Proceedings of the 2002 IEEE
International Conference on Robotics & Automation
Washington, DC • May 2002

Gauss' Least Constraints Principle and Rigid Body Simulations

Stéphane Redon, Abderrahmane Kheddar, Sabine Coquillart[1] [2] [3]

Abstract

Most of well-known approaches for rigid body simulations are formulated in the contact-space. Thanks to Gauss' principle of least constraints, the frictionless dynamics problems are formulated in a motion-space. While the two formulations are mathematically equivalent, they are not computationally equivalent. The motion-space formulation is better conditionned, always sparse, needs less memory, and avoids some unnecessary computations. A preliminary experimental comparison suggests that an algorithm operating in the motion-space takes advantage of sparsity to perform increasingly better than a contact-space algorithm as the average number of contact points per object increases.

1 Introduction

Rigid body simulations have numerous offline or online applications: virtual environments, virtual prototyping, teleoperation, assembly tasks, interactive tolerance tests, video games. Many situations require the simulation of classical physics laws. This explains why rigid body simulation is such an active research field.

While moderately complex objects can be handled in real-time by the known simulation systems[4], complex objects may require many computations as the number of contact points increases, and the strategy is often to *display* complex objects while computing the simulation on simplified objects[9]. We believe that the reason for this is that the simulation systems do not make explicit use of the number of degrees of freedom (dof) in the simulation[1]. Thus, as the average number of contact points per object increases, the simulators are not as efficient as they could be.

Note that the average number of contact points per object can increase dramatically as the objects complexity increases. Consider two contacting cubes. If the cubes are almost aligned and one cube is slightly rotated, then eight contacts occur. Now, if the cubes are replaced by n-sided polyhedral cylinders in a similar configuration, then $2n$ *contacts occur.* Other examples can easily be found: threaded screws insertion, tightly-packed objects, object clusters. Also, the fact that tolerance values are used in collision detection increases the number of contact points.

[1] S. Redon and S. Coquillart are from INRIA-Rocquencourt - FRANCE.
[2] A. Kheddar is from Université d'Evry CEMIF-SC - FRANCE
[3] Contact author : stephane.redon@inria.fr
[1] Of course, the collision detection is also slowing down the system.

To our knowledge, most of recently proposed (constraint-based) solutions able to handle unilateral constraints formulate the dynamics problems (*ie* the *constrained motion problem* and the *collision resolution problem*) as a linear complementarity problem (LCP) (see for example [2][4][17][18][19]). The LCP relates contact forces and accelerations or velocities. For example, in a frictionless system, if $\mathbf{f}$ and $\mathbf{a}_{cp}$ are two vectors in $\mathbf{R}^m$ describing the normal contact forces and the relative normal accelerations at the m contact points, then there exists a $m \times m$ matrix $\mathbf{A}$ and a vector $\mathbf{b}$ in $\mathbf{R}^m$ such as:

$$\mathbf{a}_{cp} = \mathbf{A}\mathbf{f} + \mathbf{b},\ \mathbf{a}_{cp} \geqslant 0,\ \mathbf{f} \geqslant 0,\ \mathbf{a}_{cp}^T \mathbf{f} = 0 \qquad (1)$$

As a result, the number of degrees of freedom of the unconstrained system is not explicit. In other words, problem (1) is a *contact-space* formulation[17].

This paper uses Gauss' principle of least constraints to formulate both dynamics problems in the frictionless case in a *motion-space*. In the motion-space, an algorithm solving the dynamics problems is able to make explicit use of the number of dof in the simulation and avoids unnecessary computations. Note that some authors have presented formulations that explicitly contain the number of dof in the system. Lötstedt[13] uses a formulation closely related to Gauss' principle, but doesn't handle elastic collisions. Baraff[6] presents a linear-time algorithm for acyclic articuled bodies, but then uses a contact-space formulation[4] to handle unilateral contacts. Milenkovic[14], too, makes explicit use of the number of dof by formulating the problem as a quadratic programming problem, but requires many variables to enforce all the dynamics/collision conditions.

This paper is organized as follows. Gauss' principle is recalled in Section 2. A motion-space formulation of the frictionless dynamics problems is derived from Gauss' principle in Section 3 to form similar minimization problems. Section 4 reduces the minimization problems to a well known nearest-point problem (NPP) which will be our motion-based formulation. Section 5 compares both formulations from a theoretical and practical point of view. It shows that while the formulations are mathematically equivalent, they are not *computationally* equivalent: the motion-space formulation is better conditionned, is always sparse, requires less memory to store the data and avoids some unnecessary computations. An experimental comparison is made between two typical algorithms that solve the LCP problem and the NPP problem. A preliminary comparison suggests that, thanks to a sparser formulation, the algorithm operating in the motion-space (*ie* solving the NPP problem) is able to perform increasingly better than the algorithm operating in the contact-space, as the average

0-7803-7272-7/02/$17.00 © 2002 IEEE

number of contact points per object increases. Section 6 concludes.

2 Gauss' least constraints principle

Few references about applications of Gauss' principle of least constraints[10] can be found. However, it has recently prompted a renewed interest[7][20]. Especially, Gauss' principle has been used in the special case of initially motionless objects[5] and, in robotics, to compute the dynamics of redundant manipulators[8]. Udwadia[20] shows that Gauss' principle leads to a closed-form equation of motion when all the constraints are bilateral. An immediate advantage of Gauss' principle over the principle of virtual work, used in LCP methods, is that it allows to give a very intuitive formulation of the motion of a constrained system. Actually, it is so intuitive that it is often rediscovered and/or used without mentioning Gauss ([14]).

Though it was initially expressed for a set of point masses, it applies also for a frictionless system of rigid bodies subject to geometrical (unilateral or bilateral) constraints[7][12], and can be simply expressed thanks to generalized accelerations and masses. Let's consider a *contact group*[2] of n mobile objects. A rigid body i has only six degrees of freedom, and its acceleration is split into a *translational* term $\mathbf{a}_i(G_i)$, which is the acceleration of the object's center of mass, and a *rotational* term α_i. These two vectors are in $\mathbf{R}^3$. The accelerations of the n (potentially) mobile objects can be stacked in a single $6n$-dimensional generalized acceleration vector $\mathbf{a}$, while the masses and (time-dependant) inertia tensors can be stacked in a single $6n \times 6n$ generalized mass matrix $\mathbf{M}$. This matrix is block-diagonal and is symmetric positive definite (SPD).

Now, let $\mathbf{a}_u$ denote the (generalized) unconstrained acceleration of the contact group, that is, the accelerations the contact group would have were it not subject to geometrical constraints[3]. $\mathbf{a}_u$ is in $\mathbf{R}^{6n}$. Gauss' principle states that the generalized *constrained* acceleration $\mathbf{a}_c$ minimizes the following scalar function of $\mathbf{a}$ over the set of *possible accelerations*:

$$G_a(\mathbf{a}) = \frac{1}{2}(\mathbf{a}-\mathbf{a}_u)^T\mathbf{M}(\mathbf{a}-\mathbf{a}_u) = \frac{1}{2}||\mathbf{a}-\mathbf{a}_u||^2_M \qquad (2)$$

The possible accelerations are the accelerations which are compatible with the current contact group's configuration. Note that since $\mathbf{M}$ is SPD, $||.||_M$ is a well-defined non-euclidean norm. By analogy with the kinetic energy $E = \frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v}$ of the system, this norm is usually called *kinetic norm*. Thus, Gauss' principle amounts to minimize the kinetic distance between the generalized accelerations $\mathbf{a}$ and $\mathbf{a}_u$, over the set of possible accelerations[4]. Making implicit the fact that the distance is the kinetic one, Gauss' principle can be stated even more simply: *at any moment, a contact group's constrained acceleration is the closest possible acceleration to its unconstrained acceleration.*

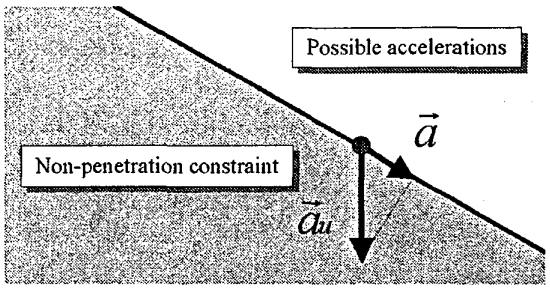

Figure 1: *Gauss' principle allows to find very easily the particle's constrained acceleration.* The particle's unconstrained acceleration $\mathbf{a}_u$ is the gravity $\mathbf{g}$. The possible accelerations are given by the *non-penetration constraint* due to the slope. The particle's constrained acceleration is the closest possible acceleration to its unconstrained acceleration.

In this formulation, the explicit unknowns are the objects' accelerations, and the contact forces are implicit: the problem is stated in the *motion-space*, and not in the contact-space, as in LCP methods[17]. Thus, the number of dof ($6n$) of the unconstrained contact group *is explicit*. Figure 1 shows an application of Gauss' principle for a particle laid on a slope and subject to gravity[5].

3 Dynamics problems formulation

3.1 Constrained motion

The constrained motion problem for a contact group of n objects is a direct application of Gauss' principle, and all we have to do is express the set of possible accelerations. This set can be derived from the traditional contact model[3][4][17]. Let's denote two contacting objects by i and j. I is the contact point, $\mathbf{n}$ is the surface normal at I, directed from j to i. Depending on the object it belongs to, I is denoted by I_i or I_j. Using this notation, the non-penetration constraint on the contact points' accelerations is[3]:

$$(\mathbf{a}_i(I_i) - \mathbf{a}_j(I_j)).\mathbf{n} + 2.(\mathbf{v}_i(I_i) - \mathbf{v}_j(I_j)).\frac{d\mathbf{n}}{dt} \geqslant 0 \qquad (3)$$

In a contact-space approach, this constraint would be used to form the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ of the introduction, which relate contact forces and normal accelerations *at the contact points*. However, using Gauss' principle, we need to express

[2]Throughout this paper, a contact group is a set of mobile objects. Two objects i and j are in the same contact group if and only if there exists a chain of contacting mobile objects from object i to object j. At any moment, any contact group is dynamically independent of the others. Therefore, in a rigid body simulation, the contact groups can be examined separately when solving the dynamics problems.

[3]For example, the unconstrained acceleration of a cube laid on a table and subject to gravity g is precisely g.

[4]Since G_a and $\sqrt{G_a}$ are minimized by the same $\mathbf{a}_c$.

[5]For clarity, the examples given in this paper will concern particles. For isolated particles, the object space and the accelerations space can be superimposed, and the kinetic distance is proportional to the euclidean distance.

this constraint on *objects*' accelerations. This is done simply: for any rigid object k, we have $\mathbf{a}_k(I_k) = \mathbf{a}_k(G_k) + \alpha_k \wedge \mathbf{G}_k\mathbf{I}_k + \omega_k \wedge (\omega_k \wedge \mathbf{G}_k\mathbf{I}_k)$, where ω_k is the (known) rotational velocity of object k. Since $\mathbf{v}_i(I_i)$, $\mathbf{v}_j(I_j)$ and $d\mathbf{n}/dt$ are known[3], inequation (3) is linear in the translational and rotational accelerations of the contacting objects. Stacking the m non-penetration constraints (3) yields a single constraint on the contact group's acceleration $\mathbf{Ja} \geqslant \mathbf{c}$, where $\mathbf{J}$ is the well-known $m \times 6n$ jacobian[17], and $\mathbf{c}$ is in $\mathbf{R}^m$. This general constraint defines the convex set of possible accelerations $\{\mathbf{a} : \mathbf{Ja} \geqslant \mathbf{c}\}$. Finally, the constrained accelerations are:

$$\mathbf{a}_c = \arg\min\left\{\frac{1}{2}||\mathbf{a} - \mathbf{a}_u||^2_M : \mathbf{Ja} \geqslant \mathbf{c}\right\} \qquad (4)$$

A common property of frictionless systems is immediately visible. Since $\mathbf{a}_c$ minimizes a (non-euclidean) distance over a convex set, it is unique. Note that equation (4) yields the constrained accelerations for a contact group containing any number of objects subject to any number of (unilateral or bilateral[6]) constraints.

3.2 Resolving collisions

Let us now derive a formulation of the collision resolution problem in the *velocities* space from Gauss' principle. Let $\mathbf{v}$ denote a generalized velocity of a contact group of n objects ($\mathbf{v}$ is in $\mathbf{R}^{6n}$), and let $\mathbf{v}^-$ denote the (generalized) velocity that occurred immediately before the collision. The problem is to find $\mathbf{v}^+$, the (generalized) velocity that will occur immediately after the collision. To do so, we make two classical assumptions[17]: the first assumption is that the collision duration is infinitesimal[16], and the second assumption is Poisson's hypothesis[15], which relates the objects' velocities at the contact points. As a result of the first assumption, we can consider that the object's positions during the collision, as well as the constraints acting on them, remain unchanged. Moreover, the exterior actions can be neglected compared to the intensity of the actions of the constraints. Accordingly, $\mathbf{a}_u = 0$. Let's assume now that the generalized acceleration is constant over the whole duration of the collision dt. This yields $\mathbf{v}^+ - \mathbf{v}^- = \mathbf{a}dt$. Since $\mathbf{a}_u = 0$, Gauss' principle allows to state that $\mathbf{v}^+$ minimizes G_v, where $G_v(\mathbf{v}) = \frac{1}{2}||\mathbf{v} - \mathbf{v}^-||^2_M$, over the velocities that can occur immediately after the collision. These *possible separating velocities* are found simply. In the case of (possibly) simultaneous collisions, Poisson's hypothesis yields the collision response constraint on the objects' velocities, for any of the m contact points (colliding or not)):

$$(\mathbf{v}_i^+(I_i) - \mathbf{v}_j^+(I_j)).\mathbf{n} \geqslant -e(\mathbf{v}_i^-(I_i) - \mathbf{v}_j^-(I_j)).\mathbf{n} \qquad (5)$$

where e is the coefficient of restitution for the objects involved. Again, the velocities at the contact points depend on the objects' translational and rotational velocities. Consequently, inequation (5) is a linear function of the translational and rotational velocities of objects i and j. Stacking the m collision response constraints yields the set of possible (separating) velocities $\{\mathbf{v} : \mathbf{Jv} \geqslant \mathbf{d}\}$, where $\mathbf{J}$ is a $m \times 6n$ jacobian and $\mathbf{d}$ is in $\mathbf{R}^m$. Finally, the velocities occurring immediately after the collision are:

$$\mathbf{v}^+ = \arg\min\left\{\frac{1}{2}||\mathbf{v} - \mathbf{v}^-||^2_M : \mathbf{Jv} \geqslant \mathbf{d}\right\} \qquad (6)$$

Once again, this result can be stated in a very elegant way: *in a contact group, the velocities occurring immediately after a collision are the closest possible velocities to the velocities that occurred immediately before the collision.* Figure 2 shows a collision resolution for a particle that has just collided a motionless object.

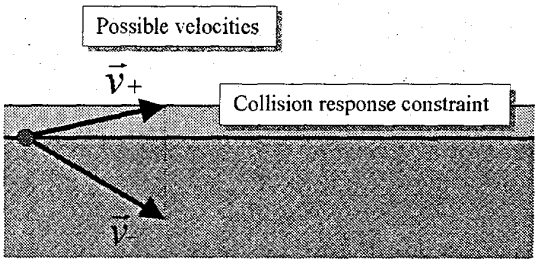

Figure 2: *A particle has just collided a motionless object.* The possible velocities are given by the *collision response constraint*, which depends on e, the coefficient of restitution. The particle's velocity immediately *after* the collision is the closest possible velocity to the particle's velocity immediately *before* the collision.

4 Solving the dynamics problems

Section 3 has reduced both dynamics problems to the same minimization problem:

$$\mathbf{x}^* = \arg\min\left\{\frac{1}{2}||\mathbf{x} - \mathbf{x}_u||^2_M : \mathbf{Jx} \geqslant \mathbf{c}\right\} \qquad (7)$$

where $\mathbf{x}_u$, $\mathbf{M}$, $\mathbf{J}$ and $\mathbf{c}$ are known. If the contact group contains n objects subject to m (unilateral) constraints, then $\mathbf{x}$ and $\mathbf{x}_u$ are in $\mathbf{R}^{6n}$, $\mathbf{c}$ is in $\mathbf{R}^m$, and $\mathbf{J}$ is in $\mathbf{R}^{m\times 6n}$. Since $\mathbf{M}$ is SPD, it can be factored as the product of two positive definite $6n \times 6n$ matrices[7]: $\mathbf{M} = \mathbf{Q}^T\mathbf{Q}$

It is now possible to make the contact forces (or impulses) visible, thanks to the lagrangian method. Hovever, *we will end with a motion-based formulation.* The lagrangian function associated to problem (7) is:

$$L(\mathbf{x}, \lambda) = \frac{1}{2}||\mathbf{x} - \mathbf{x}_u||^2_M - \lambda^T(\mathbf{Jx} - \mathbf{c}) \qquad (8)$$

The vector λ is in $\mathbf{R}^m$ and represents contact forces or impulses, according to the dynamics problem being solved.

[6] Since bilateral constraints can be expressed as pairs of unilateral constraints.

[7] The objects' *local* inertia tensors are constant and can be factored offline. The matrix $\mathbf{Q}$ can thus be computed in O(n).

Since

$$\frac{\partial L}{\partial \mathbf{x}} = 0 \Leftrightarrow \mathbf{x} = \mathbf{M}^{-1}\mathbf{J}^T\lambda + \mathbf{x}_u, \tag{9}$$

the variable $\mathbf{x}$ can be eliminated (temporarily losing sight of the objects' motion). Thus, we must minimize:

$$f(\lambda) = \frac{1}{2}\lambda^T\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\lambda - \lambda^T(\mathbf{c} - \mathbf{J}\mathbf{x}_u) \tag{10}$$

under the constraint $\lambda \geqslant \mathbf{0}$. Assuming the contact group's state is consistent, there exists a vector $\mathbf{k}$ such that $\mathbf{J}\mathbf{k} = \mathbf{c}$. We now use the factorization of $\mathbf{M}$ to retrieve a motion-based formulation. Let $\mathbf{s} = \mathbf{Q}(\mathbf{k} - \mathbf{x}_u)$ and $\mathbf{J}_Q = \mathbf{J}\mathbf{Q}^{-1}$. Minimizing (10) is equivalent to the following *non-negative least-squares problem* (NNLSP):

$$\lambda^* = \arg\min\left\{\frac{1}{2}||\mathbf{J}_Q^T\lambda - \mathbf{s}||^2 : \lambda \geqslant \mathbf{0}\right\} \tag{11}$$

Since $\mathbf{J}_Q^T\lambda$ and $\mathbf{s}$ are both in $\mathbf{R}^{6n}$ and since, from equation (9), $\mathbf{x} = \mathbf{Q}^{-1}\mathbf{J}_Q^T\lambda + \mathbf{x}_u$, this formulation is motion-based.

Geometrically, problem (11) consists in projecting $\mathbf{s}$ on the *positive cone* generated by the rows of $\mathbf{J}_Q$: $\mathcal{C} = \{\mathbf{x} : \mathbf{x} = \mathbf{J}_Q^T\lambda$ and $\lambda \geqslant \mathbf{0}\}$. Problem (11) is thus a *nearest-point problem* (NPP).

5 Comparing formulations

5.1 Mathematical equivalence

The equivalent contact-space formulation is obtained from problem (11). Indeed, the necessary and sufficient conditions of problem (11) are:

$$\begin{cases} \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\lambda + \mathbf{J}\mathbf{x}_u - \mathbf{c} \geqslant \mathbf{0}, \\ \lambda \geqslant \mathbf{0}, \\ (\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\lambda + \mathbf{J}\mathbf{x}_u - \mathbf{c})^T\lambda = 0. \end{cases} \tag{12}$$

This formulation is exactly[8] the one given in Baraff[4] or Ruspini[17]. In this formulation, the number of dof in the contact group is implicit, since all vectors are in $\mathbf{R}^m$, and since the factorization of the operational inertia matrix $\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T$ is hidden.

At this point, the reader may wonder why we did not derive problem (11) directly from the LCP formulation (12). After all, the only tricks involved are factoring the mass matrix $\mathbf{M}$ and noticing that $\mathbf{J}\mathbf{x}_u - \mathbf{c}$ is in the column space of $\mathbf{J}\mathbf{Q}^{-1}$. However, we believe that the insight into the intuitive underlying physics offered by Gauss' principle will help to understand why an algorithm solving problem (11) should be able to perform better than an algorithm solving problem (12).

[8]Note that for clarity, the formulations given in this paper are for 6-dof rigid bodies. However, Gauss' principle holds in reduced coordinates, and can be used along with the general contact model of Ruspini[17]. Besides, the mathematical equivalence given in this section is a proof of this, since all that is required is factoring $\mathbf{M}$.

5.2 Computational difference

Whereas problems (11) and (12) are mathematically equivalent, they are not *computationally* equivalent for essentially four reasons:

- *ill-conditionned matrix*: the condition number of $\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T$ is the square of that of $\mathbf{J}_Q$ (see Gill[11], for example). As a result, problem (12) is much more ill-conditionned than problem (11), and is therefore much more sensitive to round-off errors.

- *sparsity*: Since $\mathbf{M}$ and $\mathbf{Q}$ are block-diagonal matrices, and since $\mathbf{J}$ is always sparse, the matrix $\mathbf{J}_Q$ involved in problem (11) is *always sparse* as well. On the contrary, $\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T$ may be dense[6].

- *memory*: When the involved matrices are considered dense, it requires much more memory to store $\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T$ ($O(m^2)$) than to store $\mathbf{J}_Q$ ($O(nm)$) as the number of contact points per object increases. Moreover, when the matrices are considered sparse, $\mathbf{J}_Q$ *always* require $O(m)$ only, while $\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T$ may still be dense.

- *unnecessary computations*: an algorithm solving problem (12) will have to enforce conditions on both contact forces (or impulses) λ and objects' accelerations (or velocities) at the contact points $\mathbf{x}_{cp}$. As a result, this algorithm will have to maintain the m coordinates of $\mathbf{x}_{cp}$ while solving problem (12). However, the m coordinates of $\mathbf{x}_{cp}$ are not independent, since the contact points belong to rigid bodies: these coordinates could be deduced from the objects' motions. Since these objects' motions are not readily available in a contact-space formulation, the algorithm will perform unnecessary computations. In the motion-space formulation, an algorithm is able to operate on the $6n$ truly independant motion coordinates only, and on the corresponding coefficients in $\mathbf{J}_Q$.

As a preliminary conclusion, let's state the difference between a contact-space formulation and a motion-space formulation more simply: in a contact-space approach, the contact forces are computed *for themselves*, and are *then* used to compute the object's motions. In a motion-space approach, the contact forces are computed *along with the objects' motions*[9].

5.3 Experimental comparison

There are numerous ways to solve the nearest-point problem (12). However, its geometrical structure has led the optimization community to develop specific algorithms, generally split into two categories: *combinatorial* algorithms (active-set methods), and *descent* methods. Algorithms in the first category look for the solution by moving from face to face in the cone. Algorithms in the second category are interior-point methods or exterior-point methods[1]. For typical rigid body simulations, however, it is simpler to use active-set algorithms

[9]Since the objects' motions are directly related to $\mathbf{J}_Q^T\lambda$.

(as is the case in contact-space formulations: at least for frictionless systems, the LCP problems are solved by Lemke's algorithm or Dantzig's algorithm[2][4][17][18][19]).

In an attempt to make an experimental comparison of both approaches, we chose to implement Baraff's well-known algorithm[4] to solve problem (12). This algorithm, in the frictionless case, is equivalent to Dantzig's algorithm for LCPs. For the NPP problem (11), we chose Wilhelmsen's algorithm[21] for several reasons. It is an active-set method, easy to implement and, most of all, is closely related to Dantzig's algorithm, making thus the comparison more pertinent. Actually, as most pivotal methods, both algorithms have an outer loop and an inner loop. In the inner loop, both algorithms need to solve linear systems involving submatrices of $\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T$ (note that consequently, the assessment about ill-conditionning doesn't hold anymore. However, this allows to compare efficiency under equivalent robustness assumptions). The difference between the algorithms resides in the fact that Wilhelmsen's algorithm operates on $\mathbf{J}_Q^T\lambda$ and λ, while Dantzig's algorithm operates on $\mathbf{x}_{cp}$ and λ. We used the

	n=2	n=7	n=12	n=17	n=22
p=5	0.60	0.50	0.51	0.54	0.52
p=15	0.85	1.08	1.23	1.47	1.63
p=25	0.85	0.90	1.07	1.21	1.05
p=35	0.94	0.93	1.31	1.15	1.01
p=45	1.01	1.08	1.5	1.08	1.03

Table 1: Dense case: The ratio of execution times exhibits no significant behavior.

same guidelines to implement both algorithms. The matrix $\mathbf{J}_Q$ and the matrix $\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T$ were treated as dense in a version of the algorithms, and as sparse in another. Cholesky factorizations were used to solve the linear systems in both cases. However, we did not implement an incremental factorization routine, for either algorithm, as suggested in [4]. Since preliminary tests did show that both algorithms performed approximately the same number of linear systems resolutions, no algorithm would have benefitted significantly more than the other. Note that the same tolerance value was used in both algorithms ($\varepsilon = 10^{-5}$).

Note that we are totally aware that, despite all our efforts to implement both algorithms with the same guidelines, it may still be argued that comparing execution times is not always pertinent. However, we are far less interested in the ratio of execution times than in the behavior of this ratio. For a given test, a random matrix $\mathbf{J}$ was computed in the following way. For any row, two integers where randomly chosen between 1 and n. These (possibly equal) integers where used to place two sets of six coefficients in a row, resulting in a typical jacobian matrix. The fact that the chosen integers could be equal enabled to simulate object/environment contacts. To obtain a vector $\mathbf{c}$ consistent with $\mathbf{J}$, a random vector $\mathbf{k}$ was chosen in $\mathbf{R}^{6n}$ and $\mathbf{c}$ was set to $\mathbf{J}\mathbf{k}$. The entries of the equivalent LCP problem were computed as in (12), with $\mathbf{M}$ set to the $6n \times 6n$ identity matrix (which has obviously no impact on the comparison). The two parameters of the tests were the number of objects n and the (average) number of unilateral constraints per object p. For a given pair (n, p), 50 execution times were obtained for Dantzig's and Wilhelmsen's algorithms: t_i^D and t_i^W, respectively, for $1 \leqslant i \leqslant 50$. The ratio of average execution times was then:

$$r(n,p) = \frac{\sum_{i=1}^{50} t_i^D}{\sum_{i=1}^{50} t_i^W} \tag{13}$$

	n=2	n=7	n=12	n=17	n=22
p=5	0.51	0.62	0.80	0.91	0.93
p=15	0.71	1.19	1.52	1.95	2.19
p=25	0.70	1.04	1.60	1.98	1.90
p=35	0.76	1.11	2.08	2.20	2.05
p=45	0.78	1.34	2.44	2.43	2.20

Table 2: Sparse case: the motion-space algorithm takes advantage of sparsity to perform increasingly better than the contact-space algorithm, as the number of contacts per object increases.

Note that we chose n and p such as both problems could be stored in memory and didn't need disk access. The results are reported in Table 1 for the dense case (*ie* dense for both algorithms), and in Table 2 for the sparse case (*ie* sparse for both algorithms). Again, let's emphasize that we are not really interessed in the ratio of execution times *for itself*, but in its behavior. This behavior is demonstrated in Tables 1 and 2: the algorithm operating in the motion-space takes (a better) advantage of sparsity *to perform increasingly better* than the algorithm operating in contact-space, as the number of contact points per object increases. Considering now the ratio in itself, and considering the fact that we used the same guidelines to implement both algorithms, we may (carefully) state that a motion-space algorithm performs better than a contact-space algorithm (at least for our implementation and for the tests we did). It should be clear that when the objects are simple and/or that there are few contacts per object, a contact-space approach will (not surprisingly) perform better. As the objects complexity increases, however, the motion-space formulation seems to be preferable.

6 Conclusion and results

Thanks to Gauss' principle of least constraints, the dynamics problems occuring in a frictionless rigid body simulation (the constrained motion problem and the collision resolution problem) have been formulated in the motion-space. While the resulting formulations are mathematically equivalent, they are *not computationally equivalent*. The main reasons for this is that the motion-space formulation is better conditionned, is always sparse, takes less memory, and is able to avoid some unnecessary computations. A preliminary experimental study suggests that an algorithm operating in the motion-space is

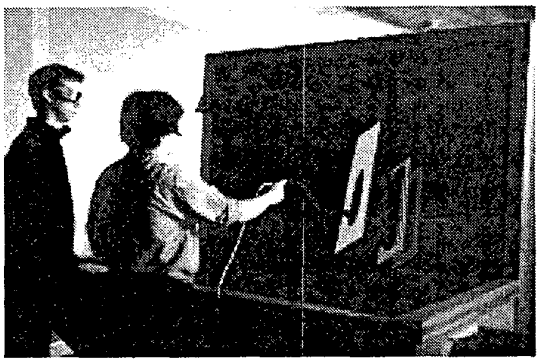

Figure 3: *Towards assembly tasks in an immersive environment (edited photograph of our simulator).* Assembly tasks should benefit from an algorithm operating in a motion-space, as the increase in the objects complexity may result in many contact points.

able to take advantage of sparsity to perform increasingly better than a contact-space algorithm as the number of contact points per object increases[10]. As a conclusion, we note that the motion-space formulations given in this paper presently hold for frictionless systems only. Whereas frictionless simulations stand on their own and have practical applications (virtual prototyping, assembly tests and fitting operations for example), we believe that the encouraging results constitute a strong incentive to develop friction-handling algorithms derived from the motion-space approach. This problem is currently under research. Meanwhile, a complete frictionless rigid body engine, enabling second-order and first-order world[11] simulations, has been implemented and successfully tested with industrial (aeronautics) data in an immersive environment (Figure 3).

Aknowledgements

The authors would like to thank Peter Spellucci for his precious help on quadratic programming. This study was funded by the French Ministry of Research through an AMX grant and the RNTL PERF-RV project. The Responsive Workbench™ was partially funded by the Regional Council of Ile-de-France.

References

[1] K. S. Al-Sultan. Nearest Point Problems: Theory and Algorithms. Ph.D. dissertation, The University of Michigan, Ann Arbor, 1990

[2] M. Anitescu, F. A. Potra and D. E. Stewart. Time-stepping for Three-dimensional Rigid Body Dynamics. Computational Modeling of Contact and Friction. Comput. Methods Appl. Mech. Engrg. 177 (1999), no. 3-4, 183–197.

[3] D. Baraff. Analytical Methods for Dynamic Simulation of Non-penetrating Rigid Bodies. In Computer Graphics (Proc.SIGGRAPH), volume 23, pages 223-232. ACM, July 1989.

[4] D. Baraff. Fast Contact Force Computation for Nonpenetrating Rigid Bodies. In SIGGRAPH 94 Proceedings, pp 23-34. ACM SIGGRAPH, 1994.

[5] D. Baraff and R. Mattikalli. Impending Motion Direction of Contacting Rigid Bodies. Technical report CMU-RI-TR-93-15, Robotics Institute, CMU, June, 1993.

[6] D. Baraff. Linear-time Dynamics using Lagrange Multipliers. In SIGGRAPH 96 Proceedings, pp 137-146. ACM, 1996.

[7] H. Baruh. Analytical Dynamics. WCB McGraw-Hill, 1999.

[8] H. Bruyninckx and O. Khatib. Gauss' Principle and the Dynamics of Redundant and Constrained Manipulators. In Proceedings of ICRA 2000, pp 2563-2568.

[9] B. Fröhlich, H. Tramberend, A. Beers, M. Agrawala and D.Baraff. Physically-Based Manipulation on the Responsive Workbench. Proceedings of the IEEE VR' 2000 Conference.

[10] K. F. Gauss. Uber ein neues allgemeines Grundgesatz der Mechanik. Journal für die Reine und Angewandte Mathematik, 4, pp 232-235, 1829.

[11] P. E. Gill, W. Murray and M. H. Wright. Practical optimization. Academic Press. 1981.

[12] L. Lilov and M. Lorer. Dynamic Analysis of Multi-rigid Body System Based on the Gauss Principle. Z. Angew. Math. Mechanik, 62(10), pp 539-545, 1982.

[13] P. Lötstedt. Numerical simulation of time-dependent contact friction problems in rigid body mechanics. SIAM Journal of Scientific Statistical Computing, vol. 5, no. 2, pp. 370- 393, 1984.

[14] V. J. Milenkovic. H.Schmidl. Optimization-Based Animation. In SIGGRAPH 2001 Proceedings, ACM SIGGRAPH, 2001.

[15] B. Mirtich and J. Canny. Impulse-based Simulation of Rigid Bodies. In Proceedings of Symposium on Interactive 3D Graphics, 1995.

[16] M. Moore and J. Wilhelms. Collision Detection and Response for Computer Animation. In Computers Graphics (Proceedings of SIGGRAPH 88), pp 289-298. ACM SIGGRAPH, August 1988.

[17] D. Ruspini and O. Khatib. Collision/Contact Models for the Dynamic Simulation of Complex Environments. IEEE/RSJ IROS'97, September 1997.

[18] J. Sauer and E. Schömer. A Constraint-based Approach to Rigid Body Dynamics for Virtual Reality Applications. In Proceedings of the ACM Symposium on Virtual reality software and technology 1998, 1998, pp 153-162.

[19] J.C. Trinkle, J.C., J.-S Pang, J.-S. Sudarsky and G. Lo. On Dynamic Multi-rigid-body Contact Problems with Coulomb Friction. Zeitschrift fur Angewandte Mathematik und Mechanik, vol.77, no.4 p. 267-79.

[20] F. E. Udwadia and R. E. Kalaba. Analytical Dynamics: a New Approach. Cambridge University Press, 1996.

[21] D. R. Wilhelmsen. A Nearest Point Algorithm for Convex Polyhedral Cones and Applications to Positive Linear Approximations. Mathematics of computation, 30, pp 48-57, 1976.

[10] Actually, a simulation in reduced coordinates[17] would benefit even more from a motion-based approach, since the only important parameter is the average number of contacts *per degree of freedom*.

[11] A first-order world formulation is straightforwardly obtained by substituting constraints on velocities to constraints on accelerations in Section 3.1 and by applying Gauss'principle to velocities instead of accelerations.