applied according to the following recursive rules (Algorithm 1):

Algorithm 1. The mode behavior predicate transformation.

Step 1. If the tree node matches the inequality (inequalityExpression), the traverse is stopped. If the inversion operation indicator was set, the inequality sign is replaced by the opposite one.

Step 2. If the tree node matches the logical negation operation ('!'), the inversion operation indicator is set to the descendant vertex. Resetting the indicator reverses the inversion, if negations are consecutive. The current node is excluded from the tree, the connection between a parent node and descendant one is built.

Step 3. If the tree node matches the conjunction operation ('&&') and the inversion operation indicator is set, the conjunction operation is replaced by the disjunction ('||'). A new vertex with logical negation ('!') is added to all outgoing branches. The inversion indicator is reset.

Step 4. If the tree node matches the disjunction operation ('||') and the inversion indicator is set, the disjunction is replaced by the conjunction ('&&'). A new vertex with logical negation ('!') is added to all outgoing branches. The inversion indicator is reset.

Computation expressions for event functions $g_i(y,t)$ are formed based on the trees. More specifically, these structures define the rules for determining an integration step based on the event functions dynamics.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 July 2024 doi:10.20944/preprints202407.0261.v1

Let's build and transform a syntactic tree for the mode behavior predicate pr_1, corresponding to the state s_1 (Figure 1) of the servo drive with the PWM controller model. Applying equations (12), we get $pr_1 = \neg p_2 \wedge \neg p_3$. Let inequalities be denoted as:

$$ne_{21} : x < 0, \quad ne_{22} : saw(t) - k_p |x| \le 0, \quad ne_{31} : k_p |x| - saw(t) \le 0,$$

then $pr_1 = \neg(ne_{21} \wedge ne_{22}) \wedge \neg ne_{31}$. The basic syntactic tree is shown in Figure 12, a. After the transforming process, according to the Algorithm 1, the tree presented in Figure 12, b is formed, where

$$\overline{ne_{21}} : x \ge 0, \quad \overline{ne_{22}} : saw(t) - k_p |x| > 0, \quad \overline{ne_{31}} : k_p |x| - saw(t) > 0.$$

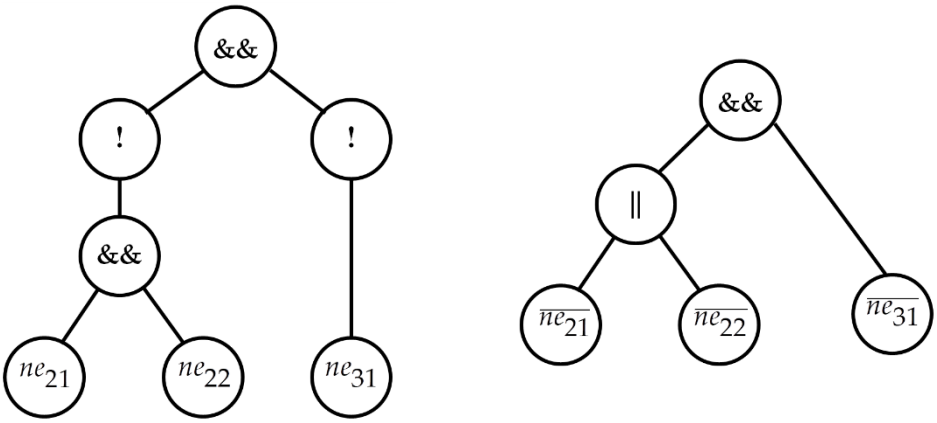


Figure 12. Syntactic trees for the mode behavior predicate.

The following algorithms are connected with the integration step determining, which is dependent on event functions dynamics. Let the mode behavior predicate for the state s_i be like pr_i: $g_i(y,t) < 0$, i.e. the syntactic tree consists of the only inequality ne_i. And the n^{th} step of numerical resolving the Cauchy problem (2) is performed, the value for the right side of the differential equation $f_n = f(y_n, t_n)$ is calculated. Besides, the value of the next integration step h_{n+1}^m is determined by the numerical integration algorithm. Then, according to the theorem of calculating the integration step for asymptotic approaching to the mode limit [8], the integration step can be limited by the value, determined by the following algorithm.

Algorithm 2. Determining an integration step for one-sided events detection.

Step 1. Calculate $g_n = g(y_n, t_n)$.

Step 2. Calculate $\frac{\partial g_n}{\partial y} = \frac{\partial g(y_n, t_n)}{\partial y}$.

Step 3. Calculate $\frac{\partial g_n}{\partial t} = \frac{\partial g(y_n, t_n)}{\partial t}$.

Step 4. Calculate $g_n' = \frac{\partial g_n}{\partial y} \cdot f_n + \frac{\partial g_n}{\partial t}$.

Step 5. If $g_n' < 0$, let $h_{n+1} = h_{n+1}^m$ and go to the next integration step.

Step 6. Calculate the 'event step size' h_{n+1}^{ev} as $h_{n+1}^{ev} = (\gamma - 1)\frac{g_n}{g_n'}$.

 doi:10.20944/preprints202407.0261.v1

Step 7. Compute the step size $h_{n+1} = \min\left(h_{n+1}^{m}, h_{n+1}^{ev}\right)$ and go to the next integration step.

In Step 5 the event function derivative is determined. As the event function approaches the event surface, g_n' is positive, and, as it moves away from the boundary $g_i(y,t)=0$, g_n' becomes negative. Then, once the event function course has been determined, no additional constraints on the integration step value have to be imposed, if the event function moves away from the mode surface.

Let $h^{ev}(g)$ be the value of the integration step determined by the discussed algorithm. In general, as it was shown earlier, the condition pr_i for the system staying to the s_i might be a complex logical function depending on the set of inequalities ne_{jk}, where $j \in J$ and $k = 1,2,\ldots$. If each of the inequalities has the form $ne_{jk} : g_{jk}(y,t) < 0$, the Algorithm 2 allows to obtain a set of values for the integration step $h_{jk}^{ev}\left(g_{jk}\right)$.

Let the syntactic tree be built using Algorithm 1. Then the value of the integration step can be determined at each stage of the calculating the mode behavior by the following recursive Algorithm 3, while traversing the tree nodes.

Algorithm 3. Determining the integration step in case of a set of event functions.

Step 1. If the tree node matches an inequality (inequalityExpression), the tree traversing is stopped. The integration step value, obtained by the Algorithm 2, is returned.

Step 2. If the tree node matches the conjunction operation ('&&'), the algorithm for the step determining is called for descendant nodes. The minimal value from all those calculated is returned.

Step 3. If the tree node matches the disjunction operation ('||'), the algorithm for the step determining is called for descendant nodes. The maximum value from all those calculated is returned.

The Algorithm 2 is justified by the requirement of the event function for the mode behavior having negative value. Using the Algorithm 3, it may turn out that event functions, evaluated at the step 1, are non-negative. It means, that at the current moment the corresponding predicate does not cause changing the system state, therefore there is no need to call the Algorithm 2. In this case, the Algorithm 3 does not limit the step value.

Let's illustrate the Algorithm processing while calculating the 'event' step in the state 1 of the servo drive with the PWM controller model. There are the following conditions: $\overline{ne_{21}} : x \geq 0$, $\overline{ne_{22}} : saw(t) - k_p|x| > 0$, $\overline{ne_{31}} : k_p|x| - saw(t) > 0$. The event functions, corresponding to the conditions, are $g_{21} = -x$, $g_{22} = saw(t) - k_p|x|$, $g_{31} = k_p|x| - saw(t)$. Then the step value is determined by the formula $h_1^{ev} = \min\left(\max\left(h_{21}^{ev}(g_{21}), h_{22}^{ev}(g_{22})\right), h_{31}^{ev}(g_{31})\right)$.

9. Calculation and Simulation

The second stage in computer modeling is implementing software models, designed at the first stage of specification. Applying various integration techniques and developed event detection algorithms, included into the libraries with numerical methods in the considered simulation platforms, for numerical computing (4) – (8) resulted in qualitatively identical solutions for phase variables $\omega(t)$ and $\varphi(t)$, as presented in Figure 13.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 July 2024 doi:10.20944/preprints202407.0261.v1

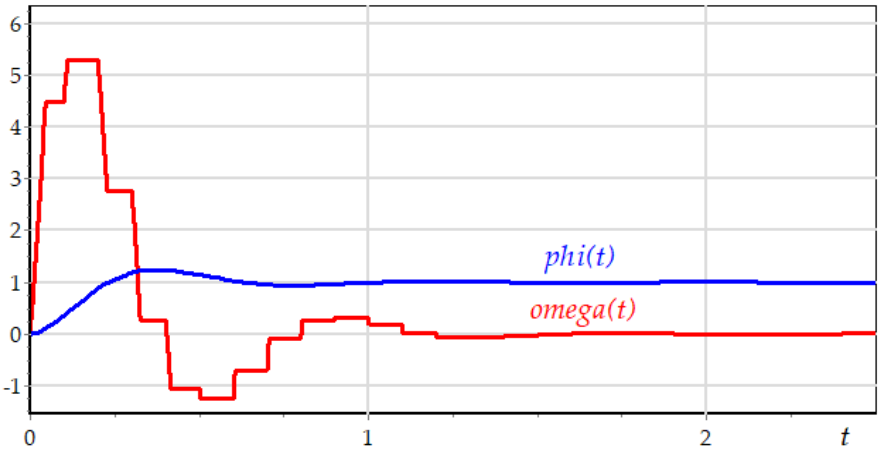


Figure 13. Transient processes of the computational experiment.

The STEKS (5,4) technique [23] combined with the algorithm for asymptotic event detection, the parameter $\gamma = 0.6$ [8], was used for calculations in ISMA. The adaptive algorithm ARK [24] and the invented technique for intersections detecting [15] with the step value refining for the signal sources and discrete units were applied in SimInTech. The integration method ode45 (Dormand-Prince) [3,9] with the algorithm for switching detection for all of the model units [25] was applied in MATLAB. The integration technique Explicit RK23 Solver [13] with double-sided event detecting was used in Ptolemy II.

Let's evaluate the accuracy of the problem solutions in ISMA, SimInTech, MATLAB and Ptolemy II, taking into account using original libraries with numerical techniques and detection algorithms. Before the experiments, the exact moments of events generating and corresponding values of phase variables $\varphi^* = \varphi\left(t_i^*\right)$ and $\omega^* = \omega\left(t_i^*\right)$ for the initial system of equations (4) were obtained by the analytical fitting the initial conditions. The summary data on the calculating error for the first and seventh events are presented in Table 1, where t_i^* is the exact time of the event i, $\tilde{t}_i$ is the obtained time, ω_i^* – the exact value of the angular velocity, $\tilde{\omega}_i$ – the calculated value of the angular velocity, φ_i^* – the exact value of the angle of rotation, $\tilde{\varphi}_i$ – the calculated value of the angle of rotation. Computing was carried out with the given accuracy $\varepsilon = 10^{-4}$, that is commensurable with the error of the integration techniques applied.

Table 1. Summary data on calculating error.

Platform	Time		Angular velocity		Angle of rotation	
	$\tilde{t}_1 - t_1^*$	$\tilde{t}_7 - t_7^*$	$\tilde{\omega}_1 - \omega_1^*$	$\tilde{\omega}_7 - \omega_7^*$	$\tilde{\varphi}_1 - \varphi_1^*$	$\tilde{\varphi}_7 - \varphi_7^*$
ISMA	1.5*10-6	1.3*10-5	1.5*10-4	8.4*10-5	6.5*10-6	7.1*10-5
SimInTech	9.5*10-6	7.4*10-6	2.0*10-4	2.7*10-5	4.3*10-5	1.7*10-5
MATLAB	-2.2*10-11	-1.6*10-10	-2.2*10-9	2.0*10-10	-4.3*10-10	-2.3*10-9
Ptolemy II	4.0*10-5	2.1*10-4	2.5*10-4	3.5*10-4	1.8*10-4	1.4*10-4

The model in MATLAB demonstrates the most accurate results. But it does not comply with the condition of the event one-sidedness [10], as it is presented in the table. The values of phase variables in detection points are calculated with the comparable accuracy using tools in SimInTech and ISMA. The largest calculating error is obtained in Ptolemy II by the double-sided detection method [13]. The detection technique in ISMA differs from the others mentioned, since it meets the requirement of one-

 doi:10.20944/preprints202407.0261.v1

sidedness [8,10], and the integration step is calculated in accordance to the condition of asymptotic approaching to the mode limit. The algorithm is based on linear extrapolation of the event function and limiting the event function dynamics at the next step, the function can change its value not more than in γ times [8]. As a result, the discreteness of the system behavior representation in the vicinity of the mode limits is increasing. The accuracy of this representation is defined by the minimal value of the integration step, which in this case is equal to $h_{\min} = 10^{-6}$.

Let's analyze the dynamics of event functions. Figure 14 demonstrates the graphs of event functions g_{jk}, that determine mode shifting in the interval $t \in [0, 0.1)$. At first, the system is in the state 1 (s_1), then, in $t \approx 0.0449$, the transition to the mode (state) s_3 is made, after that, in $t \approx 0.1$, the system returns to s_1. The further transitions and state changing are not considered, although two more shifting can be seen in Figure 14.

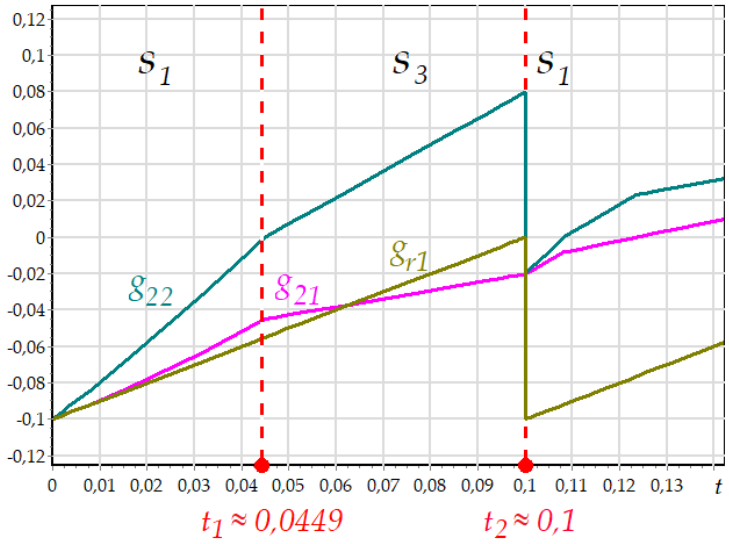


Figure 14. Event functions dynamics.

There are three event functions for the first state: $g_{21} = -x$, $g_{22} = saw(t) - k_p|x|$, $g_{31} = k_p|x| - saw(t)$. The system is in the state s_1 in the interval $t \in [0, 0.0449)$, but the event function g_{31} does not affect the mode shifting, since $g_{31} > 0$. This graph is not presented in the figure above. As a result, the integration step calculating is dependent on the dynamics of the functions g_{21} and g_{22}. The Figure 15 shows the enlarged fragment of the graph in the vicinity of the moment $t \approx 0.0449$.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 3 July 2024 doi:10.20944/preprints202407.0261.v1

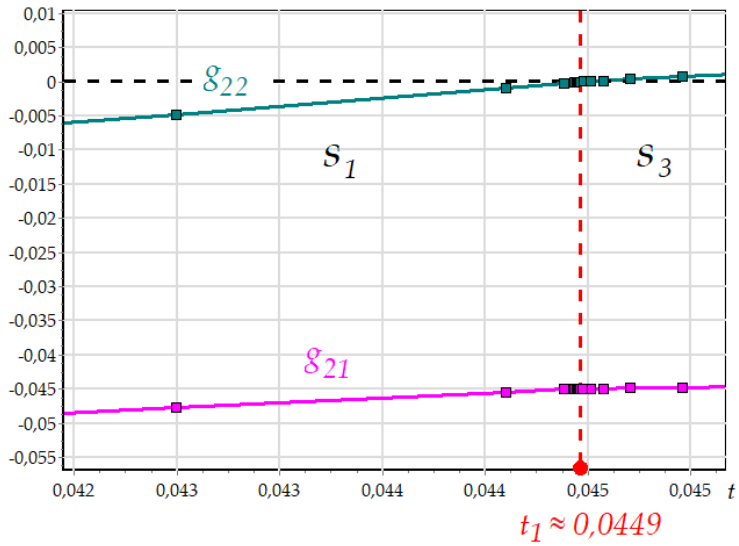


Figure 15. Transition from s_1 to s_3, when the condition $g_{22} < 0$ is not met.

Since the event function g_{22} changes faster than g_{21}, the algorithm for the integration step determining uses the value $h_{22}^{ev}(g_{22})$ for asymptotic approaching to the mode limit. The step decreasing, while approaching to the mode limit, is shown in Figure 15. As soon as $g_{22} = 0$, the condition p_3 from (9) is met, the system switch to the mode s_3. After the transition, the integration step is increased, but the detection algorithm does not limit its value: the step can be limited only by the numerical integration method.

The third state has the resulting condition for the mode $g_{r1} < 0$, where the function $g_{r1} = saw(t) - 0.1$ is obtained from the condition of the integrator reset (Figure 2). The graphs of this event function are presented in Figures 14 and 16. The enlarged fragment of the graph in the vicinity of the moment $t \approx 0.1$ also demonstrates the step decreasing while approaching to the mode limit. After the modes switching, the event function changes its value instantly, since the variable *saw* is reset to 0. The condition p_1 from (9) is met, and the system switches to the mode s_1.

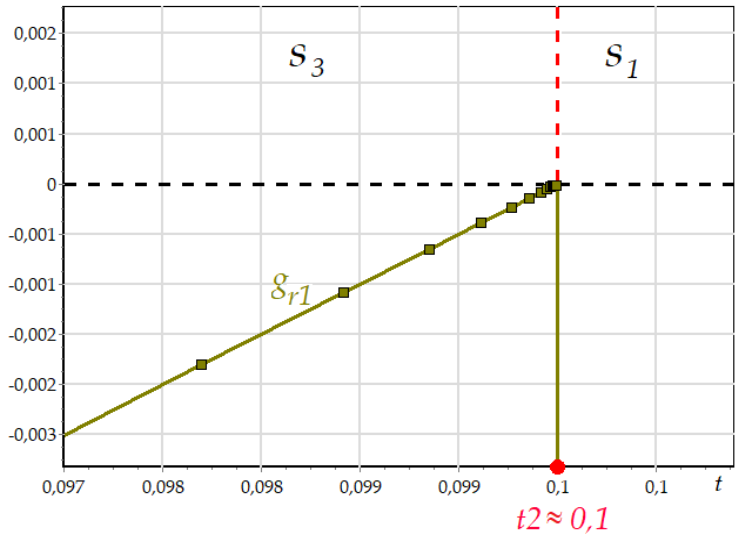


Figure 16. Transition from s_3 to s_1, when the condition $g_{r1} < 0$ is not met.

 doi:10.20944/preprints202407.0261.v1

10. Conclusions

The analysis of software platforms for simulating event systems is performed. The model of a simple servo drive with the PWM controller was tested. The system is considered as a hybrid dynamic system with event-driven continuous processes, which implies applying corresponding techniques for the model definition and numerical analysis. The specification of such systems in advanced international practice is based on the finite state machine paradigm with the handy domain-driven graphical interpretation in the form of the Harel diagrams or statecharts. Two of the considered simulation platforms, ISMA and Ptolemy II, comply with the finite state methodology, having appropriate, as close to natural mathematical as possible, description for the event-continuous processes. The SimInTech tries to use the finite state approach with structural diagrams, which is unacceptable by definition. The specification results in a complex, hard to understand software model, including hierarchical submodels, which significantly complicate, rather than simplify the event processes definition, causing the contradiction with the accepted finite state paradigm. MATLAB/Simulink, unlike SimInTech, applies only the structural approach without mentioning the finite state machine. But the structural paradigm, as the engineering practice demonstrates, is intended for continuous processes specification and is very limited in case of applying to define multi-mode hybrid systems. Numerical experiments with software models show qualitatively identical results. Quantitative differences are connected with applying certain integration techniques and detection algorithms. The presented results demonstrate the smallest error of the model in MATLAB and the relatively large one of the model in Ptolemy II. ISMA and SimInTech have the intermediate results in the comparative analysis of the calculation errors. It should be noticed, that the ISMA model is the only one, that complies with the requirement of the event one-sidedness due to the developed algorithm for asymptotic detection, applied to determining the optimal integration step value, considering the event functions dynamics (Algorithm 1 – Algorithm 3). This property is the necessary condition for the model results adequacy in simulating physical processes with the requirement of non-negativity of phase variables, for example processes of chemical kinetics.

References

1. Banerjee, S.; Verghese, G.C. (Eds.) Nonlinear Phenomena in Power Electronics; Wiley-IEEE Press: New York, USA, 2001; 441 p.; DOI:10.1109/9780470545393.
2. Cendoya, M.; Toccaceli, G. Application of Statecharts in Buck-Boost DC-DC Converter Simulation. WSEAS Transactions on Electronics, 2021, vol. 12, pp 81–88; DOI: 10.37394/232017.2021.12.11.
3. Matlab product page on the official website of MathWorks. Available online: https://www.mathworks.com/products/matlab.html (accessed on 15 April 2024).
4. SimInTech product page on the official website of 3V Service. Available online: https://en.simintech.ru/ (accessed on 15 April 2024).
5. Harel, D. Statecharts: a Visual Formalism for Complex Systems. Science of Computer Programming, 1987, vol. 8(3), pp. 231-274; DOI: 10.1016/0167-6423(87)90035-9.
6. Shornikov, Yu.V.; Bessonov, A.V. The Core Components of ”ISMA 2015” Software [Komponenty Jadra Programmnogo Kompleksa “ISMA 2015”] (in Russian). Certificate of State Registration of Computer Programs # 2015617235 [Svidetel'stvo o Gosudarstvennoj Registracii Programmy dlja JeVM # 2015617235]. Federal Service For Intellectual Property (Rospatent), Moscow, 2015, 1p.
7. Ptolemy II product page on the official website of The University of California, Berkeley. Available online: https://ptolemy.berkeley.edu/ptolemyII/index.htm (accessed on 15 April 2024).
8. Novikov, E.A.; Shornikov, Yu.V. Modeling of Stiff Hybrid Systems [Modelirovaniye Zhestkikh Gibridnykh Sistem] (in Russian); Lan: St. Petersburg, Russia [Lan': Sankt-Peterburg, Rossija], 2019; 420 p.; ISBN 978-5-8114-3523-4.
9. Hairer, E.; Nørsett, S.P.; Wanner, G. Solving Ordinary Differential Equations I: Nonstiff Problems; Springer-Verlag: Berlin, Germany, 1993; Corr. 3rd printing, 2009; 528 p.; ISBN 978-3642051630.
10. Esposito, J.; Kumar, V.; Pappas G. Accurate Event Detection for Simulating Hybrid Systems. Hybrid Systems: Computation and Control (HSCC), 2001, Volume LNCS 2034, pp. 204 – 217; DOI: 10.1007/3-540-45351-2_19.
11. Karpov, Yu.G. State Machine Theory [Teorija Avtomatov] (in Russian); Piter: Moscow, Russia [Piter: Moskva, Rossija], 2002; 206 p.; ISBN 5-318-00537-3.

12. Kasyanov, V.N.; Evstigneev, V.A. Graphs in Programming: Processing, Visualization and Application [Grafy v Programmirovanii: Obrabotka, Vizualizacija i Primenenie] (in Russian); BHV-Peterburg: St. Petersburg, Russia [BHV-Peterburg: Sankt-Peterburg, Rossija], 2003; 1104 p.; ISBN 5-94157-184-4.
13. Lee, E.A.; Zheng, H. Operational Semantics of Hybrid Systems. Hybrid Systems: Computation and Control: 8th International Workshop, HSCC, LNCS 3414, Zurich, Switzerland, March 9-11, 2005.
14. Moraleda, A.U.; Villalba, C.M. Modeling and Simulation in Engineering Using Modelica; UNED Editorial: Madrid, Spain, 2018; 298 p.; ISBN: 978-84-362-7365-6.
15. Timofeev, K.A. Module for Intersection Event Detection Block for the SimInTech Software Package (SimInTech Intersection Detection Block) [Modul' Bloka Detekcii Sobytij Peresechenija Programmnogo Kompleksa SimInTech (Blok Detekcii Peresechenij SimInTech)] (in Russian). Certificate of State Registration of Computer Programs # 2024619776 [Svidetel'stvo o Gosudarstvennoj Registracii Programmy dlja JeVM # 2024619776]. Federal Service For Intellectual Property, Moscow, 2024, 1p.
16. Blaze, E.S.; Brodovsky, V.N.; Vvedensky, V.A. et al Servo Drives: In 3 volumes [Sledyashhie Privody] (in Russian); Ed. B.K. Chemodanov; Publishing house of BMSTU: Moscow, Russia [Izdatel'stvo MGTU im. N. Je. Baumana: Moskva, Rossija], 2003; ISBN 5-7038-1384-0.
17. Shornikov, Yu.V.; Bessonov, A.V. The Component for Specification of Hybrid Systems in LISMA PDE Language [Komponenta Specifikacii Modelej Gibridnyh Sistem na Yazyke “LISMA_PDE"] (in Russian). Certificate of State Registration of Computer Programs # 2015617191 [Svidetel'stvo o Gosudarstvennoj Registracii Programmy dlja JeVM # 2015617191]. Federal Service For Intellectual Property (Rospatent), Moscow, 2015, 1p.
18. OpenModelica product page on the official website of Open Source Modelica Consortium (OSMC). Available online: https://openmodelica.org/ (accessed on 15 April 2024).
19. JuliaSim product page on the official website of JuliaHub. Available online: https://info.juliahub.com/products/juliasim (accessed on 15 April 2024).
20. HyVisual product page on the official website of The University of California, Berkeley. Available online: https://ptolemy.berkeley.edu/hyvisual/ (accessed on 15 April 2024).
21. Stateflow product page on the official website of MathWorks. Available online: https://www.mathworks.com/products/stateflow.html (accessed on 15 April 2024).
22. Antlr product page. Available online: https://www.antlr.org/ (accessed on 15 April 2024).
23. Novikov, E.A. Explicit Methods for Stiff Systems [Javnye Metody dlja Zhestkih Sistem] (in Russian); Nauka: Novosibirsk, Russia [Nauka: Novosibirsk, Rossija], 1997; 192 p.; ISBN 5-02-031245-2.
24. Skvortsov, L.M. Construction and Analysis of Explicit Adaptive Onestep Methods for Numerical Solution of Stiff Problems [Postroyeniye i Analiz Yavnykh Adaptivnykh Odnoshagovykh Metodov Chislennogo Resheniya Zhestkikh Zadach] (in Russian), Journal of Computational Mathematics and Mathematical Physics [Zhurnal Vychislitel'noy Matematiki I Matematicheskoy Fiziki], 2020, Vol. 60, pt. 7, p. 1111-1125; DOI: 10.31857/S0044466920070108.
25. Shornikov, Yu.V.; Novikov, E.A.; Dostovalov, D.N.; Myssak, M.S. Visual Modeling of Dynamical Systems by Instrumental Facilities. 7 IFAC Conference on Manufacturing Modelling, Management, and Control (MIM'2013), St. Petersburg, IFAC, 2013; Manufacturing modelling, management, and control, vol. 7, pt. 1, p. 2185-2190; DOI: 10.3182/20130619-3-RU-3018.00252.