7
Conclusion

In this work we have successfully demonstrated a complete ray-tracing solution, capable of interactively rendering dynamic scenes including illumination effects such as shadows and reflections. The chosen acceleration structure, the Uniform Grid, was capable of maintaining ray-tracing scalability while allowing full reconstruction to support unstructured movements in the scene.

In order to explore the processing power of current GPUs, we have developed a data-parallel grid reconstruction algorithm capable of obtaining fast and scalable rebuild times. Additionally, we have presented optimized ray-traversal, intersection and shading routines inside the graphics hardware.

After all tests, we can conclude that our proposed ray-tracing solution has successfully performed better than state of the art research using Uniform Grids and BVH, both on the CPU [Wald et al. 2006, Wald et al. 2007]. On the other hand, our technique has presented slightly inferior performance than both CPU [Shevtsov et al. 2006] and GPU [Zhou et al. 2008] kd-tree implementations. In this case, however, we have only been able to evaluate sparse scenes ("Toys" and "Forest"). These present a worst-case scenario for the Uniform Grid, requiring a more adaptive structure.

In addition, our proposed GPU implementation of the Uniform Grid rebuild has performed significantly faster than other state of the art results on the CPU [Wald et al. 2006]. Performance figures demonstrate that our current bottleneck is the ray-tracing step. There are several improvements that can be done in order to further optimize this implementation. These and other future work are discussed in the next section.

7.1
Future Work

Following this work, there are several research topics that can be further investigated. A major improvement to our bottleneck in the ray-tracing algorithm would be to use a more adaptive acceleration structure. For instance, our current grid construction algorithm can be modified to build multi-level Uniform Grids.

There are several ways to organize grids hierarchically, including loosely nested grids [Cazals et al. 1995, Klimaszewski and Sederberg 1997], recursive or multiresolution grids [Jevans and Wyvill 1989], and macrocells or multi-grids [Parker et al. 2005].

With knowledge of the behavior of each scene object, it is also possible to assign independent Uniform Grids to each moving and deformable object. This would guarantee tightly packed structures, with a minimal number of empty cells. In this case, rigid body movement could be simply treated by transforming the ray into local object space, as in [Wald 2004].

All these techniques share the same idea of subdividing some regions of space more finely than others, and thus traverse empty space more quickly than populated space. Another structure modification with the same goal would be to use a flag to skip empty cells along the ray. In this case, it is possible to store in each cell a value to encode the minimal step distance the ray can safely travel until finding the next non-empty cell. This is similar to [Baboud and Décoret 2006].

A further extension to this work would be to explore tracing ray bundles. As in [Wald et al. 2006], it is possible to trace packets of rays through a hierarchical Uniform Grid using several optimizations not investigated in our work. A modified GPU ray-tracing implementation with these optimizations could achieve an order of magnitude in performance gains.

Finally, the GPU hardware could be used for rebuilding other successful acceleration structures, such as the BIH or the BVH. Together with an optimized ray-traversal procedure fully inside the graphics hardware, it may be possible to achieve even better results for ray-tracing dynamic scenes.