An Approach to Implementation of Physical Simulation Models

Shpakov Vladimir
St.Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences
39, 14th line, St. Petersburg, Russian Federation

Keywords: Process-Oriented, Hybrid Simulation, Model Design, Transitive Model, Rule-based Approach.

Abstract: Advantages of a rule-based approach to dynamic system physical models specification and implementation are discussed. Much need for the approach at the modern state of system engineering is pointed out. In particular, such an approach may be useful for simulation of control and self-organizing systems. A rule-based situation formalism of an interacting hybrid processes specification is briefly stated and some ways of its use for physical simulation model implementation are shown. Facilities of the considered methods are illustrated by examples of some simple dynamic system models implementations. Specifications of these physical models and some results of simulation are presented.

1 INTRODUCTION

A mathematical approach to dynamic system simulation serves now as accepted. It consists in following. At first, mathematic model of the system works out on base of corresponding dynamic laws (mechanics, thermodynamics, electrodynamics and others). As a rule, the model is a set of differential and algebraic equations about abstract variables. Here system physical parameters form equations coefficients. Once, the coefficients constitute rather complicated dependences upon parameters. Model experiments carry out on computers using numerical methods of the equations solution.

An imperfection of such an approach is manifested already when system parameters modifying in course of model experiments. It is caused by necessity of equations coefficients recalculation. Initial system model modification or extension causes alteration of the equations and the software for their processing.

An alternative of this abstract mathematical approach is using of physical models of dynamic systems. In this case the model variables are real system process states, such as distances, velocities, forces, angles, temperatures, pressures, levels and so on. The states evolutions are executed with procedures, which realize corresponding dynamic laws, subject to satisfying of existing constraints and relations between system elements. Among physical models advantages, it is worth to note less labor expenditures on the model development, its modification and expansion. The physical approach is rather convenient for synergetic systems simulation. It enables comparatively easy to realize systems interacting with each other and with environment.

There are a lot of computer means for dynamic system simulation including that intended for physical simulation models development (Modelica, Simulink, SimMechanics, SimElectronics and others) (Avvizano, 2008). Available computer means for physical model realization (physics engines) are oriented substantially on carrying out model experiments, system analysis and computer games development (Boeing, 2007). Their usage for creation of the models embedded for example in control systems, training or teaching equipment is rather awkward. At the same time, high processing power of modern computers enables to develop relatively simple methods of such models specification and implementation.

Hereinafter a short description of a rule-based formalism is presented and capability of its using for implementation of the physical simulation models is discussed. In the third section an example of simple dynamic system specification and some results of its simulation are presented.

Vladimir S..
An Approach to Implementation of Physical Simulation Models.
DOI: 10.5220/0003981501730176
In *Proceedings of the 2nd International Conference on Simulation and Modeling Methodologies, Technologies and Applications* (SIMULTECH-2012), pages 173-176
ISBN: 978-989-8565-20-4

2 PROCESSES REALIZATION IN TERMS OF USING A RULE-BASED SITUATION FORMALISM OF PROCESSES SPECIFICATION

Real physical interacting processes almost always are hybrid ones that is they have discrete and continuous components. A base of the formalism is an abstract model of hybrid automaton (Henzinger, 1996), in which discrete states are represented by logical variables. A set of logical variables contains also a subset of predicates of the continuous states. The predicates are used to specify interactions between continuous and discrete-event components of the processes. The logic variables can determine both discrete changes of continuous component and changes of their evolution dynamic. Thus a state of the processes model may be represented by a set of real variables X for the continuous components and a set of logic variables W for the discrete components. The last consists of subset Q for the discrete states and subset G of predicates, that is $W = Q \bigcup G$. To specify processes it is necessary to define transition functions of the following types:

$\sigma : W \to Q \times \{False, True\}$ — function of discrete state transitions;

$\delta : W \times X \to X$ — function of continuous state transitions;

$\gamma : X \to G \times \{False, True\}$ — predicate value dependence of continuous process states.

The formalism expressiveness and effectiveness of the model implementation essentially depend upon specific forms of these functions. A definitional domain of the transition function σ may be represented with help of logic formulae that makes the representation rather obvious. In the general case, the model specification may need whatever logic formulae. To our mind as the condition it is convenient to use an elementary conjunction of the logic variables. Such conjunctions may be understandably interpreted as logic-dynamic situations (Shpakov, 2004). The situation S_j may be defined as following:

$S_j = s_{j_1}, \ldots s_{j_i}, \ldots, s_{j_n}$, where $s_{j_i} = w_{j_i}$, or $s_{j_i} = \neg w_{j_i}$, $w_{j_i} \in W$, $n = 1 \ldots N_w, N_w = |W|$.

Designating the set of situations S the type of the transition function of discrete states may now be defined as following: $\sigma : S \to Q \times \{False, True\}$. This function may be assigned with help of a collection of production rules "*condition* $\to$ *action*". In the rules the logic-dynamic situation is used as a condition and the procedure, which assigns defined values to logic states, is used as an action. At that the rule takes the following form:

$$S_j \to r'_{j_1}, \ldots, r'_{j_i}, \ldots, r'_{j_m}\,, \qquad (1)$$

where $r'_{j_i} = q'_{j_i}$ or $r'_{j_i} = \neg q'_{j_i}$, $q'_{j_i} \in Q$.

To implement an arbitrary logic formula it is necessary to use several rules with the different conditions and the same executive parts. It is a specification of a normal disjunction form and it enables to realize any logic formula.

The function δ must for every mode determine new values of the process states which correspond with a current process state, the system dynamic and also with current states of some processes interacting with this process. The hybrid process mode may be naturally represented by a logic situation considered above. It was proposed to find the new values of the continuous process states by calculation of their transitive relations with the current states (Alur, 2000). Algorithms of the transitive relations calculations for some elementary processes (integration, differentiation, smoothing and others) are presented in (Shpakov, 2006). Complicated processes may be implemented by different connections of the elementary ones and using arithmetic operations and functional procedures. In this case the transitive function δ may be represented with help of a collection of following rules:

$$S_j \to x'_k = \tau_k(x_k, x_i)\,, \qquad (2)$$

where $S_j \in S$ — the situation, corresponding to the mode, τ_k — transitive relation, $x_k, x_i \in X$ — states of the continuous processes.

Since the hybrid process modes very often depend on their states being at some assigned ranges it is convenient to represent the function γ by a collection of following rules:

$$((x_{j_1} \ge a_k + x_{j_2}) \wedge (x_{j_3} \le b_k + x_{j_4})) \to g_k\,, \qquad (3)$$

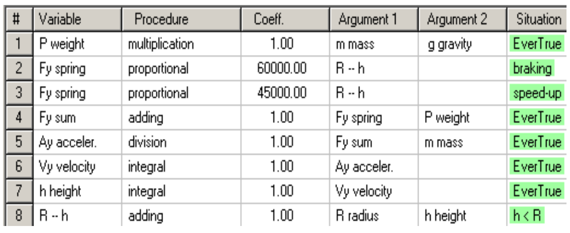
where $x_{j_1}, x_{j_2}, x_{j_3}, x_{j_4} \in X$, $g_k \in G$, a_k and b_k - constants, corresponding to the range. More complicated predicates may be received by using simple ones and rules (1).

A computer implementation of the processes specified by rules (1, 2, 3) is executed with help of an interpreter of these rules. The sets of process states variables (X and W) are represented in the interpreter with arrays of records. All rules are implemented with help of condition operators "`if…then…`". The interpreter base is an executing procedure which scanning lists of rules in a cycle. The processing algorithm calculates the value of the rule condition part. If this value is equal to `True` then a procedure of the executive part of the rule is triggered.

3 AN EXAMPLE OF PHYSICAL MODEL IMPLEMENTATION

The approach described was used when developing in SPIIRAS a prototype of computer environment oriented on a collection of interacting hybrid processes simulation. The environment has quite intuitive and obvious interface for editing rules (1, 2, 3) and processes visualization. Later, specifications in the environment of a springy hoop dropping on a plane are presented.

Physical parameters of the hoop are mass (m), a radius (R) and an elastic stiffness (k). The process states are vertical acceleration (A_y), velocity (V_y), and a center of mass height over the plane (h) and the hoop elastic deformation $(R-h)$, which originates when the hoop getting in contact with the plane. It is necessary to simulate the hoop moving in two situations: a free fall when $(h>R)$, and an elastic contact interaction when $\neg(h>R)$. The hoop free fall is occurred under the action of the weight (P), and at the elastic impact its movement is determined by the simultaneous actions of the weight and the elastic reaction force (F_y). At the situation $(h>R)$ the hoop acceleration is equal to the gravitational acceleration (g), the velocity and the height is determined by integrals of the acceleration and the velocity, respectively. The elastic reaction force is proportional to the elastic deformation $(R-h)$. In figure 1 there is a copy of part of the environment editor where the rules of the hoop state transition are presented.



#	Variable	Procedure	Coeff.	Argument 1	Argument 2	Situation
1	P weight	multiplication	1.00	m mass	g gravity	EverTrue
2	Fy spring	proportional	60000.00	R -- h		braking
3	Fy spring	proportional	45000.00	R -- h		speed-up
4	Fy sum	adding	1.00	Fy spring	P weight	EverTrue
5	Ay acceler.	division	1.00	Fy sum	m mass	EverTrue
6	Vy velocity	integral	1.00	Ay acceler.		EverTrue
7	h height	integral	1.00	Vy velocity		EverTrue
8	R -- h	adding	1.00	R radius	h height	h < R

Figure 1: State transition rules of a spring hoop dropping on a plane.

Variable names in the rules correspond to ones pointed above. The situations with names *(braking)* and *(speed-up)* are determined as follows

$$(h<R),\ (V_y<0)\rightarrow(braking),$$

$$(h<R),\ (V_y>0)\rightarrow(speed-up).$$

The rules number 2 and 3 calculate current sizes of elastic reaction forces on sections of braking and speeding up. The fourth rule calculates a sum of the forces and the fifth one does the hoop acceleration. The rules number 6 and 7 calculate the hoop current velocity and the height correspondingly. And the eighth rule calculates the size of the elastic hoop deformation.

A chart of the hoop mass center height changing is presented in figure 2.

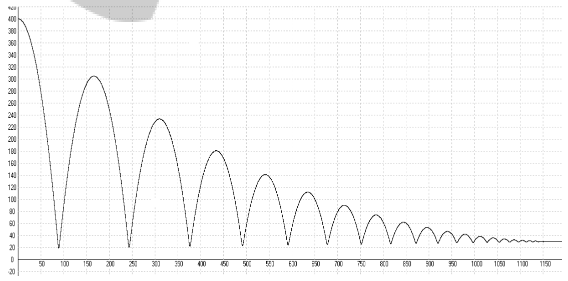


Figure 2: A chart of the hoop mass center changing.

A graphical chart of the elastic reaction force changing is presented in figure 3. The force differs from zero only when $(h<R)$. It increases on the section of braking and decreases on the section of speeding up.

Now we show how to extend the model to the case when there is a horizontal component of the velocity. In this case in the presence of friction the hoop begins to rotate as a result of its contact with the plane. At the same time the hoop horizontal velocity decreases.

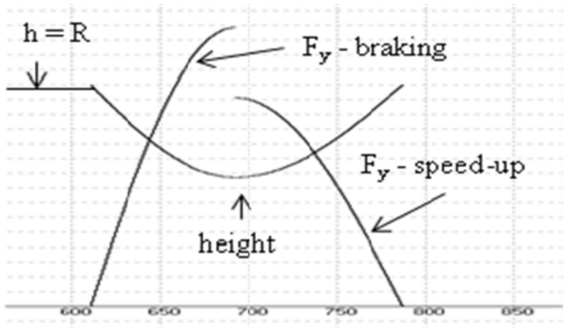
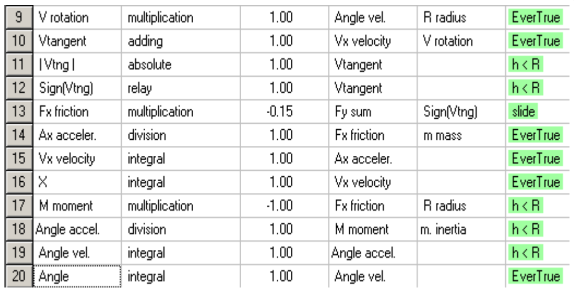


Figure 3: Change of the hoop spring force during its contact with the plane.

It is necessary to assign a hoop moment of inertia and include in the state vector variables for representation of new process states: a horizontal coordinate (X), a horizontal acceleration (A_x), a velocity (V_x), a friction force (F_x), a moment of the couple (M_{tng}), an angle, an angle acceleration, an angle rate and a velocity of the band in regard to the plane (V_{tng}). The friction force originates in a situation defined $(h < R) \wedge (|V_{tng}| > 0) \rightarrow slide$.



9	V rotation	multiplication	1.00	Angle vel.	R radius	EverTrue
10	Vtangent	adding	1.00	Vx velocity	V rotation	EverTrue
11	|Vtng|	absolute	1.00	Vtangent		h < R
12	Sign(Vtng)	relay	1.00	Vtangent		h < R
13	Fx friction	multiplication	-0.15	Fy sum	Sign(Vtng)	slide
14	Ax acceler.	division	1.00	Fx friction	m mass	EverTrue
15	Vx velocity	integral	1.00	Ax acceler.		EverTrue
16	X	integral	1.00	Vx velocity		EverTrue
17	M moment	multiplication	-1.00	Fx friction	R radius	h < R
18	Angle accel.	division	1.00	M moment	m. inertia	h < R
19	Angle vel.	integral	1.00	Angle accel.		h < R
20	Angle	integral	1.00	Angle vel.		EverTrue

Figure 4: The transition rules for simulation of the hoop moving after its contact with the plane.

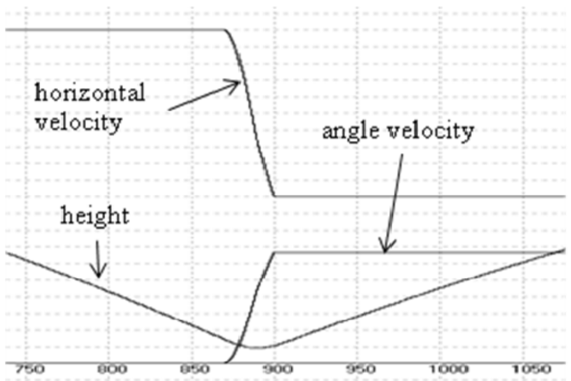


Figure 5: Changes of the hoop height, its horizontal and angle velocities during its contact with the plane.

The rules for the process states evolution specification are presented in figure 4. An addition of these 12 rules to the existing 8 ones enables to simulate an elastic hoop dropping in the presence of the horizontal component of the velocity and the hoop rotation.

Curves of the height, the horizontal and angle velocities found in result of such hoop dropping simulation are presented in figure 5.

4 CONCLUSIONS

A programming of dynamic system computer models based on use of transition rules (1, 2, 3) has a number of advantages in comparison with using of universal programming languages. A usage of the rules enables to shorten the time of the model development. The program is turned out more obvious. The model modification, extension and validation are simplified essentially. A transmission of knowledge from technicians to programmers is facilitated.

ACKNOWLEDGEMENTS

The present research was supported partly by project funded by grants 12-01-00015 of the Russian Foundation for Basic Research.

REFERENCES

Alur R., Henzinger T.A., Lafferriere G., Pappas G. J., 2000. Discrete Abstractions of Hybrid Systems, *Proceedings of the IEEE. No. 88.*

Adrian Boeing, Thomas Bräunl, 2007. Evaluation of real-time physics simulation systems, http*://www.adrianboeing.com/pal/papers/p281-boeing.pdf.*

Carlo Alberto Avvizano, 2008 Review of existing simulation tools, *http://sirslab.dii.unisi.it/I-RAS/wp-content/uploads/2008/05/simulinkreviewx.pdf.*

Henzinger T.A., 1996. The Theory of Hybrid Automata, *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS 96).*

Shpakov V.M., 2004. Executable specifications of production processes transition models, *Mechatronics, automatization, control. No. 3, (in Russian).*

Shpakov V. M., 2006. Dynamic knowledge specification on the base of continuous process transition model, SPIIRAS *Proceedings, issue 3, vol. 1, (in Russian).*