Advanced Graphics Assignment 3

Hui En Lin (8098735), Thijs van Loenhout (2498790)

February 3, 2023

Introduction

This document briefly discusses assignment 3 of the advanced graphics course (as thaught in 2022-2023). We have decided to implement photon mapping. All work on this assignment was done by us – Hui En and Thijs – and build instructions are the same as for the provided template. Aside from the material on the slides, the template and the lecturer's blog, we used the following outside recources for inspiration:

- Jensen's 1996 paper on photon mapping
- A blog post explaining photon mapping
- A GitHub repository (primarily used for as a reference for the storage of the photon map and traversal using a kD-tree).

All screenshots in this report use the same scene containing a completely specular sphere, a diffuse torus and a glass cube. For reference, Figure 1 depicts this scene using our path tracer.

Overview of functionality

We have extended the work of the previous assignments to include photon mapping functionality. In particular:

- `photon_map.h` includes the architecture for photon maps. We use kD-trees to store them.
- `createPhotonMap()` creates a global photon map (for the caustics map, see the section on Limitations).

  Each photon that is send, is send from a random position on a random light, with a random direction.

  Each photon is processed in a way similar to our path tracer, either through the `PhotonPath()` or `PhotonPathwCols()` methods (the former not including albedo info, while the latter does). These methods support Russian roulette.
- Using an adapted version of the Whitted style ray tracer and `avgPhotonPow()` to retreieve local photon information,

A photon map with 100000 photons is visualized in Figure 2. Note how the specular sphere is correctly black, as no photons are stored there. It's hard to see, but there are hints of 'caustic' photons.

A visualization with the ray tracer using this photon map is shown in Figure 3. This is a result we're not super content with, but there are some positives: all three material types are supported and there are hints of caustics (even blending the red and blue colors).

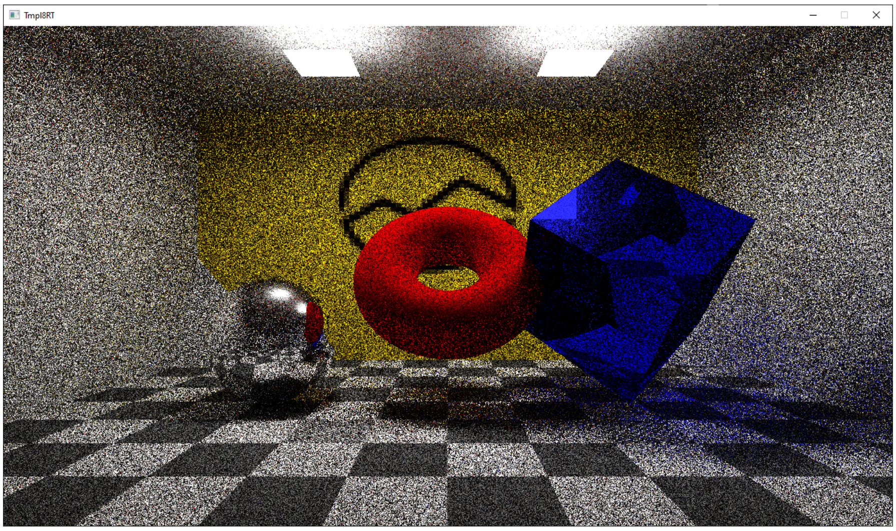

Figure 1: Path traced result after 60s of rendering.

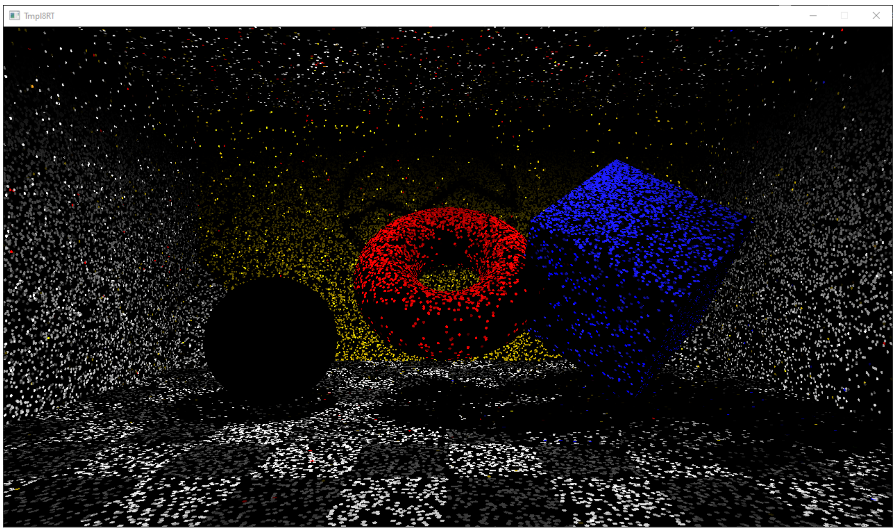

Figure 2: Visualization of photon map using 10000 photons

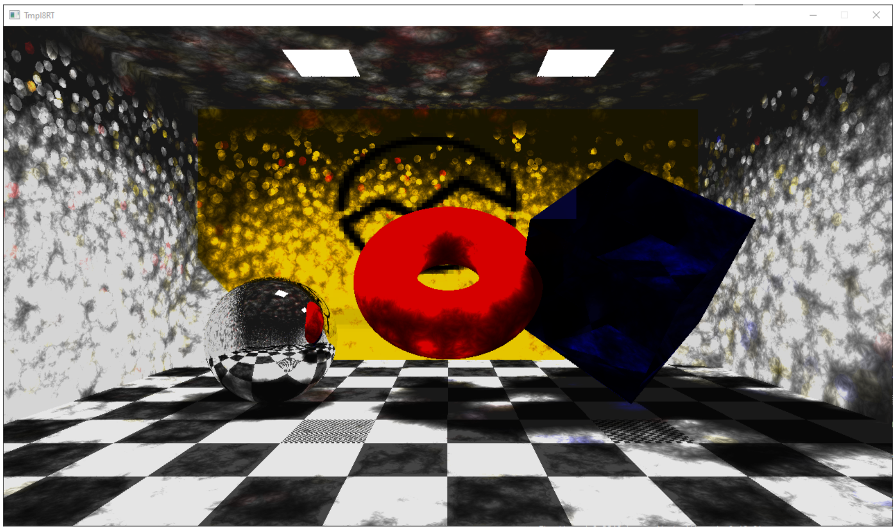

Figure 3: Ray tracing with the photon map using 10000 photons. Each vertex searches the 10 nearest photons.

Limitations

Currently, we do not split the global and caustics photon map, as the work by Jensen suggests, because were not able to figure out how to combine them directly. Instead, we use one photon map that combines light intensity (through density) and albedo information.

Also, we have not been able to properly divide flux over the photons, hence our resulting illumination levels do not match that of our path tracer. In fact, our struggle with this, is also the cause of our aforementioned problem with the caustics map.

Finally, using our photon map with our Whitted style ray tracer, creates ‘patches’. We do not know why, but suspect this has to do with the resolution of our photon map: increasing the number of photons, decreases the patchiness. However, our machines couldn’t handle more than we’re already using.