

COMP259 Spring 2004 Project Update:

Impulse based Stacking for Rigid Bodies

For this project, I have been looking into novel methods for simulation of rigid bodies that integrate well with *stacking* situations. I will focus on interactions such as collision, contact and friction. Many authors [Baraff01, Mirtich] built dynamic models that simulate rigid bodies quite efficiently when in *ballistic* and *colliding* phase, either using a force-based or an impulse-based approach. Nevertheless, these authors also argue that the simulation is more problematic in the *resting* case, often shown in the case where objects at rest start sinking into the ground. Many authors use *ad hoc* threshold velocities in an attempt to prune these cases out of the collision modeling algorithm and instead treat them with a contact model. They either have to solve a complicated quadratic system [Baraff01], or introduce artificial microcollisions [Mirtich] to prevent the bodies from interpenetrating. This causes some unwanted effects, such as blocks vibrating on a plane, or a block sliding down from a slanted plane, where it should sit still because of the friction force.

Recently, Guendelmann, Bridson & Fedkiw [Guendelmann], have come up with an elegant, more *uniform* algorithm, claiming to resolve some of the issues discussed above without the need for *ad hoc* threshold velocities. The main contribution of the approach is a **new, fixed time-stepping scheme** that separates *collision detection* and resolution from *contact resolution*. They use a dual geometric representation. Aside from the traditional mesh representation, they also keep an object space signed distance function for collision detection. The collision detection algorithm is a *sample-based* technique: they do inside-outside tests on the vertices of the triangular mesh, using the stored signed distance function. Although they claim that this is sufficient for well resolved meshes, and although they improve the approach by flagging edges that intersect the zero contour, many collisions can still be missed for fast moving objects.

The **first task** in building a system from Guendelmann's algorithm was to implement the fixed-time integration stepping scheme that separates collision detection and resolution from contact resolution, where only the velocities are updated between both stages, and

the positions are updated only after the second stage. Basically, both stages perform a collision resolution step, but the only difference between both is the coefficient of restitution used in the impulse computation. In the contact resolution phase, we model the collision as *completely plastic* ($\epsilon = 0$). I implemented this time stepping scheme, and the results are promising: for a limited number of objects, the algorithm is really stable. Objects at rest, and stacks of a few objects remain static without sinking into the ground and without slowing down simulation, because of the fixed-time stepping. This is a good improvement over the bisection method, where many close-by objects drastically increase the simulation time.

Nevertheless, as the number of objects increases, we have to adapt the size of the time steps. I was using SWIFT for collision detection, but SWIFT was primarily conceived for closest-feature determination, and hence not very well suited for deep penetration detection. Very soon, I had to take time steps of 0.001 not to miss collisions.

Therefore, we need a new way to deal with collisions. One idea that seems acceptable is to use the deepest point of penetration between a pair of objects to calculate proper response. This is also plausible for simulations with larger amount of objects, as we only have to resolve one pair of colliding points for every pair of objects. The collision resolution phase will then make sure that the objects are properly separated before going into contact resolution. Thus, as a **second step**, I changed the collision detection to *DEEP [DEEP]*. I use the returned pair of deepest penetration points to resolve the collisions, using Poisson's hypothesis ($u_{rel}' = -\epsilon u_{rel}$). This allowed me to increase the time step again.

The **third step** of implementation is to include friction. So far, I have implemented part of Mirtich's friction model. It includes static and dynamic (sliding) modes, but I have not integrated his latest work (from the thesis): *Stronge's hypothesis* and *sticking friction*. My first implementation works, but shows the issues he was having with *bouncing*: objects won't come to rest. I am currently working out why we are observing this behavior, even in Guendelmann's framework. In their paper, they stated that they use the simpler ad hoc friction model by Moore & Willems [*M&W*]. It may be necessary to plug that model in my implementation to compare both.

Many issues remain though, especially in the case of *multiple collisions*. We must account for new penetrations after a pair of objects is separated. Guendelmann suggests

just using a fixed number of collision resolution iterations, but my first tests show that this method doesn't really work. I have a few ideas for solving this:

- Use an explicit order of collision resolution, such as bottom-up: the main force in most simulations is gravity, pulling objects down. It can be beneficial to resolve collisions for the bodies on the bottom first, such that we need less iterations to come to a state of non-penetration between all pairs.
- Solve a small LCP using impulses: Baraff's paper [Baraff94] solves forces efficiently with an LCP (as opposed to his quadratic solution), but the method assumes *non-penetration*, and works with forces. On the other hand, Kawachi, Suzuki & Kimura [Kawachi] recently presented a paper that specifically deals with multiple collisions in an impulse-based framework. They too reduced to problem to an LCP, including friction in the model.

Here are the **next steps** that I will take. First off, some of the minimal requirements before the end of the project:

- Do better benchmarks, with various parameters: number of objects, mesh tessellation, time stepping, ...
- Friction:
 - Implement Mirtich's *sticking mode* and *Stronge's hypothesis*
 - Implement the simpler *Moore & Willems* friction model
 - Compare both models in complexity and quality
- Deal with multiple collisions efficiently: Use a bottom-up collision processing order
- Final **demo**: some sort of '*dynamic scene with stacking*' benchmark, maybe a scene that simulates a domino track.

Depending on the time I have left after that, I would also like to do the following:

- Deal with multiple collisions even *more* efficiently: Implement the work in [Kawachi].
- Extend to non-convex methods: DEEP can handle non-convex objects through SWIFT++ decomposition, but the resulting pair of deepest points might not really be what we are looking for (distances to virtual faces).

Acknowledgements:

Miguel Otaduy: Fruitful discussions and for the help with getting nasty memory bugs out of DEEP, that were crashing the program. DEEP does not like degenerate collision cases.

Morgan McGuire: For pointing me to Guendelmann's paper and for giving me good practical tips on the implementation.

References:

[Baraff94] **Fast contact force computation for nonpenetrating rigid bodies**

David Baraff, Proceedings of ACM SIGGRAPH 1994

[Baraff01] **Collision and contact course notes**

David Baraff, ACM SIGGRAPH Course notes 2001

[Guendelmann] **Non convex rigid bodies with stacking**

Guendelmann, Bridson and Fedkiw, Proceedings of ACM SIGGRAPH 2003

[M&W] **Collision detection and response for computer animation**

Matthew Moore , Jane Wilhelms, Proceedings of SIGGRAPH 1988

[Mirtich] **Impulse-based Dynamic Simulation of Rigid Body Systems**

Brian Vincent Mirtich, PhD. Thesis, University of California at Berkeley

[Kawachi] **Technical Issues on Simulating Impulse and Friction in Three Dimensional Rigid Body Dynamics**

K. Kawachi, H. Suzuki and F. Kimura, IEEE Computer Animation '98 Conference, Jun, 1998

[DEEP] **Fast penetration depth computation for physically-based animation**

Y.J. Kim, M. Otaduy, M.C. Lin, D. Manocha, Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation