Related Work

Data Layout

Sajadi et al [3] looked at how to lay out the geometric data on the disk. According to the paper, many of the previous attempts to speed up rendering of interactive walk-through had not considered the question of how the data layout on the hard disk itself. They proposed a paged based data stucture where the geometric primitives were grouped together in the form of disk-pages. There is a heirarchy on the disk-pages that helps organize them for efficiency. This data structure proved to have many advantages over previous attempts. Because of these results, we explored the optimal way to organize uniform data units on the disk.

Sajadi et al [4] explored the data layout for Solid-State Drives (SSD) due to their advantages over Hard-Disk Drives (HDD) and the properties of how they read and write data. They explored the properties of existing layouts. They found that high dimensional spatial data does not render efficiently with HDDs. They also found that Cache-Aware Layouts and Cache-Oblivious Layouts have similar performances on an SSD however Cache-Aware Layouts have many key advantages. They then went into detail on how to construct the cache-aware layout. This eventually led them to develop an efficient out-of-core interactive walk-through rendering application. Despite these advantages of SSDs, due to cost and size limitations HDDs still have to be used for some applications. Through this exploration though it was found that if we reduce seek time on the HDD then the performance will be greatly improved. We thus need to introduce redundancy to reduce seek time. The idea is that with redundancy, the performance of the HDD will be close to the performance of an SSD. Redundancy is not too big of a deal with an HDD due to the inexpensiveness of the space.

Yoon et al [5] explored the problem of data layout on a HDD in such a way as to reduce the seek time required. They developed a cache-oblivious metric to estimate the seek time given the data layout and access requirements for the data. Reducing the seek time then became an optimization problem. Unfortunately the problem was trying to find an optimal permutation of up to millions of data units which proved to be NP-hard. Similar to other problems where an optimal permutation needs to be computed, they developed a multi-level minimization algorithm as a heuristic to get an approximate solution quickly. We decided to take the metric they use and see how much we can reduce seek time if we add redundancy after the heuristic for the optimal layout has been applied.

Out-Of-Core Rendering

Compression-based Rendering

Image-based Rendering

Motivation

Jiang et al [2] explored solutions to the data layout problem where the disk seeks are minimized. He was able to achieve a seek time of one between frames. This was accomplished by using redundancy in the data layout. This means that multiple copies of the same geometric primitive were put into the data layout so that the seek time between primitives who would be accessed together is reduced. The redundancy factor ended up being quite large for that case so the algorithm was not practical. The performance was better though so we decided to explore the trade-off between performance and redundancy.

Jiang et al [1] later explored the trade-off between performance and redundancy. Specifically, they looked at the question of given a seek time requirement, what is the minimum amount of redundancy that can be achieved. In this way you can figure out the least amount of space required to achieve a certain seek time in your application. It ended up being an integer linear programming algorithm. After implementing their algorithm and analyzing the results, the consistency and interactivity of the system was greatly improved. The amount of redundancy using this improved algorithm was significantly less than with the single-seek layout paper, making this algorithm much more practical. The algorithm however did not compute the actual layout of the data that is optimal.

In summary, the page-based data structure gave us motivation to use uniform data units arranged linearly on a disk in order to store all the information for a walkthrough. The data management paper showed us that when we use redundancy to reduce seek time, it will improve the performance for a walkthrough stored on an HDD. The Single Seek Layout paper showed the result of using a great amount of redundancy. The optimization paper found the least redundancy factor required for a certain seek time but it did not compute the actual layout of the data. This paper computes the layout of the data and seeks to find the optimal seek time given an upper limit on redundancy.

References

[1] Jiang S. Sajadi B. Ihler A. and Gopi M. Optimizing redundant-data clustering for interactive walkthrough applications. *CGI 2014*.

[2] Jiang S. Sajadi B. and Gopi M. Single-seek data layout for walkthrough applications.

[3] Sajadi B. Huang Y. Diaz-Gutierrez P. Yoon S. and Gopi M. A novel page-based data structure for interactive walkthroughs. *ACM SIGGGRAPH 2009*.

[4] Sajadi B. Jiang S. Heo J. Yoon S. and Gopi M. Data management for ssds for large-scale interactive graphics applications. *ACM SIGGGRAPH 2011*.

[5] Yoon S. Lindstrom P. Pascucci V. and Manocha D. Cache oblivious mesh layouts. *ACM SIGGGRAPH 2005*.