Towards Using Realistic Ray Tracing in Augmented Reality Applications with Natural Lighting

Fabian Scheer[1] Oliver Abert[1] Stefan Müller[1]

[1]University of Koblenz-Landau

Abstract: Virtual objects in augmented reality applications often appear flat and thus violate the immersion. To further extend the range of illumination effects and to improve the visual quality of the renderings we combine augmented reality, ray tracing and image based lighting techniques. We explore the resulting possibilities and problems which occur in such a scenario. Beyond that we present a method to augment video images by shadows casted from virtual onto real objects.

Keywords: Augmented Reality, Ray Tracing, High Dynamic Range, Image Based Lighting, Simulating shadows on real objects

1 Introduction

Augmented reality techniques have become interesting for diverse fields of applications, for example in learning and training as well as for touristic or entertainment purposes. However, usually the computer generated content is generated by simple rasterization algorithms. On the one hand, this enables a high frame rate, which is important for the presentation. On the other hand, a good deal of realism is sacrificed, since the appearance of virtual objects is often synthetic, thus destroying any immersive impression a user could have. Ray tracing based approaches would allow a wide selection of different advanced lighting techniques, which are not feasible with rasterization. In the context of this paper we are discussing the illumination of the virtual model in detail. The intention is, that any virtual object will be illuminated by real world lighting conditions and thus appears indistinguishable from real world objects. Yet we know that this also employs tight constraints on the geometry itself, we are not focussing on this topic yet. With the recent advent of realtime ray tracing it becomes possible to combine AR and ray tracing based illumination while still ensuring an interactive frame rate.

2 Previous Work

A very good discussion of common problems for augmented reality applications can be found in [Fou95]. One of the most well known ray tracing systems, OpenRT, was extended in 2003 by Andreas Pomi [PMWS03] to support real world lighting. They focused on video textures, hence they were able to illuminate full ray traced scenes by real world video, i.e a room lit by a TV screen showing real footage. Additionally, they also simulated an HDR camera

capturing the upper hemisphere with a fish-eye lens and using the result as an environment map for ray tracing. They achieved a high frame rate of up to 19 frames per second by using a cluster of machines. However, with the employed restrictions they are not able to cast shadows on real world geometry apart of an assumed plane acting as the floor. Since a cluster of computers is required, an outdoor use is precluded. Recently, Havran et al. [HMSHPS05] presented an approach similar to ours. They sampled the hemisphere, which was captured by a HDR camera to extract a number of light sources. Using a simple multi-pass rasterization approach they were able to synthesize good result efficiently. However their approach is naturally limited to the capabilities of graphics hardware, i.e. no real shadows nor reflections/refractions.

3 Modeling the Scene Geometry

In order to enhance the realism of augmented reality environments, it is necessary to take the real lighting into account. However, in most cases only a very simple Phong based local illumination model is used, so that virtual objects appear quite unrealistic. Therefore a major challenge is to correctly simulate the lighting for virtual objects. For this purpose we imply that the geometry of the real scene is given. We distinguish between a global model (section 3.1), which contains information about the real geometry of the complete environment and a local model (section 3.2), which contains only the geometry that is close to a certain point of interest. In both cases the real objects' geometry is only used for the illumination computation as well as for occlusion queries. However, this geometry is not rendered directly.

3.1 Global Model

A global model is capable to provide information about all objects in a scene like the complete geometry, light sources or complex BRDF surface properties. Beyond that, information about the HDR lighting situation at certain points of interest in the scene could be captured by Light Probes or fish-eye cameras. Additionally, the restriction of distant illumination can be overcome by intersecting rays with the model. On the one hand, the benefit of such a model is a more accurate and complete illumination simulation for virtual objects. On the other hand, the frame rate will be reduced due to the elevated computational demand of the ray tracing process.

3.2 Local Model

In contrast to the former, a local model contains only the geometry and HDR lighting information, that is closely around a certain point of interest, i.e. in proximity of the virtual object. The incoming light at that point is captured by an omnidirectional HDR image as explained in section 4. This enables rendering virtual objects in this local context with respect to the real lighting condition. Additionally, the ray tracing process benefits from

a strongly simplified scene description. Thanks to the omnidirectional capturing, reflections, refractions and complex BRDF surface properties can be computed for virtual objects assuming a distant illumination.

4 Dynamic Omnidirectional HDR Image Capturing

To capture the complete illumination of a scene we employ high dynamic range omnidirectional capturing techniques. In our work we used a HDR video camera (HDRC IMS Chips) to overcome the restrictions of static illumination. The HDR camera captures the logarithm of incoming radiance at 12 bit precision. For the linearization and photometric calibration of these values, a MacBeth color checker was captured at different exposures. A mapping function was derived by linear optimization, which maps the real radiance values to the captured radiance values, yielding linear radiance values with float precision.
For capturing we used both a fish eye lens and a light probe, capturing 180° and approximately 360° respectively of the environment. For some effects like reflections, refractions or complex BRDFs the upper hemisphere, captured by the fish eye, is not sufficient, since a ray could possibly shoot in an arbitrary direction. However, using a light probe for capturing the whole environment introduces other drawbacks, such as the necessity of calibration, which basically prevents dynamic capturing, i.e. moving the light probe around during runtime.

5 Combining Ray Tracing and Augmented Reality

The correct registration of the model with the video image requires a stable pose estimation algorithm, ideally without the occurrence of jitter effects allowing an accurate and immersive computation of the virtual illumination. To combine augmented reality and ray tracing, the information of the transformation matrix M, given by the pose estimation algorithm, has to be integrated into the ray tracer. Usually, pose estimation algorithms deliver a transformation matrix, which describes the orientation of a marker or object in the camera coordinate system. Translating all of the geometry into the camera or a global coordinate system is too expensive. A natural approach is to invert M, effectively describing the camera in the coordinate system of the tracked marker.

5.1 Image Based Lighting

Nowadays many approaches exist for rendering considering natural lighting known as Image Based Lighting (IBL - see [RWPD06] for more details). The various methods are mainly categorized in techniques of Precomputed Radiance Transfer (PRT) and sampling-based approaches. Working with video environment maps (VEM) let us define some requirements the IBL techniques have to fulfill:

- Interactive or realtime computation
- Temporal coherence of the illumination between video frames to avoid flickering effects
- Efficient approximation of the illumination environment
- Efficient adaptation to changing illumination situations

PRT techniques convert an environment map (EM) into its frequency-space representation by using spherical harmonics and lead to a fast illumination requiring only a dot product for each vertex. Omitting the high frequencies in the illumination computation the resulting shadows will become blurry. Sampling-based approaches achieve a better quality of the shadows but generate too many samples in order to use them directly as input light sources for a ray tracer. To take advantage of both approaches we implemented the PRT technique of [RH01] and the Hierarchical Importance Sampling of [ODJ04] in our augmented reality system "*RaytracAR*". We employed the inverse spherical harmonic transform to generate an output irradiance environment map for texture mapping of a diffuse virtual object. To

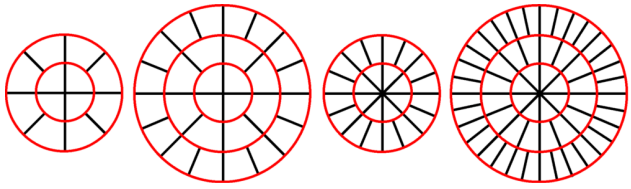

Figure 1: Adjustable initial division into regions

compute a realistic shadow we reduced the number of samples of the hierarchical sampling algorithm by a hemispherical partitioning algorithm according to the importance of the samples and used them as input light sources for the ray tracer. To adapt to restrictions of a given hardware or a desired visual quality of the user, the number of output light sources of the partitioning is adjustable. First, we divide the EM into regions, whereas the resolution is adjustable too, as shown in figure 1. Then we assign the samples to their appropriate regions. To reduce the samples we set one sample per region as a representative light source. To avoid an underestimation of the importance of bright areas and an overestimation of very dark areas, we introduce a metric based on the mean luminance of the samples in the whole EM (MLEM) and the mean luminance of the samples in a considered region (MLR). If the MLR is less than the MLEM, no light source for this region is set. If it is greater the region is further subdivided depending on the deviation of the MLR to the MLEM. The following pseudo-code demonstrates the algorithm. To control the number of output light sources (LS) the parameter $maxLS$ defines the maximum number of regions that can be created by the further subdivision. LS is acquired by an own metric and denotes the even number of regions to create.

```
if(MLR > MLEM)
    lumiRange = maxLumiEM - MLEL;
    deviation = MLR - MLEM;
    lumiFactor = lumiRange/maxLS;
    for(int k=1; k<maxLS; k++)
       if(deviation>k*lumiFactor)LS = pow(2,k);
```

The metric `LS = pow(2,k)` strongly increases the number of output light sources according to the importance of this area in the EM, but although a more moderate metric like `LS = 2 * k` delivers good results. The position and color of the output light source of a region is adopted from the brightest sample in this region to ensure a correct shadow. The mixture of a fixed division into regions and an importance based refinement allows an efficient adaptation to new lighting conditions in the VEM. The adaptation to new lighting conditions is kept locally in the regions avoiding an impact on the illumination information in the other regions and thus keeping the temporal coherence between the video frames. The irradiance environment mapping requires information of the incoming light of arbitrary directions. Therefore the capturing of a Light Probe is preferable, whereas for the sampling algorithm the usage of a the fish eye lens is sufficient.

5.2 Augmented Shadows on real Objects

A major challenge is the correct simulation of shadows. Here, a ray tracing based approach has great advantages over rasterization techniques, since shadows can be computed quite easily and straightforward, though more costly. We flag virtual objects that are visible as well as real objects that are invisible. Shadows which are cast by real objects (given in the scene model) on virtual ones are generated automatically by the ray tracing process. The other way around requires a bit more effort. To augment the video image with realistic shadows casted by virtual objects onto real objects, a scale factor *shadow* for each RGB channel has to be computed in order to darken shadowed pixel values accordingly. Once a hit point of a view ray with an invisible real object is found, the set of all light sources of the upper hemisphere has to be determined. The color values of these light sources are summed up in $UnShadowColor$ also for each channel. In a following step, shadow rays are traced to every light source of this set to determine if a virtual object casts a shadow. During this process the color values of the blocked light sources are summed up in $ShadowColor$. The ratio $ShadowColor/UnShadowColor$ determines how much light is blocked by virtual objects for a given hit point, so we can compute the scale factor *shadow* for every RGB channel by $shadow = 1 - ShadowColor/UnShadowColor$. A channel wise multiplication of the video image pixels with the scale factors augments the video image with realistic shadows. Considering that a pixel value in the video image corresponds to the result of the rendering equation, augmenting the video image with shadows expands the rendering equation by a scaling of the outgoing radiance, displayed in a pixel of the video image, with the factor *shadow*.

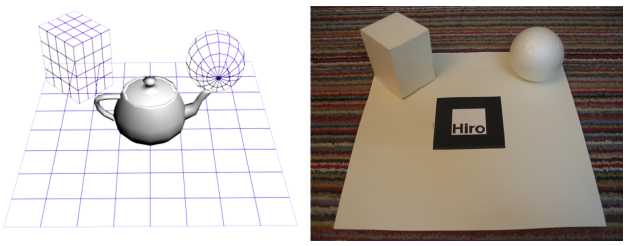

Figure 2: Local scene model and real test scene

5.3 Test Scenario

For the tests we used a local scene model because of its independence of a global scene context and its flexibility in usage. The static test scene consisted of 814 triangles organized in a bounding volume hierarchy of 379 axis aligned bounding boxes. The distance of the camera to the marker was 500 mm. Figure 2 shows the real test scene (right) and the local scene model including the virtual teapot (left). Although the underlying ray tracing system "*Dream*" [Gei05] is capable of multithreading we tested the application on a single processor AMD Mobile Athlon 3000+ system with 1 GB memory. The environment map had a resolution of 300×300 pixels. To track the marker in the test scene we used the optical tracking system ARToolKit [KB99]. To test our system with changing lighting conditions, four light sources in the real scene were turned on successively. For the illumination simulation we tested the ray tracing with the reduced number of samples as input light sources based on the fish eye EM. We further tested the combination of texture mapping the teapot with the irradiance environment map and generating the shadows by the reduced number of samples based on the Light Probe EM.

6 Results

The partitioning algorithm was set to different parameters yielding a varying number of light sources. Taking the reduced number of samples as input light sources for the ray tracer we obtained the following frame rates:

Resolution	Without Shadow	With Shadow	Light Sources
320×240	0.5 sek / 1.9 fps	1.2 sek / 0.8 fps	59
640×480	1.3 sek / 0.7 fps	4 sek / 0.2 fps	59
320×240	0.6 sek / 1.7 fps	3.7 sek / 0.2 fps	86
640×480	1.5 sek / 0.6 fps	14 sek / 0.06 fps	86
320×240	0.7 sek / 1.6 fps	6.2 sek / 0.15 fps	122
640×480	2.2 sek / 0.46 fps	24 sek / 0.04 fps	122

For the combination of irradiance texture mapping and the generation of shadows using the reduced number of samples (figure 3) we parameterized the partitioning algorithm more conservative to generate only 76 light sources and received the following results:

Resolution	Without Shadow	With Shadow	Light Sources
320×240	0.6 sek / 1.6 fps	1.3 sek / 0.7 fps	76
640×480	0.7 sek / 1.3 fps	4.6 sek / 0.2 fps	76

The frame rates mainly depend on the number of light sources that are used as an input for the ray tracer, however it depends strongly on the distance of the camera to the marker when the local scene model is used. Reducing the distance to the virtual object increases its screen filling percentage. Therefore more pixels need to be shaded leading to an increased computational effort for the ray tracing process. To overcome this limitation a global scene model should be used. Figure 3 shows resulting renderings. Note that on the left we achieved a highly realistic shadow by adjusting the parameters that even areas of medium importance are sampled by the partitioning using 178 LS ($maxLS$ was set to 64). On the right the illumination of the teapot is more realistic due to the irradiance mapping. Here, the partitioning was adjusted conservatively yielding only 76 LS ($maxLS$ was set to 1) to show the possibility to adapt to limitations of any underlying hardware by delivering although visual attractive results.

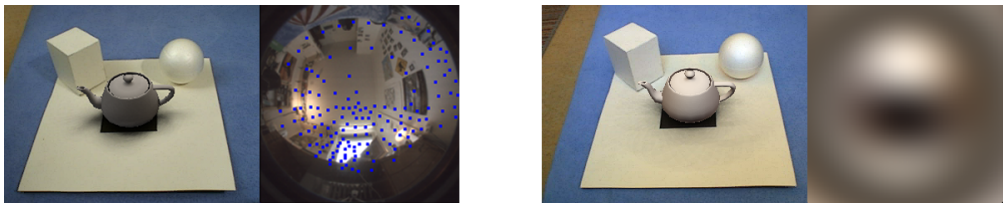

Figure 3: Rendered test scene with light source samples (left) and with combination of irradiance texture mapping and shadow generation via the reduced samples (right).

7 Conclusions and Future Work

In this paper we employed the combination of ray tracing and augmented reality. By using a model of the real scene, virtual objects can influence the appearance of the real environment displayed in the video image by casting shadows. A parameterization of the sample reduction algorithm enables us to adapt the number of input light sources for the ray tracer to a desired quality of the illumination simulation or any given hardware restrictions. Comparing the two implemented methods we achieved visually better results for the illumination of the virtual object by using the irradiance texture map. Currently the system is restricted to diffuse and mirrored surfaces, but thanks to the underlying ray tracing system it can be extended to handle specular surfaces as well. Furthermore with ray tracing even complex surface reflection properties like BRDFs or BSSRDFs could be simulated correctly, which would vastly increase the quality of the renderings. Another interesting enhancement is the augmentation of a virtual reflection in a real mirrored or transparent object which would increase the immersion of the user. The system is not optimized at the moment and we are expecting higher frame rates by optimizing the system, running the system on multi

processor systems and porting the irradiance map generation, the hierarchical sampling and sample reduction algorithm to the GPU.

References

[Fou95] Alain Fournier. Illumination Problems in Computer Augmented Reality. In *Technical Report, University of British Columbia, Department of Computer Sciene*, 1995.

[Gei05] Markus Geimer. *Interaktives Ray Tracing*. PhD thesis, University of Koblenz-Landau, August 2005.

[HMSHPS05] V. Havran, K. Myszkowski M. Smyk, G. Krawczyk, and H.-P.-Seidel. Interactive system for dynamic scene lighting using captured video environment maps. In *Eurographics Symposium on Rendering 2005*, 2005.

[KB99] Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*, 1999. San Francisco.

[ODJ04] Victor Ostromoukhov, Charles Donohue, and Pierre-Marc Jodoin. Fast hierarchical importance sampling with blue noise properties. *ACM Transactions on Graphics*, 23(3):488–495, 2004. Proc. SIGGRAPH 2004.

[PMWS03] Andreas Pomi, Gerd Marmitt, Ingo Wald, and Philipp Slusallek. Streaming Video Textures for Mixed Reality Applications in Interactive Ray Tracing Environments. In *Virtual Reality, Modelling and Visualization 2003 (VMV) , November 19-21, Munich, Germany*, 2003.

[RH01] Ravi Ramamoorthi and Pat Hanrahan. On the relationship between radiance and irradiance: determining the illumination from images of a convex lambertian object, 2001. http://graphics.stanford.edu/papers/invlamb/, Stand[27.03.2007].

[RWPD06] E. Reinhard, G. Ward, S. Pattanaik, and P. Debevec. *High Dynamic Range Imaging: Aquisition, Display and Image-Based Lighting*. Morgan Kaufmann, 2006.