ALGORITHMS OF MULTIBODY DYNAMICS SIMULATION USING ARTICULATED-BODY METHOD

M.V. MIKHAYLYUK, E.V. STRASHNOV AND P.YU. TIMOKHIN

Scientific Research Institute of System Analysis, Russian Academy of Sciences (SRISA RAS), Russia, 117218, Moscow, 36/1 Nakhimovskiy Avenue
e-mail's: mix@niisi.ras.ru, strashnov_evg@mail.ru, webpismo@yahoo.de

Summary. In this paper we consider a real-time multibody dynamics simulation related to the video simulators. The articulated-body method and its modifications based on impulses are used to solve this problem. We present an algorithm that includes the semi-implicit Euler method for numerical integartion of motion differential equations and the sequential impulses method for constraints enforcing. Proposed methods and algorithms were implemented and verified in robot simulator, developed in SRISA RAS.

1 INTRODUCTION

In computer video simulators, it is often necessary to simulate the dynamics of mechanisms, robots and manipulators, represented as multibody systems. These systems can have a different structure (open and closed kinematic chains, trees), and contain such complex control mechanisms as manipulator grasp device, wheel suspensions, Peaucellier-Lipkin linkages etc. In this paper we consider multibody dynamics simulation with collision detection and collision response of the virtual environment objects. In addition, there are joint friction and joint limits on the relative motion. In turn, virtual reality systems are imposed their limitations on methods and algorithms of multibody dynamics simulation. The basic requirement is to perform all calculations in real-time mode. In this case, it is necessary to ensure the accuracy of calculations which allows virtual robots to perform basic technological tasks with the visual plausibility. In the paper we would like to propose methods and algorithms of multibody dynamics simulation that meet the above requirements.

There are two approaches for multibody dynamics simulation, based on the coordinate formulation. In paper [1] was developed an iterative sequential impulses method which uses maximal coordinates. In this method, each link is simulated independently and each joint connection is expressed as holonomic constraint. The sequential impulses method is robust, versatile, and applicable for solution of many problems in virtual environment systems. However, this method does not always provide the required accuracy for simulation of complex mechanisms in the virtual scenes. Furthermore, if maximal coordinates are used, it is necessary to ensure a motor constraint [2], joint limits constraints [3] and constraint stabilization [1, 3]. There is an alternative approach to the description of multibody dynamics by means of generalized coordinates (also known as reduced coordinates). The advantage of this approach is the simplicity of describing certain constraints (for example, joint limits constraints) and engine simulation, and also constraint stabilization is not required in it. However, compared with the first approach, this one is less universal, tedious, and it requires the special formulation of multibody dynamics equations. In this work, methods of multibody dynamics simulation with generalized coordinates are involved.

2010 Mathematics Subject Classification: 70E55, 97R60.
Key words and Phrases: Multibody System, Real-Time Simulation, Video Simulator, Articulated-Body Method, Constraints, Sequential Impulses Method.

Depending on the solution method of forward dynamics problem (calculation of unknown accelerations through forces), there are semi-recursive [4, 5] and fully recursive formulations [6, 7]. The composition and solution of a system of N linear equations with respect to $\ddot{\overline{q}}$ are required in a semi-recursive formulation, where $\overline{q}$ is the set of generalized coordinates, N is the number of degrees of freedom of multibody system. Note that direct methods need $O(N^3)$ arithmetic calculations to solve this linear system. The fully recursive formulations, such as single body method [6] and articulated-body method [7], don't work with the linear system explicitly, and calculations are performed using small vectors and matrices. According to [8], these methods are equivalent to the tridiagonal matrix algorithm for solving a linear system, therefore just $O(N)$ calculations are required. With a small $N \sim 10$ the computation time of both approaches is almost the same [5]. However, for $N > 10$ the second approach requires less computation time.

In this paper, an articulated-body method [7] and its variations for constraints enforcing [9, 10] are used. Constraints enforcing method that uses forces was developed in [9], whereas the impulse-based approach was applied in [10]. The use of impulses is more versatile compared to forces, since the impulse application allows the body velocities to be directly changed. Furthermore, the impulse-based approach effectively handles the body interaction constraints (impact, contact and friction). In this paper we develop an algorithm for multibody dynamics simulation by using the semi-implicit Euler method [11] for solving multibody motion equations. The constraints enforcing is performed by the developed sequential impulses method [1] that handles each constraint with an impulse calculation. The proposed algorithms and methods have been implemented in the software modules written on the C++. In the dynamics subsystem of the robot simulator developed in SRISA RAS, approbation was carried out, which demonstrated the applicability of these methods and algorithms for multibody dynamics simulation.

2 THE ARTICULATED-BODY METHOD

First of all, we describe the fundamentals of the articulated-body method [7] that is used to dynamics simulation of multibody system with the tree structure. Represent the multibody system as a connectivity graph, in which the nodes correspond to links (bodies) and the arcs correspond to joints. In this case, the *regular numbering* scheme for links and joints is used. According to this scheme, the node number 0 represents the immovable fixed base of multibody system. Then the bodies and joints are numbered consecutively from the base so that the i-joint connects parent $\lambda(i)$-link with child i-link and $\lambda(i) < i$. We will consider a multibody system with one-degree-of-freedom joints (revolute and prismatic), while multi-degree-of-freedom joints will be represented as series of one-degree-of-freedom joints interconnected with links of zero mass and inertia. Then each joint is defined by its coordinate q_i, while the whole multibody system are defined by an *N*-dimensional vector $\overline{q}$ of generalized coordinates, where N is the number of multibody system joints.

The spatial algebra of 6D vectors is used to derive the motion equations of multibody system in the articulated-body method. These vectors are denoted by spatial symbol with hat (e.g. $\hat{s}$). Among these vectors there are motion-type vectors in M^6 (e.g., $\hat{v}$ is the spatial velocity) and force-type vectors in F^6 (e.g., $\hat{f}$ is the spatial force). Detailed description of the spatial vector algebra can be found in [7]. Note that each of these vectors obeys its own rule of transformation from one coordinate system to another. The matrix ${}_B X_A$ performs coordinate transformation from Cartesian frame A to B for a motion vector, and ${}_B X_A^* = {}_B X_A^{-\mathrm{T}}$ does the same for a force vector.

The articulated-body method is developed to solve the forward dynamics problem, which is the calculation of body accelerations given applying forces. This problem is reduced to computing the generalized accelerations $\ddot{\overline{q}}$ that depend on the known coordinates $\overline{q}$ and velocities $\dot{\overline{q}}$. The main idea of the method is that for the *handle* (body) i of the articulated-body there is a linear relationship [10] between the applied force $\hat{f}_i$ and the resulting acceleration $\hat{a}_i$, that is given by

$$\hat{f}_i = I_i^A \hat{a}_i + \hat{Z}_i^A, \tag{1}$$

where I_i^A is the articulated-body inertia, $\hat{Z}_i^A$ is the articulated bias force.

The calculation of I_i^A and $\hat{Z}_i^A$ is performed from terminal links to handle i by the Newton-Euler equations of motion for bodies, with transferring forces from child's to handle, and reduction the equations to the form (1). The force $\hat{f}_i$ is related to the joint torque τ_i as follows:

$$\hat{s}_i^{\mathrm{T}} \hat{f}_i = \tau_i, \tag{2}$$

where $\hat{s}_i$ is the vector of allowed joint motion. Then eliminating $\hat{f}_i$ using (1), we obtain for each i-th joint an equation for $\ddot{q}_i$ in terms of $\hat{a}_{\lambda(i)}$, I_i^A and $\hat{Z}_i^A$.

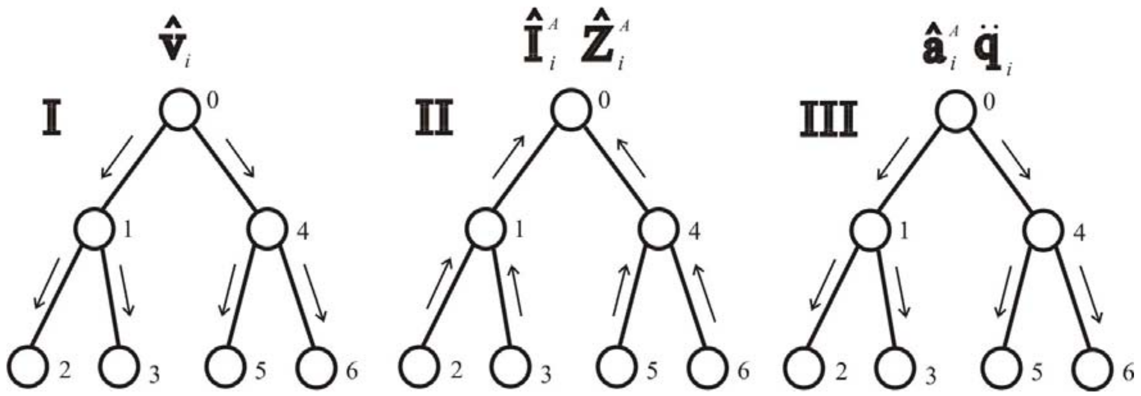

Figure 1: The articulated-body algorithm

The articulated-body algorithm consists of three passes through the tree of multibody system, as shown in Figure 1. In the first step we pass from the base to the tips of tree and for each i calculate the body velocity $\hat{v}_i$, matrix ${}_i X_{\lambda(i)}$ of coordinate transformation from parent $\lambda(i)$-link to child i-link and the matrix ${}_i X_0$ of coordinate transformation from fixed base to i-link, and also the quantities I_i^A and $\hat{Z}_i^A$ are initialized by the body inertia and applied forces and moments. At the same time, to reduce the computations, all quantities of i-link are defined in local coordinate system, located in the i-joint. At the second step, an upward pass from the tips of tree to the base is made, and for each i-link we calculate the articulated-body inertias I_i^A and bias forces $\hat{Z}_i^A$. At the last step, an outward pass through the tree is performed with the calculation of the accelerations. In addition, we assume the uniform gravitational field, therefore the base acceleration is initialized by $\hat{a}_0 = \begin{pmatrix} \bar{0} \\ -\bar{g} \end{pmatrix}$. Note that for each i the $\ddot{q}_i$ is first computed from the known quantities $\hat{a}_{\lambda(i)}$, I_i^A and $\hat{Z}_i^A$, and only then acceleration $\hat{a}_i$ is determined. Detailed description of the articulated-body method with various optimizations can be found in [7, 10].

3 CONSTRAINT ENFORCING FOR MULTIBODY SYSTEM

The articulated-body method is not general for multibody dynamics simulation. In this paper, the constraint formulation is used to handle a collision with other objects, joint friction, closed kinematic chain, etc. To provide these constraints we propose an impulse-based approach. In this case, depending on the type of constraint, it is necessary to impose constraints on linear, angular or generalized velocities.

Consider the technique of one constraint enforcing with respect to linear velocities. It is known [10] that to enforce this kind of constraint, it is necessary to calculate such an impulse p_f that

$$p_f = m_{eff} \Delta v = m_{eff} (v^d - v^c), \tag{3}$$

where m_{eff} is the effective mass, Δv denotes constraint velocity increment, v^c is the current constraint velocity, and v^d is the desired constraint velocity.

Now the constraint enforcing problem for multibody system involves a calculation of the effective mass m_{eff}, impulse p_f and its application, to calculate new velocities so that the constraint velocity will be equal to v^d.

At first, we outline the algorithm for calculating effective mass m_{eff}. According to (3), if we apply the unit test impulse $p_f = p_t = 1$ to the multibody system and calculate the constraint velocity increment Δv_t, then the effective mass can be calculated as

$$m_{eff} = \frac{p_t}{\Delta v_t} = \frac{1}{\Delta v_t}. \tag{4}$$

To calculate constraint velocity Δv_t, it is necessary to determine the changes $\Delta \hat{v}_i$ in the absolute velocities of the link and the changes $\Delta \dot{q}_i$ in generalized velocities after applying the impulse p_t. According to [10], we obtain equations for $\Delta \dot{q}_i$ by integrating of equations and realizing the passage to the limit in the equations of motion in the articulated-body method. In this case, for each link, first articulated bias impulses $\hat{Y}_i^A$, and then link velocities and generalized velocities are calculated.

The algorithm for calculating test velocities is as follows. First, articulated bias impulses $\hat{Y}_i^A$ are initialized with zero values. Then, two passes through the tree are performed, as shown in Figure 2. Proceeding from the fact that the constraint is dual and in the general case two links of the multibody system may be involved (in Fig. 2, a pair of links with numbers 2 and 5 are indicated by black circles), at the first upward pass from these links to the root link the impulses $\hat{Y}_i^A$ are calculated. The second step is a calculation of the changes in link velocities $\Delta\hat{v}_i$ and the changes in generalized velocities $\Delta\dot{q}_i$ by downward pass from root link to these links. Further, the test velocity Δv_t is calculated from the obtained velocities, and then the effective mass m_{eff} is determined by the formula (2). Note that if $\Delta v_t = 0$, then the constraint is degenerate and does not require further consideration.

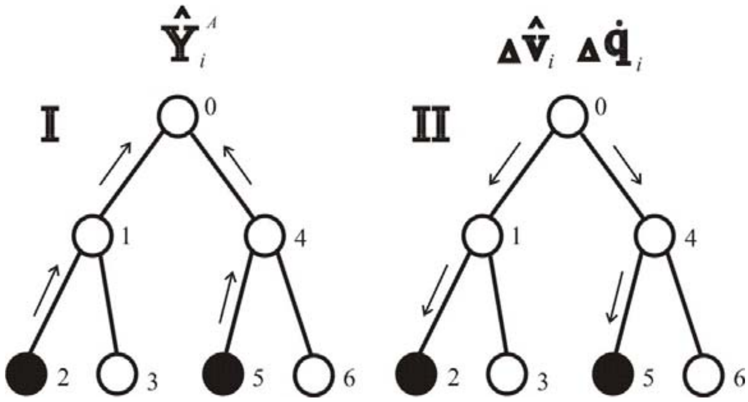

Figure 2: Algorithm for calculating test velocities

The technique of calculating the effective mass with respect to angular and generalized velocities is similar.

The equation for calculating angular impulse p_τ (similar to equation (3)) takes the form

$$p_\tau = m_{eff}\Delta\omega = m_{eff}(\omega^d - \omega^c), \tag{5}$$

where $\Delta\omega$ is the angular constraint velocity increment, ω^c is the current angular constraint velocity, and ω^d is the desired angular constraint velocity.

Similarly, the equation for calculating joint impulse p_n takes the form

$$p_n = m_{eff}\Delta\dot{q} = m_{eff}(\dot{q}^d - \dot{q}^c), \tag{6}$$

where $\Delta\dot{q}$ is the joint constraint velocity increment, $\dot{q}^c$ is the current joint constraint velocity, and $\dot{q}^d$ is the desired joint constraint velocity.

To enforce various constraints, it is required to calculate the impulses of the form (3), (5) or (6). After calculating these impulses, it is required to update the link velocities $\hat{v}_i$ and the generalized velocities $\dot{q}_i$. For this, an algorithm is used that is analogous to the algorithm for calculating test velocities, as shown in Figure 3. The only difference is that the downward calculation of $\hat{v}_i$ and $\dot{q}_i$ is performed from the root link to all child links.

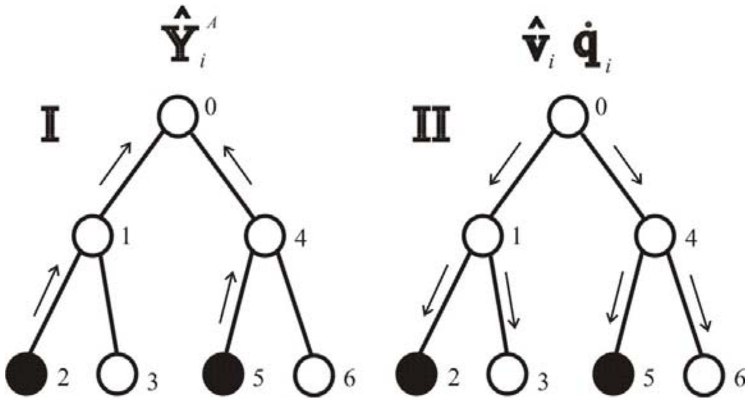

Figure 3: Algorithm for velocity calculation by the impulses

Later, these algorithms will be used to enforce several constraints by the sequential impulses method.

4 ALGORITHM OF MULTIBODY DYNAMICS SIMULATION

The algorithms considered above based on the articulated-body method, can be used to solve the problem of multibody dynamics simulation in the virtual environment systems. In this paper, we propose an approach in which semi-implicit Euler scheme [11] is used to integrate the equations of motion and the constraint enforcing is performed by the sequential impulses method [1]. The main problem to be solved here is to determine the state of the multibody system at the next time $t+\Delta t$ on the basis of the previous state corresponding to the time t. To solve this problem, an algorithm consisting of several steps is developed. The first step is the articulated-body algorithm which allows for each i-link to compute generalized acceleration $\ddot{q}_i(t)$ through the generalized coordinates $q_i(t)$ and velocities $\dot{q}_i(t)$, applying known forces and joint moments. Then, according to the Euler scheme, the generalized velocities $\dot{q}_i(t+\Delta t)$ are calculated as

$$\dot{q}_i(t+\Delta t) = \dot{q}_i(t) + \Delta t \ddot{q}_i(t) . \tag{7}$$

These velocities are used to determine the link velocities $\hat{v}_i(t+\Delta t)$ by passing through the tree from the root link to the child links (see the first step of the articulated-body algorithm).

At the next step of the algorithm, the obtained generalized velocities $\dot{q}_i(t+\Delta t)$ must be modified so that the constraints with respect to velocities are satisfied. For this, an iterative sequential impulses method with successive constraints enforcing and impulses application is proposed. Primarily, in this method for each constraint the effective masses m_{eff} are calculated using test velocities algorithm. Then, at each k-th iteration of the method, impulse p_*^k is determined for various types of constraints (respect to linear, angular, or generalized velocities). Since there are many constraints, it is necessary to perform several iterations in order to calculate the desired impulse p_*^k, but by means of equations (3), (5) and (6) we can determine just impulse increment Δp_*^k. The desired impulse p_*^k will be calculated in several iterations as follows

$$p_*^k = p_*^{k-1} + \Delta p_*^k . \quad (8)$$

The calculated impulses Δp_*^k and p_*^k can't be arbitrary for certain types of constraints such as contact and friction of interacting bodies. For example, the contact constraint is non-holonomic that requires $p_*^k \geq 0$. In turn, according to the Coulomb law for friction must be $\left|p_*^k\right| \leq p_{\max}$. Therefore, to solve this problem, **a clipping algorithm** for the impulse p_*^k by boundary values $p_{\min}$ and $p_{\max}$ is used with the subsequent correction of impulse Δp_*^k:

If $p_*^k > p_{\max}$, *then*

$\Delta p_*^k = \Delta p_*^k + p_{\max} - p_*^k$;

$p_*^k = p_{\max}$,

otherwise, if $p_*^k < p_{\min}$, *then*

$\Delta p_*^k = \Delta p_*^k + p_{\min} - p_*^k$;

$p_*^k = p_{\min}$.

Ultimately, the resulting impulse increment Δp_*^k is applied to change link velocities $\hat{v}_i(t+\Delta t)$ and generalized velocities $\dot{q}_i(t+\Delta t)$. For this, the previously described algorithm for velocities calculation through the impulses is used.

In the general case, constraints are imposed to the link coordinates, which leads to the fact that during the simulation they can be violated. Therefore, the constraint stabilization should be performed to satisfy these constraints. A split impulses method was proposed in [1], where the stabilization was performed by using so called *pseudo velocities*. In terms of a multibody system, pseudo link velocities $\hat{v}_i'(t+\Delta t)$ and pseudo generalized velocities $\dot{q}_i'(t+\Delta t)$ are introduced for each link. At each k-th iteration the constraints are provided by pseudo impulses $p_*'^k$ and $\Delta p_*'^k$. In this case, the procedure for calculating and applying pseudo impulses is analogous to usual impulses.

The iterations of the sequential impulses method continue until one of the end criteria is met. There are some possible criteria: small values of the impulses increments $\left|\Delta p_*^k\right| \leq \varepsilon$ and $\left|\Delta p_*'^k\right| \leq \varepsilon$ for all constraints (where $\varepsilon > 0$ is given small value), exceeding the maximum allowed count of iterations, the expiration of a given time, etc.

The last step of multibody dynamics algorithm is the calculation of the generalized coordinates $q_i(t+\Delta t)$ by the semi-implicit Euler scheme with respect to generalized velocities $\dot{q}_i(t+\Delta t)$ and pseudo generalized velocities $\dot{q}_i'(t+\Delta t)$ as follows:

$$q_i(t+\Delta t) = q_i(t) + \Delta t(\dot{q}_i(t+\Delta t) + \dot{q}_i'(t+\Delta t)) . \quad (9)$$

After calculating generalized coordinates, the pseudo generalized velocities zeroed, i.e. $\dot{q}_i'(t+\Delta t) = 0$.

To accelerate the convergence of the sequential impulses method, the multibody dynamics algorithm is supplemented by the procedure for applying accumulated impulses $p_*(t)$ from the previous step of simulation. For this, we use the algorithm for calculating velocities for all constraints.

Now we describe the whole algorithm for multibody dynamics simulation using the articulated-body method. Since [1] involves the sequential impulses method with maximal coordinates (the SIMMC algorithm), we call our algorithm as SIMGC (Sequential Impulses Method with Generalized Coordinates).

Algorithm SIMGC:

1. For each i-link the generalized accelerations $\ddot{q}_i(t)$ are calculated by means of the articulated-body algorithm (see Fig. 1) from the known values of $q_i(t)$ and $\dot{q}_i(t)$.
2. The generalized velocities $\dot{q}_i(t+\Delta t)$ are determined according to the formulas (7).
3. A downward pass through the tree with the calculation of the link velocities $\hat{v}_i(t+\Delta t)$ through the previously obtained generalized velocities $\dot{q}_i(t+\Delta t)$.
4. Correct the velocities $\hat{v}_i(t+\Delta t)$ and $\dot{q}_i(t+\Delta t)$ using the accumulated impulses $p_*(t)$ from the previous simulation step for active constraints. For this, algorithm for velocity calculation through the impulses is applied (see Fig. 3).
5. For each constraint the effective mass m_{eff} is calculated. Depending on the type of constraint the algorithm for calculating test velocities Δv_t (see Fig. 2) with formula (4) (or analogous formula) is applied. If $\Delta v_t = 0$, then we assume that constraint is degenerate and don't further enforce it.
6. For each i-link the pseudo velocities $\hat{v}_i'(t+\Delta t)$ and $\dot{q}_i'(t+\Delta t)$ are initialized by zero.
7. The loop by k for all constraints (with the calculation of pseudo impulses, if constraint stabilization is required):
   - depending on the type of constraint the current velocity increment Δv^k, $\Delta\omega^k$ or $\Delta\dot{q}^k$ is determined;
   - according to the type of constraint, the impulse Δp_*^k is calculated by the formula (3), (5) or (6);
   - the accumulated impulse p_*^k is determined by formula (8);
   - modify impulses p_*^k and Δp_*^k by clipping algorithm with values $p_{\min}$ and $p_{\max}$;
   - the velocities $\hat{v}_i(t+\Delta t)$ and $\dot{q}_i(t+\Delta t)$ are modified by the impulse Δp_*^k;
   - depending on the type of constraint the current pseudo velocity increment $\Delta v'^k$, $\Delta\omega'^k$ or $\Delta\dot{q}'^k$ is determined;
   - the pseudo impulse $\Delta p_*'^k$ is calculated by the formulas similar to (3), (5) or (6);
   - the accumulated pseudo impulse $p_*'^k$ is determined by the formula similar to (8);
   - modify impulses $p_*'^k$ and $\Delta p_*'^k$ by clipping algorithm with values $p'_{\min}$ and $p'_{\max}$;

- the pseudo velocities $\hat{v}_i'(t+\Delta t)$ and $\dot{q}_i'(t+\Delta t)$ are modified by the impulse $\Delta p_*'^k$;
- if the computation time $t > T$ or $\left|\Delta p_*^k\right| \le \varepsilon$ and $\left|\Delta p_*^k\right| \le \varepsilon$ for all constraints, then finish the loop.

End of loop.

8. For each i-link the generalized coordinates $q_i(t+\Delta t)$ are calculated by (9).

The SIMGC algorithm is easily extended to handle collisions of a multibody system with a single rigid body. The calculation and application of impulses for single rigid body is described in [12].

5 RESULTS

The proposed methods and algorithms have been implemented as program modules in the dynamic libraries for Windows OS. For this, a library of spatial vectors and operations over them was developed (written in object-oriented C++), including six-dimensional motion-type and force-type vector, spatial 6×6 matrix, spatial inertia with mass, inertia tensor and center of mass. In addition, a class of spatial transformations with a rotation matrix and a position vector was created that transforms the spatial motion-type and force-type vectors with minimal computational costs. The developed library of spatial vectors has been used for multibody dynamics simulation.

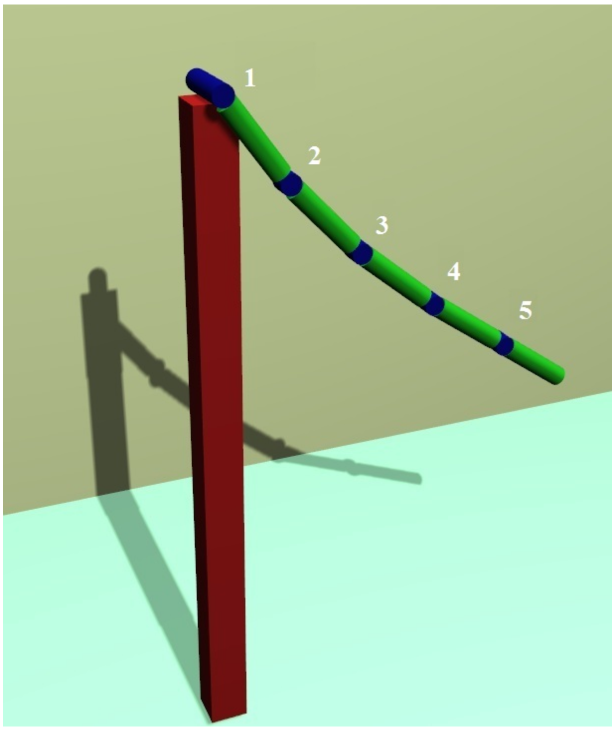

Figure 4: Five-link chain

The Figure 4 shows a kinematic chain consisting of five links connected by hinge joints with friction (they are numbered 1-5). The first link of the chain is attached to the fixed base (column) through the hinge joint 1. In this scene, the chain swing under the action of gravity is simulated. For this, the initial horizontal position of the chain is selected. Due to the presence of joint friction, in some time the chain is stabilized vertically.

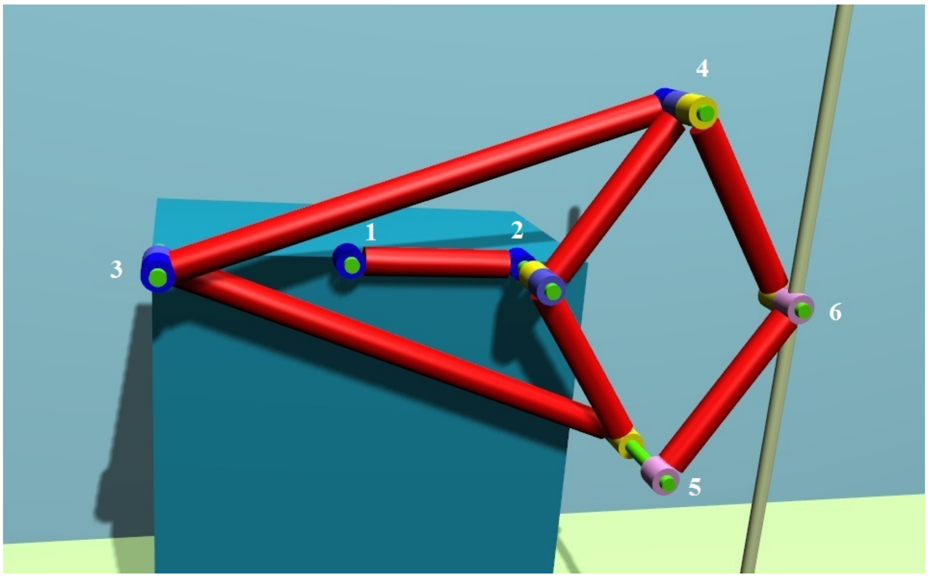

Figure 5: Peaucellier–Lipkin linkage

The second virtual scene shows a Peaucellier–Lipkin linkage (see Fig. 5) that transforms the rotational motion into translational motion. This mechanism contains of 3 closed kinematic chains with 7 links, 1 hinge with engine (in Fig. 5 it’s a joint 1) and 5 hinge joints (in Fig. 5 they are joints 2-6). In this scene the point of hinge joint number 6 moves along a straight line while the joint 1 rotates.

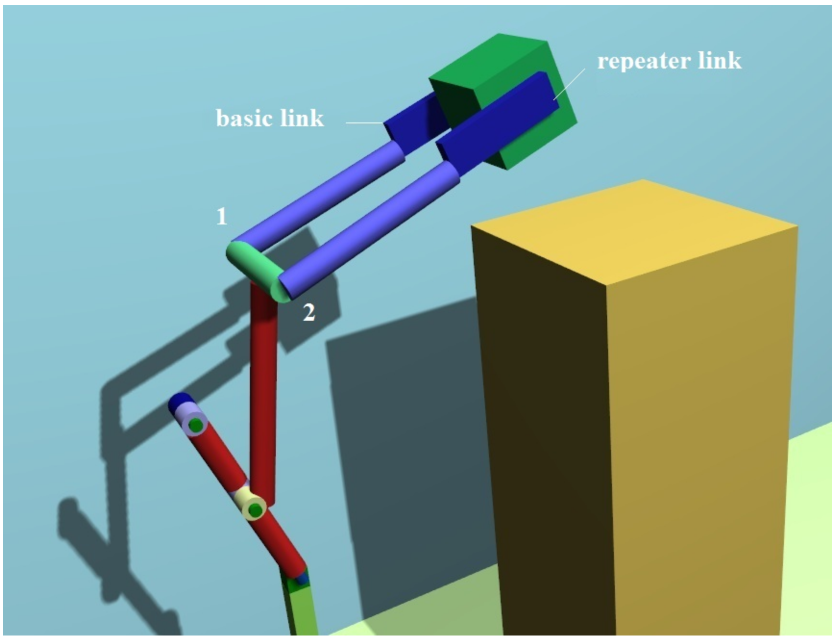

Figure 6: The manipulator grasps an object

In the last scene (see Fig. 6), the object capture by manipulator is considered. The grip mechanism is realized by means of so-called repeater [2]. Here one engine (in Fig. 6 it's a joint 1) controls simultaneously two links through a repeater (in Fig. 6 it's a joint 2). This scene shows simulation of grasping and moving of the object by manipulator.

Simulation in these test scenes has been performed with two algorithms: the SIMMC (Sequential Impulses Method with Maximal Coordinates) algorithm and the SIMGC (Sequential Impulses Method with Generalized Coordinates) algorithm. The simulation step Δt is variable and is limited to 10 ms. As the criterion for termination of iterations, the value $\varepsilon = 10^{-6}$ for the impulses was selected. The algorithms have been tested on a 3 GHz Pentium D with 4GBytes of RAM. Table 1 shows average CPU time *t* of simulation step and average iterations number *N* in sequential impulses method for two algorithms.

	SIMMC	SIMGC
Five-link chain	t = 0.5 ms N = 100	t = 0.1 ms N = 6
Peaucellier-Lipkin linkage	t = 16 ms N = 2300	t = 1.2 ms N = 40
Capture an object with a manipulator	t = 3 ms N = 200	t = 0.6 ms N = 10

Table 1 : CPU time and number of iterations

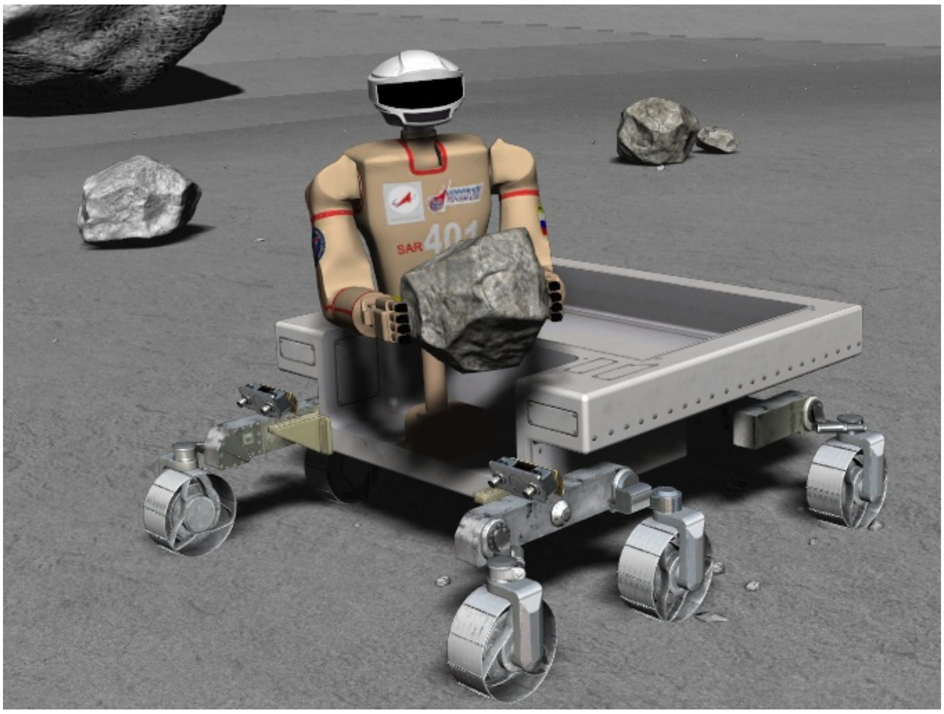

Figure 7: The anthropomorphic robot grasps a stone

The table shows that the SIMGC algorithm provides the necessary accuracy of computation in less time and requires less number of iterations, than the SIMMC algorithm. But iteration in the SIMGC is performed faster than in the SIMMC. The most significant benefit of the proposed algorithm was found on the virtual scene with Peaucellier–Lipkin linkage, where the SIMMC requires more than 10 ms for calculations. But to provide real time, the iterations of sequential impulses method must be limited to a maximum computation time $T = 5$ ms.

The simulation results show that the application of the articulated-body method and its modifications in the SIMGC algorithm can improve the quality and speed of simulation in comparison with the SIMMC algorithm for most of virtual scenes with multibody systems.

The algorithms for multibody dynamics simulation proposed in this paper have been implemented in the dynamic subsystem of robot simulator, developed in SRISA RAS. The Figure 7 shows the anthropomorphic robot that grasps a stone.

6 CONCLUSIONS

The SIMGC algorithm developed on the basis of the articulated-body method for multibody dynamics simulation is suitable for many tasks in virtual environment systems. However, the methods used in the work have some problems. These include the presence of a singular configuration in the simulation of a spherical joint (similar to a "gimbal lock"), the stability problem in case of a large simulation step, and so on. Therefore, further development of the proposed methods and algorithms for multibody dynamics simulation is expected. It includes the simulation of a spherical joint by quaternion and an implicit scheme for integrating the equations of motion with the representation of the problem in the form of differential algebraic equations [13].

The work was supported by the Russian Foundation for Basic Research, project No. 16-37-00107-mol_a.

REFERENCES

[1] M.V. Mikhaylyuk and E.V. Strashnov, “Simulation of articulated multibody system using sequential impulses method”, *Proceedings of SRISA RAS*, 4(2), 54-60 (2014).
[2] E.V. Strashnov and M.A. Torgashev, “Simulation of the actuator dynamics of the virtual robots in the training complexes”, *Mechatronika, Avtomatizatsiya, Upravlenie*, 17(11), 762-768 (2016).
[3] E.V. Strashnov and M.V. Michaylyuk, “Simulation of restrictions for the relative motion of the articulated rigid bodies in the virtual environment systems”, *Mechatronika, Avtomatizatsiya, Upravlenie*, 16(10), 678-685 (2015).
[4] S.-S. Kim, “A subsystem synthesis method for efficient vehicle multibody dynamics”, *Multibody System Dynamics*, 7, 189-207 (2002).
[5] J.I. Rodriguez, J.M. Jimenez, F.J. Funes and J.G. de Jalon, “Recursive and residual algorithms for the efficient numerical integration of multi-body systems”, *Multibody System Dynamics*, 11, 295-320 (2004).
[6] A.F. Vereshchagin, “Computer simulation method of complex robot manipulator mechanisms dynamics”, *Proceedings of the Academy of Sciences of the USSR. Technical cybernetics*, 6, 89-94 (1974).
[7] R. Featherstone, *Rigid body dynamics algorithms*, New Jork: Springer-Verlag, (2008).
[8] V.N. Ivanov, I.V. Dombrovskiy, F.V. Nabokov, N.A. Shevelev and V.A. Szymanowskiy, “Classification of multibody system models used in numerical calculations of dynamic behavior of machine-building constructions”, *Bulletin of the Udmurt University*, 2, 139-155 (2012).
[9] E. Kokkevis, “Practical physics for articulated characters”, *Proceedings of Game Developers Conference 2004* (2004).
[10] B. Mirtich, *Impulse-based dynamic simulation of rigid body systems*, PhD thesis, University of California, Berkeley (1996).

[11] E. Catto, “Iterative dynamics with temporal coherence”, *Proceedings of Game Developers Conference*, 1-24 (2002).
[12] A.M. Trushin, “Collision response of virtual objects using sequential impulses method”, *Proceedings of SRISA RAS*, 4(2), 95-105 (2014).
[13] S. Hadap, “Oriented strands: dynamics of stiff multi-body system”, In *SCA’06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 91-100 (2006).

The results were presented at the 16-th International seminar "Mathematical models &modeling in laser-plasma processes & advanced science technologies" (5 - 10 June, 2017, Petrovac, Montenegro).

Received March 5, 2017.