INTERACTIVE RIGID BODY MANIPULATION WITH OBSTACLE CONTACTS

Matthias Buck
Daimler-Benz AG
Research and Technology
P.O.Box 2360
D-89013 Ulm
Germany

phone: +49 731 505 2109
fax: +49 731 505 4224
email: m.buck@dbag.ulm.DaimlerBenz.COM

Elmar Schömer
Universität des Saarlandes
Lehrstuhl Prof. Hotz
Im Stadtwald
D-66123 Saarbrücken
Germany

phone: +49 681 302 3264
fax: +49 681 302 4812
email: schoemer@cs.uni-sb.de

ABSTRACT

The interactive manipulation of rigid objects in virtual reality environments requires an object behaviour which is at least physically plausible to be useful for applications like interactive assembly simulation or virtual training. Physically plausible behaviour implies that collisions between simulated solid objects are taken into account, and that the motion of objects with obstacle contacts can be controlled without force feedback mechanisms in an intuitively correct manner. We present a real-time framework which enables the simulation of interactively controlled solid objects with a dynamically changing set of contact constraints. In this paper, all contact configurations are replaced by a canonical set of point contacts, which is updated dynamically. The basic step to determine the contact forces and the object motion consists in the solution of a non-linear complementarity problem (NCP), which results from the unilateral contact conditions together with an adequate discretization of the corresponding differential equations of motion.

Keywords: Virtual reality, motion simulation, contact constraints, non-linear complementarity problem, NCP-functions, Newton iteration

1 INTRODUCTION

Certain virtual reality (VR) techniques have already been used for many years in industrial applications, such as in flight simulators or drive simulators. In these cases, the navigation in virtual worlds and the real time visualization of these worlds have been major issues, while interaction has not taken place in virtual but in real environments, i.e. in car or aircraft cabins with real instruments and real control devices.

Recently, new applications have been considered which require the user to directly interact with and to manipulate simulated objects within virtual worlds. Among these are ergonomy studies, digital mockup, assembly simulation, simulation of co-working, teleoperation, and training applications.

Such interaction in virtual environments requires more than just realistic visualization - it requires also realistic physical behaviour of the virtual objects. Realistic here not necessarily means physically correct behaviour, but at least physically plausible behaviour. For interactions in virtual worlds, this means in particular that solid objects do not interpenetrate if they collide, that there is friction if two touching objects slide along each other, and that objects have appropriate gravitational, mass and inertia properties. In our paper, we address the motion simulation of rigid bodies subject to frictionless contact constraints. This requires first a suitable mathematical representation of the contact situations between two or more touching bodies, together with a mechanism which automatically updates this representation if the contact situation changes during the simulation. This further requires a method to determine the motion of each involved moving object such that the contact conditions are enforced at each contact situation. As we have interactive applications in mind, all this has to be achieved under real-time conditions. Finally, input data from an external device has to be connected to the motion simulation to enable interactive motion control of virtual objects.

1.1 Previous Work

The simulation of the kinematics and dynamics of rigid bodies in the presence of contact constraints has been studied by several authors. Early papers are by [Loets82] with important contributions to the simulation of friction, by [Cai87], and by [Hahn88], who pioneered dynamic simulation and modeling of contacts. A thorough study on the dynamics of non-penetrating rigid bodies was presented by [Baraf92], who is one of the most active contributors to this field ([Baraf94, Baraf96]). Cremer, Stewart, and Vanecek [Creme89, Vanec94] use similar methods in their dynamic simulation systems *Newton* and *Isaac*.

The reported approaches generally set up the contact conditions for each contact in terms of the local *contact acceleration*. As long as the normal component of this acceleration remains zero, the contact persists. However, during numerical integration of the resulting motion equations, small numerical errors accumulate, which leads to a drifting problem: the contact condition is not enforced in terms of the *contact distance*, and the touching objects drift away from each other or even interpenetrate. As one of the main contributions of this paper, we will describe a method to remedy this problem.

While these approaches are based on the solution of continuous dynamic equations and contact forces, Mirtich and Canny [Mirti94] proposed an impulse-based method. Here any impact among contacting bodies is exchanged through trains of impulses.

The formulation of motion constraints of rigid bodies due to contact is discussed in a number of papers [Cai87, Monta88, Shan95, Baraf96]. Quite a number of formulations have been proposed to determine the dynamic behaviour of objects which are subject to motion constraints, as there are Newton-Euler, D'Alembert, or Gauss' principle of least constraint. In [Baraf96], the author states that ultimately all these approaches differ mainly in one basic point. Either constraints are modeled by reducing the number of coordinates which are necessary to describe the remaining degrees of freedom, or forces have to be introduced to maintain the constraints. A very good overview over the pros and cons of this choice is given in that paper.

The geometric representations of objects in VR and simulation systems in most cases are polyhedral surface descriptions. Such geometries yield simple contact constraint conditions, as long as the touching geometric elements (vertices, edges, faces) remain the same. If objects in contact are sliding along each other, contact conditions are changing discontinuously as the set of touching geometric elements is changing. In [Bouma93], the authors give a detailed analysis of collision contacts between polyhedral objects. In the case of face-face contacts, the area of touch is represented by a set of point contacts. If there are more than the 3 necessary contact points, they classify the remaining contacts as inactive. The set of 3 active contact points is determined through solving a quadratic programming problem.

Although collision detection mechanisms are necessary to detect new contacts, we do not treat this topic in our paper. We just want to mention here, that the main research direction in this field is to find methods which speed up the detection of object intersections, by using efficient data structures or by using suitable bounding volumes which allow fast collision tests. A good state-of-the-art paper is by Gottschalk et.al. [Gotts96].

1.2 Contributions of This Work

In this paper, we first discuss the representation of any contact situation by a minimal set of point contacts. This is the basis for a uniform mathematical treatment of the motion simulation with contact constraints. We further discuss how this minimal set of point contacts updates in the course of changing contact configurations.

Further we present a reformulation of the kinematic contact conditions in terms of contact distances, as opposed to the classical formulation in terms of contact accelerations. This approach allows to enforce the contact conditions without any drift problem due to error accumulation.

In the classical approach, the motion equations with contact constraints are formulated using a linear complementarity problem (LCP), which can be solved with the Lemke algorithm. In our case, we encounter a non-linear complementarity problem instead, which we propose to solve using so-called NCP-functions.

Finally, we will discuss the performance of our approach, and present practical applications and simulation results.

1.3 Contact Forces and Contact Impulses

In the simulation of object motion with contacts, two situations can be distinguished, in both of which the contacting objects have to be prevented from interpenetration. The first situation occurs at the time of collision, when the two objects are touching but have a non-zero normal velocity at the contact point. Generally, in this situation a **contact impulse** $\mathbf{p} = \Delta t \mathbf{f}$ is assumed to cause the instantaneous velocity change $\mathbf{p} = m\Delta \mathbf{v}$ necessary to prevent object intersection. The time of impact Δt here is generally assumed very short (however it will never be zero in reality as unbounded contact forces $\mathbf{f}$ do not exist).

The second situation concerns the case where a contact is already established, and the normal velocity at the point of contact is zero. Here normal **contact forces** are assumed to maintain the contact without interpenetration. The magnitude of such contact forces depends on the external or dynamic forces acting on the objects. In interactive applications however, the user interface provides desired object velocities, which have to be modified in the presence of contact constraints. So we handle contact situations similarly as collision situations, and take contact impulses here as well to prevent interpenetration.

In the context of time-discrete numerical simulation, the impact of an impulse is always distributed over an entire simulation time interval. In this sense, we do not have to distinguish between forces and impulses. Velocity changes are seen as being caused by forces $\mathbf{f}$ which act over the simulation interval Δt, which correponds to an impulse of $\mathbf{p} = \Delta t \mathbf{f}$.

2 MOTION CONSTRAINTS

2.1 Mechanical Constraints due to Obstacle Contacts

Contact configurations between polyhedra can be composed of a subset of 9 contact primitives, which correspond to the pairwise combinations of the 3 geometric primitives vertex, edge, and face. Contact primitives which are not point contacts by themselves can be replaced by an appropriate number of point contacts. To replace a face-face contact we need 3 point contacts, and for an edge-face contact we need 2. (In fact, in the physical world, there are no perfectly planar faces or perfectly straight edges, and on a microscopic level all contacts can be modeled as a composition of point contacts.) For our problem, the formulation of motion constraints, it is useful to determine a canonical set $\mathcal{S}$ of point contacts, that is a non-redundant (and thus minimal) set of point contacts which represents a given contact situation. The number k of point contacts in such a set equals the number of degrees of freedom (DOF) which are removed from the moving objects due to contacts. This set can be translated directly into a set of k scalar constraint equations for the motion simulation.

How can we efficiently obtain the set $\mathcal{S}$? It is not sufficient to replace for example each face-face contact with three point contacts, because this can lead to redundancies. Furthermore, $\mathcal{S}$ has to be updated each time the contact situation changes during the motion simulation, i.e. if new contacts are established or existing contacts vanish.

In the following, we present an algorithm which dynamically determines a canonical set $\mathcal{S}$ of point contacts during motion simulation. This requires to register new contact points as well as vanishing contact points.

2.2 Updating the Set of Point Contacts

To keep the set of point contacts $\mathcal{S}$ canonical, we must avoid to include any redundant contact point. Let $\mathcal{S}$ be canonical at the beginning of a given simulation interval. If during this interval the two objects start to intersect at some location, the collision detection mechanism (which we treat as a black box in this paper) reports one or several points of collision. Each point of collision may be either of the type vertex-face, face-vertex, or edge-edge. Furthermore, for each point of collision the time of collision is reported by the collision detection module. Now we are only interested in the earliest point of collision, because this is the location where both objects start to intersect, and where a new contact constraint should be placed. Accordingly, the simulated motion is stopped at the time instance t_c at which the intersection starts, and the corresponding new contact point is added to $\mathcal{S}$, which is still canonical. Subsequently, the motion simulation continues exactly where the current simulation step was interrupted by the collision.

Obviously the new contact point cannot be redundant, otherwise an intersection could not have been occurred at this location. Note that only the earliest contact point is guaranteed to be non-redundant, therefore it is important to add only this single new contact point to $\mathcal{S}$. If there are several simultaneous earliest collision points, just one of them is selected (according to some rule or arbitrarily) for the new contact point. This is however no severe restriction, as further contact points may follow shortly one after the other in subsequent simulation intervals, if necessary.

To find the earliest new contact point between the two involved objects, the collision detection module looks for all points of collision between these objects, and determines the corresponding collision times. The determination of the exact collision times and of the earliest point contact is not trivial, because the objects may move along complicated trajectories. In our present implementation, we are using linear interpolations to determine the collision times, which is appropriate as long as the rotational components of the specified motion during each timestep are sufficiently small.

The second mechanism required to update $\mathcal{S}$ is to determine breaking contact points. Contact points break either if attractive contact forces would be necessary to maintain an existing contact, or because a contact point moves outside a contact region (e.g. an object vertex slides along an obstacle face and leaves it at a convex edge). In both cases, the corresponding contact point is eliminated. Note that no redundancies can be introduced into $\mathcal{S}$ by a removal of contact constraints.

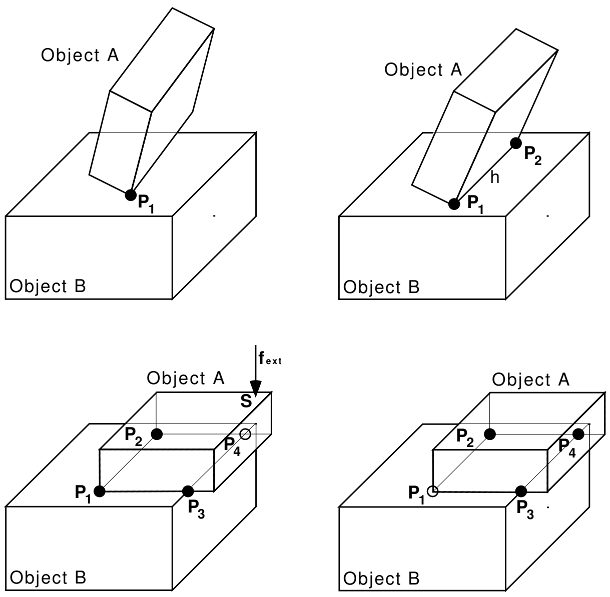

Figure 1: Several steps of a changing contact configuration (from top left to bottom right). Used contact points are marked by black dots.

The mechanism described above is illustrated in fig.1, where at the beginning object A has already one vertex-face contact with the fixed object B at point P_1. As object A rotates around P_1, a collision is detected at a second vertex-face contact, P_2. Now, object A may rotate around the axis h through $\overline{P_1P_2}$, until two edge-edge point collisions occur at P_3 and P_4. These two collisions virtually occur simultaneously, however one of them (say P_3) is selected according to some rule or arbitrarily as a new contact point. As long as the motion restrictions originated by the three contact points P_1, P_2, and P_3 are maintained, no interpenetration can occur at P_4, and thus no redundant contact point will be established there.

Now let an additional force f_{ext} be applied to object A at point S, the projection of which lies outside the triangle (P_1, P_2, P_3). This will cause object A to rotate slightly around axis $\overline{P_2P_3}$, and the point contact at P_1 will break. Further, a collision will occur at P_4, and a new point contact will be established there. Note that the contact point at P_4 now is non-redundant, because the contact constraint at P_1 no longer exists. It should be mentioned that only in this example the new contact point is close to the other contact points and even belongs to the same object faces. In general, new contact points can be established anywhere on the surfaces of the involved objects, if the earliest collision between them occurs there.

This example indicates how the canonical set of point contacts can change automatically in the course of the simulation as necessary to maintain the dynamic force and torque equilibrium.

2.3 Unilateral Motion Constraints

Contacts, in contrast to joints or hinges, represent unilateral constraints, as they restrict the local object motion only in one direction (the one which would lead to interpenetration). In the mathematical formulation, such unilateral constraints correspond to inequality conditions, in contrast to the equality conditions of bilateral constraints. This makes the simulation of contact constraints mathematically more complicated. Consider two objects which are touching without friction at a set of k point contacts, and with external forces and torques acting on them. The resulting motion of these objects depends on the external forces as well as on the contact forces. A contact force however can be only repulsive, otherwise the contact vanishes. The relation between contact forces and the kinematics of contact i is generally described by the following complementary set of conditions:

$$a_i \geq 0, \qquad f_i \geq 0, \qquad f_i a_i = 0 \tag{1}$$

which says that neither the normal acceleration a_i nor the normal contact force f_i may be negative, and at least one of both must be zero. If we collect all a_i and f_i in the respective vectors $\mathbf{a}$ and $\mathbf{f}$, we obtain

$$\mathbf{a} \geq \mathbf{0}, \qquad \mathbf{f} \geq \mathbf{0}, \qquad \mathbf{f}^T\mathbf{a} = 0 \tag{2}$$

Since $\mathbf{a}$ can be expressed as a linear function of $\mathbf{f}$ this leads to a linear complementarity problem (LCP) (see [Cottl68, Cottl92]) for the determination of $\mathbf{a}$ and $\mathbf{f}$. An efficient way to determine the solution of this LCP is the Lemke algorithm (see [Lemke65]).

As already mentioned, a disadvantage of this classical approach is that it constrains the normal contact accelerations a_i, and not directly the normal contact distances w_i, which leads to numerical error accumulation during the integration of the motion equations. To avoid this problem, we reformulate eq. 2 as

$$\mathbf{w}(\mathbf{f})^{t+\Delta t} \geq \mathbf{0}, \quad \mathbf{f} \geq \mathbf{0}, \qquad \mathbf{f}^T\mathbf{w}^{t+\Delta t} = 0 \tag{3}$$

where $\mathbf{w}(\mathbf{f})^{t+\Delta t}$ indicates that the contact distances at the end of each simulation time interval Δt are a function of the unknown contact forces $\mathbf{f}$. In contrast to the classical approach, this function however is non-linear in our case, and methods to solve an LCP like the Lemke algorithm cannot be used.

In the following section, we will derive this function $\mathbf{w}(\mathbf{f})^{t+\Delta t}$. In section 4, we will employ so-called NCP-functions to set up an equation system in the unknown forces $\mathbf{f}$, which can be solved efficiently with the Newton iteration. Once these forces are determined, it is trivial to derive the resulting object motion.

3 MOTION SIMULATION WITH CONSTRAINT FORCES

The motion of a rigid body subject to external forces is described by the Newton-Euler motion equations:

$$\dot{\mathbf{v}} = \frac{1}{m}\sum_{i=1}^{k}\mathbf{f}_i \tag{4}$$

$$\dot{\omega} = \mathbf{I}^{-1}\left(\sum_{i=1}^{k}\mathbf{r}_i \times \mathbf{f}_i - \omega \times \mathbf{I}\omega\right) \tag{5}$$

where $\mathbf{f}_i$ are the external forces (including contact forces), $\mathbf{r}_i$ are the vectors which point from the center of mass to the points where the forces apply, $\mathbf{I}$ denotes the inertia tensor, and m the object mass.

As we want to integrate these equations numerically, we switch from the differential formulation to a discrete one. Let us for the moment assume, that only the k constraint forces are present (other external forces can be added without difficulty), then we obtain

$$\mathbf{v}^{t+\Delta t} = \mathbf{v}^t + \Delta t\,\frac{1}{m}\sum_{i=1}^{k} f_i \mathbf{n}_i^t \tag{6}$$

$$\omega^{t+\Delta t} = \omega^t + \Delta t\,(\mathbf{I}^t)^{-1}\left(\sum_{i=1}^{k} f_i \mathbf{r}_i^t \times \mathbf{n}_i^t - \mathbf{J}_I^t\right) \tag{7}$$

where $\mathbf{J}_I^t = \omega^t \times \mathbf{I}^t\omega^t$ represents the gyroscopic forces. In (6) and (7), f_i are the force magnitudes and $\mathbf{n}_i$ are the force directions, which in the frictionless case are identical with the contact normals. The position of the moving object is given by the vector $\mathbf{c}$, and its orientation by the quaternion $\mathfrak{q}$. (See appendix A for details on quaternions).

Another integration step yields the position and orientation of the moving object:

$$\mathbf{c}^{t+\Delta t} = \mathbf{c}^t + \Delta t\,\mathbf{v}^{t+\Delta t} \tag{8}$$

$$\mathfrak{q}^{t+\Delta t} = \mathfrak{q}^t + \frac{1}{2}\Delta t\,(0,\,\omega^{t+\Delta t})\cdot \mathfrak{q}^t \tag{9}$$

where $(0,\,\omega^{t+\Delta t})$ and $\mathfrak{q}$ are quaternions, and the dot represents the quaternion product (see appendix A).

Note that for (6) and (7) we choose forward differentiation, and for (8) and (9) backward differentiation. This allows to plug (6) into (8) and (7) into (9), which results in an equation system for the position $\mathbf{c}^{t+\Delta t}$ and orientation $\mathfrak{q}^{t+\Delta t}$ of the moving object at the end of the simulation interval, with the contact forces $f_1,\ldots,f_k$ as unknowns.

In the following we want to derive the contact distances w_i at each contact as a function of $\mathbf{c}^{t+\Delta t}$ and $\mathfrak{q}^{t+\Delta t}$. To this end, we first rewrite (9) by replacing the quaternions by their equivalent rotation matrices:

$$\mathbf{R}(\mathfrak{q}^{t+\Delta t}) = \mathbf{R}(1,\,\frac{1}{2}\Delta t\,\omega^{t+\Delta t})\cdot\mathbf{R}(\mathfrak{q}^t) \tag{10}$$

This allows us to express the position $\mathbf{p}^{t+\Delta t}$ of a general point P of a moving object in terms of its coordinates $\hat{\mathbf{p}}$ in an object fixed coordinate system:

$$\mathbf{p}^{t+\Delta t} = \mathbf{R}(\mathfrak{q}^{t+\Delta t})\hat{\mathbf{p}} + \mathbf{c}^{t+\Delta t}$$

3.1 The Contact Distances and Contact Normals

As described in section 2, all contact configurations between polyhedra can be represented by a set of the two basic point contacts vertex-face and edge-edge. In this section we give the contact distances and contact normals for these contacts as a function of the global motion parameters $\mathbf{R}_i$ and $\mathbf{c}_i, i = 1,2$ of the two touching objects.

Vertex-face contact: Let $\mathbf{a}$ be a vertex of object 1 which is in contact with a face of object 2. Assume that this face lies in the plane with equation $\mathbf{n}^T\mathbf{x} = n_0$. We describe the motion of the vertex and the plane as follows:

$$\mathbf{a} = \mathbf{R}_1\hat{\mathbf{a}} + \mathbf{c}_1, \qquad \mathbf{n} = \mathbf{R}_2\hat{\mathbf{n}}, \qquad n_0 = \hat{n}_0 + \mathbf{c}_2^T\mathbf{n},$$

where $\mathbf{R}_i$ denotes the current orientation matrix of object i, and $\mathbf{c}_i$ the current position of its center of mass. $\hat{\mathbf{a}}$, $\hat{\mathbf{n}}$ and $\hat{n}_0$ describe the position of the vertex and the plane in the objects' fixed coordinate systems.

The contact normal of a vertex-face contact is given by the face normal $\mathbf{n}$ and the contact point lies at $\mathbf{a}$. The contact distance is given by:

$$w = \hat{\mathbf{n}}^T\mathbf{R}_2^T(\mathbf{R}_1\hat{\mathbf{a}} + \mathbf{c}_1 - \mathbf{c}_2) - \hat{n}_0 \tag{11}$$

Edge-edge contact: Let $\mathbf{a}_1$ and $\mathbf{b}_1$ be the endpoints of an edge of object 1 which is in contact with an edge of object 2 with endpoints $\mathbf{a}_2$ and $\mathbf{b}_2$. We describe the motion of the endpoints as $\mathbf{a}_i = \mathbf{R}_i\hat{\mathbf{a}}_i + \mathbf{c}_i$ and $\mathbf{b}_i = \mathbf{R}_i\hat{\mathbf{b}}_i + \mathbf{c}_i$. The contact normal in this case is given by $\mathbf{n} = (\mathbf{b}_1 - \mathbf{a}_1)\times(\mathbf{b}_2 - \mathbf{a}_2)$. Without a detailed derivation, we give the contact distance for this case as:

$$w = \frac{(\mathbf{b}_2 - \mathbf{a}_2)^T(\mathbf{a}_1 \times \mathbf{b}_1)}{|\mathbf{n}|} + \frac{(\mathbf{a}_2 \times \mathbf{b}_2)^T(\mathbf{b}_1 - \mathbf{a}_1)}{|\mathbf{n}|} \tag{12}$$

Finally, we can express the contact distances $w_i^{t+\Delta t}$, $i = 1...k$ at the k contact points as functions of the normal contact forces $f_1,\ldots,f_k$, if we plug eqs. (8, 10) into (11) respectively (12). We refer to this function in the following using the vector notation

$$\mathbf{w} = \mathbf{g}(\mathbf{f}) \tag{13}$$

4 SOLVING THE NON-LINEAR COMPLEMENTARITY PROBLEM

In the previous section, we derived the contact distances $\mathbf{w} = \mathbf{g}(\mathbf{f}), \mathbf{g} : \mathbb{R}^k \to \mathbb{R}^k$. Together with the complementarity conditions

$$\mathbf{w}(\mathbf{f}) \geq \mathbf{0}, \quad \mathbf{f} \geq \mathbf{0}, \qquad \mathbf{f}^T\mathbf{w} = 0$$

we obtain a non-linear complementarity problem (NCP) for the determination of $\mathbf{f}$. The mathematical literature proposes for the solution of this problem the application of so-called NCP-functions ([Kanzo96, Kanzo97]). These functions transform the NCP into a non-linear equation system, which can be solved with standard methods like the Newton iteration .

The class of NCP-functions $\varphi(a,b) : \mathbb{R}^2 \to \mathbb{R}$ is defined by the property

$$\varphi(a,b) = 0 \iff a \geq 0,\, b \geq 0,\, ab = 0 \qquad (14)$$

Particularly interesting properties have been reported for the following NCP-function, which is also called Fischer function ([Fisch92]):

$$\varphi(a,b) \;=\; \sqrt{a^2+b^2} - a - b \qquad (15)$$

With this auxiliary function, we define the operator $\mathbf{F}_\varphi : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ as follows:

$$\mathbf{F}_\varphi(\mathbf{f},\mathbf{w}) \;=\; \begin{pmatrix} \mathbf{g}(\mathbf{f}) - \mathbf{w} \\ \phi(\mathbf{f},\mathbf{w}) \end{pmatrix}. \qquad (16)$$

where

$$\phi(\mathbf{f},\mathbf{w}) \;=\; (\varphi(f_1,w_1),\ldots,\varphi(f_n,w_n))^T$$

A set of $(\mathbf{f},\mathbf{w})$ in consequence solves the above NCP exactly if $\mathbf{F}_\varphi(\mathbf{f},\mathbf{w}) = \mathbf{0}$ holds.

The equation system $\mathbf{F}_\varphi(\mathbf{f},\mathbf{w}) = \mathbf{0}$ has the degree $2n$. As pointed out in [Kanzo96], the solution of this system can be reduced to the solution of an equivalent equation system of degree n, as the n components of $\mathbf{w}$ basically don't represent independent variables.

With the Jacobian

$$\frac{d\mathbf{F}_\varphi}{d(\mathbf{f},\mathbf{w})}(\mathbf{f}^j,\mathbf{w}^j) \;=\; \begin{bmatrix} \frac{d\mathbf{g}}{d\mathbf{f}}(\mathbf{f}^j) & -\mathbf{I} \\ \frac{d\phi}{d\mathbf{f}}(\mathbf{f}^j,\mathbf{w}^j) & \frac{d\phi}{d\mathbf{w}}(\mathbf{f}^j,\mathbf{w}^j) \end{bmatrix}$$

we can transform the Newton-approach

$$\frac{d\mathbf{F}_\varphi}{d(\mathbf{f},\mathbf{w})}(\mathbf{f}^j,\mathbf{w}^j)\begin{pmatrix} \Delta\mathbf{f}^j \\ \Delta\mathbf{w}^j \end{pmatrix} \;=\; -\mathbf{F}_\varphi(\mathbf{f}^j,\mathbf{w}^j)$$

into the system

$$\left(\frac{d\phi}{d\mathbf{f}}(\mathbf{f}^j,\mathbf{w}^j) + \frac{d\phi}{d\mathbf{w}}(\mathbf{f}^j,\mathbf{w}^j)\frac{d\mathbf{g}}{d\mathbf{f}}(\mathbf{f}^j)\right)\Delta\mathbf{f}^j$$
$$= \; -\phi(\mathbf{f}^j,\mathbf{w}^j) - \frac{d\phi}{d\mathbf{w}}(\mathbf{f}^j,\mathbf{w}^j)(\mathbf{g}(\mathbf{f}^j) - \mathbf{w}^j)$$

To determine $\Delta\mathbf{f}^j$ we have to solve a linear equation system of degree n. Subsequently, $\Delta\mathbf{w}^j$ follows by direct substitution:

$$\Delta\mathbf{w}^j \;=\; \frac{d\mathbf{g}}{d\mathbf{f}}(\mathbf{f}^j)\Delta\mathbf{f}^j + \mathbf{g}(\mathbf{f}^j) - \mathbf{w}^j$$

The iteration

$$\mathbf{f}^{j+1} \;=\; \mathbf{f}^j + \Delta\mathbf{f}^j$$
$$\mathbf{w}^{j+1} \;=\; \mathbf{w}^j + \Delta\mathbf{w}^j$$

continues until $|\mathbf{F}_\varphi(\mathbf{f}^j,\mathbf{w}^j)|$ falls below a threshold value.

This Newton-type method converges quadratically, and we generally have good estimates for $(\mathbf{f}^0,\mathbf{w}^0)$ from the previous simulation interval, such that only few iterations are necessary. Experimental results are given is section 5.

4.1 Extension to Multibody Systems

In the previous sections we assumed that only contacts between two rigid objects have to be taken into account. In many practical applications however this is not true. Imagine for example that one interactively controlled object is a kind of tool which is used to manipulate several other moveable objects.

If more than two objects are in contact with each other, their motion cannot be simulated independently, as they are mutually constraining their relative motion. Instead, they have to be simulated as one articulated combined object. A cluster of n connected movable objects can have up to $6n$ non-redundant point contacts, which corresponds to a system of $1 \leq k \leq 6n$ scalar motion constraints.

The mathematics for the motion simulation of such a cluster of touching objects is basically the same as described in the previous sections. For each object, there is a set of motion equations (4) ... (10), and for each point contact there is an entry in the set of complementarity conditions, eq. (3).

The numerical effort to determine all k contact forces is dominated by the solution of linear equation systems. Since this step has complexity $O(k^3)$ great values of k are inhibitive for real-time solutions. So far, we limited ourselves to applications with few moveable objects, which can be handled in real-time as indicated in section 5.

5 APPLICATIONS AND EXPERIMENTAL RESULTS

The methods described in the previous sections have been tested in different interactive applications. To enable the interactive control of one of the virtual objects, we translated input data from a spaceball or a data glove into a virtual force and torque. These external forces and torques have been applied to the selected object by simply adding these values to the motion equations (4, 5). A gravitational effect is easily included by just adding a negative vertical acceleration to equation (4).

The first example we want to present consists of two moveable objects, one of which is interactively controlled, and a fixed horizontal ground plane with

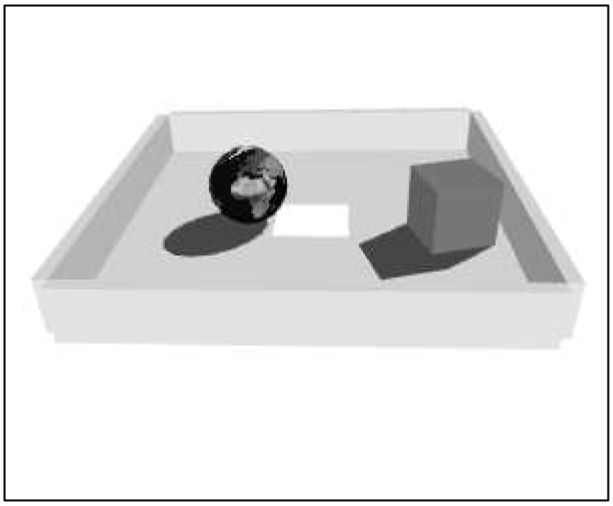

Figure 2: Virtual scene with two movable objects, one of which is controlled interactively, and a static obstacle.

bounds and a hole (see fig.2). The interactively controlled object can be moved around, and in case of mechanical contacts behaves intuitively correct, i.e. the touching objects do not interpenetrate, but slide along each other. The second moveable object can be pushed around when touched by the first one.

The second example, illustrated in fig.3, is an assembly simulation, where the user has to insert a virtual car radio into the console of the car, using a data glove. In this example, the simulation of sliding contacts is essential in order to perform the task in a realistic and intuitive manner.

Average computing times on an SGI O2^{TM} (R10000, 150MHz) for the iterative solution of the equation system for different numbers of contacts are given in the following table:

contacts	1	2	3	5	10
time (ms)	0.3	0.4	0.6	2.1	11.5

For a visualisation frame rate of 30 Hz, 33 ms are available for the motion simulation. In this application, we are far below that limit. So the computing times for the motion simulation with up to 10 point contacts are clearly adequate for real-time solutions.

6 SUMMARY AND FURTHER RESEARCH

In this paper, we presented a real-time framework which enables the simulation of interactively controlled solid objects with contact constraints. Due to the chosen physically oriented simulation model with contact forces, the resulting object motion in case of mechanical contacts is realistic and intuitively correct. This is an important prerequisite for practical applications like assembly simulations or virtual training. The computing times for the motion simulation with up to 10 point contacts are clearly adequate

Figure 3: Ergonomy study and assembly simulation in a virtual car interior. The user has to install the car radio.

for real-time solutions. The framework scales with any number of involved objects, however for large contact clusters with many mutual contacts the simulation times will go up significantly.

So far, only frictionless contacts have been taken into account. As friction is an important property of real environments, a planned direction for future extentions is to add this feature to the simulation system. As our contact model is based on point contacts, and contact forces are already implemented, it should be possible to introduce friction forces to the presented framework.

If the complementarity condition eq. (3) for unilateral motion constraints is replaced by the contact condition $\mathbf{w}(\mathbf{f})^{t+\Delta t} = \mathbf{0}$, bilateral constraints (like joints or hinges) can be included as well.

Another potential field of improvements concerns the mechanical user interface. For a perfectly realistic user interaction with virtual environments, force feedback mechanisms would be necessary. Otherwise, even though the motion behaviour within the simulated environment is correct, the user does not feel the reaction forces, and mainly depends on the visual feedback, which is a significant limitation. Nevertheless, as long as practical force feedback devices are not available, the described techniques allow a satisfactory solution for interactive manipulation tasks in virtual environments.

APPENDIX A: DESCRIPTION OF ROTATIONS AS QUATERNIONS

A quaternion $\mathbf{q} = (q_0, \mathbf{q}) \in \mathbb{R}^4$ is composed of a vector component and a scalar, which together describe the orientation or rotation of an object by the rotation axis $\mathbf{r} \in \mathrm{IR}^3$, $|\mathbf{r}| = 1$, and the rotation angle φ as follows:

$$\mathfrak{q}_{\mathbf{r},\varphi} = \left(\cos\frac{\varphi}{2}, \sin\frac{\varphi}{2}\mathbf{r}\right)$$

The rotation matrix which corresponds to the quaternion $\mathfrak{q} = (q_0, \mathbf{q})$ is given by

$$\mathbf{R}(\mathfrak{q}) \quad = \quad \frac{(q_0^2 - \mathbf{q}^2)\mathbf{E} + 2\mathbf{q}\mathbf{q}^T + 2q_0\mathbf{q}^\times}{q_0^2 + \mathbf{q}^2} \qquad (17)$$

The concatenation of rotations is determined by the multiplication of the corresponding quaternions $R(\mathfrak{a}) \cdot R(\mathfrak{b}) = R(\mathfrak{a} \cdot \mathfrak{b})$, where the quaternion product $\mathfrak{a} \cdot \mathfrak{b}$ is defined as

$$(a_0, \mathbf{a}) \cdot (b_0, \mathbf{b}) \quad = \quad \left(a_0 b_0 - \mathbf{a}^T\mathbf{b}, a_0\mathbf{b} + b_0\mathbf{a} + \mathbf{a} \times \mathbf{b}\right)$$

For further details on quaternion derivation and arithmetics, refer to [Glass90].

References

[Baraf92] D. Baraff: *Dynamic simulation of non-penetrating rigid bodies*, Ph.D. Thesis 92-1275, Cornell University, March 1992

[Baraf94] D. Baraff: *Fast contact force computation for nonpenetrating rigid bodies*, SIGGRAPH 94, Orlando, July 1994

[Baraf96] D. Baraff: *Linear-time dynamics using Lagrange multipliers*, Technical Report CMU-RI-TR-95-44, Carnegie Mellon University, January 1996

[Bouma93] W. Bouma, G. Vaněček: *Contact Analysis in a Physically Based Simulation*, ACM Symposium on solid Modeling and Applications, Montreal, Canada, May 1993

[Cai87] C. Cai, B. Roth: *On the spatial motion of a rigid body with point contact*, IEEE International Conference of Robotics and Automation, pp.686-695, 1987

[Carr95] K. Carr, R. England (edts): *Simulated and Virtual Realities - Elements of Perception*, Taylor & Francis, 1995

[Cottl68] R.W. Cottle, G.B. Danzig: *Complementarity pivot theory of mathematical programming*, Linear Algebra and its Applications, **1**:103-125, 1968

[Cottl92] R.W. Cottle, J.S. Pang, and R.E. Stone: *The Linear Complementarity Problem*, Academic Press, 1992

[Creme89] J.F. Cremer, A.J. Stewart: *The Architecture of Newton, a general-purpose dynamic simulator*, IEEE International Conference of Robotics and Automation, pp.1806-1811, 1989

[Fisch92] A. Fischer: *A special Newton-type optimization method*, Optimization 24, pp. 269-284, 1992

[Glass90] P.-G. Maillot: *Using quaternions for coding 3d transformations*, Graphic Gems (A.S. Glassner, edt.) pp. 498-515, Academic Press, Boston, 1990

[Gotts96] S. Gottschalk, M.C. Lin, D. Manocha: *OBBTree: A Hierarchical Structure for Rapid Interference Detection*, SIGGRAPH 96, Computer Graphics Proceedings, pp. 171-179, August 1996

[Hahn88] J.K. Hahn: *Realistic animation of rigid bodies*, Computer Graphics 22(4):299-308, August 1988

[Kanzo96] C. Kanzow: *Global Convergence Properties of some Iterative Methods for Linear Complementarity Problems*, SIAM Journal of Optimization, Vol.6, No.2, pp.326-341, May 1996

[Kanzo97] C. Kanzow, H. Kleinmichel: *A New Class of Semismooth Newton-Type Methods for Nonlinear Complementarity Problems*, Manuskript, Institut für angewandte Mathematik, Universität Hamburg, Januar 1997

[Lemke65] C.E. Lemke: *Bimatrix equilibrium points and mathematical programming*, Management Science **11**:681-689, 1965

[Loets82] P. Lötstedt: *Mechanical systems of rigid bodies subject to unilateral constraints*, SIAM Journal of Applied Mathematics, 42(2):281-296, 1982

[Monta88] D.J. Montana: *The kinematics of contact and grasp*, The International Journal of Robotics Research, Vol.7, No.3, June 1988

[Mirti94] B. Mirtich, J. Canny: *Impulse-based dynamic simulation*, in K. Goldberg, D. Halperin, J.C. Latombe, and R. Wilson (edts.) The algorithmic foundations of robotics. A.K. Peters, Boston, MA, 1995

[Shan95] Y. Shan, Y. Koren: *Obstacle accomodation motion planning*, IEEE Transactions on Robotics and Automation, Vol.11, No.1, Febr. 1995

[Vanec94] G. Vanecek, Jr., J.F. Cremer, *Project Isaac: Building Simulations for virtual environments*, Technical Report, Purdue University, June 1994