Advanced Graphics 2022/2023 – Assignment 3 (FINAL)

Introduction

For this assignment, you have considerable freedom. ***Assignment 1*** was about light transport fundamentals and setting up the framework for basic ray tracing. ***Assignment 2*** dealt with efficient ray/scene intersection: the low-level core of any ray tracing based renderer. For ***Assignment 3*** you either build on this by implementing recent research.

Assignment

Broadly speaking, there are four areas to work on, linked to the theory presented in the lectures:

1. Acceleration structures:
    a. Expanding on assignment 2, implement the paper "Spatial Splits in Bounding Volume Hierarchies" by Stich at al. (click for download). This yields a high-quality BVH, at the expense of construction time. Note: Proof that the resulting tree is of a high quality is required. *Difficulty: medium/hard.*
    b. Or, go the opposite route: implement the paper "Bonsai: Rapid Bounding Volume Hierarchy Generation using Mini Trees" by Ganestam et al., which maximizes build performance. Note: build performance must be roughly in line with the numbers in the paper. *Difficulty: medium/hard.*
    c. Starting from a TLAS/BLAS builder from the tutorial series, add 'partial rebraiding', based on the paper "Improved two-level BVHs using partial re-braiding" by Benthin et al. Prove that the resulting TLAS/BLAS performs in line with the findings in the paper. *Difficulty: easy*.
    d. Other options exist. There is a paper by NVIDIA that explains how to efficiently traverse a (CPU-built) 8-wide BVH on the GPU for state-of-the-art #RTXoff performance *(hard)*. There exist other heuristics than SAH; an implementation and an in-depth analysis may provide you with an interesting project *(easy)*.
    e. *Feel free to propose something interesting!*

2. Physically based rendering, options:
    a. Implement a basic but correct bidirectional path tracer, as described by Veach in his Ph.D. thesis (as well as in many other sources). Proof the correctness of your implementation by comparing energy levels. Warning: *hard*.
    b. Implement photon mapping, as described by Henrik Wann Jensen ("Global Illumination using Photon Maps", 1996). Your implementation should correctly handle caustics. Verify your result against ground truth produced by a path tracer. Warning: must produce correct energy levels. *Difficulty: easy – medium.*
    c. Implement a basic but correct volumetric path tracer for non-homogeneous media, e.g. a cloud. Provide a test scene that clearly demonstrates the functionality. *Difficulty: medium.*
    d. *Other options: feel free to propose something interesting.*

3. Filtering & reprojection, options:

    a. Implement a filter that combines reprojection and spatial filtering to improve image quality. Suggestion: use a simple scene to make reprojection worthwhile. *Difficulty: easy.*
    b. Implement Adaptive Sampling (see e.g. "A Survey of Adaptive Sampling in Realistic Image Synthesis", M. Šik. *Difficulty: easy.*
    c. Add the "Open Path Guiding Library" to your existing renderer and report on the integration process. *Difficulty: easy, I think.*
    d. Implement a recent paper on the topic of filtering (see 4).

    Filtering algorithms can be conveniently implemented on the CPU. You will receive a (modest) bonus for a GPU implementation. Do take into account that this significantly complicates the work.

4. Implement a recent paper in the field of graphics. Some options:

    a. "Path Guiding in Production", by Vorba et al., 2019, see: https://cgg.mff.cuni.cz/~jirka/path-guiding-in-production/2019/index.htm Path guiding is a class of 'learning algorithms', which estimate importance of directions over the hemisphere based on earlier transport results. *Difficulty: medium.*
    b. "Direct Ray Tracing of Smoothed and Displacement Mapped Triangles", Smits et al., 2000. The paper describes a method for directly rendering displacement maps in a ray tracer. The method is computationally expensive, which made it impractical in 2000. Now, in 2020, things have changed. *Difficulty: I think hard. (2023 update: yes, hard).*
    c. "Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting", Bitterli et al., 2020. As discussed in the slides: a method for importance sampling thousands of lights. *Difficulty: medium, but potentially a lot of work.*
    d. Implement Wavefront Path Tracing on the GPU.

If you have a different project in mind that matches the intended scope and general topic of the course, feel free to discuss this with me.

Starting Point

The expected starting point for this assignment is the code you handed in for assignment 2. However, you are free to use a different starting point, e.g. code from the BVH articles, or an open source renderer such as Lighthouse 2, Mitsuba or PBRT. Make sure to reference the code you use and make clear what extension you added.

Language Notes

This assignment may be executed in a programming language of choice. Support on the implementation side will be mostly limited to C++ and C# however, and performance is expected to be optimal for C++ code. Choice of programming language will not play a role in grading.

Practical Details

The deadline for this assignment is ~~*Wednesday February 1st*~~ **Friday February 3rd, 17.00**. You may hand in your assignment up to 24 hours late in exchange for a 1 point penalty. The materials to submit are:

- your project, including sources and build instructions (if these are not obvious);
- a brief report, detailing implemented functionality, division of work, references and other information relevant to grading your submission.

As with the previous assignments, you may work on this assignment alone, or with one other student.

Feel free to discuss practical details on Teams. You are not supposed to share complete ray tracers there, but if everyone uses the same optimal ingredients, that would be considered 'research'.

Tasks & Grading

Given the increased freedom for this assignment, grading is going to be somewhat subjective. Generally speaking, a passing grade (6) for this assignment requires successful implementation of a somewhat recent paper.

To obtain additional points:

1. deliver a high-quality implementation (in terms of robustness / performance);
2. implement a relatively challenging technique;
3. ~~combine challenges~~; *(don't do this; not fair to one-person teams)*
4. implement a technique on the GPU;
5. produce an interesting demo (in terms of scene or animation or produced image);
6. analyze the characteristics of your implementation (image quality, speed, etc.).

Obviously, many other options exist. Contact me if you want to discuss an idea of which you are not sure whether it is worthwhile.

Purpose

After successfully completing this assignment, you have obtained theoretical and practical knowledge on algorithms for efficient physically based rendering. This is a solid foundation for further research in the field of graphics.

May the Light be with you,
- Jacco.