sap_0279

GPU-based light wavefront simulation for real-time refractive object rendering

Gernot Ziegler[1]
Christian Theobalt[1]
Ivo Ihrke[1]
Marcus Magnor[2]
Art Tevs[1]
Hans-Peter Seidel[1]
[1] MPI Informatik, Saarbrücken, Germany
[2] Technical University Braunschweig, Braunschweig, Germany

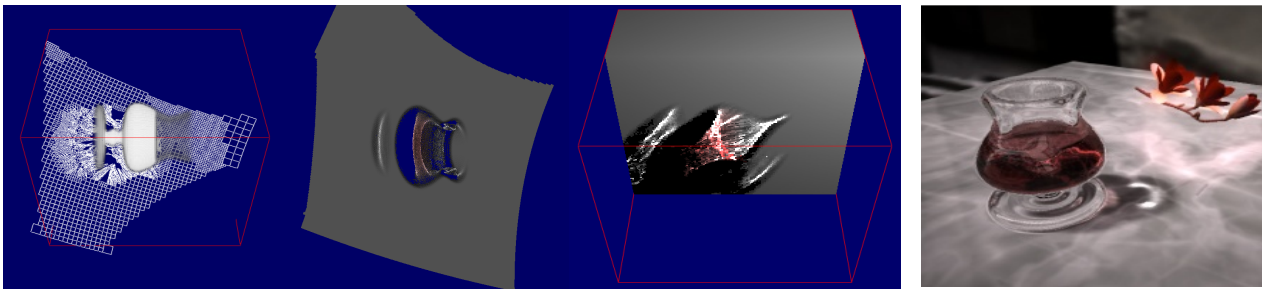

Figure 1: Left to right: Adaptive light wavefront simulation calculates the modified light distribution and protocols it into an irradiance volume. The real-time viewer finally embeds the refractive object and its lighting into a 3D-scene.

1 Introduction

In our paper on Eikonal Rendering [1], presented in the SIGGRAPH 2007 main paper program, we propose a novel algorithm to render a variety of sophisticated lighting effects in and around refractive objects in real-time on a single PC. Our method enables us to realistically display objects with spatially-varying refractive indices, inhomogeneous attenuation characteristics, as well as spatially-varying reflectance and anisotropic scattering properties. We can reproduce arbitrarily curved light paths, volume and surface caustics, anisotropic scattering as well as total reflection by means of the same efficient theoretical framework. In our approach, scenes are represented volumetrically. One core component of our method is a fast GPU particle tracer to compute viewing ray trajectories. It uses ordinary differential equations derived from the eikonal equation. The second important component is a light simulator, which utilizes a similar ODE-based framework to adaptively trace light wavefronts through the scene in a few seconds. While the conference paper [1] focuses on the development of the theory and the validation of our results, this sketch describes in detail the new concepts and data structures that we developed to implement view rendering and light wavefront tracing on the GPU (see figure 1 for a stage overview).

2 Exposition

To efficiently map our concepts onto the GPU we have developed new algorithms and data structures that enable us to strongly accelerate light simulation and achieve real-time view rendering performance. We use a volumetric scene representation to store all relevant data in textures, such as refraction indices, their local gradients and attenuation values describing the local energy loss due to material properties.

Our off-line light wavefront simulation implements fast ODE-based particle tracing using the ray equation of geometric optics, where each particle represents a ray path. The main difference to classic particle tracing is that four particles (the corners of the patches), are bound into one entity, a wavefront patch. The light's approaching wavefront is partitioned into a list of such wavefront patches, and stored in a number of textures. During simulation, patches might grow larger than one voxel due to ray divergence, which is why they must be tesselated occasionally. The energy of each patch is calculated by a fast update rule derived from the intensity law of geometric optics. Patches may also leave the volume or drop below a minimum energy threshold, and thus become irrelevant for the simulation. For these two reasons, we apply HistoPyramid techniques [2] to conduct dynamic patch list administration right on the GPU. HistoPyramids are hierarchical 2D data structures, similar to mipmaps, which assist in the compaction and targeted expansion of 2D data arrays, like our patch list. This way, the wavefront can adaptively tesselate and still keep a low memory footprint while it traverses the refraction volume. The final component in light simulation is wavefront protocolling. Since we can warranty each patch to be smaller than a voxel, we treat them as OpenGL points and scatter them into a flattened 2D version of the output volumes. During the write, all passes can be accumulated, or selective writing can e.g. store only the highest-energy pass or a certain pass count. Simulation ends when all patches have been culled or left the volume.

When simulation is complete (typically after a few seconds, depending on volume resolution and lighting complexity), the results can be viewed as part of a 3D scene. The employed rendering techniques resemble volume raycasting, but keep using the same set of ODE's as mentioned before to deflect the incoming viewing rays. On the path through the volume, each viewing ray accumulates light contributions from the irradiance volume according to a complex image formation model described in [1] and implemented in OpenGL fragment shaders, until it encounters an opaque surface or finally leaves the volume, finishing with an environment map lookup.

We hope that an explanation of our research prototype inspires novel computer graphics concepts for graphics hardware, and foresee interesting extensions into global illumination and raytracing.

References

[1] I. Ihrke, G. Ziegler, A. Tevs, C. Theobalt, M. Magnor and H-P. Seidel: Eikonal Rendering: Efficient Light Transport in Refractive Objects. *Proc. ACM SIGGRAPH 2007 (to be published)*.

[2] G. Ziegler, A. Tevs, C. Theobalt and H-P. Seidel. GPU Point List Generation through Histogram Pyramids. *Technical Reports of the MPI for Informatics*, June 2006, MPI-I-2006-4-002.