R. Kasper, D. Vlasenko

COMPARISON OF SIMULATION OF CONSTRAINED MULTIBODY DYNAMICS USING RELATIVE AND ABSOLUTE COORDINATES

Institute of Mobile Systems (IMS), Otto-von-Guericke University

Magdeburg, Germany

Dmitri.Vlasenko@Masch-Bau.Uni-Magdeburg.DE

The process of development of mechanical systems, shown in Fig. 1, is an iterative process, starting from the requirement of users and finishing by a complete product. And during all this procedure we need to perform modelling and testing of developed parts.

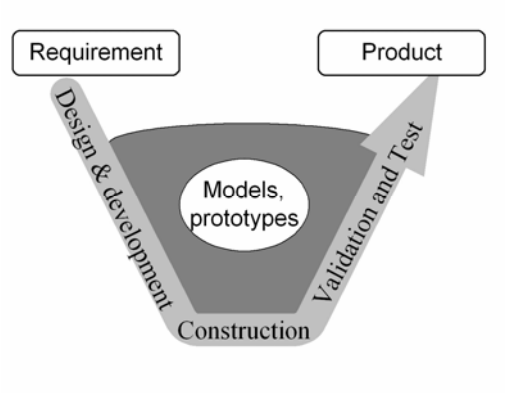

Fig. 1. V-scheme of product development

The virtualisation of this process, i.e. the development and tests of computer models instead of real prototypes, has the following significantly advantages:

- *Decreased development cost.* The building of real mechanical systems can be very expensive because the costs of the materials, assembling, starting-up and adjustment. Numerical simulation of virtual systems frees engineers from these procedures and significantly reduces the manufacturing costs.
- *Decreased development time*. Usually the construction of prototypes is a long process, where many people and firms take part. Mechanical,

electrical and software components are delivered by different companies. Special departments perform the assembling, starting-up and adjustment. Sometimes an absence of a component or a component redesign delays the prototypes' tests on a long term. Clear that the development of virtual models avoids these problems.

- *Increasing quality of the product.* Because of the high development costs firms do not have possibilities to build up many prototypes. Therefore, they are forced to limit the number of product tests. The simulation of models lets engineers comprehensively investigate the models' dynamics for different values of parameters.

Simulation of multibodies has grown rapidly in recent years with the advent of computers since engineers need to analyze increasingly complex mechanical systems. The dynamic simulation of constrained multibody systems is essential in robotics, biomechanics, vehicle and machinery design etc.

1. Simulation tool requirement In general, methods, implemented in simulating tools, can be divided in two main approaches. In the first approach (implicit joint formulation), the configuration of systems is identified using absolute coordinates, describing position and orientation of bodies relative to the inertial frame. In the second approach (explicit joint formulation), relative or joint coordinates are used to formulate a minimum number of dynamic equations which are written in terms of the systems' degrees of freedom.

The method's choice depends on the simulation tool requirements that differ in many implementation areas. Universal simulation software should requests such contradictory desires as:

1.1. **Stability** The tool should simulate the dynamics of multibodies on different time intervals. The systems of equations, describing the motion of multibodies, in general, are complicated for integration because they include either algebraic equations, describing equations of constraints, or discontinuities, appearing because of dry frictions, impact phenomena, computer-control etc. The stable solution of these equations is connected with the development of various types of stabilization and projection methods, switching algorithms and DAE solvers.

1.2. Flexibility In the last time the demand for pre-fabricated goods, with lots of options that the customer can choose from, grows significantly. The markets for flexible manufacturing depend heavily on the ability of the producer to maximize flexibility, while keeping the cost down and providing as fast a response time as possible on customized orders. This goes hand in hand with a demand for flexible modelling and simulation

tools, whereby hardware components are described by corresponding software modules that must be combinable in at least the same flexible manner as the hardware components themselves.

1.3. Usability The software should minimize the time of simulating model's redesign. Modelling should be much closer to the way an engineer builds a real system, first trying to find standard components like motors, pumps and valves from manufacturers' catalogues with appropriate specifications and interfaces [1]. The main factors that help reduce both cost and development time of software are: the reusability of software components, a high abstraction level, a distributed development and test of subsystems.

1.4. Interaction with other tools In the last few years the importance of mechatronics significantly grows [2]. The huge numbers of modern machines are complex mechatronic structures consisting of electronic units, electromechanical transformers such as sensors, actors, pure data processing units as controllers and mechanical structures. The popularity of mechatronic structures grows enormously: Computer hardware, cars, home electronics like clothes washers and video equipment, robots, airplanes etc.

That is why nowadays one of the most important requirements for a mechanical simulation tool is its interaction with electrical and control tools. Today the world's largest automotive companies estimate that 80-90% of future innovations are based on the integration of electronics and information processing in their classical mechanical products.

But special problems appear when coupling several components from different disciplines to one new system and the methodical limits of the used tool are reached, because of the different engineering domains. One possibility is the translation by analogy consideration [3]. The other way is to couple different simulation tools, but then there is no direct view to the real system components. Every result and modification has to be translated and very often this can only be done by the model developer. It is clear that this is a major source of errors.

The interaction with other tools is one of the most important parameter of simulation software.

1.5. Numerical efficiency The simulation of multibody systems should be performed in a feasible period of time. Dynamics equations based on classic Lagrange approaches are of the order $O(n^4)$, which means that the number of floating point operations grow with the fourth power of the number of bodies n in the system. Many fast algorithms have been formulated within the last two decades. For systems with small number of closed loops, numerical efficiency of some algorithms has the order $O(n)$ [4].

Still more important is the numerical efficiency in real-time systems, where the maximal time of simulation on each time period is strictly limited. The detailed review of problems of the development of real-time simulators can be found in [5].

Significantly increases the numerical efficiency of the method the distribution of calculations during the simulation. Parallelization of computations can significantly decrease the simulation time. It seems that the popularity of distribute simulation algorithms will quickly increases.

2. Explicit joint formulation In the explicit joint formulation, relative or joint coordinates are used to formulate a minimum number of dynamic equations which are written in terms of the system degrees of freedom. In this approach, closed loops are cut to obtain various open loops and the solution is required to satisfy additional algebraic equations, i.e. the equations of the loop-closing constraints. In Fig. 2 a 4-bar model developed in Dymola software is shown, based on the explicit joint formulation. Model's bodies are called marked by "b0", "b1", "b2" and "b3", cut-joint is marked by "sphereC" and other joints are marked by "j1", "j2", "rev" and "rev1".

The significant advantage of this approach is the little dimension of equations of motions that improves the numerical efficiency of simulation. The second advantage is that in the case of the simulation of systems with the tree structure the additional stabilization is not needed.

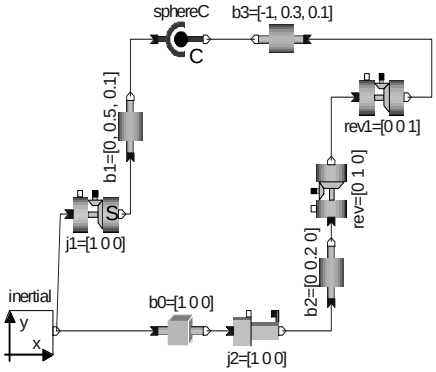

Fig. 2. Dymola model of the 4-bar pendulum

The disadvantage of this approach is the complexity of models' development. A design engineer should choose what constraints will be closing. The graph of the model is a tree (except loop-closing constraints),

therefore each link has its own orientation: it has an “input”, i.e. a body, that is nearer to the root of the tree and an “output”, i.e. a body, that is farther from the tree’s root. From that follows that a design engineer during the development of a subsystem needs to set the orientation of subsystem’s links, compatible with the orientation of the complete system. This significantly complicates the development of submodels.

The second disadvantage of this approach is that CAD models need redesign. The choice of loop-closing constraints and the definition of the direction inside of each submodel do not let use CAD models inside of the simulation software “as it is”. Taking into account that CAD-tools are very popular in the engineering community, this disadvantage should be considered as significant.

The next problem is the complication of components’ reuse. If a design engineer changes the choice of loop-closing constraints, then the structure of the model’s graph will change and the engineer needs to change the orientation of submodels’ links.

Also it is difficult to integrate simulation software, based on the explicit joint formulation, with other tools. Usually during the interaction the values of absolute coordinates and absolute velocities are needed, that desires the additional computations.

3. Implicit joint formulation In the implicit joint formulation the configuration of the system is identified using absolute coordinates, describing position and orientation of bodies relative to inertial frame. Since the equations of motion are defined using the non-minimum set of coordinates, therefore, the following two drawbacks occur: big size of equations and the necessity of stabilization of all constrained systems, both with closed loops and with the tree structure.

But on the other hand, using implicit joint formulations, we obtain important advantages during the models’ development. The modelling in simulation software using absolute coordinates is much more quickly and easier than the modelling using relative coordinates. A design engineer does not need to set the orientation of links therefore the modelling of models’ components is performed much more independently. CAD models can be used as models inside of the simulation software and it does not need additional work. Redesign and reuse of components is simple and has no limitations.

Also it is possible to improve the numerical efficiency of algorithms based on the implicit joint formulation. Matrices in equations of motion are sparse, therefore the multibody dynamics can be efficiently simulated using sparse solvers (e.g. Lapack or Umfpack) or using the symbolic decomposition of matrices.

Finally, it is easier to integrate the simulation software, based on the explicit joint formulation, with other tools. No additional computation of absolute coordinates and absolute velocities is needed.

4. Virtual Systems Developer It was developed the software Virtual Systems Developer (VSD) based on the component-oriented method for simulation of multibodies [6], using the implicit joint formulation. Unlike of a huge number of other methods, the method uses the block-module concept during simulation. This approach has many advantages:

- Subsystems can be modelled, tested and compiled. Then they can be used in a way similar to software components that encapsulate their internal structure and can be connected via interfaces.
- Critical effects like coulomb friction, backslash etc. can be encapsulated inside a subsystem.
- Subsystems are ideal candidates for the partitioning of large systems on multiple processors.
- The calculation of good-partitioned models has $O(n)$ time complexity, where n is the total number of simulated bodies

In VSD a wide set of objects, describing different types of constraints and forces was developed: revolute joint, ball joint, stiff connection, gravity force, torque, springs etc. Using Autodesk Inventor API, it was also performed an integration of VSD with Autodesk Inventor. Design engineers can specify geometric and material data of simulation models inside Inventor and then translate it into the simulation tool. This approach minimizes the model's development cost and instruction of end-users.

5. **Example of Simulation** It was performed a simulation of the car model shown in Fig. 3. This example illustrates all advantages of VSD: the object-oriented simulation of multibodies, the stabilization of a closed-loop system and the numerical efficiency of the distributed simulation.

The complete model consists of 17 bodies coupled in several subsystems: four dampers, four suspensions, four wheel subsystems. The bodies are connected by 20 joints. The hierarchy of submodels has three levels: the subsystems of the first level are dampers and wheels, the subsystems of the second level are suspensions, on the highest level is situated the complete car.

The car’s tyres are simulated by springs with dampers. The model is stable by design because additional springs are placed between wheels and ground acting in x and y direction.

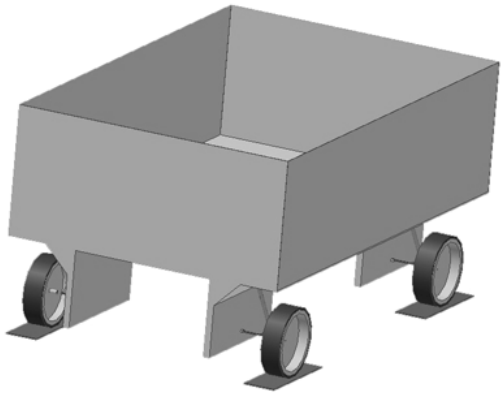

Fig. 3. Car model

We performed the emulation of the passenger that gets into the car by the additional force f (measured in Newtons) acting in z-direction, whose value depends on time t:

$$f_z = \begin{cases} 0 & \text{when } t < 2.5 \\ 1000 \cdot (t\text{-}2.5)/0.1 & \text{when } t \in [2.5,\ \ 2.6] \\ 1000 & \text{when } t > 2.6 \end{cases} \quad (1)$$

The car was modelled in Autodesk Inventor and converted to VSD. The simulation time interval was chosen to be [0s, 6s]. The integrator uses Runge-Kutta algorithm of the fourth order with the fixed time step equal to 0.001s. Fig. 4 shows the changes of z-coordinate of the car body, measured in metres.

Experimental data show that the algorithm is stable and the drift of the model has order 10^{-10}, this is equal to the accuracy of Autodesk Inventor model's definition.

For the validation of our simulations results we have built up the same model in Simpack. The comparison shows that the dynamics of the model was calculated correctly. The absolute difference between z-accelerations of the car body in the VSD and in Simpack has order 10^{-4}, absolute difference between z-coordinates has order 10^{-5}. This is much more than the error of the Autodesk model's definition because the values of wheels' spring

constants (close to 10^5 *N/m*) are large and the definition of the Simpack model was performed using generalized coordinates.

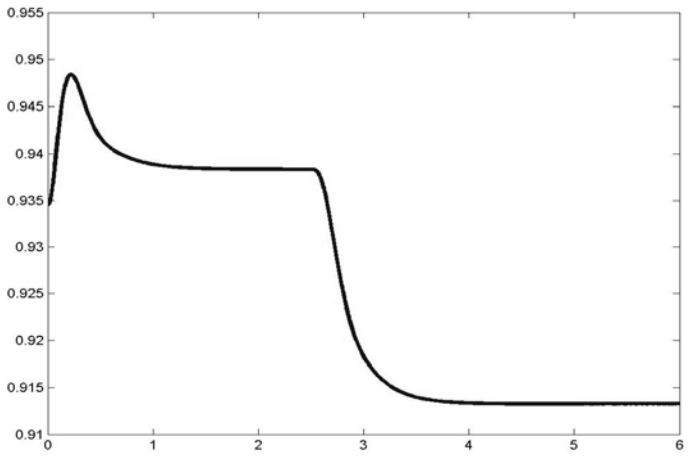

Fig. 4. *Z*-coordinate of the car body

In comparison with the undistributed simulation based on equations of motion, the component-oriented simulation of the model is about four times faster.

6. **Conclusion** In this paper we made a short review of requirements for simulation tools. Then, comparing methods, based on the implicit and explicit joint formulations, we came to the conclusion that the use of absolute coordinates is more suitable for the basement for the multibody simulation software.

It was developed VSD software simulating forward dynamic of constrained mechanical systems. It is based on an exact, non-iterative method, using the implicit joint formulation. The method is applicable to mechanisms with any joint type and any topology, including branches and kinematic loops. The simulation of good-partitioned systems has complexity O(*n*), where *n* is the total number of simulating bodies.

We reviewed the advantages of our tool: quick development of models, numerical efficiency and integration with Autodesk Inventor.

Experimental data show the stability of the method. The drift of the car model with the closed-loop structure is limited for a long period of time and has order 10^{-10} that is equal to the accuracy of Autodesk Inventor model's definition. The comparison of simulations results with results, obtained in

Simpack software, shows that the dynamics of the example was calculated in VSD correctly and accurate.

Thus, it is showed that VSD implements advantages of implicit joint formulation and is suitable for the simulation of large constrained multibody systems

References

[1] *Elmqvist H., Mattsson S. E., Otter M.* Object-Oriented and Hybrid Modeling in Modelica // Journal Européen des systèmes automatisés, 2001, Volume 35,1, pp. 1 – 10

[2] *Kasper R.* Mechanical structures in mechatronic systems // Proceedings of NAFEMS Seminar: "Mechatronics in Structural Analysis", 2004, Wiesbaden

[3] *Kasper R., Koch W.* Object-Oriented Behavioural Modelling of Mechatronic Systems // Proceedings of the Third Conference on Mechatronics and Robotics, 1995, Stuttgart: Teubner

[4] *Vlasenko D.,* Component-oriented method for simulation of multibody dynamics // PhD thesis, Institute of Mobile Systems, 2006, Otto-von-Guericke-University Magdeburg

[5] *Haug E. J., Negrut D., Serban R., Solis D.,* Numerical Methods for High-Speed Vehicle Dynamic Simulation // Mechanics of Structures and Machines, 1999, Vol. 27(4

[6] *Vlasenko D., Kasper R.,* Modular Forward Dynamic Simulation of Constrained Mechanical Systems // Proceedings of ECCOMAS Thematic Conference on Advances in Computational Multibody Dynamics, 2005, Madrid, Spain