A Static Load Balancing Scheme for Parallel Volume Rendering on Multi-GPU Clusters

Shusen Liu1, Venkatram Vishwanath2, Joseph Insley2, Mark Hereld2, Michael E. Papka2, Valerio Pascucci1
1Scientific Computing and Imaging Institute, University of Utah 2Argonne National Laboratory

Introduction

GPU-based clusters are an attractive option for parallel volume rendering. One of the key issues in parallel volume rendering is load balancing, wherein keeping a balanced workload per node is essential for improving performance.

Methods

In our work, the data is divided into much finer blocks in comparison to prior efforts. Each node renders a selected group of these blocks, which span the entire domain. By re-arranging the blocks intelligently, we can potentially make every node have a similar workload profile even when camera parameters vary dramatically.

Results & Future Works

1. Load balance experiment results

We evaluate our approach on the GPU cluster “Tukey” at Argonne National Laboratory. Each Tukey node is equipped with 2 Tesla M2070 GPUs, and the test volume is a 2048x2048x2048 regular grid (8GB - unsigned char). In order to measure the load balancing performance of different approaches, we measured the rendering time for a number of unique views on every node - a 280 degree camera rotation, with two zooming operations along the camera path.

Due to a direction correlation between camera position and volume rendering time, during a volume rendering session changing the view leads to significant load imbalance. For interactive parallel rendering (where a lot of view changing exist), it is even more important to keep a balanced load. Our proposed algorithm is intended to address this while eliminating the data transfer overhead exists in other dynamic load balancing schemes.

In our approach, the blocks rendered by one particular node are interleaved with blocks from other nodes.

Compositing will be more complicated. Instead of generate one single image per-node every small image buffer rendered from each individual block will need to be stored. However, for GPU based volume renderer the compositing time is very limited due to small node count (less than 200).

We evaluate with two mechanisms to generate the interleaved block layout. A “linear” index and an z-order index. A comparison between single block method and these approaches is depicted in next section.

Figure 1-3, we compare the single block scheme with our proposed multi-block schemes. It clearly demonstrates the efficacy of our method in achieving static load balancing under drastically changing views. From 1 to 3, we gain better load balance by introducing multi-blocks per node and by reorganizing the multi-block block layout pattern. In Figure 4, we notice that improving the load balancing dramatically improves the overall rendering performance.

2. Future works

We plan to extend our work to handle adaptive mesh refinement (AMR) datasets, and adaptive volume rendering for extremely large regular grid dataset(STB rabbit retina dataset ). We also plan to improve the block layout algorithm by adopting objective function based optimization techniques.

Reference


