Progressive Photon Mapping

HANSUNG KIM, University of California, Berkeley, United States
LINYUE SONG, University of California, Berkeley, United States
RAHUL ARYA, University of California, Berkeley, United States
SEAN MURPHY, University of California, Berkeley, United States

ACM Reference Format:
Hansung Kim, Linyue Song, Rahul Arya, and Sean Murphy. 2018. Progressive Photon Mapping. In . ACM, New York, NY, USA, 2 pages.
https://doi.org/XXXXXXX.XXXXXXX

1 CURRENT PROGRESS

We currently finished an implementation of a basic path tracer in CUDA on GPU. The path tracer features diffusive and specular materials, importance sampling from the material BSDF, and configurable path length and samples per pixel. We ensured that the implementation outputs visually correct renders when compared to the previous CPU-based assignments. Figure 1 shows an example render of a cornell box using our current CUDA path tracer implementation.

Our current progress also includes a partial implementation of progressive photon mapping. We have first modified our base code so that each ray-tracing pass will identify "hit points" in the scene that contribute to each pixel of the rendered output. We have then implemented a photon mapping pass that is separate from the ray tracing pass, where a configurable number of photons are generated and shot from the light source to determine its hit point with the scene geometry. We have also implemented a simple array-based global data structure that stores the hit point data obtained from the ray tracing pass for use in the photon mapping pass, where the structure is queried with each photons to accumulate flux and update radius value around each hit point. During the rendering phase, we will use the accumulated flux on each hit point to render image.

2 FUTURE PLANS

We are roughly half a week behind our originally planned schedule, but we still expect to be able to achieve the original goal of having a complete implementation of progressive photon mapping and possibly a machine learning-based improvement. We adjusted schedule by skipping implementation of a non-progressive photon mapping renderer.

We plan to move to a kd-tree data structure to store hit points during photon mapping (rather than using a list.) Moreover, we want to implement a BVH to accelerate our ray tracing and improve our caustics - this will be important when rendering more complex scenes. We have rough designs for how to implement both of these data structures in a manner that supports fast access on GPU as well as thread-safety, and are presently working on the implementation.

Manuscript submitted to ACM

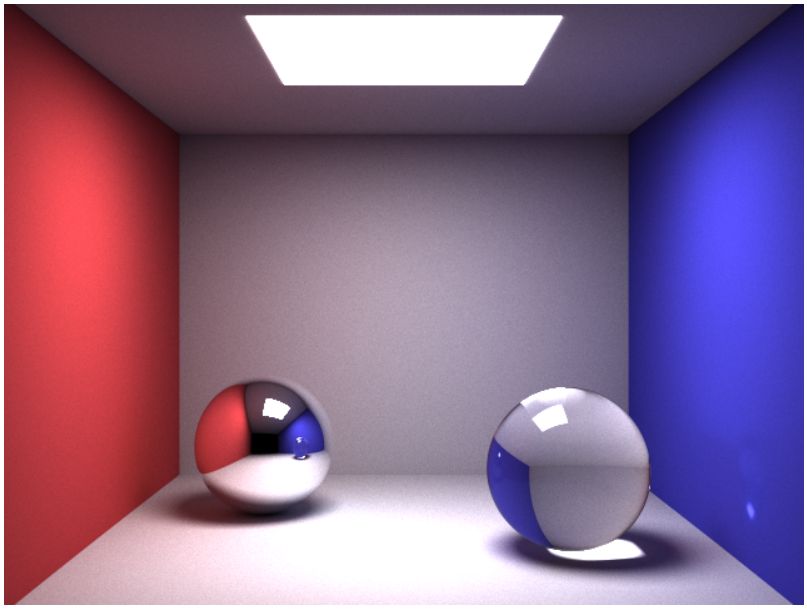

Fig. 1. A Cornell box render using our current path tracer implementation.

We have also begun initial drafting on the ML-acceleration portion of this project, where we will train a model to predict the flux on each hit point based on a smaller sampling of neighboring photon. Our hope is that this will reduce the number of photons needed to obtain a realistic image.