Andrew Johnson

CS 475

Project 7

Commentary

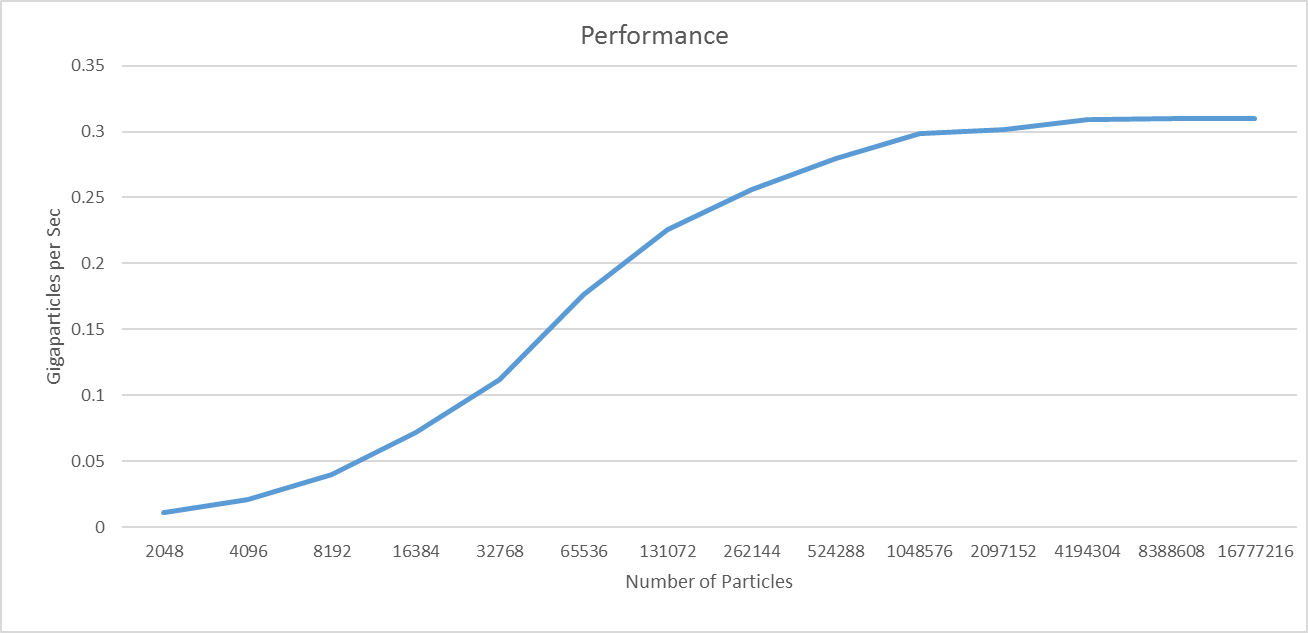
To start, I’d like to say I did this on my own computer using Visual Studio. I’m finding out I may need to clean out some dust in my computer because these projects are really heating up my computer.

How I affected the color is using the reciprocal of the velocity and subtract it from a half. The code looks like, “cp = .5 - 1/(DT\*v);”. I saw that it should stay between 0 and 1 and this was the best way I could think of. I took a screen shoot of using around 16000 particles with my two bumpers.

I kept the local size at 32 because it worked well in the past project. Then I gathered the performance for number of particles from 2048 to around 16 million.

Here is the table and graph for the performance, where the performance is measured in gigaparticles per second.

|  |  |
| --- | --- |
| Num of Particles | Giga/Sec |
| 2048 | 0.010928 |
| 4096 | 0.021005 |
| 8192 | 0.039737 |
| 16384 | 0.071853 |
| 32768 | 0.111771 |
| 65536 | 0.176311 |
| 131072 | 0.225429 |
| 262144 | 0.256225 |
| 524288 | 0.27988 |
| 1048576 | 0.29817 |
| 2097152 | 0.30182 |
| 4194304 | 0.30894 |
| 8388608 | 0.31003 |
| 16777216 | 0.310051 |



The curve looks to be a logarithmic type where it maxes out around .31 gigaparticles per second. I’m guessing this maxing out is the GPU’s limit in organizing all the particles so it can run it swiftly. After all, all computers have their limitations. So it seems it’ll be able to compute just as fast as around 8 million, as it will with 16 million and so on. That makes GPU’s quite powerful in computation when having to mess with a large number of particles. For example, if you were trying to create snow in a program. You would have to take into account all the small snowflakes and treat them all independently.

