Advanced Graphics - Assignment 2: Interactive Ray Tracer (BVH)

Dustin Meijer(5726328) & Luuk van de Wiel(4088212)

Introduction

In this report we will describe our project for Assignment 2 of Advanced Graphics. We used a BVH to achieve real-time rendering for many triangles.

Implemented functionality:

First of all, our camera was a bit broken in the first assignment, and it sadly is still broken. We hope that this will not result in a deduction of our grade, since the assignment is about BVH construction and traversal.

We implemented a BVH, which can be built in multiple ways. We use the binning [1] approach combined with SAH [2]. In our code it's also possible to use "normal" SAH or a median-split BVH. As expected, the median-split BVH gives the worst results. The "normal" SAH is too slow to use for kind of big scenes, such as the Stanford Bunny (~70k triangles). Therefore, it only is logical that we use the combined SAH and binning approach, where SAH is used in each bin.

We use 16 bins, because in [2] it is stated that "16 bins seem to be very close to the optimum".

This gave us a tremendous speedup. With f16.obj (4056 tri's), we got ~0.025 fps when not using the BVH. When using the SAH binned BVH, we got ~15fps.

Our ShadowRays have an early out when traversing through the BVH, to increase efficiency.

When building the BVH, we use "smart" bounds calculation for Partitioning, which consists of adjusting the bounds of a candidate left and right node, when a primitive is added to the candidate node, instead of calculating the bounds afterwards. This sped up BVH construction a lot.

We tried to implement ordered traversal, but we weren't able to achieve speedups, sadly. We implemented option 1 and option 3, as described in the slides. Both caused a slight drop of the fps, actually. This could be caused by the extra work that needs to be done per traversal, although we are not certain. We even implemented option 3 with saved computations (which means that we only do the computations which node needs to be traversed once, and using that value in further frames), but this also wasn't faster (probably because of big arrays which need to be retrieved from memory).
We acknowledge that we were not able to fix these slowdowns, but we added all these 3 traversal options in our BVH.cpp, to show our work. Preprocessor directives guide which one needs to be used (the one with 1 is used, the other ones with 0 aren't).

We created 3 scenes (with different models[3]) to test our BVH and our renderer. These are also easily changeable with pre-processor directives. These can be found in Scene.h.

Division of work:

BVH framework – Dustin
BVH traversal – Dustin
BVH construction - Luuk
BVH partition - Luuk
SAH – Luuk
Binning – Luuk
Ordered Traversal – Luuk
Report – Luuk
Scenes and models - Luuk

References

[1] On fast Construction of SAH-based Bounding Volume Hierarchies, Wald, 2007

[2] Heuristics for Ray Tracing using Space Subdivision, MacDonald & Booth, 1990

[3] http://www.prinmath.com/csci5229/OBJ/index.html

Relevant note

As you are probably aware, Dustin received some tragic news regarding his father. Therefore, I (Luuk) did all of the later work on our practical assignment, as is indicated by the division of work. Since these are unusual circumstances, I would request that Dustin not get a deduction in his grade because of this.