CSCE 645 Research Project Update 2

"Interesting Interactions with Granular Material"

Gabriel Stella

1 Summary

As of the last update, I had the following goals:

1. Implementation efficiency
2. User interaction support
3. Representation compression

I have completed the first by implementing a faster grid-based particle proximity checker, reducing the runtime of each update step from $O(n^2)$ to $O(n)$. The increase in performance is noticeable, allowing it to scale so at least 2000 particles now (even running on Javascript in a browser).

The biggest issue right now is a glitch that I call "popcorn", since it makes the particles jump around. What seems to happen is that when a particle spins fast enough, the forces somehow cause it to *pull in* other particles rather than repel them, forming a tight cluster. Then, the rotational friction forces somehow cause those other particles to spin up without slowing down the central particle, and the entire mass begins to rotate. Eventually, once a certain energy level is reached in the cluster, it explodes, launching high-energy particles across the simulated area. As soon as these high-energy particles contact anything else, the behavior is repeated.

While I have now fixed the previous problems with my polysphere implementation, the popcorn glitch is even more pronounced when these particles are used rather than circles, and the polysphere simulation is drastically less stable; particles are strangely bouncy, taking far too long to settle down. As I have little (no) experience implementing physics engines, I don't have much intuition for how these types of problems arise, so I've had to spend quite some time investigating it.

I have done some work on better user interaction, in the form of increased visualization and a better particle addition tool (allowing more particles to be created at once). I plan to add a particle deletion tool and a particle manipulation tool, but since these will likely be simple to implement, they are not the current focus.

Representation compression is a slightly bigger problem. I have been using soft-body ("molecular dynamics") particle physics, as described in the paper "Particle-Based Simulation of Granular Materials". However, even at rest, the system behaves like a fluid, with visible energy waves moving through the collection of particles. In contrast, proper sand piles are almost completely static; they definitely do not undulate. Due to this, I am beginning to think that I may have to transfer to a rigid-body circle system, though the literature said that these can be highly costly and unstable, especially when simulating scenarios with prolonged contact between particles. This is relevant to my representation compression / computational efficiency modifications, since no particle in the current system is ever still enough to be deactivated. Furthermore, the rules that will govern interaction between the two types of particles (active vs. inactive/compressed), as well as the amount of space compression that can be achieved without losing simulation fidelity, are unclear.

2 Planning

My current focus is on increasing the stability of the simulation. First, I'm going to take a close look at the forces that occur in small 2- and 3-particle interactions to try to see if I can determine what causes the seemingly-unbounded energy increases. Then, I'll put back the new polysphere implementation and see what is causing the unwanted bounciness.

While it is fun to have a live demo available on my website, the limitations of Javascript are apparent. As such, after getting the logic of this simulation where I want it, I'll be reimplementing it in C++. This new version should be able to scale to at least $10\times$ more particles, if not $100\times$ thanks to faster data structures in addition to the increased baseline speed of machine executables.