Joint Virtual Reality Conference of EGVE - EuroVR (2013)
B. Mohler, B. Raffin, H. Saito, and O. Staadt (Editors)

LightSkin: Real-Time Global Illumination for Virtual and Mixed Reality

P. Lensing[1] and W. Broll[1]

[1]Ilmenau University of Technology, Germany

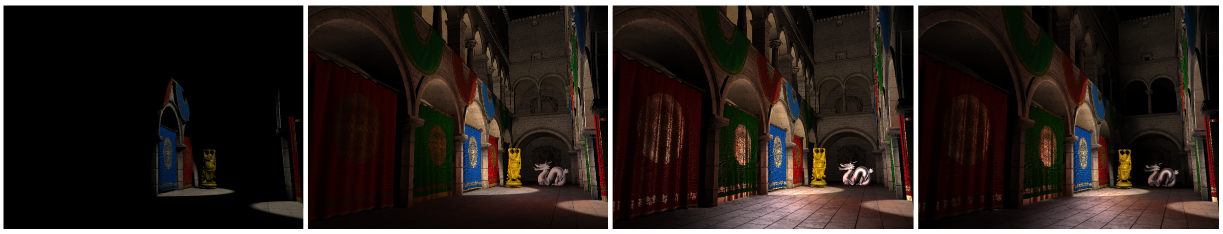

Figure 1: *Our real-time global illumination approach for diffuse and glossy reflections. From left to right: scene without indirect illumination, indirect diffuse reflections added, glossy reflections added, occlusions added.*

Abstract

Synthesizing global illumination effects is a vast field of research for both offline and real-time rendering. While the most important goals for offline rendering are realism and physical correctness, real-time rendering approaches additionally need to be sufficiently fast. In this paper we present a fast and novel global illumination approach capable to realize indirect illumination for diffuse and glossy surfaces based on thousands of virtual area lights even for dynamic scenes. To achieve real-time performance we calculate indirect light influence only on sparse scene points in model-space and interpolate the results for the entire visible scene. A novel shading technique is proposed to support high-frequency indirect lighting effects such as view-dependent glossy reflections without introducing temporal incoherence in dynamic scenes. Since our approach does not require any pre-computation it may be applied to Mixed Reality applications improving the visual integration of virtual content.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Shading

1. Introduction

Applying global illumination (GI) approaches to virtual reality (VR) applications is a difficult task. Real-time constraints and the need for high resolution images (e.g. for cave rendering) restrict the number of feasible GI approaches to a few dozen. This number further decreases if full support for dynamic models and light is required. Therefore, many VR rendering systems do not support global illumination although it might dramatically improve their rendering quality and by that immersion. Especially in the field of Mixed Reality (MR) mutual GI effects between the real environment and the virtual content advance the integration of the artificial content. However, applying GI to MR also introduces further problems due to the incomplete representation of the real scene.

In this paper we introduce a novel shading approach for two-bounce illumination supporting diffuse and glossy reflections, while preserving filigree surface details and fully supporting model and light animation. This is achieved using a novel sparse shading method. One major goal of our approach is the minimization of temporal incoherence for dynamic scenes. In particular, animations must not introduce undesirable flickering upon the change of indirect lighting conditions. Since our approach runs entirely on the GPU, making efficient usage of the rasterization pipeline and texture caching, we are able to provide >60fps for complex scenes on state-of-the-art graphics hardware (even for high resolutions) without precomputation.

Furthermore we demonstrate how our approach can be used in a MR application to apply indirect light gathered from real surfaces to artificial objects. Since our approach

© The Eurographics Association 2013.

DOI: 10.2312/EGVE.JVRC13.017-024

delivered by EUROGRAPHICS DIGITAL LIBRARY www.eg.org diglib.eg.org

does not need any pre-computations we are able to capture the geometry of the real scene on-the-fly using a RGB-D sensor (Kinect) and apply it instantly to the virtual model. Basically we treat each captured pixel as an indirect area light source for the virtual scene.

The paper is structured as follows: First, we provide an overview of common real-time GI (RTGI) approaches as well as approaches adapted to Mixed Reality (MR) for enhanced object integration. In section 3 we describe the basic ideas of our global illumination approach, while section 4 reveals specific implementation details. The application of our approach to a MR scenario is presented in chapter 5. Chapter 6 discusses the results and limitations in terms of quality and performance before concluding and providing a look into future work in the final section.

2. Related work

Here we focus on the most common and recent RTGI approaches. Furthermore, since our approach is also related to Mixed Reality (MR), we provide an overview of state-of-the-art GI techniques used in MR environments.

Pre-computed Radiance Transfer (PRT) as introduced by Sloan et al. [SKS02] is the foundation for several recent RTGI approaches. Here, the indirect light is represented by a few, pre-computed coefficients of low ordered spherical harmonics (SH) basis functions. Global illumination effects can be evaluated in real-time for dynamic lighting as long as the order of the basis function stays low. This implies that the light is restricted to low-frequency behavior, since high-frequency light would need SH of higher order. Another drawback of this approach is the pre-calculation of the SH coefficients. This step is time consuming and restricts the approach to static scenes. This restriction was leveled by Iwasaki et al. [IDY*07].

Cascaded Light Propagation Volumes (LPV) introduced by Kaplanyan et al. [KD10] use SH to propagate indirect light within a discrete, cascaded volume, which encompasses the entire scene to approximate indirect illumination. The indirect light is injected into the volume using Reflective Shadow Maps (RSM) [DS05] and the propagation through the volume is evaluated each frame, making expensive pre-computations unnecessary. While the light propagation considers occlusions at a coarse level, it also introduces some view dependent artifacts, due to the imprecise representation of the scene.

A very promising RTGI approach is Voxel Cone Tracing as introduced by Crassin [CNS*11]. Here, the entire scene is represented by a prefiltered sparse voxel octree for indirect illumination. The indirect light of a surface point can be obtained by tracing a bundle of cones within the octree and gathering the incoming light. However, the memory usage of the approach is pretty high, even if the voxel representation is sparse. Thus, a rather complex memory management system has to be used for large scenes. Furthermore, dynamic objects are rather expensive since they induce dynamic octree updates. Further, thin objects may cause problems. However, the visual results of the approach are considerably good.

Another class of RTGI approaches is based upon Instant Radiosity [Kel97], which allows for diffuse indirect illumination without pre-computations. The basic idea is to create a secondary virtual point light (VPL) at each point where the primary light hits a surface. The positions and intensities of the VPLs can be obtained via ray-tracing. Dachsbacher and Stamminger [DS05] proposed an efficient GPU-based approach to obtain the VPLs of the first bounce using RSMs.

Later Ritschel et al. [RGS09] introduced Imperfect Shadow Maps (ISM), which can be seen as extension to RSMs. Here each secondary VPL maintains its own shadow map approximation (ISM), which allows for considering occlusions while applying indirect light to the scene. However, since the ISM is rather coarse and the number of VPLs is rather low, we can observe temporal incoherence for complex dynamic scenes.

Applying VPLs obtained from a RSM directly to visible surfaces is an expensive task and cannot be performed in real-time without either reducing the number of receiver points or reducing the number of VPLs. Nichols et al. [NW09] and Dachsbacher et al. [DS06] proposed to reduce the receiver points in screen space, while we [LB13] recently proposed to reduce the points in model space. Since our new approach uses some main ideas of our previous work [LB13], we will discuss it in depth in the next section. If the number of receiver points cannot be reduced, reducing the number of VPLs may be possible. Therefore, Prutkin et al. [PKD12] and Nichols et al. [NSW09] propose VPL clustering schemes.

In the field of Mixed and Augmented Reality Knecht et al. [KTM*10] introduced Differential Instant Radiosity to simulate mutual diffuse light transport between virtual and real objects exploiting ISMs. The approach is capable to run in real-time but the real scene has to be re-constructed manually in advance. Furthermore, the approach produces some flickering artifacts for dynamic scenes based on the relatively low number of ISMs that can be rendered simultaneously. Recently, Knecht et al. [KTW*13] extended their approach towards supporting reflective and refractive surfaces by combining different independent rendering methods.

In our previous work [LB12] we proposed a method for dynamic real scenes, applying a RGB-D sensor to obtain the geometry data of the real scene. We used VPLs at very high counts to simulate mutual diffuse reflections between real and virtual objects. The VPL influence was spatially restricted, thus glossy reflections were not considered.

3. Our Global Illumination Approach

In general an ideal GI approach would provide a solution to Kajiya's rendering equation:

$$L(x,\omega_r) = L_e(x,\omega_r) + \int_{\Omega} f_r(x,\omega_r,\omega_i) L_i(x,\omega_i)\, cos\vartheta_i d\omega_i \qquad (1)$$

where L is the luminance reflected from point x in direction ω_r and ϑ_i is the angle between the surface normal and the incident light vector ω_i. For calculating the reflection L it is necessary to convolute the incoming light L_i against the BRDF f_r of the surface point over the entire hemisphere (Ω) of the surface point (x). For multiple light bounces this equation is recursively applied.

For indirect illumination the incident light L_i is gathered from other surfaces, which receive direct light. Since there is no analytic solution for equation (1) we can only provide an approximation to the ideal solution. In our approach we approximate incident indirect light by a number of virtual area lights (VAL), which are located at surface points, which receive direct light. This can be easily achieved using reflective shadow maps [DS05]. Each virtual light is described by its position l, surface normal n, luminous flux ϕ and its area a. Thus the reflected indirect Luminance L is approximated by a sum:

$$L(x,\omega_r) = \sum_{j=1}^{m} f_r(x,\omega_r,\omega_i) L_i(x,l_j,\phi_j,n_j,a_j) cos\vartheta_i \qquad (2)$$

Solving this equation for each surface point x is still too expensive, since the number of VALs has to be very high to achieve pleasing results. Especially for dynamic scenes m has to be >1000 to provide good temporal coherence. Even with deferred shading, where the number of receiver points is limited to the number of pixels in the G-Buffer, the calculation would be too expensive to be calculated in real-time.

Therefore several interpolation schemes exist which are able to work with a reduced number of receiver points. Approaches like irradiance caching or multi-resolution gathering [NSW09] evaluate the indirect light only at a sparse subset of receiver points, which are distributed in screen-space. Since these approaches interpolate the indirect luminance L directly, they are restricted to low-frequency diffuse light behavior. Furthermore they produce view-dependent artifacts and reduce filigree surface details.

Recently, we [LB13] introduced another sparse shading/interpolation algorithm, which can be used to calculate indirect diffuse light, without introducing view-dependent incoherence while retaining complex surface details. This is achieved in two steps: first, instead of distributing the sparse receiver points in screen space they are distributed in model space. Thus, the distribution does not change when the camera moves. Second, instead of calculating the luminance for each receiver point directly only three intermediate vectors are calculated. They build up an averaged VPL based on all indirect virtual lights influencing the receiver point. For dense shading, this averaged VPL is applied to each corresponding receiver point stored as pixel in the deferred shading buffers.

Our new approach is based on our previous work [LB13], but we propose a more sophisticated, extended shading model, improving the temporal coherence, while minimizing false interpolation results and supporting glossy reflections (without increasing the computational costs). Further, we add an approximation for occlusions.

3.1 Our Indirect Shading Approach

Several indirect illumination approaches interpret an RSM texel as a virtual point light that emits light towards one hemisphere. On the one hand this has the advantage that the light calculation is fast and easy to evaluate, on the other hand is this approximation physically incorrect and introduces artifacts. Therefore Prutkin et al. [PKD12] (among others) proposed for their RSM light clustering algorithm to use virtual area lights instead of point lights.

While applying arbitrary area lights to surface points would result in complex form factors, simple homogeneous disc area lights can be applied without appreciable performance penalty. Thus we use disc shaped area lights for indirect illumination. The form factor between a disk surface and an infinitesimal surface element can be calculated using the following equation:

$$F_{disk} = \frac{r^2 cos\vartheta_j}{r^2 + \|x-l\|^2} \cos\vartheta_i = w_j \cos\vartheta_i \qquad (3)$$

where r denotes the radius of the disc, and ϑ_j the angle between the disc normal and the vector from the disc center l to the surface point x. Thus, the diffuse reflection from point x based on a disc shaped area light may be described by

$$L_{d,j} = f_{r,d} \phi_j w_j \cos\vartheta_i \qquad (4)$$

While (according to the Phong reflection model) glossy reflections may be approximated by:

$$L_{s,j} = f_{r,s}\, c_k \phi_j w_j \cos^{k}\gamma \qquad (5)$$

k denotes the spec ular exponent, and γ is the angle between the ideal reflection vector and the observer direction (ω_r). c_k acts as factor to approximate energy

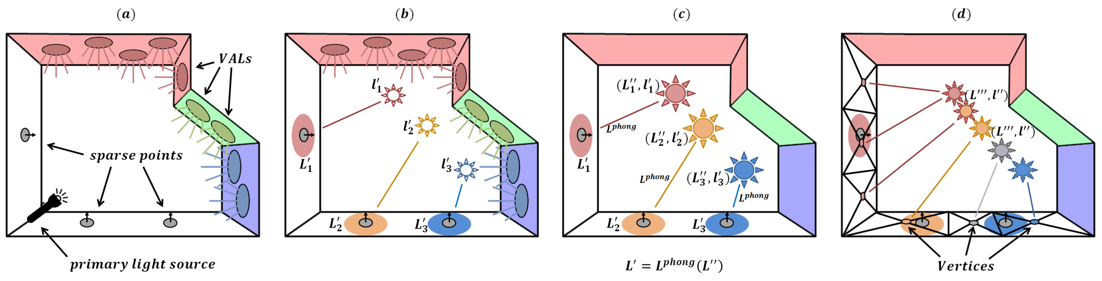

Figure 2: *(a) The VALs are distributed using RSM; (b) each VAL is applied to each sparse scene point resulting in an averaged VPL position (l') and the luminance at the sparse point (L'); (c) the normalization step, the luminance of the averaged VPLs is adjusted that $L' = L^{phong}(L'')$; (d) for each vertex a new averaged VPL (L''', l'') is built, which is an interpolation of the VPLs from the sparse points.*

conservation, where $c_k = c(k+1)$ while c is a user defined parameter.

Our entire indirect illumination approximation may now be described as (n is the number of indirect area lights):

$$L = \sum_{j=1}^{n} (L_{d,j} + L_{s,j}) \tag{6}$$

3.2 Interpolation

Applying equation 6 to each sparse scene point would provide the indirect luminance L. As already mentioned in section 3, a simple in terpolation of the indirect luminance would destroy high-frequency surface details and specular highlights. Therefore, we propose another interpolation scheme (see also [LB13]). It is performed in four steps (See also figure 2):

1. For each sparse scene point x we average two intermediate VPLs, based upon all indirect VALs and the properties of x (position, normal and specular exponent).
2. We renormalize the intensities of the VPLs so that their application to the sparse scene point x will result in the indirect luminance L (eq. 6).
3. We weight the averaged, renormalized VPLs to each vertex v, based upon similarities of the corresponding sparse point x (position, normal).
4. We apply the interpolated VPLs on a per pixel basis to the entire visible scene.

It follows a more detailed description of the different processing steps:

Step 1: For each sparse scene point x we maintain two virtual point lights: One VPL for specular and one for diffuse indirect light. These point lights are weighted with respect to the sparse receiver points as follow:

$$\begin{aligned} L'_d &= \sum_{j=1}^{n} (\phi_j w_j cos\vartheta_i)\,, \quad l'_d = \frac{1}{W_d} \sum_{j=1}^{n} l_j (\varepsilon_l + w_j cos\vartheta_i) \\ L'_s &= \sum_{j=1}^{n} (\phi_j c_k w_j \cos^k \gamma)\,, l'_s = \frac{1}{W_s} \sum_{j=1}^{n} l_j (\, \varepsilon_l + w_j \cos^k \gamma) \end{aligned} \tag{7}$$

$$W_d = \sum_{j=1}^{n} (\varepsilon_l + w_j cos\vartheta_i), \;\; W_s = \sum_{j=1}^{n} (\, \varepsilon_l + w_j \cos^k \gamma)\,.$$

(L'_d, l'_d) builds up the VPL for diffuse, while (L'_s, l'_s) builds up a VPL for glossy reflections (L denotes the luminance, while l denotes the light position). ε_l is a very small constant value that prevents l'_d and l'_s from being undefined (in the coordinate origin). This may happen if the receiver point does not receive any indirect light. We will see that this can cause temporal incoherence during the averaging on per vertex basis (step 3).

Step 2: If we consider that $f_{r,d}$ and $f_{r,s}$ are constant within each surface point and independent to the incident light direction the following equation is valid (according to equations 4, 5, 6 & 7):

$$L = \sum_{j=1}^{n} (L_{d,j} + L_{s,j}) = f_{r,d} L'_d + f_{r,s} L'_s \tag{8}$$

Thus, to obtain the correct luminance L while applying our averaged VPLs (L'_d, l'_d) and (L'_s, l'_s) to the sparse surface points, we just have to renormalize L'_d and L'_s so that their application to x will result in L_d and L_s (see figure 2). Please note, that this renormalization is only performed once for each sparse shading point x per frame, thus it is pretty inexpensive with respect to the overall computation time.

The application of the averaged VPLs to the surface points can be performed by each suitable local shading model. We propose to use a Phong-based shading model:

$$L^{phong}(L'_d, L'_s) = f_{r,d} L'_d \frac{cos\vartheta}{\|x - l'_d\|^2} + f_{r,s} L'_s \frac{\cos^k \gamma}{\|x - l'_s\|^2} \tag{10}$$

Thus the renormalized L'_d and L'_s are:

$$L''_d = L'_d \frac{\|x - l'_d\|^2}{cos\vartheta}, \qquad L''_s = L'_s \frac{\|x - l'_s\|^2}{\cos^k \gamma} \tag{11}$$

For each sparse surface point x the equation $L = L^{phong}(L''_d, L''_s)$ is now valid and each point x carries now our averaged, renormalized VPLs (L''_d, l'_d) and (L''_s, l'_s).

Step 3: This step is mostly similar to [LB13]. Based upon the VPLs of the sparse receiver points we calculate two new VPLs for each vertex. This VPLs are weighted averages. Each vertex stores references to their eight nearest sparse surface points x (calculated in advance). Using eight surface points has shown to provide a good compromise between speed and quality. According to the similarity of the sparse points (position and normal) each corresponding VPL (L''_d, l'_d) and (L''_s, l'_s) is weighted to the vertex v. We propose the following weighting, which has proven to provide good results for most distributions:

$$w_{v,j} = (1 - \frac{d_j}{d_{max}})(n_v \cdot n_{x,j}) \tag{12}$$

where d_j is the distance between the vertex v and the sparse surface point x_j. d_{max} is $\max(d_j)$ for $j = 0, \dots, 7$. n_v and $n_{x,j}$ are the normals from the vertex and the sparse surface point x. The dot-product is clamped between [0,1]. Each vertex VPL can now be described as:

$$L'''_d = \frac{1}{W_v}\sum_{j=0}^{7} w_{v,j} L''_{d,j}, \qquad l''_d = \frac{1}{W_v}\sum_{j=0}^{7} w_{v,j} l'_{d,j},$$
$$L'''_s = \frac{1}{W_v}\sum_{j=0}^{7} w_{v,j} L''_{s,j}, \qquad l''_s = \frac{1}{W_v}\sum_{j=0}^{7} w_{v,j} l'_{s,j} \tag{13}$$

with $W_v = \sum_{j=0}^{7} w_{v,j}$. ε_l from equation 7 assures that l''_d and l''_s do not evalutate wrong VPL positions, if the accumulation of w_j is zero.

Step 4: In the last processing step we evaluate for each visible pixel eq. 11 $L^{phong}(L'''_d, L'''_s)$ with the averaged positions l''_d and l''_s.

3.3 Occlusion

Indirect light occlusion is crucial to provide soft-shadowing, but unfortunately, calculating exact occlusions in real-time is not yet possible. Thus, we propose an occlusion technique, which benefits from several approximations to provide perceptually convincing soft-shadows without introducing too much calculation overhead.

Our occlusion approach works well for glossy and diffuse reflections, but for a better understanding we focus in the following description on diffuse reflections. The approach is performed in two steps:

First, we calculate for each sparse scene point the standard derivation σ of the averaged light position l'_d for each dimension (x, y, z) leading to an axis aligned bounding box centered around l'_d. The dimensions of the bounding box indicate whether the indirect light source is either spatially compact or wide spreaded in the scene. Based upon this box we assume a sphere shaped area light S_o with radius=max(box.width, box.height, box.depth). This sphere is exclusively used for occlusion calculation.

Second, for each possible occluder we determine how much it occludes S_o as seen from each sparse scene point. Therefore it is necessary to calculate the overlapping area of the occluder and S_o. This can be performed by double-projecting both: first we project them on the unit hemisphere and after this we orthogonally project them on a unit circle (see figure 3). Unfortunately this double projection may lead to complex planar shapes, which cannot be easily evaluated. Thus, we restrict our occluders to simple shapes. For our implementation we use disk-shaped occluders. This allows us to re-use our sparse receiver points as occluder disks. The orientation and position of the disk is already known and the radius is set to the maximum distance between the sparse point and a model vertex within the Voronoi region of the sparse point.

Determining the exact overlapping area of a projected disk and sphere is still a too time consuming. Thus, we propose using the following approximation: based on the Nusselts analogon we know that F_{disk} from eq 3 multiplied by π is exactly the area of a disk projected on the unit circle and if we consider $cos\vartheta_j = 1$ we obtain the projected area of a sphere. Thus, we know the exact areas for both projections, but we do not know their positions and shapes. The positions can be approximated by simply projecting the center of the disk and the sphere on the unit circle:

$$c_{proj} = (\frac{c - x}{\|c - x\|} \cdot t, \frac{c - x}{\|c - x\|} \cdot b) \tag{14}$$

Here c denotes the center position and x the position of the sparse scene point. t is the tangent perpendicular to the normal n_x of the sparse scene point and b is the binormal perperpendicular to n_x and t.

Since the shape of the projection remains unknown, we assume that it can be approximated by a quad with a side length of $\sqrt{\pi F_{disk}}$. Thus, one has to determine the overlapping area of two quads, which can be calculated pretty fast in a shader program (see figure 3).

The amount of occlusion o can now be calculated by dividing the overlapping area by the projected area of S_o. It

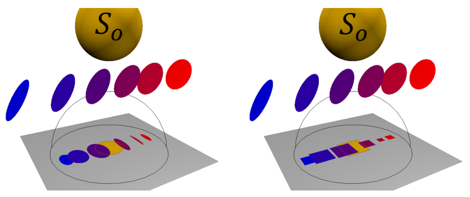

Figure 3: *Left: the correct double projection of S_o and the occluder disks; right: our approximation using quads*

is directly applied to the luminance of the sparse point:

$L'_d = L'_d(1 - o)$.

Please note, that occluders behind S_o (seen from the sparse point), would also wrongly occlude light. T herefore it is neccessary to scale the occlusion amount by distance. We scale the amount linearly to zero if the occluder lies in S_o, and between the sparse point and the center of S_o.

4. Implementation

Our overall implementation is similar to [LB13], but since our indirect illumination approach maintains two VPLs for each sparse scene point – one for the diffuse and one for the specular channel – our indirect lighting is in general more expensive. Therefore we provide a more efficient implementation for applying the indirect light.

4.1 Creating Virtual Area Lights from RSM

Our RSM implementation uses two 4xfloat textures to represent the VALs. The position and normal is stored in 3xfloat each, while the luminous flux is compressed to one float value as proposed in [DS06]. The remaining float channel stores the area of the light. We assume for our area calculation that each VAL points towards the primary light source, which is in generally not true, but it simplifies the calculation. While this approximation is not very exact it solves one important problem for our approach: The averaged VPL positions l'_d and l'_s are now decoupled from the VAL distribution density.

4.2 Applying Virtual Area Lights

In [LB13] we propose to use a splatting algorithm to apply lights obtained from the RSM to the sparse scene points. Using this technique makes it necessary to perform four texture lookups for each indirect light: sparse point position, sparse point normal, VAL position, VAL normal & flux. Furthermore, four four-channel targets are written (L'_d, l'_d, L'_s, l'_s) and the z-test is performed for each VAL (the z-test is used to mask unnecessary sparse scene points).

Knowing that memory access is pretty expensive even for modern graphic cards we propose to use a gathering approach which is tweaked for the graphic card texture cache size. Our shader implementation looks like follows:

Listing 1

```
$initialize\ targets\ L'_d, L'_s, l'_d\ and\ l'_s\ to\ zero$
$fetch\ sparse\ point\ position\ and\ spec.exp.\ (4xfloat)$
$fetch\ sparse\ point\ normal\ (4xfloat)$
$for(j = 0;\ j < K_{cache};\ j++)$
  $fetch\ VAL\ position\ \&\ area\ (4xfloat)$
  $fetch\ VAL\ normal\ \&\ flux\ (4xfloat)$
  $calculate\ w_j$
  $w_{d,j} = \varepsilon_l + w_j cos\vartheta_i$
  $l'_d.xyz += lw_{d,j};\ \ l'_d.w += w_{d,j}$
  $w_{s,j} = \varepsilon_l + w_j cos^k \gamma$
  $l'_s.xyz += lw_{s,j};\ \ l'_s.w += w_{s,j}$
  $L'_d += (calculate\ L_{d,j})$
  $L'_s += (calculate\ L_{s,j})$
$write\ targets$
```

This shader reduces the number of lookups, target writes and z-tests to a minimum and efficiently uses the texture cache of the graphic card because each lookup in the loop can be shared for each target pixel. Thus, K_{cache} is tweaked that a maximum cache hit rate is achieved (for us $K_{cache} = 128$ works best, 97% hit rate). The number of applications of this shader is reduced by $1/K_{cache}$. Please note, that this shader is additively applied to the targets. The final positions of the VPLs can be obtained by dividing l'_d.xyz by l'_d.w. This is performed in the renormalization step (see section 3.2).

Compared to our previous splatting approach in [LB13], our new approach is an order of a magnitude faster and it is still 1.35 times faster than a group shared memory optimized compute shader version.

5. Application to Mixed Reality

We use an RGB-D sensor for capturing the real scene. Each pixel stores an RGB color and depth value. Applying a standard tracking approach to obtain the camera position we are able to calculate the position, area, and normal for each sensor pixel. Since we handle each VAL as lambertian emitter we can conclude the luminous flux for each pixel based on its RGB value. Thus, we are able to approximate near field indirect illumination for the artificial content.

However, we have to consider some restrictions for our Mixed Reality approach:

1. Sparse scene points are only available for the virtual content since they are distributed in model space, thus only the virtual content can receive indirect illumination for now.
2. Using an RGB-D sensor only provides VALs facing towards the camera, thus rear-sides of real objects are not considered for indirect illumination.

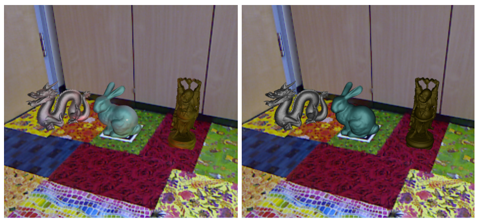
Figure 4: *AR, Left: Scene with GI; right: without.*

3. Moving the camera changes the indirect illumination if we use the most recent captured image only.
4. Bilateral filtering is necessary to reduce depth noise.

Despite these restrictions, applying indirect illumination improves the integration of the virtual content (see fig. 4).

6. Results and Discussion

In figure 5 (left column) we compared our approach to an implementation without interpolation, where each RSM VAL is applied to each visible surface point without considering occlusion. It can be seen that our approach is very close to the brute-force solution and that filigree surface details in the image are preserved. While our Approach renders the scene in figure 5 at 150 fps, the brute-force solution provides 0.5 fps only.

Figure 6 shows a comparison to ground truth data rendered with the offline renderer Cinema 4D. While the perceived differences between the images are pretty low we can observe some wrong dark areas. These are caused by our performance optimized occlusion approach, which double occludes the incoming indirect light if the occluders lie in one line towards the indirect area light. This “double occlusion” is in generally wrong since the nearest occluder should be considered only. Furthermore, our occlusion approach does not change the received color spectrum of a sparse surface point. Instead, it just damps the existing spectrum.

To illustrate the improvements of our approach to [LB13], we compare them in terms of quality (figure 5, center & right column) and performance (table 1). Some further performance metrics can be found in table 2.

Our approach achieves perceptual convincing results in real-time, but there are some restrictions requiring further consideration. One restriction is that the mesh tessellation has to be finer than the sparse scene point distribution, otherwise indirect light details will get lost. Thus, if we consider a large wall which consists only of two triangles and is populated by 64 sparse scene points our interpolation would produce non-satisfying results. Another important issue is that the spectrum of the specular reflection is depending on the sparse scene point density. This does not mean that the specular reflection is limited to a certain specular exponent, but if we want to approximate ideal reflections we have to provide additional scene points. Thus mirroring materials are inefficient, while glossy materials are well approximated even with a relatively low number of sparse scene points. The last limitation we observed concerns our occlusion approximation: As we attenuate the luminance at the sparse scene points directly, on one hand, we cannot provide sharp shadows. On the other hand, this limitation is somewhat useful for our approach, as our occluders are approximated by disks and therefore sharp shadows would reveal this approximation.

7. Conclusion and Future Work

We introduced a fast global illumination approach that produces appealing images in real-time, without restricting the scene to be static and without expensive precomputations. Furthermore, our approach consumes very little graphics card memory and thus does not need any special virtual texture treatments.

The applications of our approach are manifold and it can be used for VR and MR. Even high resolutions, which are especially important for cave rendering do not add significant processing time.

Furthermore, our approach is independent of the indirect light source distribution. Thus, it is not effected when the lights are spatially independently moved and spread, while light clustering approaches would suffer in terms of performance. This allows our approach also to be used for direct lighting. One possible application could be to calculate the net effect of ten thousands of particle lights in a scene (see our provided video).

In our future work we plan to add multiple indirect light bounces to our algorithm. We may also consider using a variation of micro-rendering for our occlusion approach to limit the “double occlusion” effect. Further, we will look into possibilities for adaptive sparse scene point distributions.

Table 1: *Comparison between [LB13] and new approach in FPS. For the new approach we measured three different setups: diffuse only; diffuse & glossy; diffuse & glossy & occlusion. SP denotes the sparse points. Used graphics card: AMD Radeon 7870; Resolution: 1920x1080 pixel.*

	[LB13]			Our New Approach		
Light count	16k	64k	512k	16k	64k	512k
200k Tri, 4k SP	71	19	4	126/89/66	70/46/38	25/16/14
200k Tri, 16k SP	22	5	1	69/45/32	24/15/12	7/4/4
730k Tri, 4k SP	41	11	3	73/51/39	40/27/23	15/9/9
730k Tri, 16k SP	19	4	1	53/36/23	22/14/12	7/4/4

Table 2: *Timings for different processing steps measured in a scene with 200k Triangles and 4k sparse points.*

Processing step	Timing
RSM generation	0.47 ms
Direct rendering / direct lighting	1.69 ms
Indirect lighting of sparse scene points	4.02 ms
Normalization	0.10 ms
Occlusion	3.98 ms
Interpolation	3.22 ms
Total	13.48 ms

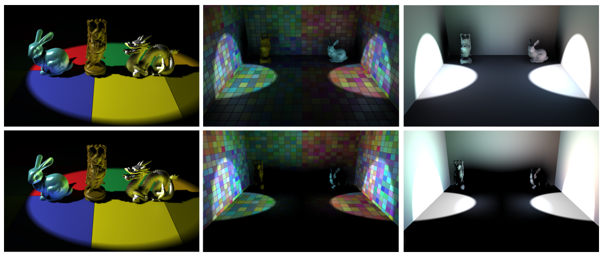

Figure 5: *Left column: Our approach (upper row) with 256 sparse surface points per model compared to brute-force indirect illumination where each VAL is applied to each visible pixel. Center & right column: our approach (upper row) compared to [LB13] (lower row). The right column shows the diffuse illumination only. Our previous interpolation method [LB13] produces errors for opposite light sources due to the normal interpolation (see the lower right image). This wrong interpolation results in dark surfaces, while our new approach nicely supports soft-shadows, diffuse and glossy reflections.*

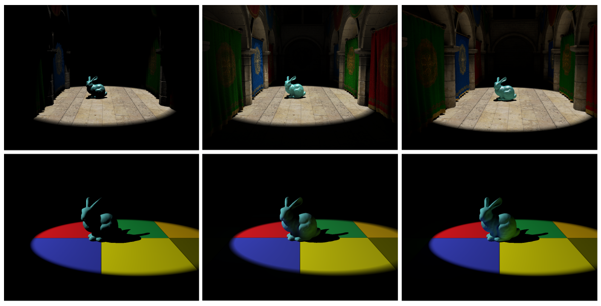

Figure 6: *Comparison between our approach and ground-truth data rendered with cinema 4d. Left column: scene without GI; center column: our approach; right column: ground-truth. In the lower row we can observe energy loss at the ears of the bunny due to our occlusion. The offline rendering took about 60 seconds, while our approach takes less than 16 ms.*

References

[CNS*11] CRASSIN, C., NEYRET, F., SAINZ, M., GREEN, S., EISEMANN, E. Interactive Indirect Illumination Using Voxel Cone Tracing, 2011, Computer Graphics Forum, Pacific Graphics, v. 30.

[DS05] DACHSBACHER, C., AND STAMMINGER, M., 2005. Reflective shadow maps. In symposium on Interactive 3D graphics and games. ACM, 203-231.

[DS06] DACHSBACHER, C., AND STAMMINGER, M., 2006. Splatting indirect illumination. In Proceedings of the 2006 symposium on Interactive 3D graphics and games (I3D '06). ACM, New York, NY, USA, 93-100.

[IDY*07] IWASAKI K., DOBASHI Y., YOSHIMOTO F., AND NISHITA T., 2007. Precomputed Radiance Transfer for Dynamic Scenes Taking into Account Light Interreflection. In Proc. of the Eurographics Symposium on Rendering, 35–44.

[KD10] KAPLANYAN, A. AND DACHSBACHER, C., 2010. Cascaded light propagation volumes for real-time indirect illumination. In Proceedings of the 2010 ACM symposium on Interactive 3D Graphics and Games. ACM, 99-107.

[Kel97] KELLER, A., 1997. Instant radiosity. In Proceedings of the 24th annual conference on Computer graphics and interactive techniques (SIGGRAPH '97). ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 49-56.

[KTM*10] KNECHT, M.; TRAXLER, C.; MATTAUSCH, O.; PURGATHOFER, W.; WIMMER, M., 2010. Differential Instant Radiosity for mixed reality, *Mixed and Augmented Reality (ISMAR),* 99-107.

[KTW*13] KNECHT, M., TRAXLER, C., WINKL-HOFER, C., WIMMER, M., 2013, Reflective and Refractive Objects for Mixed Reality, IEEE Transactions on Visualization and Computer Graphics, vol. 19, no. 4, 576-582

[LB12] LENSING, P.; BROLL, W., 2012., Instant indirect illumination for dynamic mixed reality scenes, Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on, pp.109,118, 5-8

[LB13] LENSING, P., BROLL, W., 2013, Efficient shading of indirect illumination applying reflective shadow maps. In ACM Symposium on Interactive 3D Graphics and Games '13

[NW09] NICHOLS, G., AND WYMAN, C., 2009. Multiresolution splatting for indirect illumination. In Proceedings of the 2009 ACM symposium on Interactive 3D graphics and games (I3D '09), 83-90.

[NSW09] NICHOLS, G., SHOPF, J., AND WYMAN, C., 2009, Hierarchical Image-Space Radiosity for Interactive Global Illumination, Computer Graphics Forum 28(4), 1141-1149. (June 2009)

[PKD12] PRUTKIN R., KAPLANYAN, A., AND DACHSBACHER, C., 2012, Reflective Shadow Map Clustering for Real-Time Global Illumination, Eurographics 2012 short paper track

[RGK*08] RITSCHEL, T., GROSCH, T., KIM, M. H., SEIDEL H.-P., DACHSBACHER C., AND KAUTZ, J. 2008. Imperfect shadow maps for efficient computation of indirect illumination. ACM Trans. Graph. 27, 5, Article 129 (December 2008), 8 pages.

[RGS09] RITSCHEL, T., GROSCH, T., AND SEIDEL, H.-P., 2009. Approximating dynamic global illumination in image space. Symposium on Interactive 3D graphics and games (I3D '09). ACM, 75-82.

[SNR*12] SCHERZER D., NGUYEN H., C., RITSCHEL T., SEIDEL, H-P., 2012, Pre-convolved Radiance Caching, Computer Graphics Forum (Proc. EGSR 2012), vol. 4

[SKS02] SLOAN, P.-P., KAUTZ J., AND SNYDER, J., 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. ACM Trans. Graph. 21, 3 (July 2002), 527-536.