Near-Field Illumination for Mixed Reality with Delta Radiance Fields

Tobias Alexander Franke + **Fraunhofer** IGD + **TU** Darmstadt + **tobias.franke**@igd.fraunhofer.de

SIGGRAPH2013 Fraunhofer IGD TECHNISCHE UNIVERSITÄT DARMSTADT

INTRODUCTION

Motivation

Fusing synthetic objects with a real context requires believable interaction of real and virtual light to convince an observer that the rendered result is not merely augmented but part of the scene. Such a mixed reality (MR) system has applications in movie production, gaming, advertisement of unfinished products or cultural heritage visualization. Current interactive MR systems however often disregard proper lighting entirely or greatly simplify shading, excluding light interaction such as indirect light bounces.

Related Work

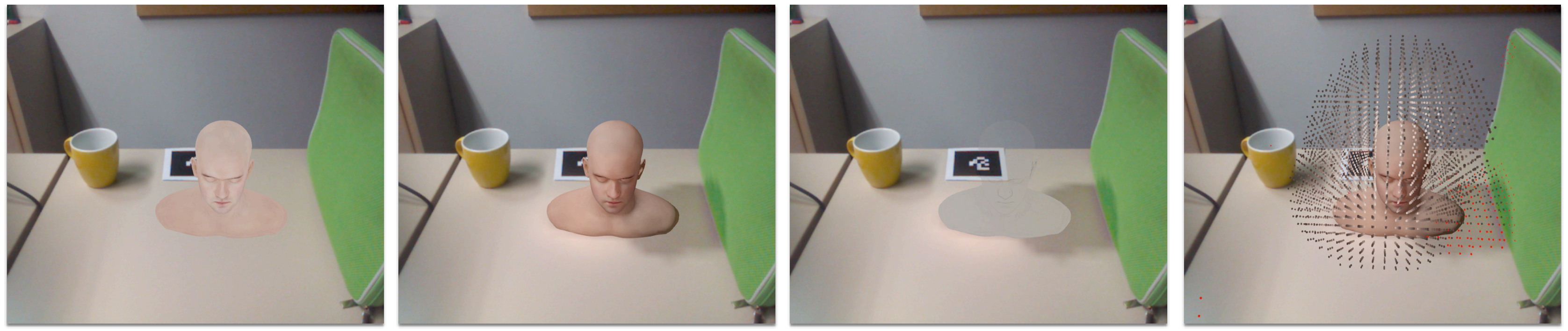
Several relighting solutions assume either rigid synthetic objects, static scenery (i.e. photographs) or have no real-time requirement when calculating a GI solution. Attempts have been made to resolve this issue with Instant Radiosity [3][4]. To suppress flickering, a large number of virtual point lights (VPL) is necessary, drastically taxing execution speed. I propose to model virtual and real light in a unified radiance field to avoid performance issues from oversampling and to maintain temporal coherence.

Theoretical Framework

Given a reconstructed real scene with light sources and geometry and a non-emissive object O to insert, we can imagine a snapshot of the radiance field of the real scene L^μ. We can furthermore think of another radiance field L^ρ of the same scene containing O. The difference between both - the **Delta Radiance Field** - contains the residual light introduced by the additional object as well as antiradiance, compensating for the light which is blocked by it. The expansion of both radiance fields into a Neumann series yields a new linear transport operator $\mathbf{T}_\Delta$, which can be used to relight an existing radiance field.

$$L^\Delta = L^\rho - L^\mu = \sum_{i=0}^{\infty} \mathbf{T}_\rho^i L_e - \sum_{i=0}^{\infty} \mathbf{T}_\mu^i L_e = \sum_{i=0}^{\infty} \mathbf{T}_\Delta^i L_e$$

Figure 2: Setup overview. A RGB-D sensor (Microsoft Kinect) is used to track a reconstructed model of the scene and geometrically register a synthetic object. A second fisheye camera (uEye UI 2230-C) captures incident real light. Marker tracking is done with AR Toolkit Plus and the final image is rendered with Direct3D 11 on a nVIDIA GTX 470.



(a) (b) (c) (d)

Figure 1: Head model (courtesy of Infinite-Realities) inserted into a real scene with one reconstructed light source: (a) a synthetic object is inserted without illumination, (b) visible first bounce around the base as well as low resolution shadow (32 propagations, 512^2 VPLs, 11ms per frame), (c) indirect effects without synthetic object for better visualization, (d) visualization of the DLPV (red dots indicate negative values).

METHOD

Practical Framework

To simulate L^Δ and operator $\mathbf{T}_\Delta$ I use a small volume (32^3 voxels) centered around the object O with dimensions twice the size of the largest bounding box edge. Similarly to *Light Propagation Volumes* (LPV) [2], indirect bounces created from Reflective Shadow Maps (RSM) are injected into the volume and then propagated. Additionally to the indirect bounces however, direct light is also injected into the volume in order to retrieve shadowed areas from blocked direct light sources. The result is called *Delta Light Propagation Volume* (DLPV).

Setup

- Required hardware → Figure 2
  - RGB-D sensor: view of the scene where object O is inserted
  - Fisheye camera: pointed upwards to record incident real light

Reconstruction

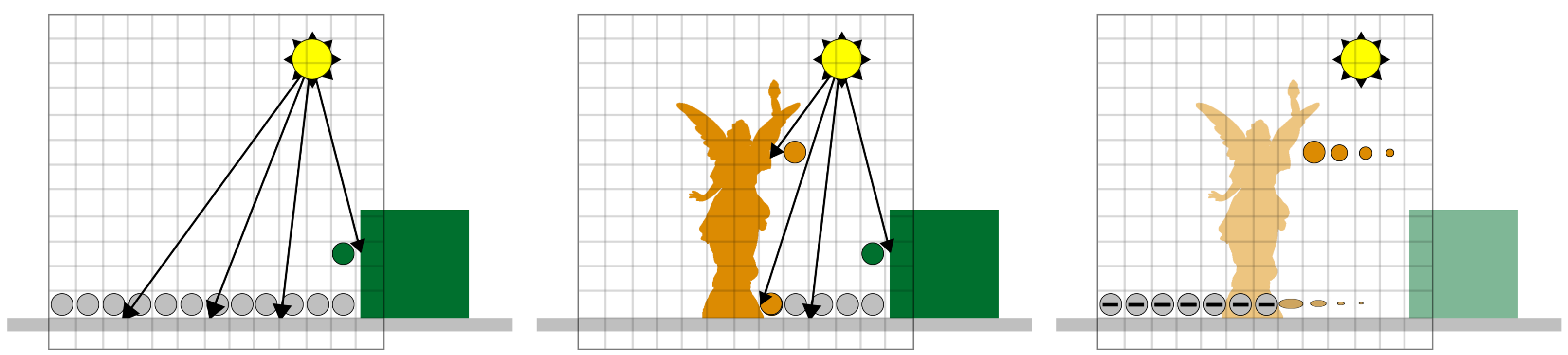
- Real light sources
  - Extract a fixed number of point light sources with a variance minimized median cut algorithm from the fisheye image
  - Alternatively track a fixed number of light sources with markers in the camera image
- Real scene
  - Track a manually reconstructed ghosting scene with a marker
  - Estimate simple diffuse albedo parameters with the help of reconstructed real light sources

Creating the DLPV V^Δ

- Render an RSM R^μ of the reconstructed scene
- Render an RSM R^ρ of the reconstructed scene with the additional object O included
- Indirect injection
  - Inject VPLs generated from R^ρ into a volumetric texture V^Δ as spherical harmonic encoded coefficients → Figure 3a
  - Set current blending mode to subtraction
  - Inject VPLs generated from R^μ into V^Δ → Figure 3b
  - Propagate the existing delta in the volume → Figure 3c
- Set current blending mode to addition
- Direct injection
  - Inject direct light at voxels generated from R^ρ into V^Δ with flux and direction of a real light source
  - Set current blending mode to subtraction
  - Inject direct light at voxels generated from R^μ into V^Δ

Rendering

- Compose final image → Figure 1b, 4
  - Create mask to differentiate real and virtual geometries
  - Simulate indirect bounces from real geometry on O with additional LPV V^ρ or other method (e.g., Precomputed Radiance Transfer with SH encoded fisheye image)
  - Query the DLPV V^Δ for real reconstructed geometry, multiply it with the reconstructed surface albedo and superimpose onto the background image



(a) (b) (c)

Figure 3: Creating a DLPV from two RSMs: (a) a volume V^μ after VPLs and direct light is injected (b) and similarly V^ρ. (c) The propagated difference between both yields the DLPV. The residue indirect light from the synthetic object remains as positive contribution, while the shadowed space behind it now has negative values.

RESULTS

Observations

- Diffuse indirect bounces are stable and almost free of flickering
- Rendering for the entire pipeline adds approximately twice the injection cost of a regular LPV for direct and indirect light
- Shadows, due to the low resolution of the DLPV, are heavily aliased. Increasing the volume size adds prohibitively large evaluation costs.
- Thin geometry can be covered by an entire voxel, leading to light-/shadow-bleeding. Shadow-bleeding artifacts are more severe due to subtraction of energy from wrong surfaces (e.g., in Figure 1b a thin purple line along the green bag).
- On average the DLPV creation consumes 6ms each frame

Conclusion

Delta Radiance Fields can be used to relight real scenes in mixed reality applications with superimposition. The first implementation using DLPVs provides efficient and temporally coherent indirect bounces, but suffers from aliasing and light-/shadow-bleeding due to the coarse volume. I would like to combat these problems with higher resolution volumes and introduce indirect specular bounces using *Voxel Cone Tracing* [1].

REFERENCES

1. Crassin, C., Neyret, F., Sainz, M., Green, S., and Eisemann, E. 2011. ***Interactive indirect illumination using voxel cone tracing***. Computer Graphics Forum.
2. Kaplanyan, A., and Dachsbacher, C. 2010. ***Cascaded light propagation volumes for real-time indirect illumination***. In Proc. of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games, ACM, I3D '10.
3. Knecht, M., Traxler, C., Mattausch, O., Purgathofer, W., and Wimmer, M. 2010. ***Differential Instant Radiosity for Mixed Reality***. In 2010 IEEE International Symposium on Mixed and Augmented Reality (ISMAR).
4. Lensing, P., and Broll, W. 2012. ***Instant indirect illumination for dynamic mixed reality scenes***. In 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR).

Figure 4: Sample images created with DLPVs featuring indirect bounces from real to virtual and vice versa, as well as shadows.