Asynchronous BVH Reconstruction on CPU-GPU Hybrid Architecture

Jin-Woo Kim*[1] Jung-Min Kim[1] MinWoo Lee[1] Tack-Don Han†[1]

[1]Department of Computer Science, Yonsei University, Korea

1 Introduction

We present a CPU-GPU hybrid architecture with an extended asynchronous bounding volume hierarchy(BVH) construction scheme [Ize et al. 2007] for ray tracing dynamic scenes. Because the performance of a ray tracer greatly depends on the acceleration data structure, it is necessary to efficiently update the AS of dynamic scenes. To achieve this, Asynchronous BVH construction simultaneously performs both BVH reconstruction and bounding volume(BV) refitting. However, it may degrades the overall performance because it exploits the CPU to simultaneously perform both the BVH construction and rendering. To solve this problem, we improve this multi-core-CPU-based scheme by utilizing a CPU-GPU hybrid architecture. This hybrid approach is implemented based on [Aila and Laine 2009]. It improves the performance by 254~303% compared to a key-frame renderer implemented on NVIDIA OptiX.

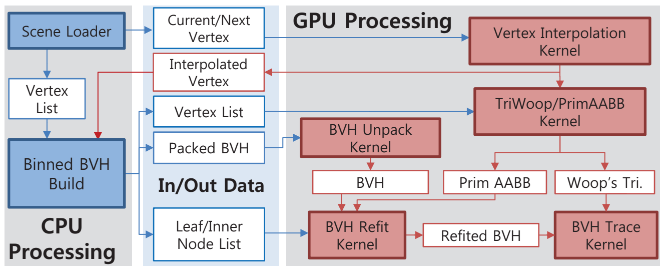

Figure 1: *CPU-GPU hybrid approach*

2 Our Approach

In our architecture, the CPU only performs scene management and BVH reconstruction, while the GPU performs BV refitting, triangle preprocessing and rendering. The ray generation and shading are processed with the same kernel in [Aila and Laine 2009]. To apply the asynchronous BVH construction to our hybrid approach, other processes are altered as follows.

BVH construction on CPU The CPU constructs the BVH using interpolated vertices received from the GPU. A BVH update based on refitting can degrade the tree quality. Thus, the CPU should rebuild the tree as quickly as possible. To do this, the BVH reconstruction utilizes a binned Surface Area Heuristic(SAH) algorithm, which has an advantage related to the speed of the reconstruction rather than the tree quality. This BVH construction eventually results in a deep tree that has only one primitive per leaf node to eliminate the loop for intersecting with primitives of each leaf node.

BVH conversion on CPU The constructed BVH is a typical depth-first tree, and is not adequate for hierarchically parallelized BV refitting. Thus, conversion to a breadth-first tree is needed. In this conversion, range information is extracted from the leaf node list and inner node list, which are required for the refit kernel. In addition, the axis-aligned bounding box (AABB) in the BVH is eliminated because it is updated in the refit kernel in the GPU, eliminating the need to contain it in the BVH in the CPU memory. Removing the AABB from the BVH reduces the total amount of data transfers between the CPU and the GPU by 87.5%.

Vertex interpolation on GPU To generate an intermediate frame between key frames, the GPU interpolates the vertices in the current scene and next scene, proportional to the execution time. This process is very simple and proceeds rapidly in the GPU.

Triangle preprocessing/AABB on GPU With the interpolated vertices, the GPU preprocesses triangles that is needed to accelerate Woop's intersection test. This process uses non-sequential memory access. To improve this, triangle preprocessing and the AABB operation, which have the same input, are performed simultaneously.

Bounding volume refitting on GPU The refitting kernel refits the leaf nodes first, using the leaf node list. After that, it does inner node refitting using the inner node list. A leaf node always contains only one primitive, which eliminates the primitive-tracing loop and subsequently increases the efficiency of GPU operations.

Scene(triangle)	Rendering time(ms)		
	Ours	*NVIDIA Optix*	*Intel Embree*
FairlyForest(147K)	20.68	52.67	103.43
Dragon&Bunny(253K)	10.99	28.00	55.76
Breaking Lion(1.6M)	27.91	84.62	65.62

Table 1: *Experimental results(primary+shadow rays), Resolution: 1920x1200*

3 Results and Conclusions

Table1 lists the results[1] of an experiment carried out using our proposed method on a GTX780, i7-4770K 3.5GHz environment, utilizing the full Kepler kernel. Our CPU-GPU hybrid scheme improves the performance of GPU-based OptiX and CPU-based Embree by up to 303% and 507%, respectively.

References

AILA, T., AND LAINE, S. 2009. Understanding the efficiency of ray traversal on gpus. In *Proceedings of the Conference on High Performance Graphics 2009*, ACM, New York, NY, USA, HPG '09, 145–149.

IZE, T., WALD, I., AND PARKER, S. G. 2007. Asynchronous bvh construction for ray tracing dynamic scenes on parallel multi-core architectures. In *Proceedings of the 7th Eurographics Conference on Parallel Graphics and Visualization*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, EG PGV'07, 101–108.

NAH, J.-H., KIM, J.-W., PARK, J., LEE, W.-J., PARK, J.-S., JUNG, S.-Y., PARK, W.-C., MANOCHA, D., AND HAN, T.-D. 2014. HART: A hybrid architecture for ray tracing animated scenes. *submitted to IEEE TVCG*.

*e-mail:jwkim@msl.yonsei.ac.kr
†e-mail:hantack@msl.yonsei.ac.kr

[1]The experimental results is used as a comparison data in [Nah et al. 2014]. The detailed experimental results are included in our technical report(`http://msl.yonsei.ac.kr/TR/msl2014-01.pdf`).