Interactive Cinematic Scientific Visualization in Unity

Jasmine Y. Shih
Kalina Borkiewicz
AJ Christensen
Donna Cox
Advanced Visualization Lab, National Center for Supercomputing Applications

ACM Reference Format:
Jasmine Y. Shih, Kalina Borkiewicz, AJ Christensen, and Donna Cox. 2019. Interactive Cinematic Scientific Visualization in Unity. In *Proceedings of SIGGRAPH '19 Posters.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3306214.3338588

1 INTRODUCTION

Cinematic scientific visualizations turn complex scientific phenomena and concepts into stunning graphics and make them easier for the general public to comprehend. Adding interactivity to cinematic scientific visualizations is highly beneficial especially for educational purposes, as it keeps the viewers engaged and promotes active learning [Cano et al. 2017]. Although there are existing software tools such as VisIt that are capable of handling large data sets and allow for interactive exploration, they are usually designed for scientists and not meant for producing cinematic visualizations for the general public. Creating aesthetically pleasing visualizations of scientific data helps to better communicate the scientific concepts, increase impact, and reach a broader audience [Borkiewicz et al. 2018]. As existing examples of visualizations that are both interactive and cinematic have mainly been produced with custom software, there is a lack of easily accessible tools for developing this type of scientific visualization.

Game engines have been gaining popularity in fields outside of the video game industry primarily because they provide a feature-rich development environment for real-time applications with high-end graphics needs. As game engines are designed to efficiently manage resources, produce high quality graphics, and handle user input, we believe that they have the potential to serve as a good tool for developing interactive cinematic scientific visualizations.

2 OUR APPROACH

To research the feasibility and value of using game engines to create interactive cinematic visualizations, we experimented with subsampled scientific particle data in Unity as a proof of concept. We chose to work with particle data because it is a very common data format in the field of physical sciences, and we decided to use Unity because it has a powerful built-in particle system and is known for having a user-friendly interface.

We went through three stages of research:

(1) Testing the performance of and experimenting with Unity's particle system using large datasets
(2) Developing an interactive cinematic visualization of a scientific model of moon formation
(3) Studying the effects of our visualization on young viewers

3 BENCHMARKING AND OPTIMIZATION

A typical scientific particle dataset obtained through simulation includes particle position and additional attributes at each timestep. To visualize particles in real time at 24 FPS using Unity's particle system, we would need to update the particles with particle data within 41 milliseconds in each frame. In this stage, we benchmarked the file reading and particle update time to find the maximum number of particles we could afford to visualize. We first compared two file reading methods — binary and string, and we found that the binary method performed approximately 100 times faster than the string method. Next, we recorded the amount of time it would take to update various numbers of particles with a timestep of scientific particle data. We found a linear relationship between the particle update time and the number of particles. Updating particle color in addition to position nearly doubled the update time.

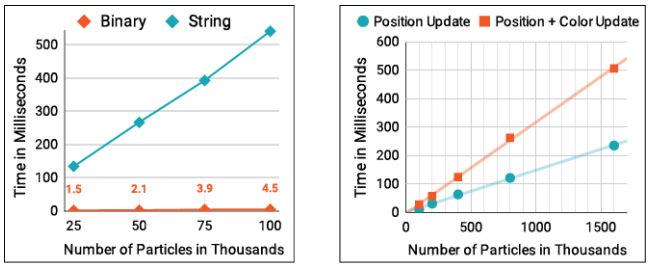

Figure 1: Results from performance testing of data reading methods (left) and particle updates (right).

As assigning the positions of the particles with particle data at every frame proved to be costly, we decided to make use of the particle system's ability to approximate physics to reduce the frequency of particle updates with particle data. We assigned both the particle position data and velocity data to the particles in the particle system every set number of frames, thereby letting the particle system interpolate the positions of the particles for the in-between frames. As an experiment, we fixed the number of particles to 300 thousand and recorded the average FPS for varying particle update frequencies. While we saw a performance improvement with decreased particle update frequency, this optimization technique is a trade-off between performance and accuracy.

SIGGRAPH '19 Posters, July 28 - August 01, 2019, Los Angeles, CA, USA

ACM ISBN 978-1-4503-6314-3/19/07.
https://doi.org/10.1145/3306214.3338588