CS 448B: Topics in Modelling
Winter 2000

Critique: Fast Contact Force Computation for Non-Penetrating Rigid Bodies

Adam Phelps
amp@cs.stanford.edu
Stanford University

March 1st, 2000

1. Paper Citation

Fast Contact Force Computation for Non-Penetrating Rigid Bodies, David Baraff. Proc. Siggraph 1994, pp. 23-34.

2. Synopsis

In this paper, Baraff presents an algorithm for computing the contact forces between objects with static and dynamic friction. Unlike previous methods used to solve the problem where the problem was transformed into an optimization problem, Baraff's algorithm can be implemented by programmers with little experience in numerical programming. Further, his approach is faster and more reliable than the previous approaches.

3. Summary

3.1 Introduction

In this paper, Baraff address two goals. The first goal was to address the concerns of the graphics and simulations community with the practical and theoretical difficulties in using analytical techniques for systems with contact. He does so by presenting a simple, reliable, and fast method that can be implemented by individuals without extended experience with numerical analysis or optimization. His second goal was to extend and improve upon previous methods of computing contact forces with friction.

The algorithm presented is an extension of the one described by Cottle and Dantzig [1] for solving linear complementary problems. Dantzig's algorithm can be used with little modification for systems with frictionless contact, and is then extended first for static (known as dry friction) and then dynamic friction (known as sliding friction). Using this algorithm, Baraff demonstrated the first known system for the interactive simulation of contact with a correct model of Coulomb friction.

3.2 Background and Motivation

Previous methods for simulating contact in frictionless systems have been based on linear programming, quadratic programming algorithms, and constrained linear least-squares algorithms, all of which required numerical optimization software packages. These are poor solutions, as they tend to be difficult to interface with, as well as being slow and somewhat unreliable. Further, in order to make use of these packages requires translating the very specific problem of contact force computation into a very general problem, thus eliminating the specific structure of the problem.

3.3 Contact Model

Baraff presents a fairly simple model for a system of contact points. Each contact point $\mathbf{p}_i$ has an corresponding relative normal acceleration a_i and normal force f_i between the two bodies. This paper doesn't deal with impact, so the normal velocity is zero. The bodies are nonpenetrating, thus they mat not accelerate towards each other, so $a_i \geq 0$, and the normal forces must be repulsive, thus $f_i \geq 0$. Further, frictionless contact forces are conservative, so $f_i a_i = 0$. The collection of all a_i's is represented by the vector **a**, and the collection of all f_i's is represented by **f**. **a** and **f** are related by **a=Af + b**, where **A** represents the masses and contact geometries of the objects, and **b** represents the external and inertial forces. Together, the constraints $\mathbf{Af} + \mathbf{b} \geq \mathbf{0}$, $\mathbf{f} \geq \mathbf{0}$, and $\mathbf{f}^T(\mathbf{Af}+\mathbf{b})=0$ form a linear complementary problem that can be solved using Dantzig's algorithm.

3.4 Frictionless Systems

The implementation of Dantzig's algorithm as presented is relatively straightforward. Initially, **f** is set to **0** and **a** is set to **b**. For each d where the constraint $a_d \geq 0$ doesn't hold, the algorithm attempts to increase f_d just enough so that $a_d = 0$. Altering f_d may affect some of the previously established values, and thus they must be adjusted to re-establish the constraints.

In order to adjust these values, the contact points are divided into two distinct sets, C and NC. If $i \in C$ then $a_i = 0$ and $f_i \geq 0$, and the point i is said to be "clamped". If $i \in NC$ then $a_i > 0$ and $f_i = 0$ and the point i is said to be "unclamped". For a unit increase of f_d, each f_i must be adjusted by Δf_i. For all $i \in NC$ Δf_i is set to 0 and the Δf_I's for $i \in C$ are set so that $\Delta a_i = 0$, where $\Delta\mathbf{a} = \mathbf{A}\Delta\mathbf{f}$. However, the conditions $f_i \geq 0$ and $a_i \geq 0$ must be maintained. As f_d is increased, it may become necessary to switch a point between C and NC, which is referred to as pivoting. Thus, the algorithm also computes the smallest scalar $s > 0$ such that increasing **f** by $s\Delta\mathbf{f}$ either causes a_d to reach zero, or causes some point to switch between C and NC. This process repeats until a_d reaches zero. The precise process needed to compute $\Delta\mathbf{f}$ and s is described in detail in the paper. Further, Baraff provides a proof that the algorithm terminates.

The implementation of this algorithm is fairly straightforward, the only complex part being in the function which computes $\Delta\mathbf{f}$, as doing so requires solving a linear system. This can be implemented using simple (though not particularly efficient) Gaussian elimination or a more complicated Cholesky decomposition,

3.5 Static Friction

In order to introduce friction to the algorithm, the contact model above must be slightly modified. First, a_i and f_i are relabelled a_{Ni} and f_{Ni}. Also, the frictional force tangential to the contact surface, f_{Fi}, is added, as is the acceleration in the tangent plane, a_{Fi}. Initially, only static friction is added, so the tangential velocity at a contact point must be zero. A solution is presented for a two dimensional system and then quickly expanded for a three dimensional system.

In addition to the previous constraints, $a_{Ni} \geq 0$, $f_{Ni} \geq 0$ and $f_{Ni}\, a_{Ni} = 0$, the effect of all the forces must maintain $v_{Fi} = 0$, so a_{Fi} must be zero. Thus, for frictional coefficient μ, $|f_{Fi}| \leq \mu f_{Ni}$. Also, $a_{Fi} f_{Fi} \leq 0$ and $a_{Fi}(\mu f_{Ni} - |f_{Fi}|)=0$, which forces $|f_{Fi}| = \mu f_{Ni}$ if a_{Fi} is not zero. These three constraints are called the static friction conditions.

Much like in the algorithm for a frictionless system, the algorithm with static friction starts with all forces equal to zero and iterates over the contact points. If for some index d, a_{Fd} is nonzero then the algorithm attempts to increase $|f_{Fd}|$ to compensate, until either $a_{Fd} = 0$ or $|f_{Fi}| = \mu f_{Ni}$ and the static friction coefficients are re-established.

Three new discrete sets are introduced, C_F, NC^+, and NC^-, If $i \in C_F$ then $a_{Fi} = 0$, if $i \in NC^+$ then $a_{Fi} < 0$ and $f_{Fi} = \mu f_{Ni}$, if $i \in NC^-$ then $a_{Fi} > 0$ and $f_{Fi} = -\mu f_{Ni}$. When adjusting f_{Nd} and f_{Fd}, the changes to the previously established forces and the maximum step size is computed as in the frictionless case, and points are moved between C and NC or between C_F, NC^+ or NC^- as above. The algorithm first establishes the normal force conditions at a given point before establishing the corresponding static friction conditions.

In the case of static and dynamic friction, however, the algorithm cannot be proven to terminate. It is possible that the algorithm will take step of zero size, resulting in some point alternating between N and NC. Baraff presents a work around to this problem, in which a point alternating between the two sets is removed from both sets, and the algorithm is forced to reestablish the conditions at some later time. This allows for the theoretical possibility that the algorithm will fail to terminate, although Baraff stipulates that he has not yet encountered a system configuration that results in an infinite loop.

3.6 Dynamic Friction

Dynamic friction occurs in the case where the tangential velocity at a contact point is nonzero. In this case, the friction force must satisfy $|f_{Fi}| = \mu f_{Ni}$ with direction opposite the tangential velocity. Thus f_{Fi} is no longer an independent variable, and all occurrences of f_{Fi} can be replaced by $\pm\mu f_{Ni}$. This change can cause the system to fail to have solutions for the contact force magnitudes, requiring the use of an impulsive force, which Baraff dealt with in a previous work. In practice, however, Baraff has only witness this to occur when μ was unrealistically large.

3.6 Results

Although the worst case running times for the presented algorithm are exponential, Baraff claims that in practice it tends to be O(n). Compared with previous methods for frictionless systems, this algorithm runs five to ten times faster on problems up to n = 150, however there is no comparable solution for systems with friction.

4. Comments

4.1 Contributions

The major contribution of this paper was the presentation of a algorithm for the computation of contact forces which is relatively simple and fast compared to previous solutions for frictionless systems. Further, the algorithm correctly handles systems with Coulomb friction, for which there was no previous method. Baraff's attempt to make the algorithm sufficiently simple so that it could be implemented by a non-specialist would appear to be correct, as I believe I could implement it (however, I have minors in both physics and mathematics, so I'm perhaps not the most appropriate individual to gauge the algorithm's simplicity).

4.2 Issues

One of the major downfalls of this paper is it's lack of clarity. The algorithm in the paper requires a good deal of mathematics to describe, however the author fails to provide any clarifying illustrations when such would have been of great use to help the reader decipher the meaning of the various terms. Further, although the base algorithm for a frictionless system is outlined in pseudocode, the extensions for static and dynamic friction are only outlined in text, making it somewhat unclear precisely how the pseudocode would be modified to implement the system with friction.

The other major issue with the paper is the lack of an form of concrete results. Although the author claims that his algorithm is five to ten times faster than previous algorithms for up to 150 contact points, he gives no details as far as actual running times.

4.3. Questions

This paper was presented in 1994, so it is possible that the algorithm from this paper has seen a good deal of use since then. Has it been implemented in any systems of note? What would be the running time of the algorithm on modern hardware?

6. Bibliography

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