QUANTIZATION–BASED SIMULATION OF HYBRID SYSTEMS

Ernesto KOFMAN†

†*Laboratorio de Sistemas Dinámicos. FCEIA – UNR – CONICET.*
Riobamba 245 bis – (2000) Rosario – Argentina
Email: kofman@fceia.unr.edu.ar

***Abstract*— This paper extends the use of quantization–based integration methods to the simulation of hybrid systems. Using the fact that these methods approximate ordinary differential equations (ODEs) and differential algebraic equations (DAEs) by discrete event systems, it is shown how hybrid systems can be approximated by pure discrete event simulation models. In this way, the treatment and detection of events representing discontinuities – which constitute an important problem for classic ODE solvers– is efficiently solved. The resulting advantages are illustrated and discussed through the simulation of two examples.**

***Keywords*— Hybrid systems, ODE integration, Discrete Event Systems.**

I. INTRODUCTION

Continuous system simulation is a topic which has advanced significantly with the appearance of modern computers. Based on classic methods for numerical resolution of ODE's like Euler, Runge–Kutta, Adams, etc., several variable–step and implicit ODE solver methods were developed.

Although there are several differences between the mentioned ODE solver algorithms, all of them share a property: they are based on time discretization. That is, they give a solution obtained from a difference equation system (i.e. a discrete–time model) which is only defined in some discrete instants.

The complexity of most technical systems yields models which often combine a continuous part (described by ODEs or DAEs) and discrete components. The interaction between these subsystems can produce sudden changes (discontinuities) in the continuous part which must be handled by the integration algorithms.

The mentioned sudden changes are called *events* and two different cases can be distinguished according to the nature of their occurrence. The events which occur at a given time, independently of what happens in the continuous part are called *Time Events*. On the other hand, events which are produced when the continuous subsystem state reaches some condition are called *State Events*.

The integration along discontinuities can cause severe inefficiency because the non–smoothness violates the theoretical assumptions on which solvers are founded (Barton, 2000). Thus, time and state events must be detected in order to perform steps at their occurrence.

The event detection techniques have been being studied since Cellier's Thesis (Cellier, 1979) and many works can be found in the recent literature (Park and Barton, 1996; Taylor and Kebede, 1996; Schlegl *et al.*, 1997; Esposito *et al.*, 2001).

Although these ideas work quite efficiently, the techniques do not say how to represent discrete parts and how to schedule the time events in general cases. Moreover, the state event detection requires performing some iterations to find the time of the event occurrence. Due to these facts , the simulation of hybrid systems constitutes one of the most difficult topics in the numerical integration area.

A completely different approach for ODE numerical simulation has been being developed since the end of the 90's where time discretization is replaced by state variables quantization. As a result, the simulation models are not discrete time but DEVS (Zeigler *et al.*, 2000).

The origin of this idea can be found in the definition of Quantized Systems and their representation in terms of DEVS models (Zeigler and Lee, 1998). Quantized Systems weres reformulated with the addition of hysteresis and formalized as a simulation method for ODE's in (Kofman and Junco, 2001) where the Quantized State Systems (QSS) were defined.

This new method was improved with the definition of the Second Order Quantized State Systems (QSS2) (Kofman, 2002a) and then extended to the simulation of DAEs (Kofman, 2002b).

Despite their simplicity, the QSS and QSS2 methods satisfy some strong stability, convergence and error bound properties. They can also reduce the number of calculations, their parallel implementation is straightforward and they can exploit sparsity in a very efficient fashion.

This work attempts to show that, due to the discrete event nature of QSS and QSS2, the main difficulties related to discontinuity handling in hybrid systems are solved with these methods.

II. QUANTIZATION–BASED INTEGRATION

A. QSS–Method

Consider a time invariant ODE in its State Equation System (SES) representation:

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

where $x(t) \in \mathbb{R}^n$ is the state vector and $u(t) \in \mathbb{R}^m$ is an input vector, which is a known piecewise constant function.

The QSS–method (Kofman and Junco, 2001) simulates an approximate system, which is called Quantized State System:

$$\dot{x}(t) = f(q(t), u(t)) \tag{2}$$

where $q(t)$ is a vector of *quantized variables* which are quantized versions of the state variables $x(t)$. Each component of $q(t)$ is related with the corresponding component of $x(t)$ by a hysteretic quantization function, which is defined as follows:

Definition 1. *Let $Q = \{Q_0, Q_1, ..., Q_r\}$ be a set of real numbers where $Q_{k-1} < Q_k$ with $1 \leq k \leq r$. Let Ω be the set of piecewise continuous real valued trajectories and let $x_i \in \Omega$ be a continuous trajectory. Let $b : \Omega \to \Omega$ be a mapping and let $q_i = b(x_i)$ where the trajectory q_i satisfies:*

$$q_i(t) = \begin{cases} Q_m & if \quad t = t_0 \\ Q_{k+1} & if \quad x_i(t) = Q_{k+1} \wedge \\ & \quad \wedge q_i(t^-) = Q_k \wedge k < r \\ Q_{k-1} & if \quad x_i(t) = Q_k - \varepsilon \wedge \\ & \quad \wedge q_i(t^-) = Q_k \wedge k > 0 \\ q_i(t^-) & otherwise \end{cases} \tag{3}$$

and

$$m = \begin{cases} 0 & if\ x_i(t_0) < Q_0 \\ r & if\ x_i(t_0) \geq Q_r \\ j & if\ Q_j \leq x(t_0) < Q_{j+1} \end{cases}$$

Then, the map b is a hysteretic quantization function.

The discrete values Q_k are called *quantization levels* and the distance $Q_{k+1} - Q_k$ is defined as the *quantum*, which is usually constant. The width of the hysteresis window is ε and, as it was shown in (Kofman *et al.*, 2001), it is better to take it equal to the quantum.

In (Kofman and Junco, 2001) it was proven that the quantized variable trajectories $q_i(t)$ and the state derivatives $\dot{x}_i(t)$ are piecewise constant and the state variables $x_i(t)$ are piecewise linear. As a consequence, those trajectories can be represented by sequences of events and then the QSS can be simulated by a DEVS model. There, it is also shown that the use of hysteresis in QSS is necessary to ensure those properties.

The mapping of a QSS like (2) into a DEVS model can be done in several ways and one of the easiest is based on coupling principles. A generic QSS can be represented by the block diagram of Fig.1. That block diagram is composed by static functions f_i, integrators and quantizers.

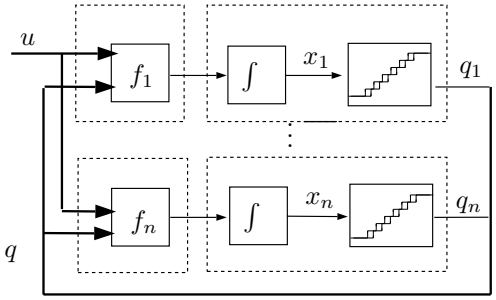

Figure 1: Block Diagram Representation of a QSS

Each pair formed by an integrator and a quantizer is called *quantized integrator* and it is equivalent to a simple DEVS model. Similarly, the static functions have DEVS equivalents and consequently, the entire block diagram has an equivalent coupled DEVS which represents it. The mentioned DEVS models can be found in (Kofman and Junco, 2001).

Some simulation programs –PowerDEVS (Pagliero and Lapadula, 2002) for instance– have libraries with DEVS models representing quantized integrators and static functions. Thus, the implementation of the QSS–method consists in building the block diagram in the same way that it could be done in Simulink.

B. QSS2–Method

QSS only performs a first order approximation. Due to accuracy reasons, a second order method was proposed in (Kofman, 2002a) which also shares the main properties and advantages of QSS.

The basic idea of the new method, (called QSS2) is the use of first–order quantization functions instead of the quantization function given by (3). Then, the simulation model can be still represented by (2) but now $q(t)$ and $x(t)$ have a different relationship. This new system is called Second Order Quantized State System or QSS2 for short.

A first–order quantization function can be seen as a function which gives a piecewise linear output trajectory, whose value and slope change when the difference between this output and the input becomes bigger than certain threshold (Fig. 2)

In that way, the quantized variable trajectories are piecewise linear and the state trajectories are piecewise parabolic[1].

As before, the system can be divided into quantized integrators and static functions like in Fig.1. However,

[1] In nonlinear systems this is only approximated.

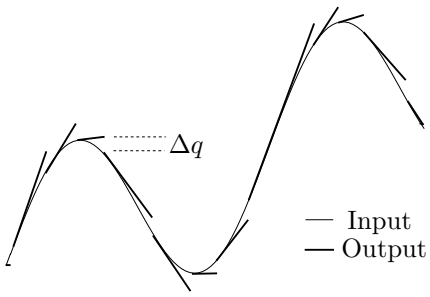

Figure 2: I/O trajectories in a *First Order* quantizer

the DEVS models of the QSS2 quantized integrators are different due to the new behavior of the quantizers. Similarly, the DEVS models of the QSS2 static functions are also different since they should take into account the slopes of the piecewise linear trajectories.

The formal definition of first order quantization functions and the DEVS models associated to the QSS2 integrators and static functions can be found in (Kofman, 2002a).

Thus, the QSS2–method can be applied to ODE systems in a similar way to QSS, i.e., building a block diagram composed with the blocks representing integrators and static functions.

C. Properties of QSS and QSS2

There are properties –which were proven in (Kofman and Junco, 2001) and (Kofman, 2002a)– that relates the solutions of Systems (1) and (2). These properties not only show theoretical features but also allow deriving rules for the choice of the quantization according to the desired accuracy.

The mentioned properties are stability, convergence and error bound and the corresponding proofs were built based on perturbation studies. In fact, defining $\Delta x(t) = q(t) - x(t)$, System (2) can be rewritten as

$$\dot{x}(t) = f(x(t) + \Delta x(t), u(t)) \tag{4}$$

From the definition of the hysteretic and the first order quantization functions, it can be ensured that each component of Δx is bounded by the corresponding quantum adopted. Thus, the QSS and QSS2 methods simulate an approximate system which only differs from the original SES (1) due to the presence of the bounded state perturbation $\Delta x(t)$.

The Convergence Property ensures that an arbitrarily small error can be achieved by using a sufficiently small quantization. A sufficient condition which guaranties this property is that the function f is locally Lipschitz.

The Stability Property relates the quantum adopted with the final error. An algorithm can be derived from the proof of this property which allows the choice of the quantum to be used in the different state variables.

Finally, the Error Bound is probably the most important property of quantization based methods. Given a LTI system $\dot{x}(t) = Ax(t) + Bu(t)$ where A is a Hurwitz and diagonalizable matrix, the error in the QSS or QSS2 simulation is always bounded by

$$|\tilde{\phi}(t) - \phi(t)| \leq |V||\mathbb{R}\mathrm{e}(\Lambda)^{-1}\Lambda||V^{-1}|\Delta q \tag{5}$$

where Λ and V are the matrices of eigenvalues and eigenvectors of A (Λ is diagonal), that is, $V^{-1}AV = \Lambda$ and Δq is the vector of quantum adopted at each component[2].

Inequality (5) holds for all t, for any input trajectory and for any initial condition.

III. HYBRID SYSTEM SIMULATION

A. Continuous Part Approximation

There is not a unified representation of hybrid systems in the literature. Anyway, the different approaches coincide in describing them as sets of ODEs or DAEs which are selected according to some variable which evolves in a discrete way (Taylor, 1993; Branicky, 1994; Broenink and Weustink, 1996; Barton, 2000).

Here, it will be assumed that the continuous subsystem can be represented by

$$\dot{x}(t) = f(x(t), u(t), z(t), m(t)) \tag{6a}$$
$$0 = g(x_r(t), u_r(t), z(t), m(t)) \tag{6b}$$

being $m(t)$ a piecewise constant trajectory coming from the discrete part, which defines the different modes of the system. Thus, for each value of $m(t)$ there is a different DAE representing the system dynamics.

It will be considered that the implicit equation (6b) has a solution for each value of $m(t)$ (which implies that the system (6) has always index 1). Variables x_r and u_r are reduced versions of x and u.

Independently of the way in which $m(t)$ is calculated, the simulation sub-model corresponding to the continuous part can be built considering that $m(t)$ acts as an input.

Then, the QSS and QSS2 methods applied to this part will transform (6) into:

$$\dot{x}(t) = f(q(t), u(t), z(t), m(t)) \tag{7a}$$
$$0 = g(q_r(t), u_r(t), z(t), m(t)) \tag{7b}$$

This model corresponds to the QSS or QSS2 associated to a generic DAE (Kofman, 2002b) which only differs from the scheme shown in Fig.1 in the presence of a new block which solves the implicit equation.

B. Discrete Part Representation

One of the most important features of DEVS is its capability to represent all kind of discrete systems. Taking into account that the continuous part is being approximated by a DEVS model, it is natural representing also the discrete behavior by another DEVS model.

[2]Symbol $|\cdot|$ denotes the componentwise module of a complex matrix or vector and symbol "$\leq$" in (5) also denotes a componentwise inequality.

Then, both DEVS models can be directly coupled to build a unique DEVS model which approximates the whole system.

In presence of only Time Events, the DEVS model representing the discrete part will be just an event generator, i.e. a DEVS model which does not receive any input and produces different output events at different times carrying the successive values of $m(t)$

Taking into account the asynchronous way in which the static functions and quantized integrators work, the events will be processed by the continuous part as soon as they come out from the generator without the need of modifying anything in the QSS or QSS2 methods.

When it comes to state events, the discrete part is ruled not only by the time advance but also by some events which are produced when the input and state variables reach some condition.

Here, the QSS and QSS2 methods have a bigger advantage: The state trajectories are perfectly known for all time. Moreover, they are piecewise linear or piecewise parabolic functions which implies that detecting the event occurrence is straightforward. The only thing which has to be done is to provide those trajectories to the discrete part so it can detect the event occurrence and it can calculate the trajectory $m(t)$.

Taking into account the trivial form of the trajectories, the discrete part can receive the state derivatives and then integrate them. It is simple and does not require computational effort since the derivative trajectories are piecewise constant or piecewise linear (in QSS2) and their integration only involves the manipulation of the polynomial coefficients.

Using these ideas, the simulation model of a hybrid system like (6) using the QSS or QSS2 method can be a coupled DEVS with the structure shown in Fig.3.

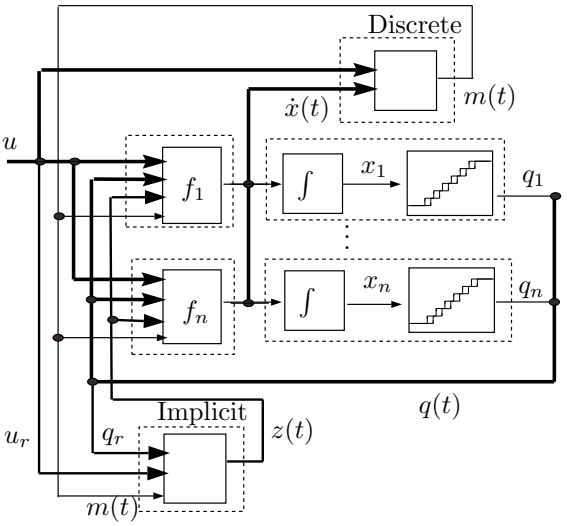

Figure 3: Coupling scheme for the QSS simulation of hybrid systems

Here the discrete part is a DEVS model which receives the events representing changes in the state derivatives as well as changes in the input trajectories.

Taking into account the generality of DEVS, the scheme of Fig.3 can simulate any systems like (6) in interaction with any discrete model.

There are cases in which this scheme can be simplified. As we mentioned before, when only Time Events are considered, the DEVS model of the discrete part will not have inputs.

Usually, the event occurrence condition is related to a zero (or another fixed value) crossing of some state variable. In this case, if the simulation is performed with the QSS–method the event condition can be detected directly by the corresponding quantized integrator. This can be easily done provided that the quantization functions contain quantization levels at the given fixed crossing values.

IV. SIMULATION EXAMPLES

A. DC-AC inverter circuit

In the inverter circuit shown in Fig.4 the set of switches can take two positions. In the first one the switches 1 and 4 are closed and the load receives a positive voltage while the other is the opposite.

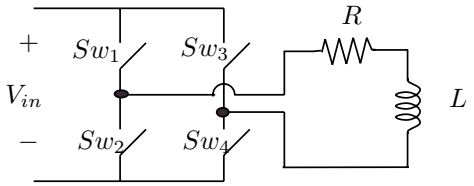

Figure 4: DC-AC Full Bridge Inverter

The system can be represented by the following differential equation:

$$\frac{d}{dt} i_L = -\frac{R}{L} \cdot i_L + s_w \cdot V_{in} \tag{8}$$

where s_w is 1 or -1 according to the position of the switches.

The system was simulated with the QSS2–method, using a scheme like the shown in Fig.3 but the discrete block was just an event generator producing events when the variable sw changes.

These event times where calculated according to a PWM strategy with a carrier frequency of 1.6kHz and a modulating sinusoidal signal of the same amplitude and a frequency of 50Hz. Thus, the number of events per cycle was 64.

Using parameters $R = 0.6\Omega$, $L = 100$mHy and $V_{in} = 300$V the simulation starting from $i_L = 0$ and taking a quantization $\Delta i_L = 0.01$A gave the result shown in Figs.5–6.

The final time of the simulation was 1 second and then the number of cycles was 50. This gives a total of

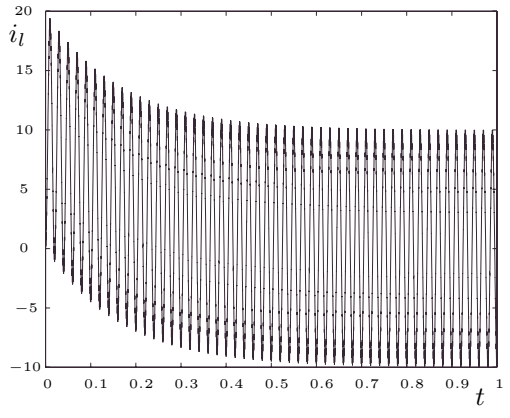

Figure 5: Load current with PWM

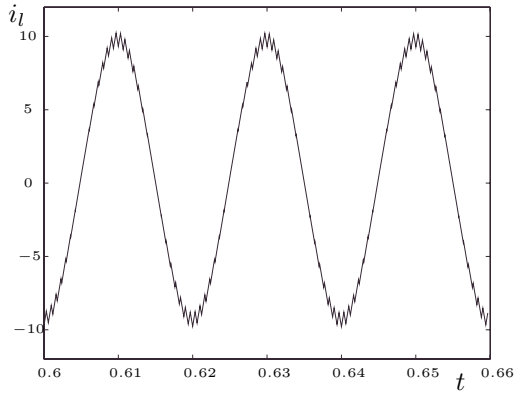

Figure 6: Detail of Fig.5

3200 changes in the position of the switches. Despite this number of events, the simulation was completed after only 3100 internal transitions quantized integrator. Thus, the total number of steps was 6300.

In this case, since the commutations do not produce any structural change (they only affect the input voltage sign), Ineq.(5) can be applied and it can be ensured that the error in the trajectory of i_L obtained is always less than 10mA.

The same system was simulated with all the discrete time methods implemented in Simulink. The best result was obtained with the fixed step ode5 algorithm (5th order), which needed more than 50000 steps to obtain an acceptable result.

The simulations were repeated with variable step methods enforcing additional calculations at the event times. Using the tolerance obtained with QSS2, the ode23 (which now showed the best performance) needed more than 20000 steps to complete the simulation.

However, this trick –enforcing calculations at predetermined time instants– cannot be used in general cases since often the event times are not known before the simulation starts.

B. A ball bouncing downstairs

Consider a ball moving in two dimensions (x and y) and bouncing downstairs. It will be assumed that the ball has a model when it is in the air –with the presence of friction– and a different model (a spring–damper) in the floor.

According to this idea, the model can be written as

$$\begin{aligned}
\dot{x} &= v_x \\
\dot{v}_x &= -\frac{b_a}{m} \cdot v_x \\
\dot{y} &= v_y \\
\dot{v}_y &= -g - \frac{b_a}{m} \cdot v_y - \\
&\quad -s_w \cdot [\frac{b}{m} \cdot v_y + \frac{k}{m}(y - \text{int}(h + 1 - x))]
\end{aligned}$$

where s_w is equal to 1 in the floor and 0 in the air. Function $\text{int}(h + 1 - x)$ gives the height of the floor at a given position (h is the height of the first step). Note that we are considering steps of $1m$ by $1m$.

The *state events* are produced when x and y verify the condition $y = \text{int}(h + 1 - x)$

Thus, the simulation model structure results similar to the one shown in Fig. 3 but without the implicit block. The discrete model should receive the events with the derivatives of x and y and send events when the event condition is achieved (to calculate that, it just has to find the roots of a second degree polynomial).

The system was then simulated using parameters $m = 1$, $k = 100000$, $b = 30$, $ba = 0.1$, initial conditions $x(0) = 0.575$, $v_x(0) = 0.5$, $y(0) = 10.5$, $v_y = 0$ and a quantum of 0.001 in the horizontal position, 0.0001 in the vertical position and 0.01 in the speeds.

The first 10 seconds of simulation were completed after 2984 internal transitions at the integrators (39 at x, 5 at v_x, 2420 at y and 520 at v_y). The trajectories do not differ appreciably from what can be obtained with a fixed step high order method using a very small step size. Fig.7 shows the simulation results.

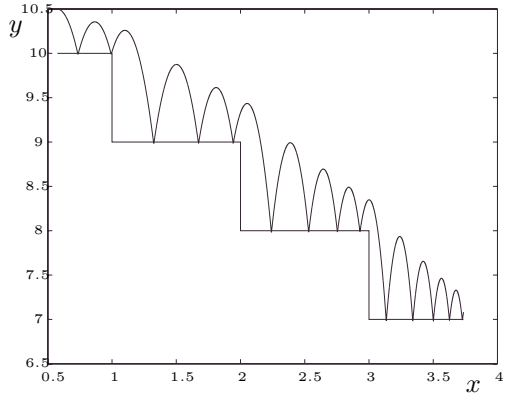

Figure 7: x vs. y in the bouncing ball example

It is important to remark that each step only involves very few calculations and the sparsity is well

exploited. In fact, the internal transitions in x does not affect any other subsystem. The steps in v_x give events to itself, to the integrator which calculates x and to the discrete model which predicts the next event occurrence. Similarly, the internal events of y only provoke external events to the integrator corresponding to v_y when the ball is in the floor and finally, the events produced in v_y are propagated to itself, to the integrator which calculates y and to the discrete model.

As a result, the discrete model receives 525 events and it produces only 26 internal transitions.

The same model was simulated with Simulink, using all the fixed and variable step algorithms. To get similar results, fixed step methods require more than 10000 steps (ode5) while variable step methods need more than 5000 steps (ode23). Here, each step involves calculations in the whole system.

V. CONCLUSIONS

The use of QSS and QSS2–methods offers an efficient and very simple alternative for the simulation of hybrid systems. Their discrete event nature and the facilities to detect discontinuities make the difference with respect to classic algorithms.

In the examples analyzed, the methods showed a performance clearly superior to all the complex implicit, high order and variable step methods implemented in Simulink.

Future work should extend the theoretical stability and error bound analysis to general discontinuous systems in order to establish conditions which ensure the correctness of the simulation. What was done in the DC-AC inverter example might constitute a first step in this direction which could be extended for general systems with time events.

Finally, in the bouncing ball example, the use of a bigger quantum in the position while the ball was in the air would have resulted in an important reduction of the number of calculation without affecting the error.

This observation leads to the convenience of using some kind of adaptive quantization. If such a result can be obtained together with the use of higher order approximations (a third order approximation QSS3 could be easily imagined) the quantization–based approximations may become a powerful tool for the simulation of general hybrid systems.

REFERENCES

Barton, P., "Modeling, Simulation, and Sensitivity Analysis of Hybrid Systems: Mathematical Foundations, Numerical Solutions, and Sofware Implementations," *Proc.of the IEEE International Symposium on Computer Aided Control System Design*, Anchorage, Alaska 117–122 (2000).

Branicky, M., "Stability of Switched and Hybrid Systems," *Proc. 33rd IEEE Conf. Decision Control*, Lake Buena Vista, FL 3498–3503 (1994).

Broenink, J. and P. Weustink, "A Combined–System Simulator for Mechatronic Systems," *Proceedings of ESM96*, Budapest, Hungary 225–229 (1996).

Cellier, F. *Combined Continuous/Discrete System Simulation by Use of Digital Computers: Techniques and Tools.* PhD thesis, Swiss Federal Institute of Technology (1979).

Esposito, J., V. Kumar, and G. Pappas, "Accurate Event Detection for Simulating Hybrid Systems," *HSCC* **2034**, 204–217. Springer (2001).

Kofman, E., "A Second Order Approximation for DEVS Simulation of Continuous Systems," *Simulation* **78**(2), 76–89 (2002a).

Kofman, E., "Quantization–Based Simulation of Differential Algebraic Equation Systems," Technical Report LSD0204, LSD, UNR, To appear in *Simulation* (2002b).

Kofman, E. and S. Junco, "Quantized State Systems. A DEVS Approach for Continuous System Simulation," *Transactions of SCS* **18**(3), 123–132 (2001).

Kofman, E., J. Lee, and B. Zeigler, "DEVS Representation of Differential Equation Systems. Review of Recent Advances," *Proceedings of ESS'01* (2001).

Pagliero, E. and M. Lapadula (2002). "Herramienta Integrada de Modelado y Simulación de Sistemas de Eventos Discretos,". Diploma Work. FCEIA, UNR, Argentina.

Park, T. and P. Barton, "State Event Location in Differential-Algebraic Models," *ACM Trans. Mod. Comput. Sim.,* **6**(2), 137–165 (1996).

Schlegl, T., M. Buss, and G. Schmidt, "Development of Numerical Integration Methods for Hybrid (Discrete-Continuous) Dynamical Systems," *Proceedings of Advanced Intelligent Mechatronics*, Tokio, Japan (1997).

Taylor, J., "Toward a Modeling Language Standard for Hybrid Dynamical Systems," *Proc. 32nd IEEE Conference on Decision and Control*, San Antonio, TX 2317–2322 (1993).

Taylor, J. and D. Kebede, "Modeling and Simulation of Hybrid Systems in Matlab," *Proc. IFAC World Congress*, San Francisco, CA (1996).

Zeigler, B., T. Kim, and H. Praehofer, *Theory of Modeling and Simulation. Second edition*, Academic Press, New York (2000).

Zeigler, B. and J. Lee, "Theory of quantized systems: formal basis for DEVS/HLA distributed simulation environment," *SPIE Proceedings* 49–58 (1998).