Impulse-based Approach for Rigid Body Collisions Simultaneous Resolution

Egor Ermolin, Alexey Kazakov[1]
[1]LEDAS Ltd
6 Prospekt Lavrentieva, Novosibirsk, Russia 630090
{hawk, lex}@ledas.com

Abstract

The paper proposes an enhanced version of the impulse-based method for numerical simulation of collisions between rigid bodies. This approach can resolve several collisions simultaneously, allows inelastic collisions with energy loss, and supports assembly constraints.

Keywords: *Physical simulation, collision resolution, move under constraints.*

1. INTRODUCTION

Simulation of the motion of a system of rigid bodies in accordance with the laws of mechanics has been studied in numerous recent papers; a good survey can be found in [1]. This is caused by the need to use this simulation for a variety of applications, from computer games to CAD/CAM/CAE systems.

Modeling the motion of such a system involves two important tasks: detection and resolution of collisions between rigid bodies. The term "collision detection" refers to determining the points of contact between rigid bodies, while "collision resolution" means calculating the action (the adjustment of the bodies' velocities) that is needed to ensure that they do not penetrate one another.

There are several well-researched approaches to collision detection that allow real-time, fine-mesh simulation of the dynamics of a system. This paper does not cover collision detection techniques, and the reader is referred to [1], where a detailed review of the existing methods is given. However, we will assume that we can use a collision detection method that provides us with all the necessary data on the colliding bodies, including the coordinates of contact points and the normal vectors to the boundaries at these points.

It should be noted that even the improved collision detection methods cannot provide precise information about the time or location of a collision. Collisions are usually determined for a snapshot of the system, rather than dynamically, and as a result the bodies may penetrate one another; these assumptions simplify the problem of collision detection, but at the same time undermine accuracy of calculation of collision point coordinates. As a consequence, no matter how good and accurate the collision resolution method, the subsequent behavior of the system will differ from reality.

We believe that some of the collision detection methods are particularly promising. In accordance with the approach proposed in [2, 3], the problem of simultaneous collision at n points can be reduce to the linear programming problem $A\vec{f} \geq \vec{b}, \vec{f} \geq 0$. The matrix A and the vector $\vec{b}$ are determined from the initial data of the problem. The unknown variables are the lengths of the force vectors f_i, which are oriented along the normals at the points of contact. A clear advantage of this method is that it takes into account all of the contact points simultaneously; nevertheless, a solution that ensures no penetration at each contact point and does not increase total energy of the system is sought with the help of an iterative technique, which limits the speed and accuracy of the computation.

The approach of [4], conversely, supplies the formulas that allow obtaining an exact solution, but only if the bodies have only one point of contact. We do not provide these formulas here and only note that this equation is fully identical to the equation produced by our approach in the case of a single contact point. For simultaneous contact of several bodies at several points, consecutive resolution of all contact points is proposed. In some cases, this approach produces acceptable results, but it is easy to provide an example where collisions at several points must be resolved simultaneously.

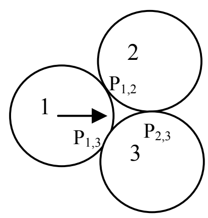

Figure 1: Simultaneous collisions

Fig. 1 shows three billiard balls of the same size and mass. Ball 1 has a certain velocity, while balls 2 and 3 are stationary. Suppose that we know the contact points $P_{1,2}$, $P_{1,3}$, and $P_{2,3}$. If the collisions are resolved consecutively, as proposed in [4] for example, if we resolve the contact point $P_{1,2}$ first, and then resolve $P_{1,3}$, then the magnitude of momentum of body 2 will be greater than the magnitude of the momentum of body 3, while the symmetry argument implies that they should be equal. Note that resolution of the collision at point $P_{2,3}$ is not necessary, because the bodies will not penetrate each other at this point. Finally, the author of [4] points out that if one attempts to analyze a system of four or more stacked boxes, then the collision resolution module almost freezes—so many momentum exchanges between the bodies are needed.

In this paper, we propose a method that synthesizes the ideas of [2, 3] and [4], i.e., takes into account all of the contact points and nevertheless finds the exact solution even in complex cases.

In the next section we provide the equations of motion for a rigid body with external forces, as well as the difference scheme that we use for simulation, and briefly describe the operation of the general motion simulation algorithm that takes into account collisions between the bodies. Next, in Section 3 we present certain assumptions that allow us to model the collisions and construct the principal equations for their calculation. In Section 4, we describe the application of our approach to simulation of assembly constraints (e.g., "point-point").

2. RIGID BODY DYNAMICS

The following equations describe the motion of a rigid body in the coordinate system connected with the center of mass of the body; the axes are parallel to the axes of the world coordinate system:

$$\begin{cases} \frac{dM\,\vec{v}_C}{dt} = \sum_i \vec{F}_i \\ \frac{dJ\,\vec{\omega}}{dt} = \sum \vec{r}_i \times \vec{F}_i \end{cases} \quad (1)$$

Here M is the mass of the body, J is the tensor of inertia of rank 2, represented by a symmetric matrix of dimension 3×3, $\vec{r}_i$ is the radius-vector of the point where the force $\vec{F}_i$ is applied, and $\vec{F}_i$ are the external forces. We assume that the position of the body is determined by the coordinates of the center of mass and the rotation matrix R that defines the transformation from the local coordinate system of the body to the world coordinate system. One should bear in mind that the tensor of inertia J is changed by a rotation of the body; if the current rotation matrix R is known for the body, then the new tensor of inertia is calculated from the following formula: $J^{'} = R^T JR$. For bodies of complex shapes, finding the tensor of inertia is often difficult. A method for finding the tensor was proposed in [5] for the case where the geometry of the body is defined by a mesh; we used this approach for testing.

For the equations of motion (1) we used the explicit first-order difference scheme.

$$\begin{cases} \frac{\vec{v}^{\,n+1} - \vec{v}^{\,n}}{dt} = \sum_i \vec{F}_i^{\,n} / M \\ \frac{\vec{\omega}^{\,n+1} - \vec{\omega}^{\,n}}{dt} = \sum_i J^{-1} \vec{r}_i^{\,n} \times \vec{F}_i^{\,n} \\ \vec{r}^{\,n+1} = \vec{r}^{\,n} + \vec{v}^{\,n+1} dt \\ R^{n+1} = Q(\vec{\omega}^{\,n+1} dt) R^n \end{cases} \quad (2)$$

Here the matrix $Q(\omega^{n+1}dt)$ is the matrix describing the rotation by $|\,\omega^{n+1}dt\,|$ around the axis $\vec{\omega}^{n+1}$. Recall that if the rotation angle is φ and the direction of the rotation axis is (n_1, n_2, n_3), then the rotation matrix looks as follows:

$$Q(\vec{n},\varphi) = \begin{pmatrix} c\varphi + (1-n_1^2)c\varphi & (1-c\varphi)n_1n_2 - n_3 s\varphi & (1-c\varphi)n_1n_3 + n_2 s\varphi \\ (1-c\varphi)n_1n_2 + n_3 s\varphi & c\varphi + (1-n_2^2)c\varphi & (1-c\varphi)n_2n_3 - n_1 s\varphi \\ (1-c\varphi)n_1n_3 - n_2 s\varphi & (1-c\varphi)n_2n_3 + n_1 s\varphi & c\varphi + (1-n_3^2)c\varphi \end{pmatrix}$$

where $c\varphi = \cos\varphi,\ s\varphi = \sin\varphi$.

To eliminate numerical error in some iterations, orthogonalization of the matrix R^{n+1} can be used.

Schemes with higher order of accuracy can be used to improve computation accuracy, but since we are working with non-smooth solutions, i.e., allow collisions between the bodies, using such schemes may be difficult.

Simulation of the motion of a system of rigid bodies is performed as follows: in each time step, we calculate the new coordinates of all bodies of the system with the help of the difference scheme (2). Next, we use some method of collision detection to find pairs of colliding bodies, if any, and resolve the collisions so as to avoid interpenetration of the bodies. The next section describes the method of collision resolution.

3. COLLISION RESOLUTION

3.1 Base assumptions

First, we list the assumptions that we make before resolving the collisions. These assumptions are common to many collision resolution approaches; a different formulation can be found in [2, 3] and [4]. We assume that at each point of contact there are forces that act on the bodies, and the forces have the following properties:

- the magnitude of a force is infinitely large,
- the time of action of the force is infinitely small,
- the impulse transferred to the system as a result is finite, and
- the direction of the vector of the force $\vec{F}$ at the collision point coincides with the direction of the normal vector $\vec{n}$ to the boundaries of the bodies at this point.

Thus, to resolve a collision we need to determine the forces acting at the points of contact between the bodies and calculate the new velocities of the bodies after the collision. We will rely on the equations of motion (1) and make the following assumptions that will allow us to determine the interaction forces:

- in a perfectly inelastic collision, the forces acting on the bodies will be such that the normal velocities of the bodies after the collision will be identical, i.e., the bodies will not penetrate one another;
- if the collision is not perfectly inelastic, then the process of collision can be divided into two phases, perfectly inelastic collision and restoration. It is assumed that the force acting on the bodies is $\vec{F}$, while in the second phase it is $e\vec{F}$, where e is the coefficient of restitution.

We will assume that the force at the point of contact is constant. This assumption allows us to simplify the equations of motion of the rigid body and use finite impulses $d\vec{p} = \vec{F}dt$.

3.2 Description of the method

Suppose that we have a set of bodies $B_1..B_q$. For each body, we know the current coordinates of its center of mass (x_i^C, y_i^C), the vector of velocity of the center of mass $\vec{v}_i^{\,0}$, angular velocity $\vec{\omega}_i^{\,0}$, and the inertia characteristics, mass M_i and tensor of inertia J_i. The following information is also assumed to be known:

- the list of collision points $P_1,...,P_n$; for each point P_k, its coordinates (x_k, y_k) and the direction of the normal vector $\vec{n}_k$ are known. The normal is assumed to be external to the body with index i_k and internal to the body with index j_k;
- the list of pairs of colliding bodies $S_1=(i_1,j_1),...S_n=(i_n,j_n)$, where i_k and j_k are the indexes of the bodies that collide at the point P_k.

In addition, the initial data of the problem include the coefficients of restitution e_k, for the collisions at the point P_k. Those pairs of bodies S_k for which $\vec{v}_r\vec{n}_k \geq 0$, where $\vec{v}_r$ is the relative velocity of the bodies at the contact point and $\vec{n}_k$ is the normal vector that is external to the first body in the pair, are eliminated from the examination, because at this points the bodies do not penetrate one another. Without loss of generality we can assume that the remaining points are the points $P_1...P_m$.

Now we introduce the interaction between the bodies at the contact points. Define the force of reaction F_k at each contact point P_k so as to transfer the impulse p_k over an infinitely small time interval: $\vec{F}_k = p_k\vec{n}_k / dt$.

A body B_s is affected only by those forces $\vec{F}_l$ for which there exists a pair of indices $S_l = (i_l, j_l)$ such that either $s = i_l$ or $s = j_l$. The direction of this force is for the first body (denoted by the first index in the pair) and for the second body (by Newton's Third Law of Motion). The magnitude of the impulse p_l is yet unknown. Under these assumptions, the velocities of the centers of mass and the angular velocities of the bodies after the collision will have the following form:

$$\vec{v}_k = \vec{v}_k^0 + \frac{\sum_{l=1}^{m}\text{sgn}_k(l)F_l}{M_k}dt = \vec{v}_k^0 + \frac{\sum_{l=1}^{m}\text{sgn}_k(l)p_l\vec{n}_l}{M_k}$$

$$\vec{\omega}_k = \vec{\omega}_k^0 + \sum_{l=1}^{m}\text{sgn}_k(l)J_k^{-1}(\vec{r}_{k,l} \times \vec{F}_l)dt = \tag{3}$$

$$\vec{\omega}_k^0 + \sum_{l=1}^{m}\text{sgn}_k(l)J_k^{-1}(\vec{r}_{k,l} \times p_l\vec{n}_l),$$

where the function sgn determines which element of the pair S_l is the body B_s:

$$\text{sgn}_s(l) = \begin{cases} 1, & \text{if } s = i_l \\ -1, & \text{if } s = j_l \\ 0 & \text{otherwise.} \end{cases}$$

Now, using the base assumption that the normal relative velocities of the bodies at the contact points coincide, we can write down the following system of equations:

$$(\vec{u}_{j_l,l} - \vec{u}_{i_l,l}) \cdot \vec{n}_l = 0, \qquad l = 1\ldots m \tag{4}$$

There are m equations in this system; each equation corresponds to one contact point; $\vec{u}_{j_l,l}$ is the velocity of the body B_{j_l} at the point P_l. This velocity is a function of the velocity of the center of mass of the body and the angular velocity of rotation,

$$\vec{u}_{j_l,l} = \vec{v}_{j_l} + \vec{\omega}_{j_l} \times \vec{r}_{j_l,l} \tag{5}$$

Using (6), we can express the velocities of the bodies at the contact points via the velocities of their centers of mass and angular velocities after the collisions, as defined by (3). Since the relationships (3) depend on p_l linearly, the relationships (4) are transformed into a system of linear equations with respect to p_l that has the form

$$Ap = b \tag{6}$$

where A is a matrix of dimension $m \times m$. The coefficients of the matrix have the following form:

$$a_{ik}^1 = \text{sgn}_{i_k}(k)\Big(\frac{\vec{n}_k}{M_{i_k}} + (J_{i_k}^{-1}(\vec{r}_{i_k,k} \times \vec{n}_k)) \times \vec{r}_{i_k,k}\Big) \cdot \vec{n}_i$$

$$a_{ik}^2 = \text{sgn}_{j_k}(k)\Big(\frac{\vec{n}_k}{M_{j_k}} + (J_{j_k}^{-1}(\vec{r}_{j_k,k} \times \vec{n}_k)) \times \vec{r}_{j_k,k}\Big) \cdot \vec{n}_i \tag{7}$$

$$a_{ik} = a_{ik}^1 - a_{ik}^2$$

Hence it follows that the equation for the contact point P_l will include all the forces $\vec{F}_k$ for which $\text{sgn}_{i_l}(k) \neq 0$ or $\text{sgn}_{j_l}(k) \neq 0$, i.e., all the forces acting on the bodies that collide at the point P_l. The components of the right-hand side vector b are represented as follows:

$$b_l = (\vec{v}_{i_l}^0 + \vec{\omega}_{i_l}^0 \times r_{i_l,l}) \cdot \vec{n}_l - (\vec{v}_{j_l}^0 + \vec{\omega}_{j_l}^0 \times r_{j_l,l}) \cdot \vec{n}_l \tag{8}$$

The proposed technique allows modeling collisions of moving bodies with fixed bodies. To this end, it suffices to assume the mass and components of the tensor of inertia of all the fixed bodies in the system infinitely large and, therefore, drop the terms containing infinitely large values in the denominator from the relationships (7).

Further, suppose that the values p_l are defined as the solution to the system (6), then the velocities of the bodies after a perfectly inelastic collision are given by the formulas (3). Recalling our assumption that the collision can be separated into two phases, one of perfectly inelastic collision and one of restoration, we conclude that, for inelastic collision at the point P_i with the coefficient of restitution e_i, the transferred impulse will be equal to $p_i' = (1 + e_i)p_i$; substituting the values of p_i' into the formulas (3), we obtain the velocities of the bodies after the inelastic collision.

Recall that we only examined a subset of the set of all collision points, dropping the points P_k such as $\vec{v}_r\vec{n}_k \geq 0$. Once we solve the system, we can guarantee that the bodies will not penetrate one another at the selected points, but we cannot assert the same about the rest of the points. Therefore, after the recalculation of velocities by the formulas (3), we can check again that the condition $\vec{v}_r \vec{n}_k \geq 0$ holds for all the points $P_1 ... P_n$, and if it is violated at any point, we form again the matrix for all these points. The process is continued until there are no points P_k with $\vec{v}_r \vec{n}_k \geq 0$.

A natural question arises: does the system of linear equations (6) have a solution? No complete answer to this question has been given yet; we can only say that the matrix of the system will probably be degenerate if the number of collision points is greater than what is needed to resolve the collisions correctly. For example, if three collision points are found in a collision of a line segment with another, longer segment, i.e., two points at the endpoints and one in the middle, then the matrix *A* will be degenerate and the result will be a one-dimensional space of solutions.

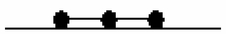

Figure 2: Overdetermined system

Nevertheless, there are methods, for example, SVD-decomposition (see [6]), that produce an acceptable solution even in this case. In the above example, the method would find a solution with the minimal norm—in the language of mechanics this means sharing the impulse transferred to the body between three points.

4. MODELING ASSEMBLY CONSTRAINTS

The approach described above allows modeling various connections between rigid bodies called assembly constraints. We will examine in detail just one type of connections, "point-point". A connection of this type assumes that two points in different rigid bodies will have identical coordinates during motion.

Suppose that the coordinates of the two connected points coincide at the initial time. For the connection not to be broken, it is necessary that the velocities of the two points coincide in each time step or, equivalently, that the relative velocity of the bodies at the point of connection equal zero. In the calculation of perfectly inelastic collision, through solving the system of equations (6) we were able to find adjustments to the velocities of the bodies such that the projection of relative velocity on the direction of the normal vector to the surface of one of the bodies equal zero. We can apply the same idea here: in each time step, along with the usual contact points we consider three additional ones with coordinates equal to the coordinates of the connection point, the normal vectors to the bodies are chosen to be perpendicular to one another, and the coefficient of restitution at these points is set to zero. By adding these points we ensure that the velocities of the connected points will coincide in each step.

However, this method of supporting connections is not accurate enough. Experiments show that for time step dt = 0.01, the difference between the coordinates of the connected points becomes unacceptable after 20 seconds of simulation; for smaller time steps, the difference of coordinates is not growing as fast, but for calculations with smaller time step efficiency is undermined.

To increase accuracy in the calculations of connections, the scheme proposed above can modified. We should determine the impulse transferred at the connection point not from the condition that the relative velocity is zero, but from equality of coordinates of the points after collision. Note that this approach does not require exact equality of coordinates at the previous time step, and it allows running calculations with greater time step.

The coefficients of the modified matrix in the lines corresponding to the usual collision points will not change, and for the additional connection points they will have the following form:

$$
\begin{aligned}
a_{ik}^{1} &= \operatorname{sgn}_{i_k}(k)\Big(\frac{\vec{n}_k}{M_{i_k}} + (J_{i_k}^{-1}(\vec{r}_{i_k,k} \times \vec{n}_k)) \times \vec{r}_{i_k,k}\Big) \cdot \vec{n}_i \\
a_{ik}^{2} &= \operatorname{sgn}_{j_k}(k)\Big(\frac{\vec{n}_k}{M_{j_k}} + (J_{j_k}^{-1}(\vec{r}_{j_k,k} \times \vec{n}_k)) \times \vec{r}_{j_k,k}\Big) \cdot \vec{n}_i \qquad (9) \\
a_{ik} &= (a_{ik}^{1} - a_{ik}^{2})dt
\end{aligned}
$$

The elements of the right-hand side that correspond to the additional points will undergo slightly greater changes. Now they will look as follows:

$$
\begin{aligned}
b_l^1 &= r_{i_l}^C + r_{i_l,l} + (\vec{v}_{i_l}^0 + \vec{\omega}_{i_l}^0 \times r_{i_l,l}) \cdot \vec{n}_l dt \\
b_l^2 &= r_{j_l}^C + r_{j_l,l} + (\vec{v}_{j_l}^0 + \vec{\omega}_{j_l}^0 \times r_{j_l,l}) \cdot \vec{n}_l dt \qquad (10) \\
b_l &= b_l^1 - b_l^2
\end{aligned}
$$

In this formula, $r_{i_l}^C$ and $r_{j_l}^C$ denote the coordinates of the centers of mass of the corresponding bodies B_{i_l} and B_{j_l}.

This approach can be used to support constraints of other types. For example, supporting coaxiality constraints requires introduction of two points with perpendicular normals that must be perpendicular to the axis.

5. NUMERICAL EXPERIMENTS

The present work has been performed in the frame of the first stage of the Physical Simulation project of LEDAS. In the first place this project is intended for a creation of an industrial physical simulation engine with a broad spectrum of applications including CAD/CAM/CAE systems. Initial engine implementation represents two-dimensional prototype for mechanical simulation of rigid bodies dynamics written in C++. The prototype supports geometrical representation for rigid bodies with boundary consisted of segments or circular arcs, models a motion of rigid bodies system under the influence of external forces, detects and resolves collisions, supports assembly constraints of "point-point" type and also takes sliding friction forces into account.

At present moment a full-scale three-dimensional engine is under intensive development. It is assumed that this product will have a flexible architecture and will represent a set of interacting replaceable modules. One of them is the Collision Resolution module that implements the algorithm described in the present work. By now the implementation of this module has been finished and below we provide performance information for this module.

For performance testing we used a set of scalable tests containing arbitrary number of collision points or "point-point" connections. The table below presents the measurements of time needed to compute adjustments of velocities of rigid bodies after inelastic collisions or adjustments preserving connections.

	Bodies	Collision points	Connections	Time (sec.)
Line	10	10	0	0.00085
	100	100	0	0.2415
Pyramid	6	9	0	0.00013
	10	18	0	0.00017
	55	135	0	0.0023
	210	570	0	6.501

Along with performance testing we have tested the Collision Resolution module for a compliance with physical laws. So, in simple cases for which an analytical solution is known (e. g. two balls collision) computational and theoretical results coincided with high precision. In complex cases we considered compliance with momentum conservation law and, in cases of absolute elastic collisions, energy conservation law as quality criterions. Testing results allow us to conclude that the proposed method also satisfies these criterions.

6. CONCLUSION

There has long been a need for a method for simultaneous collision resolution, to be used in simulation of various mechanical systems, be it game applications or engineering design. These methods certainly already exist, but an advantage of the approach proposed in this paper is its simplicity and efficiency, since collision resolution requires solving a system of linear equations with a symmetric sparse matrix.

Experiments show that simulation of perfectly elastic collision with this method preserves both the momentum and the energy of the system, while in an inelastic collision the energy loss agrees with the one produced by analytical calculations. In addition, the method can be used for systems with various assembly constraints, for example, with "point-point" constraints.

Validity of this method was confirmed in a large number of two-dimensional tests. Three-dimensional implementation has currently been completed and verified with tests.

It should be noted that this technique can handle rather complex situations, for example, with one body sliding over another. In this case, however, correct operation of the method depends strongly on the information returned by collision detection, in particular, on correct calculation of normal vectors at the points of contact between the bodies. In the future, we are planning to focus on developing a collision detection method that would handle these situations correctly.

References

[1] **O'Sullivan, C., Dingliana, J., Ganovelli, F., Bradshaw, G**. *"Collision Handling For Virtual Environments", – Image Synthesis Group, Trinity College, Dublin.*

[2] **Baraff D.** *"Analytical Methods for Dynamic Simulation of Non-penetrating Rigid Bodies", Computer Graphics (SIGGRAPH'89 Proceedings), volume 23, pages 223-232, July 1989.*

[3] **Baraff D.** *"Fast Contact Force Computation for Non-penetrating Rigid Body Simulation", Proceedings of SIGGRAPH'94, Computer Graphics Proceedings, Annual Conference Series, pages 23-34, ACM SIGGRAPH, ACM Press, July 1994.*

[4] **Mirtich B.** *"Impulse-Based Dynamic Simulation of Rigid Bodies" PhD Thesis, University of California, Berkeley, 1996.*

[5] **Mirtich B.** *"Fast and Accurate Computation of Polyhedral Mass Properties", University of California, Berkeley, 1996.*

[6] **Forsythe G., Malcolm M., Moler C.** *"Computer methods for mathematical computations", Englewood Cliffs, N.J., Prentice-Hall, 1977.*

About the authors

Egor Ermolin graduated from Novosibirsk State University in 2004 with specialization in computer science. He has been working at LEDAS company since 2003. Now he is a leading developer of the project of physical simulation. His contact email is hawk@ledas.com

Alexey Kazakov graduated from Novosibirsk State University in 2004 with specialization in applied mechanics. He has been working at LEDAS since 2001. Currently he is senior engineer in charge of developing of a new functionality in a geometrical solver project. He also participates in the project of physical simulation as an expert. His contact email is lex@ledas.com

LEDAS Ltd. is an independent software company founded in 1999 and located in Novosibirsk, Russia. LEDAS developed a state-of-the-art proprietary technology based on constraint programming, and applies it for PLM (Product Lifecycle Management) tasks, including CAD/CAM domain.