Journal Report 5
9/30/19-10/4/19
Mikail Khan
Computer Systems Research Lab
Period 5, White

Daily Log

Monday September 30

On Monday I worked on diagnosing why collisions didn't work as well as they used to even with CCD disabled. I couldn't find the issue, but I didn't want to git revert because I wanted to keep the work I'd done on the CCD algorithm itself.

Wednesday October 2

I git reverted but kept the important parts of CCD by copy pasting. There wasn't very much of an improvement so I thought the algorithm was incorrect and worked on it a lot. One thing I realized is that the collision detection library I'm using doesn't incorporate rotational velocity into the time of impact calculation, so I tried dividing the times of impact based on how fast the object to compensate for that. It really didn't make much of a difference, so I did some more research on CCD in other game engines and found out that it's really only needed when things are moving very fast.

Friday October 4

While CCD wasn't the solution to my problem I didn't misdiagnose the problem itself. Right now, my simulation is pretty much perfect when there's only one contact on a given shape in the timestep, as evidenced by circles. Rotation makes it a bit worse but the main problem is that when there's more than one contact the impulse should propagate and influence the other impulses, and it ends up being kind of recursive. To solve this, I need to implement a simultaneous collision solver. Simultaneous collision solvers seem to be the hardest part of rigid body simulation, and there are research articles on it by Nvidia (for PhysX) from as recently as 2015, although I shouldn't have to implement anything anything as near as complicated as theirs because the hard part of that was making it parallel for GPUs, and multithreading is kind of weak for WebAssembly anyways. I tried implementing a super simple version of simultaneous collision detection but it doesn't make a big difference. I've found some good resources from the creator of Box2D that I think will help.

Timeline

Date	Goal	Met
Today minus 2 weeks	Functional 2D mechanics and input	Input but shapes jitter as a result of tunneling.
Today minus 1 weeks	CCD	Refactoring done to make CCD practical done
Today	CCD	CCD is done, but it doesn't solve the jittering/tunneling problem like I'd thought it would.
Today plus 1 week	Graphs and improved UI	I want to move on from collisions for now and I think graphs will be helpful for debugging anyway.
Today plus 2 weeks	Better simultaneous collision resolution	I'll probably be able to set concrete goals by looking at the frequency of the velocity graphs since higher frequency corresponds to more jittering, but that's probably overkill.

Reflection

In narrative style, talk about your work this week. Successes, failures, changes to timeline, goals. This should also include concrete data, e.g. snippets of code, screenshots, output, analysis, graphs, etc.

I've been working on essentially the same problem for three weeks now and it's getting kind of stale. Based off of by gravity simulator from last year, I assumed I'd be able to basically just plug the (pretty complicated) equations into a computer and put some graphics on it and then the UI would be the hard part, but rigidbody jittering is a problem that researchers have been working on for a while. I'll probably be able to get rid of most of the jittering though. The graphs will be pretty interesting to see, and I don't think they'll be too difficult. I'll have to figure out how the window layout will be and how to handle user selection of what data to record since I can't record every property of every object.