Advanced Topics in Computer Graphics I

Universität Bonn
Institut für Informatik II
June 17, 2025

Summer term 2025
Prof. Dr. Reinhard Klein
Domenic Zingsheim

Sheet R09 - Bidirectional Path Tracing

Hand in your solutions via eCampus by Tue, 24.06.2025, **12:00 p.m.**. Compile your solution to the theoretical part into a single printable PDF file. For the practical part, hand in a single ZIP file containing only the exercise* folder within the src/ directory. Please refrain from sending the entire framework.

In this exercise, we will take a look at bidirectional path tracing from a practical and theoretical point of view. Because this algorithm is not so straightforward to implement, we decided to only count half of this sheet's points towards the total maximum. This way, we want to encourage you to really work on these exercises and if you submit a good solution (even if it is not complete or not everything is working correctly), you can still get full points. If you did everything correct, you can get 4 additional bonus points in each exercise.

Assignment 1) Bidirectional Path Tracing *(4 + 4 Bonus Pts)*

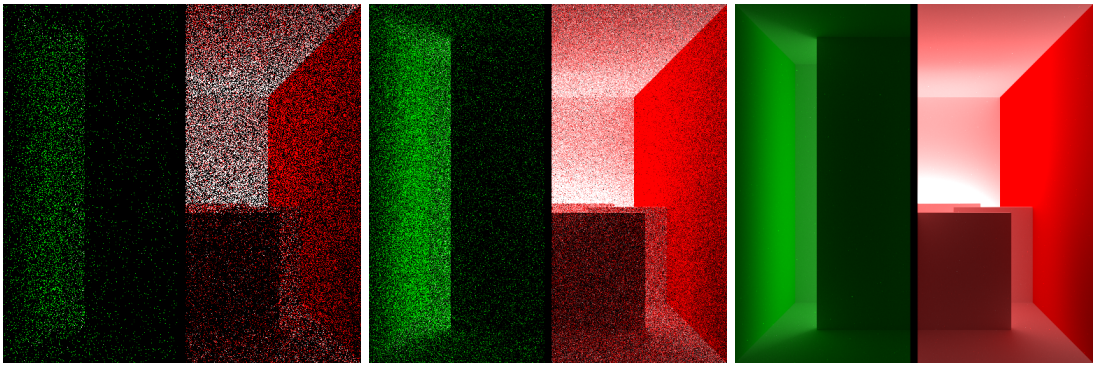

Figure 1: Test scene containing mostly long light paths rendered with BDPT using 1, 10 and 10^5 samples per pixel, respectively. A single sample entails the full connection of a camera and light subpath of length 8, each. To obtain these renderings, run `./bin/exercise09_BDPT -s data/exercise09_BDPT/bdpt.xml`.

In this exercise, you will complete a bidirectional path tracer. Instead of simply tracing rays through the camera and accumulating radiance, the bidirectional path tracing ray generator is more complex. The integration is split across three kernels:

a) A path starting at the camera is created for each pixel in the image.
b) A path starting at an arbitrary light source is created for each pixel in the scene. The photon sampling routine from the previous exercise sheet is reused to initialize the path.
c) All possible subpaths of the two generated paths are combined to connect the light source to the camera. Note that due to practical constraints, we can only create light paths of an apriorily determined maximum length since we need to store all path vertices in steps 1 and 2.

For the two path generation steps, a stripped down version of the main loop of a path tracer is used, that does not accumulate any radiance form light sources (for example via next-event estimation). Your

task is to complete the accumulation of radiance when connecting paths in `__raygen__combine()` in `bdptraygenerator.cu`. In the two nested for loops iterating over all feasible subpath combinations, connect the selected vertex on the camera subpath (index s) to the selected vertex on the light subpath (index t), and compute its "throughput" from the light source to the camera, i.e. the transported radiance divided by the sampling probability. Before accumulating the throughput, compute the MIS weights that account for the probability of creating the same path by connecting vertices $s+i$ and $t-i$, which yields a path of the same length. For further information we recommend reading chapter 10 of Veach's course at Stanford [1].

Theoretical Assignments

Assignment 2) Bidirectional Path Tracing with NEE *(4 + 4 Bonus Pts)*

In the practical part we did not accumulate any radiance when a camera path lands on a light source, nor do we perform any next event estimation.

a) Assume that the bidirectional path tracing algorithm that you implemented in the practical part is extended with radiance accumulation on the camera subpath, whenever a path vertex x_L is generated on a light source due to BSDF importance sampling, yielding the path $\bar{x}_L = (x_0, \dots x_L)$ with non-zero light transport. From a theoretical standpoint, define the updated MIS weights for each BDPT path sample $\bar{x}_{s,t} = (x_0, \dots, x_s, y_t, \dots, y_0)$, that accounts for the possibility of such an event, and the MIS weight that should be used for the contribution of the path $\bar{x}_L$.

b) Additionally assume that we also perform next-event estimation to an arbitrary light source on the camera subpath, yielding paths $\bar{x}_{NEE} = (x_0, \dots, x_i, x_L)$, where x_L is a vertex on a light source due to next-event estimation. From a theoretical standpoint, define the updated MIS weights $\bar{x}_{s,t}$, $\bar{x}_L$ and $\bar{x}_{NEE}$, that accounts for the possibility of *both* additional events.

You can assume that there are no perfectly specular materials in the scene, light sources are emitting light diffusely, and you are not connecting light paths directly to the camera vertex x_0. Use expressions like $p(x_i \mid x_{i-1}, x_{i-2})$ to denote the probability of sampling vertex x_i via BSDF importance sampling at x_{i-1}, for example, or $p(x_{L-1} \mid x_L)$ if x_{L-1} is the result of diffusely emitted light at x_L, or $p(x_L)$ for the probability of the position x_L being selected as a first vertex on the subpath starting at a light source.

Hint: It is recommended to exclusively work with pdfs with respect to surface area measure (probabilities of positions x_i on surfaces) instead of solid angle measure (probabilities of directions ω_i on the hemisphere).

References

[1] Eric Veach. Chapter 10: Bidirectional path tracing. In *CS348b-03*. URL: `https://graphics.stanford.edu/courses/cs348b-03/papers/veach-chapter10.pdf`.

Good luck!