

CONTACT PROCESSING IN THE SIMULATION OF CLAWAR

JUHASZ¹, KONYEV², RUSIN¹, SCHMUCKER¹

¹*Department Virtual Engineering, Fraunhofer Institute for Factory Operation and Automation, Sandtorstrasse 22, 39106 Magdeburg, Germany*

²*Institute for Electrical Energy Systems, University of Magdeburg, P.B. 4120, 39106 Magdeburg, Germany*

Contact processing, including collision detection and collision response, is one of the most difficult, but most important areas in simulation of the multi-body systems. However, the most widespread multi-body simulators, like Matlab/SimMechanics or Modelica/Dymola, don't support the contact processing. Other multi-body simulators, like Vortex or ODE, support the contact processing, but are more limited in the rest of the functionality. This paper presents the implementation of the contact processing in both Matlab/SimMechanics and Modelica/Dymola and the comparison of the implemented functionality with Vortex. It was performed through an example of contact tasks for a six-legged robot.

1. Introduction

In mechanical systems certain machine elements usually interact with each other. When a mathematical model of such system is designed, the interactions between the parts can be divided into two following categories: *-Mechanical joints* are used for defining permanent constraints of motion. *-Mechanical contacts* are almost instantaneous, typically short-time interactions caused by non-penetration contact forces arising between the bodies in the model. The forces occur when the surfaces of bodies touch each other. Two major phenomena occur in the mechanical contacts: collision contacts (causing collision response forces) and friction contacts (causing static or dynamic friction forces).

Contact processing is a difficult task [1,5,6]. The bodies can move in a complicated way, and they can have complex geometries. In the case of contacting bodies, the penetration of them must be prevented. There is a tradeoff between efficiency and accuracy. Accurate methods for computing contact forces are based on finite element methods. Such methods are based on subdivision of bodies into very small fragments. The surfaces of two colliding bodies are to be covered by a mesh and the relevant forces in the contact are to be computed for each point on the mesh. The resulting forces can be defined by integration of all

forces acting on the contact surface. These methods are implemented in software packages for FEM-analysis (ANSYS, Nastran etc) or in multi-body simulation (MSC Adams etc). Experiments [2] show that these methods are accurate, but require tremendous computing resources and therefore are very slow. However, many simulation applications do not require extreme accuracy and additional assumptions are taken into account providing high simulation speed, but decreasing the accuracy. As a matter of fact, different assumptions lead to different computation methods but with the same (or nearly the same) computation results. In such cases, it's not important for the application what assumptions and methods were used.

The goal of this paper is to present the ways of adding contact processing (chapter 2) to existing mechanical multi-body simulators Matlab/SimMechanics and Modelica/Dymola and to compare them with contact processing in Vortex, which is characterized by internal optimization loop with considering of energy and impulse conservation law. The results are presented though the example of contact tasks for a six-legged robot (chapter 3).

2. Contact Processing

Implementation of the contact processing is based on the following steps. (1) Mechanical models, which describe physical bodies, should be extended to describe contacting physical bodies. (2) There should be a routine that can detect collisions and can return detailed information regarding contact parameters, such as contact points and their velocities. (3) A special routine should calculate the contact response from contact parameters. (4) Each of these components should have an interface that allows replacing its implementation without doing major redesign of the other components.

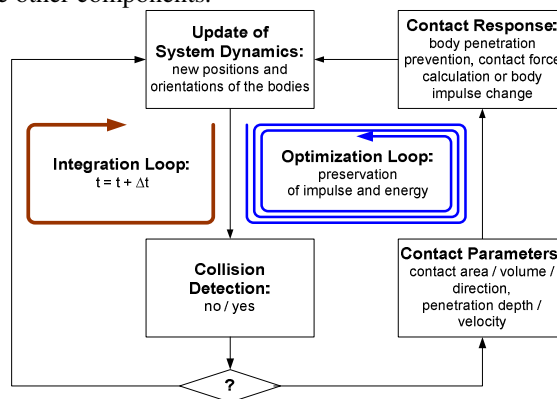


Figure 1. General scheme of contact processing in the mechanical multi-body simulation.

Figure 1 presents four basic components of contact processing, which are derived from the discussion above and will be described more detailed later.

2.1. Collision Detection

The collision detection between bodies, which are indirectly described, for example using sets of points in the space, means that the contact points or trajectories between two bodies are to be defined.

In the first part (*collision culling*), the pairs of non-colliding objects should be excluded. These can be performed using the methods of space division like Quadtree/Octree, BSP-Tree, Sweep-and-Prune. The whole space should be divided and the potentially non-colliding objects should be excluded.

In the second part (*broad collision detection*), the possibility of two objects collision by means of so-called Bounding Volumes like AABB, sphere, OBB, k-DOP should be defined. These covers simplify the complex geometry objects and make the collision detection simpler and faster.

The third part (*exact collision detection*) defines the collision between the contacting objects. Spatial, hierarchical data structure, as AABB-tree, OBB-tree, sphere-tree or k-DOP-tree should be used for faster collision definition.

2.2. Contact Parameters

The intersection test follows the third part of collision detection and finds the geometrical contact parameters, as contact plane, contact volume, contact normal and penetration, with the aim of the following methods: *intersection point of ray and sphere*, *intersection point of ray and plane*, *intersection point of ray and triangle*, *intersection line of two planes*, *intersection line (point) of two triangles*. The presented on the market algorithms (Lin-Canny Closest Features Algorithm, I/Q-COLLIDE, V-Clip, OBB-Tree, QuickCD, KDS, GJK, GJK-based EPA) combine “collision detection” and “contact parameters detection” tasks [7-11] and are implemented in the leading software, e.g. SWIFT, SOLID, ODE and others.

2.3. Contact Response and Update of System Dynamics

The contact response seems to be the most problematic and controversial part of the contact processing, since many computation approaches exist, which require different input information and may produce quite different numerical results. The following two methods are commonly used in the contact

processing: the impulse-based method and the force-based method. Both assume that the bodies are rigid. The update of system dynamics is closely connected with calculation of contact response and therefore both parts should be explained together.

2.3.1. *Impulse-based Approach*

The impulse-based approach uses collision impulses between the bodies and changes the velocity vector of the bodies during the contact [3,4]. This method based on an impact law such as Poisson's hypothesis. It considers the impulse conservation law and operates with the impulses of the colliding bodies before and after the collision as well as with the restitution coefficient of materials.

The main advantages of this method are that only a few constants are needed for description of the impact law and that the integrator step size is not influenced by the response calculation because it is performed during an infinitely small time instant. However, since the velocity is not continuous in the impulse-based model, the traditional ODE solvers can't be used. The continuous integration process in the solver should be stopped at the instant of collision and should be resumed with a new velocity. The impulse-based approach can be easily used in MBS-based models if the collision impact on the other bodies in the system is negligible (i.e. in the system of free-flying bodies). In the other words, this approach can't be used in the cases of a static objects and structures consisting of several bodies connected by joints. Furthermore such idealized impact laws are only useful for stiff collisions. These properties restrict the applicability of the impulse-based method of the dynamical analysis.

2.3.2. *Force-based Approach*

An alternative approach of contact processing in multi-body mechanical systems is based on the force and torque model of collision. It is assumed that the contacting bodies penetrate each other and the separation forces are caused by this penetration. These forces try to prevent further penetration and to separate the contacting bodies.

The calculation of contact force magnitude is difficult task and is sometimes not motivated by physics, but rather by numeric analysis. The overall result of collision should match physical laws (i.e. preservation of impulse, and preservation of energy). In addition it should be chosen so smooth that numerical methods used in simulation could handle these functions. The many existing methods for the calculation of the force in the mechanical joints and contacts are divided into two following groups:

- *Force based methods with Lagrange multipliers formulation* models the mechanical constraints (contacts and joints) with the reactive forces, which are presented as Lagrange multipliers λ . The constraint forces perform no work on the environment and the physical meaning of the mechanical contact is lost. The mechanical interaction of the bodies caused by the contact is represented by these reactive forces λ , which should be optimized between the simulation steps in the additional optimization loop (s. Fig.1) under consideration of the energy or/and impulse preservation laws.
- *Force based methods with penalty formulation* models the mechanical contact with the strong poss. nonlinear spring. The active contact/friction forces (s. Fig.2) perform work on the environment and the physical background of the mechanical contact is not lost. The mechanical interaction of the bodies caused by the contact is represented by the active forces $F_{CONTACT}$ and $F_{FRICTION}$, without any additional optimization between the simulation steps.

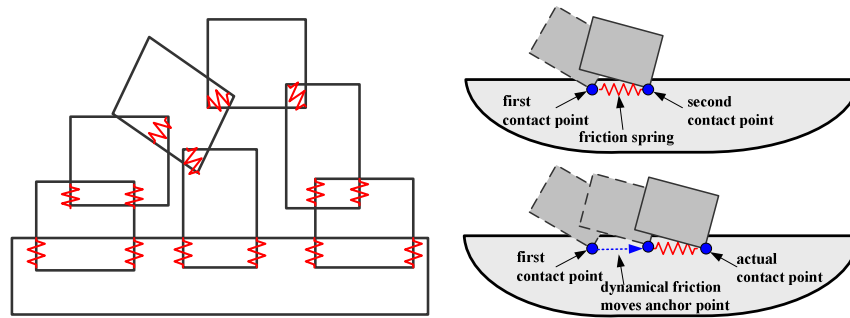


Figure 2. Force based methods with penalty formulation for contact and friction forces calculation.

There are many different equations for determining the contact/friction force magnitudes, which depend on the penetration depth p , on the penetration velocity dp/dt , on the frictional penetration l , and on the Coulomb friction coefficient μ . For the purpose of this work the following equation system is chosen for the calculation of the force magnitudes, which is very stable in the wide range of the simulation sample time and corresponds to all above mentioned requirements. Eq.1 describes the contact force magnitude depending on the stiffness $S_{CONTACT}$ in the contact area, the restitution factor ε of the materials and the collision velocity $v_{COLLISION}$ at the first time instant of intersection:

$$\begin{aligned}
F_{CONTACT} &= \begin{cases} 0, & p < 0 \\ \left[1 + \frac{1-\varepsilon}{\varepsilon \cdot v_{COLLISION}} \cdot \dot{p} \right] \cdot S_{CONTACT} \cdot p, & p \geq 0 \end{cases} \quad (1) \\
F_{FRICTION} &= \begin{cases} S_{CONTACT} \cdot l, & S_{CONTACT} \cdot l < \mu \cdot F_{CONTACT} \\ \mu \cdot F_{CONTACT}, & S_{CONTACT} \cdot l \geq \mu \cdot F_{CONTACT} \end{cases}
\end{aligned}$$

The main advantages of the force-based approach are the simplicity and the possibility of using it for stiff and soft contacts. This approach works reasonably well if several contact points are also present at the same instant time. The disadvantage of this approach is that the integrator step size should be reduced in the contact phase in order to catch the rapidly changing contact forces and torques. And similar to the impulse-based method there is necessary to choose the contact parameters (spring, damping, restitution) because the contact force is proportional not only to the penetration depth/velocity but also to the contact area and the contact volume.

Update of system dynamics take place each integration step depending of the acting contact forces/torques.

3. Simulation of Contact Tasks for Six-Legged Robot

The force based contact processing method *with penalty formulation*, represented in chapter 2, has been implemented according to Fig.1 into the multi-domain simulation environments Matlab/Simulink (s. Fig.3) and Modelica/Dymola (s. Fig.4) by means of Solid as collision detection software and has been compared with the force based contact processing method *with Lagrange multipliers formulation*, implemented into multi-body simulation environment Vortex. The six-legged robot “SLAIR2”, developed at the University of Magdeburg and the Fraunhofer Institute for Factory Operation and Automation, has been used as a test subject. Two modes, *staying on the ground* and *moving on the surface*, have been investigated and compared in the mentioned simulation environments (s. Fig.5).

Investigations show that the contact processing in Vortex is very fast, robust and reality-near. It is caused by the first-order integration method with adaptive step and by the optimization loop. However in dynamic mode the calculation of the contact forces is not precise, because the number of the optimization steps is limited to 35 by developers.

The developed contact processing environments in Matlab/Simulink and Modelica/Dymola are free from aforementioned disadvantages, can use the integration methods of higher order, and are sufficiently precise by calculation of

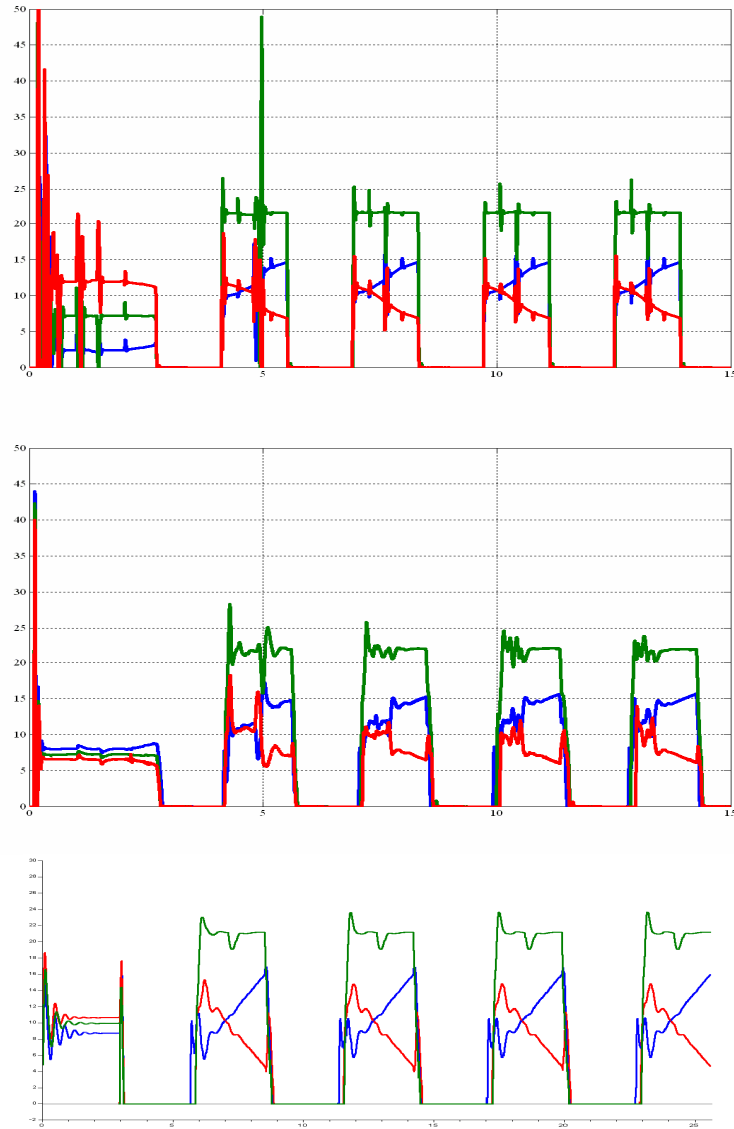


Figure 5. Comparison of the normal forces under the feet of the six-legged robot “Slair2” staying on the ground and then doing four steps on the surface with tripod gait in (top)Vortex, (middle) SimMechanics/Solid, and (down)Dymola/Solid: blue – front leg, green – middle leg, red – rear leg.

References

1. V. Engelson. *Integration of Collision Detection with the Multibody System Library in Modelica*. Thesis, PELAB, IDA, Linkoping University, 2000.
2. D. Fritzson, P. Fritzson, P. Nordling, T. Persson. *Rolling Bearing Simulation on MIMD Computers*. Int. Journal of Supercomp. Appl. and High Performance Computing, 11(4), 1997.
3. B. Mirtich: *Impulse-based Dynamic Simulation of Rigid Body Systems*. Ph.D. thesis, University of California, Berkeley, 1996.
4. P. Zhang. *Physically Realistic Simulation of Rigid Bodies*. Thesis, Department of Computer Science, Tulane University, 1996. Available via <http://www.eecs.tulane.edu/www/Zhang/>.
5. M. Otter, H. Elmqvist, J. Diaz Lopez. *Collision Handling for the Modelica MultiBody Library*. 4th International Modelica Conference, pp.45-53, March 2004.
6. N. Galoppo, M. Otaduy, P. Mecklenburg, M. Gross, M. Lin. *Fast Simulation of Deformable Models in Contact Using Dynamic Deformation Textures*. ACM SIGGRAPH Symposium on Computer Animation, 2006.
7. M. C. Lin: *Efficient Collision Detection for Animation and Robotics*. Ph.D. thesis, University of California, Berkeley, 1992.
8. K. Chung and W. Wang: *Quick Collision Detection of Polytopes in Virtual Environments*, , ACM Symposium on Virtual Reality Software and Technology 1996, 1-4, July, 96, University of Hong Kong, Hong Kong.
9. D. Schmalstieg, R. F. Tobler: *Real-time Bounding Box Area Computation*. Institute of Computer Graphics, Vienna University of Technology, 1999.
10. J. Erickson, L.J. Guibas, J. Stolfi and Li Zhang: *Separation-sensitive collision detection for convex objects*; Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms, Pages 327 – 336, 1999.
11. G. van den Bergen: *Collision Detection in Interaction 3D Environments*. 2003.