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

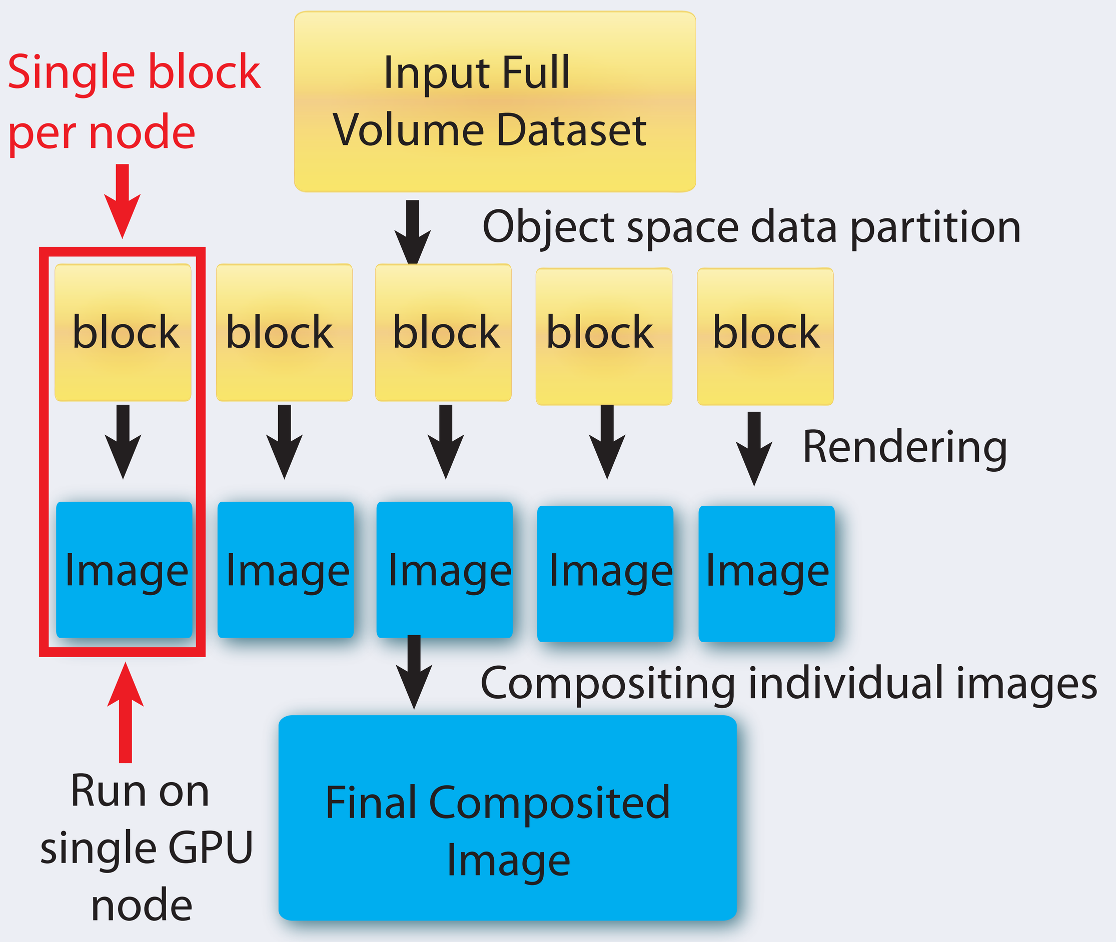
Shusen Liu[1], Venkatram Vishwanath[2], Joseph Insley[2], Mark Hereld[2], Michael E. Papka[2], Valerio Pascucci[1]

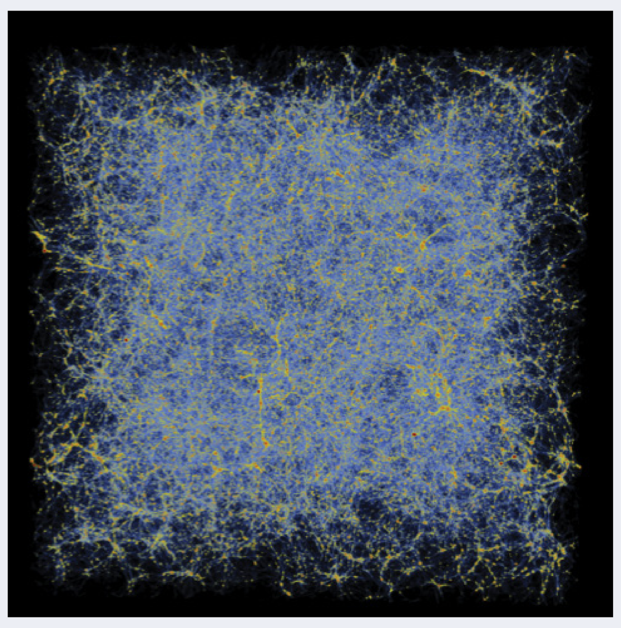
[1]Scientific Computing and Imaging Institute, University of Utah [2]Argonne National Laboratory

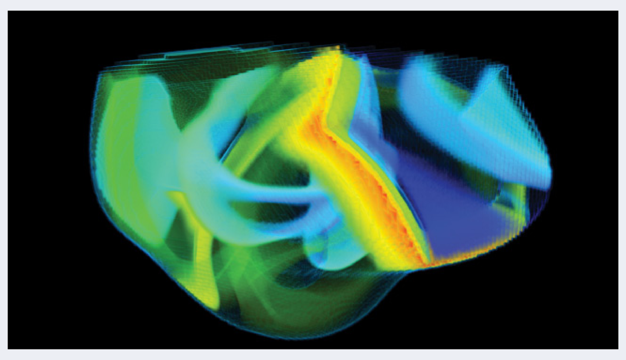
SCI X 2012

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.



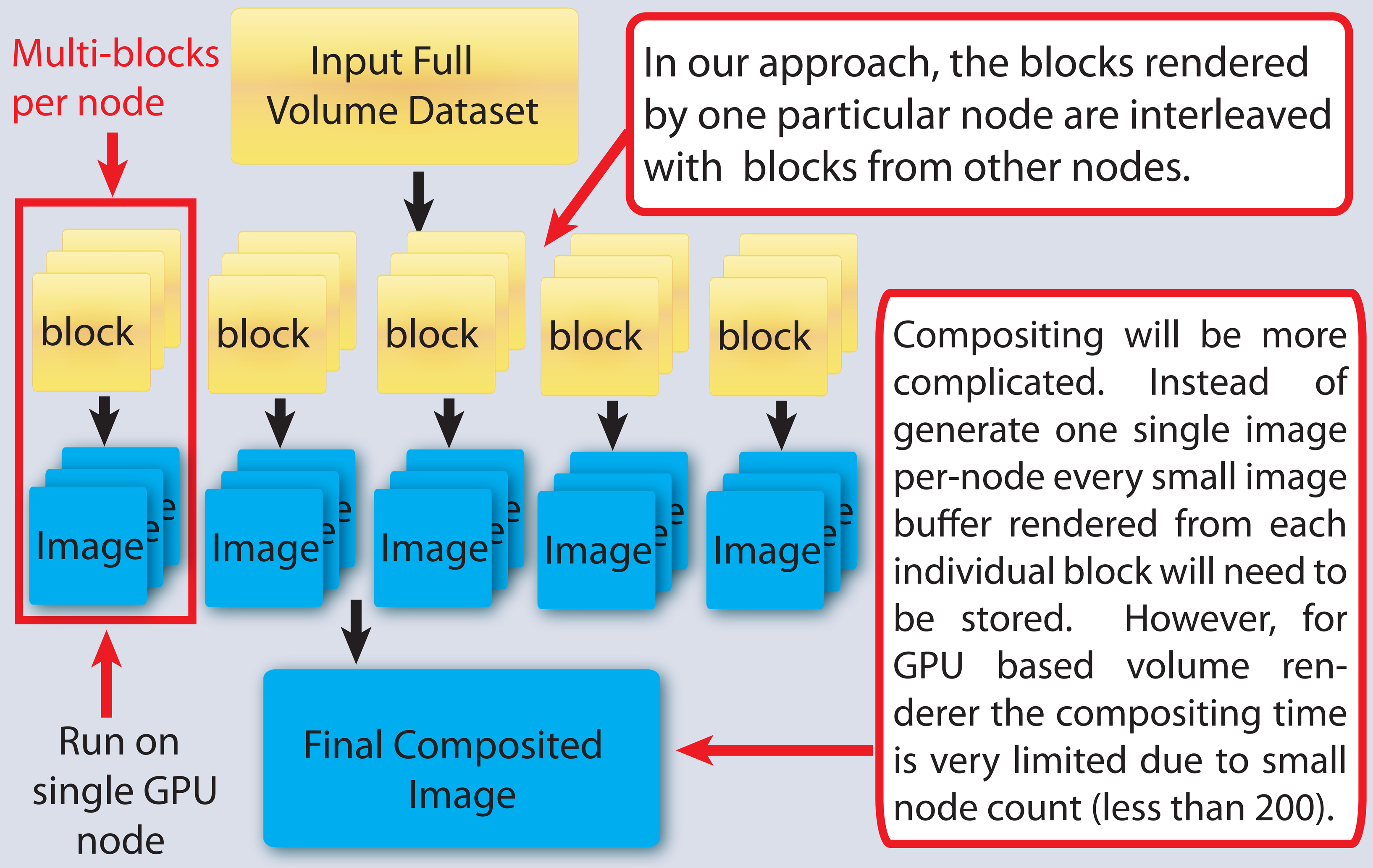




Due to a direction correlation between camera position and volume rendering time, during a volume rendering session changing the view leads to significant load imbalance. For interactive parallel rendering (where a lot of view chaning exist), it is even more important to keep a balanced load. Our proposed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

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 varying dramatically.



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.

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, which correspond to the two peaks seen in the following figures.

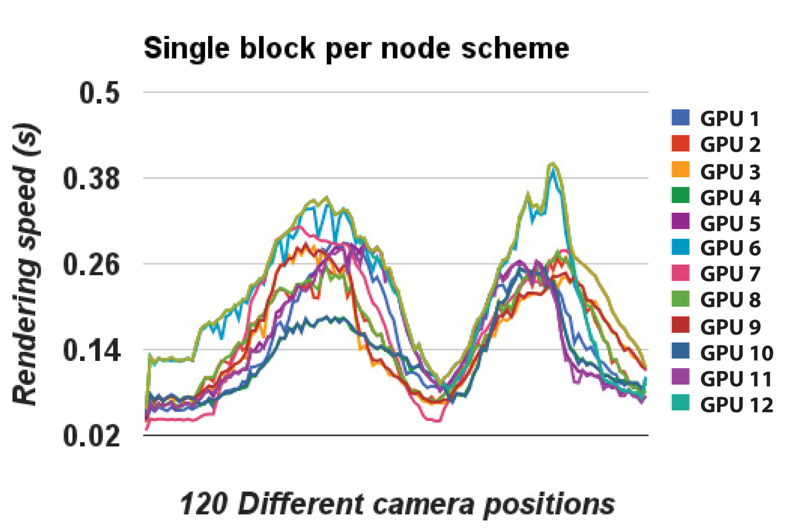


Figure 1

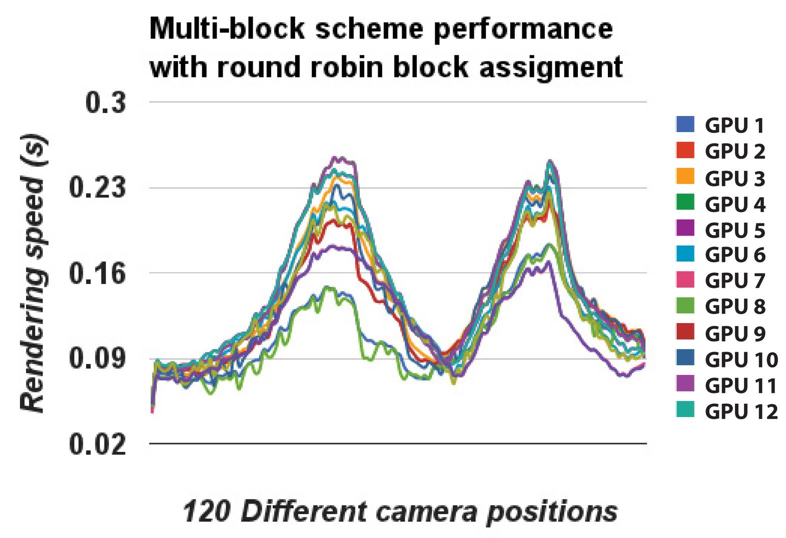


Figure 2

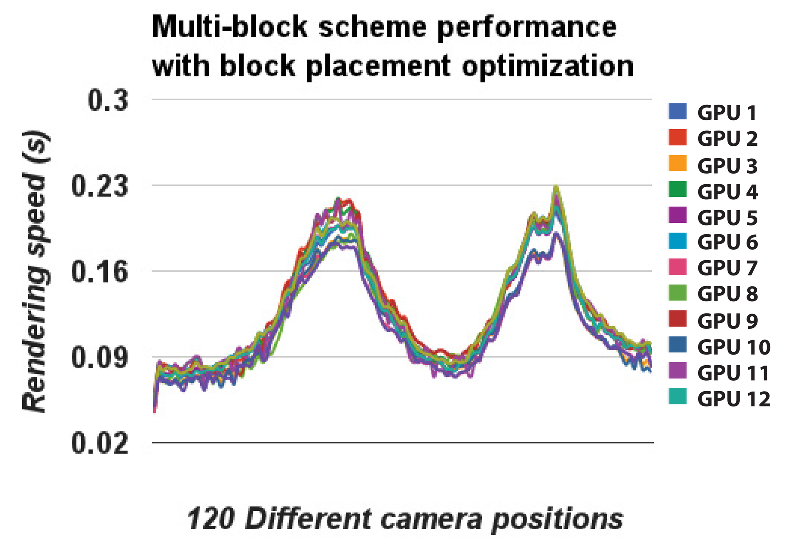


Figure 3

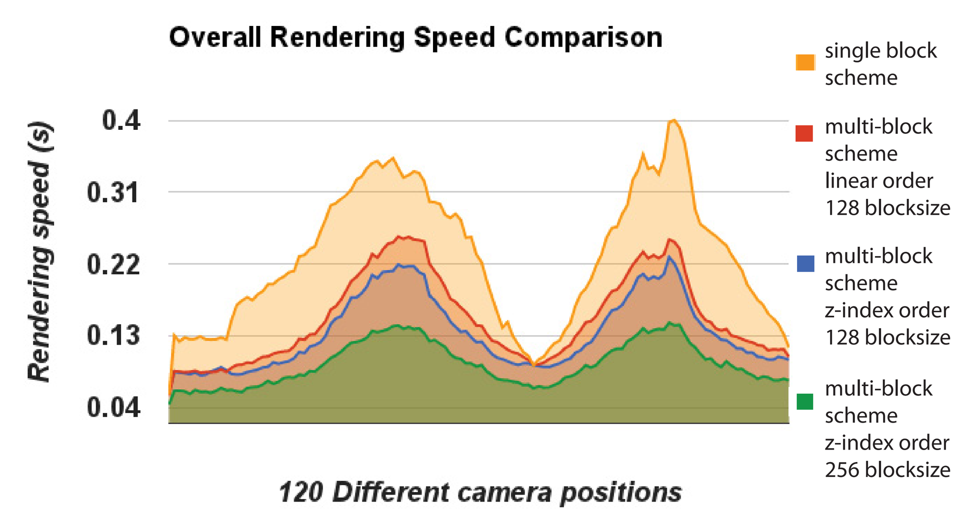


Figure 4

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(5TB rabbit retina dataset). We also plan to improve the block layout algorithm by adopting objective function based optimization techniques.

Reference

T. Fogal, H. Childs, S. Shankar, J. Kr¨uger, R. D. Bergeron, and P. J. Hatcher. **Large data visualization on distributed memory multi-gpu clusters.** In High Performance Graphics, pages 57–66, 2010.

M. Howison, E. W. Bethel, and H. Childs. **Hybrid parallelism for volume rendering on large-, multi-, and many-core systems.** IEEE Trans. Vis. Comput. Graph., 18(1):17–29, 2012.

K.-L. Ma, J. Painter, C. Hansen, and M. Krogh. **A data distributed, parallel algorithm for ray-traced volume rendering.** In Parallel Rendering Symposium, 1993, pages 15 –22, 105, oct 1993.

S. Marchesin, C. Mongenet, and J.-M. Dischler. **Dynamic load balancing for parallel volume rendering.** In EGPGV, pages 43–50, 2006.

CEDMAV

Argonne NATIONAL LABORATORY