Multiresolution Splatting for Indirect Illumination

Greg Nichols*
University of Iowa

Chris Wyman†
University of Iowa

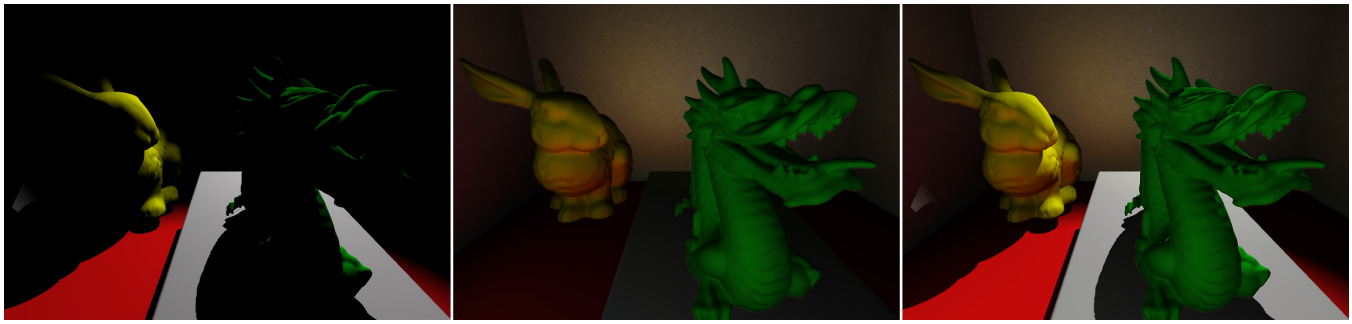

Figure 1: *Direct light only (left); indirect light generated with our method (center); the combined image (right). This scene is generated at 29 fps with fully dynamic lighting, geometry, and camera.*

Abstract

Global illumination provides a visual richness not achievable with the direct illumination models used by most interactive applications. To generate global effects, numerous approximations attempt to reduce global illumination costs to levels feasible in interactive contexts. One such approximation, reflective shadow maps, samples a shadow map to identify secondary light sources whose contributions are splatted into eye-space. This splatting introduces significant overdraw that is usually reduced by artificially shrinking each splat's radius of influence. This paper introduces a new, multiresolution approach for interactively splatting indirect illumination. Instead of reducing GPU fill rate by reducing splat size, we reduce fill rate by rendering splats into a multi-resolution buffer. This takes advantage of the low-frequency nature of diffuse and glossy indirect lighting, allowing rendering of indirect contributions at low resolution where lighting changes slowly and at high resolution near discontinuities. Because this multi-resolution rendering occurs on a per-splat basis, we can significantly reduce fill rate without arbitrarily clipping splat contributions below a given threshold—those regions simply are rendered at a coarse resolution.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture; I.3.3 [Computer Graphics]: Hardware Architecture—Graphics Processors

Keywords: global illumination, interactive rendering, hardware-assisted rendering

*e-mail: gbnichol@cs.uiowa.edu
†e-mail: cwyman@cs.uiowa.edu

1 Introduction

Indirect illumination represents light reaching a surface after previously interacting with other surfaces. While this lighting adds tremendously to visual richness and scene realism, the costs to track multi-bounce light reflections often prove prohibitive. Due to this expense, interactive applications frequently forgo complex global illumination entirely or use approximate techniques such as ambient occlusion [Zhukov et al. 1998], light maps, or precomputed radiance transport [Sloan et al. 2002]. Unfortunately these techniques impose limits of their own, often ignoring color bleeding or restricting the motion of geometry, lights, or viewer.

However, physically accurate global illumination may be unnecessary in most contexts. Tabellion and Lamorlette [2004] found that even in visually demanding applications, such as feature films, single bounce indirect illumination provides plausible lighting. Accepting this avoids tracing arbitrarily complex light paths that add little to a scene and dramatically simplifies the rendering equation.

One single bounce approach uses an augmented shadow map, called a reflective shadow map (RSM), to either gather during a final deferred render pass [Dachsbacher and Stamminger 2005] or scatter indirect illumination via splatting [Dachsbacher and Stamminger 2006]. Both techniques build on the idea of instant radiosity [Keller 1997], where pixels in the shadow map represent virtual point lights (VPLs) used as secondary light sources for computing indirect lighting. This approach effectively reformulates the rendering equation from a complex integral over surfaces to a sum over all texels in the shadow map. The key to achieving performance then lies in reducing the costs of this summation.

This paper proposes a novel multiresolution splatting technique that reduces costs for RSM-based indirect illumination. Previous techniques either gathered light from a subset of the virtual point lights or splatted light into limited regions. Our approach instead recognizes that each virtual light potentially affects the whole scene, but due to the low-frequency nature of indirect illumination many pixels receive radiance quite similar to their neighbors and can be processed as a group. This idea is similar to hierarchical radiosity approaches [Hanrahan et al. 1991], but instead works in image-space. Once per frame, the image is divided into subsplats at multiple resolutions, which are then splatted for each VPL. The results

are additively blended, and a new interpolation technique removes discretization artifacts from the final indirect illumination.

2 Previous Work

Widespread illumination research has enabled numerous techniques for interactive complex lighting. Generally, the simplest approaches add global effects by precomputing lighting, for instance using radiosity [Cohen and Wallace 1993], and baking in the results to surface textures. Clearly this precludes dynamic lighting and significant scene modifications, but quality depends on precomputation time and runtime evaluation is cheap.

Ambient occlusion [Zhukov et al. 1998] approximates indirect illumination by darkening direct lighting based upon occlusion from neighboring geometry. Although this produces only a coarse approximation, it cheaply reproduces effects such as darkened corners that arise from indirect lighting. Precomputed ambient occlusion provides a cheaper alternative to complex radiosity solutions, but maintains the assumption of static geometry. Bunnell [2005] describes an iterative ambient occlusion approximation for simple dynamic models. Other techniques [Kontkanen and Laine 2005; Malmer et al. 2007] precompute occlusion "fields" for rigid objects, allowing these objects to move while occluding nearby geometry.

Another class of techniques precomputes light transport and combines it with dynamic illumination at runtime using a simple dot product. Using a spherical harmonic basis to store the precomputed transport, Sloan et al. [2002] allow rendering of low-frequency lighting on static geometry. This works best with environmental lighting, though further work [Kristensen et al. 2005] also allows local lights. Transport fields [Zhou et al. 2005; Iwasaki et al. 2007] extend this approach to allow scenes with a few rigid, dynamic objects. Related techniques [Kautz et al. 2004; Ren et al. 2006] allow simple deformable models, but without indirect illumination.

Instant radiosity [Keller 1997] introduces the concept of point lights, which are emitted from scene illuminants using a quasi-random walk. Multiple hardware rendering passes use each VPL in turn as a point light, accumulating lighting and using shadow maps to account for indirect visibility. This technique directly demonstrates the cost of visibility queries in global illumination—each object is rasterized once for each virtual light. Laine et al. [2007] reduce visibility costs for instant radiosity by reusing shadow maps between some VPLs and Ritschel et al. [2007] precompute coherent shadow maps, allowing for faster visibility queries on rigid objects. Dachsbacher et al. [2007] and Dong et al. [2007] explore techniques that avoid explicitly computing visibility, instead relying on implicit visibility computations.

2.1 Reflective Shadow Maps

Rather than computing or approximating visibility, reflective shadow maps [Dachsbacher and Stamminger 2005] entirely ignore visibility for indirect rays, assuming that viewers will not notice incorrect visibility for secondary illumination. Reflective shadow maps build on the idea of instant radiosity, using shadow mapping hardware to generate virtual lights directly instead of using quasi-random path tracing. The map itself consists of a standard shadow map augmented by additional buffers to store surface normals, positions, and reflected flux (see Figure 2)—essentially a light-space G-buffer [Saito and Takahashi 1990].

The original method [Dachsbacher and Stamminger 2005] uses a gathering approach to sample nearby locations in the reflective shadow map for each pixel in the final image; eye-space interpolation is used to help reduce illumination artifacts. Later work reformulates this as a shooting algorithm [Dachsbacher and Stamminger 2006], splatting contributions from each VPL onto a nearby region in eye-space. This technique extends to render glossy materials and simple caustics by elongating the splat size based upon the material's BRDF, and importance sampling the shadow map allows selection of a good set of virtual lights based upon flux distribution.

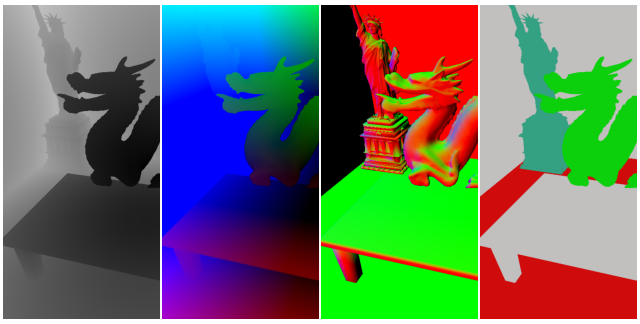

Figure 2: *Components of a reflective shadow map (left to right): a linear distance from the eye, a world space fragment position, a surface normal, and a reflected flux.*

One of the problems with splatting illumination from VPLs is excessive overdraw. In theory, each VPL can affect final illumination everywhere in the scene; using 1000 point lights requires computing contributions for 1000 splats at each eye-space pixel. Dachsbacher and Stamminger [2006] reduce overdraw by restricting splat sizes. Beyond a certain distance from each VPL, indirect contributions are ignored. This gives an ideal parameter for tuning performance, but darkens illumination significantly as splat sizes shrink.

2.2 Splatting and Multiresolution Approaches

Interactive techniques often rely on splatting, as gathering frequently proves less amenable to GPU acceleration. In the context of global illumination, Gautron et al. [2005] propose a splat-based renderer computing illumination with a radiance cache. Shanmugam and Arikan [2007] use billboards as splats to compute ambient occlusion on surfaces within the splat's influence. Sloan et al. [2007] use splats to accumulate indirect illumination from spherical proxy geometry. And caustic mapping [Shah et al. 2007] frequently uses splats to represent photon energy, varying splat size to account for divergent photons and reduce sampling noise [Wyman and Dachsbacher 2008].

Even offline illumination rendering has investigated splatting as an alternative to gathering techniques, allowing more accurate illumination on high-frequency geometry and removal of low-frequency noise [Herzog et al. 2007].

However, most interactive splat-based illumination algorithms simply assume splats must be rendered at full resolution, or clamp splats to a "reasonable" size to maintain performance. Point-based rendering [Rusinkiewicz and Levoy 2000] and volume rendering [Laur and Hanrahan 1991] use multi-resolution splatting to achieve interactive speeds, and offline rendering techniques frequently use multi-resolution and hierarchical techniques to reduce computational costs (e.g., hierarchical radiosity [Hanrahan et al. 1991]). We draw inspiration from recent caustic work [Wyman 2008] that renders illumination from splats into a multi-resolution image. This allows capturing illumination from very large splats into coarse buffers and fine illumination details in high resolution buffers while maintaining small splat sizes that dramatically reduce the overhead introduced by overdraw.

2.3 Min-Max Mipmaps

Our technique makes use of the min-max mipmap, which is similar to a subdivided quad-tree [Samet 1990]. Recent uses of min-max mipmaps include the rendering of soft shadows [Guennebaud et al. 2006], geometry intersection [Carr et al. 2006], and dynamic height field rendering [Tevs et al. 2008].

3 Algorithm

Similar to Dachsbacher and Stamminger [2006], we splat illumination from each virtual point light onto the scene. We observe that indirect lighting from a point light generally falls off quite smoothly, so splatting into a coarse buffer should suffice. Because splats are rendered in eye-space, however, depth discontinuities and sharp normal variations introduce high-frequency illumination changes ignored when splatting into a coarse buffer.

Instead of rendering a splat for each VPL into a single resolution buffer, we render these splats to a multi-resolution buffer. We start by splatting into a 16^2 buffer, allowing every VPL to affect illumination in all parts of the final rendering. In regions where this sampling appears too coarse, we subdivide the splat into *subsplats*, with some subsplats rendered in the 16^2 buffer, and some refined and output to a 32^2 or 64^2 buffer, and some refined all the way to the final output resolution.

This idea has roots in various previous techniques. It could be seen as a variant of hierarchical radiosity [Hanrahan et al. 1991], where patches are chosen based upon image-space rather than object-space constraints. Radiance caching [Gautron et al. 2005] typically focuses illumination samples near edges, and Tole et al. [Tole et al. 2002] rely on image-space criteria to better select cache samples for an interactive render. Ultimately, our algorithm simply allows splatting-based techniques to reduce fillrate costs by avoiding redundant computations, grouping them, and rendering to the coarsest buffer allowable for each group. In our algorithm each subsplat covers a single texel, though that texel might lie in a 16^2 or 64^2 buffer and thus affect hundreds of pixels in the final image.

The rest of this section describes, in greater detail, the steps of our algorithm. A quick breakdown follows:

1. Compute reflective shadow map and direct lighting,
2. Select VPLs used to splat indirect illumination,
3. Generate mipmap for use in detecting discontinuities,
4. Create and iteratively refine the list of subsplats,
5. Render subsplats to multi-resolution illumination buffer,
6. Upscale and combine buffer layers for total indirect light,
7. Add direct and indirect light for final result.

Steps 1 and 2 are described in Section 3.1, steps 3 through 5 are described in Sections 3.2 through 3.4, and steps 6 and 7 are described in Section 3.5.

3.1 Reflective Shadow Map and VPLs

We begin by generating a reflective shadow map [Dachsbacher and Stamminger 2006] by rendering from the light, storing world-space position, distance from the light, surface normal, and reflected flux for each texel (e.g., Figure 2). Next, we render from the eye, using only direct light with shadow mapping. During this step, we generate a G-buffer [Saito and Takahashi 1990] containing data needed for deferred shading (i.e., world-space position, normal, and distance from the eye).

We then sample the reflective shadow map to select points to use as VPLs for indirect illumination. We select VPLs by uniformly sampling the reflective shadow map on a regular grid, though a flux-based importance sampling or quasi-random sampling may ultimately give better quality.

3.2 Min-Max Mipmap Creation

To correctly refine subsplats, we need to identify image-space discontinuities. To do this we use a *min-max mipmap*, where each element stores maximum and minimum values rather than the average depths in a standard mipmap. To compute a min-max mipmap for depth, we start from the full-resolution linear depth buffer (computed in Section 3.1) and run the mipmap generation process. We halve the resolution at each step, computing for each output element the maximum and minimum values of the four input elements. When completed, sampling a texel within the min-max mipmap gives us the minimum and maximum depth values in that texel's image-space area, allowing efficient detection of depth discontinuities. If the difference between these values is greater than a threshold, then a depth discontinuity exists within that texel (although this method can detect spurious discontinuities in surfaces viewed at a steep angle).

Detecting normal discontinuities using a min-max mipmap is less straightforward. We generate three sets of min-max mipmaps, one for each component of the unit surface normal. If the difference between the max and min values of *any* of the normal coordinates exceeds a threshold, we consider it a normal discontinuity. In practice this method works well for detecting sharp surface features.

3.3 Processing and Refining the List of Splats

To generate the list of sub-splats, we create a single full-screen splat at the coarsest level in our *illumination buffer*, the multi-resolution image used to accumulate our indirect illumination. This splat is split into one subsplat for every texel in this coarse resolution. In our implementation, which uses a 16^2 buffer for the coarsest resolution, we therefore start with 64 subsplats.

Each subsplat can either be rendered as a point into the coarse resolution or refined into four new subsplats corresponding to the next layer in the illumination buffer. Since these subdivided subsplats each represent a pixel in a higher resolution layer, each refinement quadruples the required fillrate. Thus, the key is determining when to refine subsplats and when a coarser one suffices.

3.3.1 Refinement

In general indirect illumination changes slowly, based upon a distance-squared falloff from the light and cosine falloffs dependent on surface patch orientation. In simple scenes, relatively coarse sampling and bilinear interpolation gives plausible lighting. However, complex models create depth discontinuities along silhouettes and normal discontinuities along creases that introduce rapid changes to the indirect illumination seen by a viewer.

We detect these discontinuities by sampling the min-max mipmaps. Sampling these mipmaps at the subsplat's resolution allows us to detect significant depth or normal variations within the subsplat. If the difference between the max and min depth values exceeds a threshold, or the difference in any of the normal components exceed a similar threshold, the subsplat is refined into four higher-resolution subsplats (see Figure 4). Refinement occurs iteratively until the subsplat contains no discontinuities or we reach the maximal refinement level. A separate list of subsplats can be refined for each VPL, or a single list can be refined and used for all VPLs.

Figure 3: *Each illumination buffer level contains pieces of the final indirect illumination at a different resolution. On the right: artifacts from combining the illumination buffers using nearest neighbor upsampling and naive bilinear interpolation.*

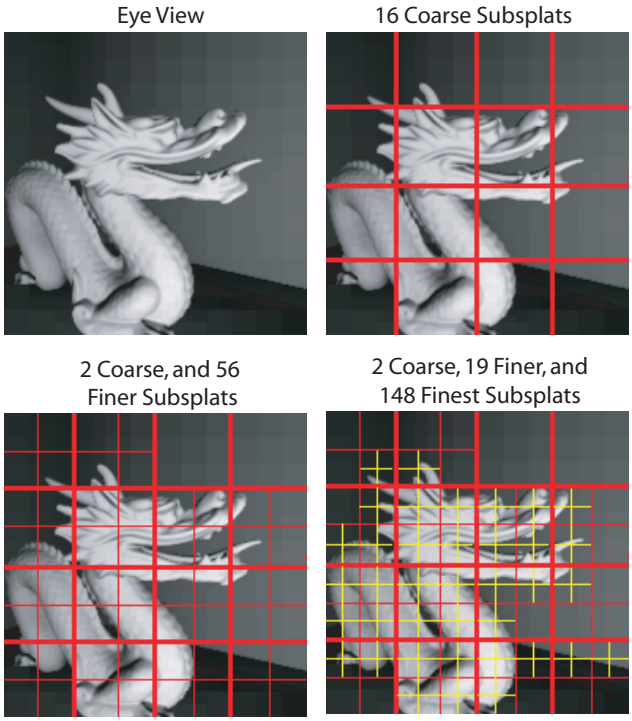

Figure 4: *Subsplat refinement occurs in areas with depth or normal discontinuities. In this case, we demonstrate how a splat from a single virtual light might be refined twice.*

3.4 Rendering Indirect Illumination

After iteratively refining, we have a large list of subsplats. Our implementation requires a three-tuple to store each subsplat, with one value specifying the subsplat's output resolution, and two values specifying its screen-space location.

During rendering, a vertex shader positions each subsplat in the correct layer of the illumination buffer, a fragment shader computes indirect illumination for each subsplat, and additive blending accumulates all the contributions. For each subsplat, our indirect illumination is computed as:

$$I(x_s, x_l) = \rho_l \rho_s \Phi_l \frac{\max\{\vec{V_{ls}} \cdot \vec{N_l}, 0\} \max\{\vec{V_{sl}} \cdot \vec{N_s}, 0\}}{|\vec{V_{ls}}|^2}, \quad (1)$$

where x_s and x_l are the shaded point and the virtual light point, $\vec{N_s}$ and $\vec{N_l}$ are the surface normals at x_s and x_l, Φ_l is the flux to the VPL at x_l, ρ_s and ρ_l are the BRDFs at x_s and x_l, and $\vec{V_{ls}}$ and $\vec{V_{sl}}$ are respectively the vectors from x_l to x_s and from x_s to x_l. These values are retrieved from the appropriate location either in the RSM or the G-buffer.

After rendering all subsplats, the illumination buffer contains the total indirect illumination split into disjoint components at various resolutions. Naively summing the layer contributions gives a blocky representation of total indirect light (see Figure 3) that requires interpolating before use.

3.5 Upsampling and Combination

Figure 3 demonstrates the artifacts from naive methods of combining multiresolution illumination: blocky illumination when using nearest neighbor upsampling, and strange multiresolution haloing and ringing when using bilinear interpolation. Clearly, neither is acceptable.

The problem lies in the complex structure of the illumination buffer. Each texel contains either all of the illumination for that eye-space location, or none at all. Linear interpolation between a texel containing all of its relevant light and one containing no illumination makes little sense, as it spreads energy from texels containing energy to those deemed too coarse or too fine. Given that roles may be reversed at other levels in the illumination buffer, multiresolution linear interpolation leads to ringing and haloing. Essentially, this arises from the varying regions of support for the interpolation filter at multiple scales.

To deal with this, we utilize a unique upsampling scheme. Upscaling progresses from coarsest to finest layers in the illumination buffer. At each resolution, every texel that contains illumination information is linearly interpolated with neighboring texels of the same resolution, whether they were originally rendered at that resolution or upsampled from a lower resolution. To avoid the haloing and ringing shown in Figure 3, each upsampling pass only outputs interpolated texels in locations where the illumination buffer

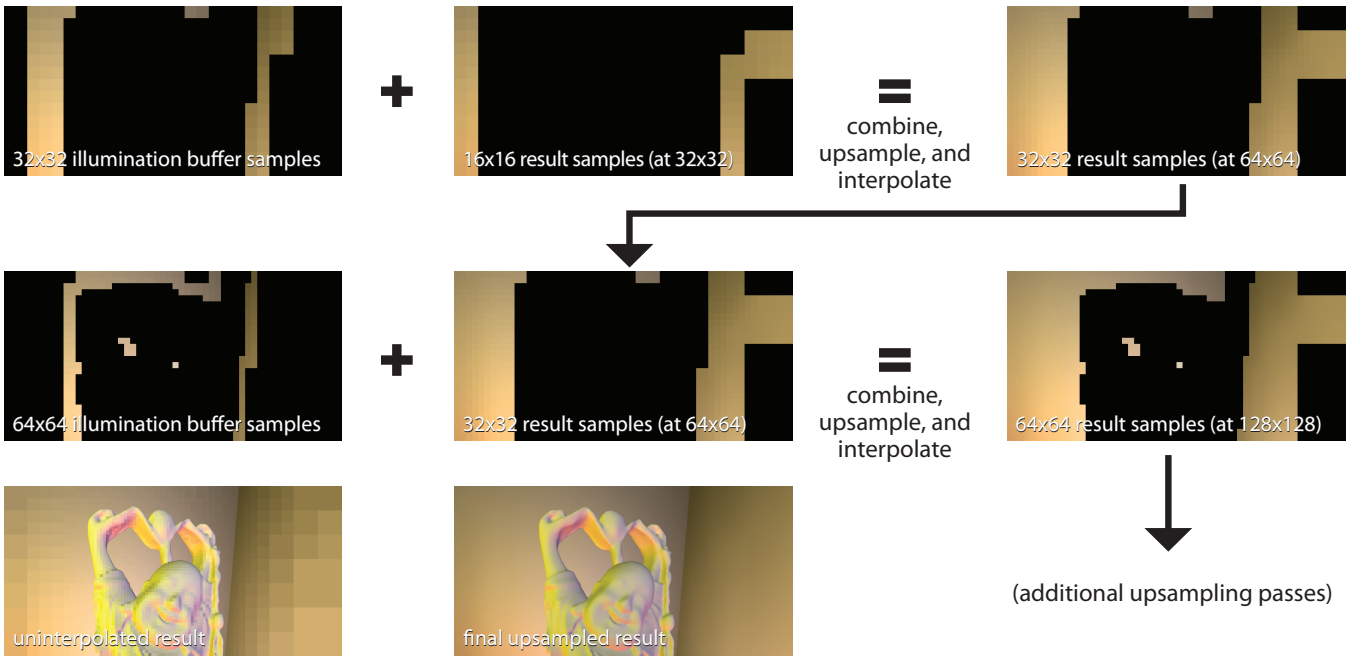

Figure 5: *Two upsampling passes, from 32^2 to 64^2 to 128^2. At each level, missing samples are combined with interpolated data from previous levels. The result is interpolated to higher resolution, and used as the input for the next upsampling pass. Lower left: the non-interpolated combination, and the final interpolated result.*

Figure 6: *Indirect lighting from a Buddha with a Phong BRDF; here, subsplats are refined separately for each VPL. To emphasize the contribution, a blue tint was added to indirect light reflected by the Buddha.* (Left to right): *direct lighting, indirect lighting, and combined result. Indirect illumination rendered at 256^2 at 23 fps.*

already contained data for that resolution: energy is never pulled from empty areas nor spread into them, and all texels that contained no energy at the start of each pass remain empty. Figure 5 demonstrates this process graphically through two upsampling steps.

After processing all the layers, we have a single combined and upsampled image that varies smoothly, without ringing artifacts. Furthermore, this method does not spread energy across major discontinuities. Our refinement process ensures that areas with these discontinuities are refined into high resolution subsplats. Because each texel is interpolated only with texels of equal resolution and those upsampled from coarser resolutions, energy stays on the correct side of a discontinuity.

4 Implementation

We implemented our method using OpenGL and GLSL on a machine with a dual-core 3GHz Pentium 4 and a GeForce GTX 280. Our implementation uses OpenGL's geometry shader and transform feedback extensions. All images and results in this paper were generated using a final output resolution of 2048^2, which was downsampled to a 1024^2 window for an antialiased rendering.

4.1 VPLs and Subsplat Refinement

Our initial implementation refined a separate list of subsplats for each VPL, using discontinuities in computed illumination to guide the refinement process. This enabled the use of our technique with arbitrary BRDFs (such as the Phong material shown in Figure 6), where sharper highlights arising from material properties may require additional refinement in areas that cannot be determined from normal or depth discontinuities alone.

With this approach, generating and refining a list of subsplats for each VPL is the most expensive part of our method, often outpacing subsplat rendering by a factor of 3 or more. The two major sources of this expense are the high bandwidth requirements for processing large number of subsplats, and the high cost of the geometry shader that does the splitting. In addition to sampling for normal and depth discontinuities, this shader must compute indirect illumination at multiple points when determining whether to split a subsplat.

For diffuse scenes we found that splats were being split almost identically for each VPL. Noting this, we implemented an alternate approach that performs splat refinement just once per frame, refining solely based on normal and depth discontinuities. All VPLs then reuse the same subsplats for rendering. In diffuse scenes, this approach yields a significant increase in performance while maintaining similar image quality. Unless otherwise stated, all images and results in this paper are generated using this second approach.

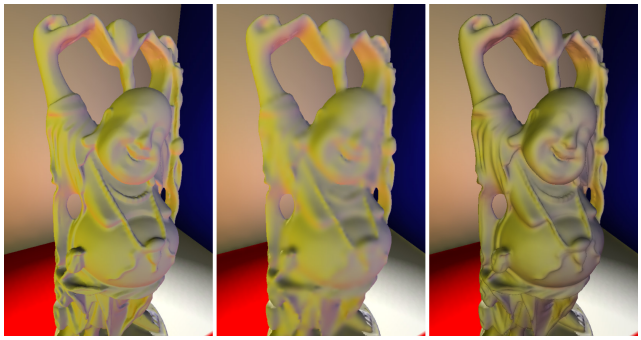

Figure 7: *The Buddha model rendered at 1024^2. The left image, rendered with a normal splitting threshold of 0.25, runs at 8 fps; the two images on the right use a threshold of 1.4 and run at 32 fps. The rightmost image approximates surface detail with the technique described in Section 4.2, using $\alpha = 0.5$.*

4.2 Adding Surface Detail

Given that a large number of subsplats must be rendered for each VPL, the number of subsplats generated has a significant impact on the performance of our technique. Since this number is heavily influenced by thresholds within the refinement process, these thresholds can serve as a parameter for turning performance. This is particularly true of the threshold used to detect normal discontinuities, which ranges from $[0..2]$: in the flying bunny scene, with an indirect resolution of 1024^2, we found that tweaking this parameter alone resulted in framerates anywhere between 5 and 35 fps. A higher threshold allows faster framerates, at the expense of fine surface details. This can be seen in Figure 7: with high normal thresholds, some surface features of the Buddha are not significant enough to trigger refinement. These areas are then rendered at low resolution, effectively blurring them out.

One solution is to approximate the missing detail, modulating the indirect illumination using the surface normal's alignment towards the camera. For each image-space pixel of indirect illumination c, with view vector $\vec{V}$ and surface normal $\vec{N}$ at the same location, and a parameter α (ranging from $[0..1]$) controlling the intensity of the modulation effect:

$$c_{out} = c * (\alpha * \max\{\vec{V} \cdot \vec{N}, 0\} + (1 - \alpha)) \qquad (2)$$

This reintroduces surface details to the indirect illumination in a visually plausible manner, allowing the use of higher normal thresholds; this in turn reduces the number of splats and improves framerates. This approximation may cause some areas to appear overly dark, which may be objectionable in some circumstances. The third panel of Figure 7 illustrates the effect of replacing surface detail using this method.

5 Results and Discussion

Like other splatting approaches for indirect illumination, the performance bottleneck in our technique is the cost of rendering the indirect subsplats. The resolution of each subsplat does not matter: in our multiresolution approach each subsplat covers just a single texel, so the cost of rendering each subsplat is independent of resolution. This also allows us to render subsplats as points, rather than triangle meshes, quads, or point sprites. We achieved a 5% speed increase simply by switching from quads to points.

Indirect Resolution	Simple Teapot (6.3K triangles)	Spinning Buddha (250K triangles)	Dragon & Bunny (405K triangles)	Flying Bunnies (417K triangles)
128^2	68 fps	60 fps	59 fps	52 fps
256^2	50 fps	42 fps	42 fps	41 fps
512^2	33 fps	30 fps	28 fps	31 fps
1024^2	22 fps	24 fps	21 fps	25 fps

Table 1: *Framerates at various resolutions of indirect illumination.*

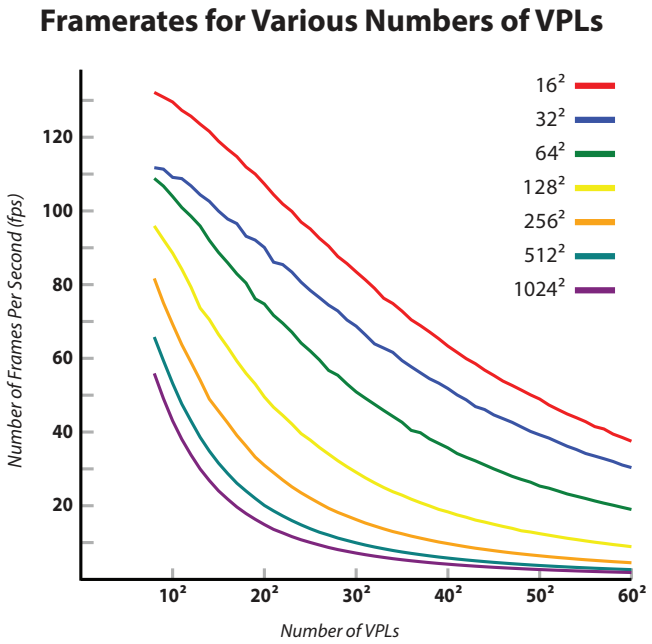

Figure 8: *Framerates of our method in the dragon scene using increasing numbers of VPLs, at various resolutions.*

As demonstrated by the data in Table 1, our performance has little dependency on geometric scene complexity: when rendering indirect illumination at 1024^2, performance is similar whether the scene contains a single teapot or many complex models. Geometry is rasterized twice, one rendering from the light and one from the eye. Only at fairly low resolutions does the geometric complexity of a scene have a significant effect on performance. However, our method is sensitive to the *visual complexity* of a scene. When looking at a flat wall, few subsplat refinements are required; complex geometry requires refinement around edges, creases, and crevices; and high frequency random geometry may require a uniformly dense refinement, where a naive splatting technique would outpeform our multiresolution approach.

As with other splatting techniques, we ignore visibility considerations when accumulating indirect illumination, which can result in an overly bright image. VPLs on a scene's ceiling illuminate not only the surface of a table, but also the floor below the table. One approach might approximation visibility independently, e.g., using ambient occlusion, and modulate the results. In the future, we hope to explore multiresolution techniques that account for visibility.

Figure 1 demonstrates the dragon and bunny scene rendered with direct illumination only, and with indirect illumination subdivided to a maximum refinement of 512^2. Figure 9 demonstrates several different configurations of indirect illumination, rendered with a

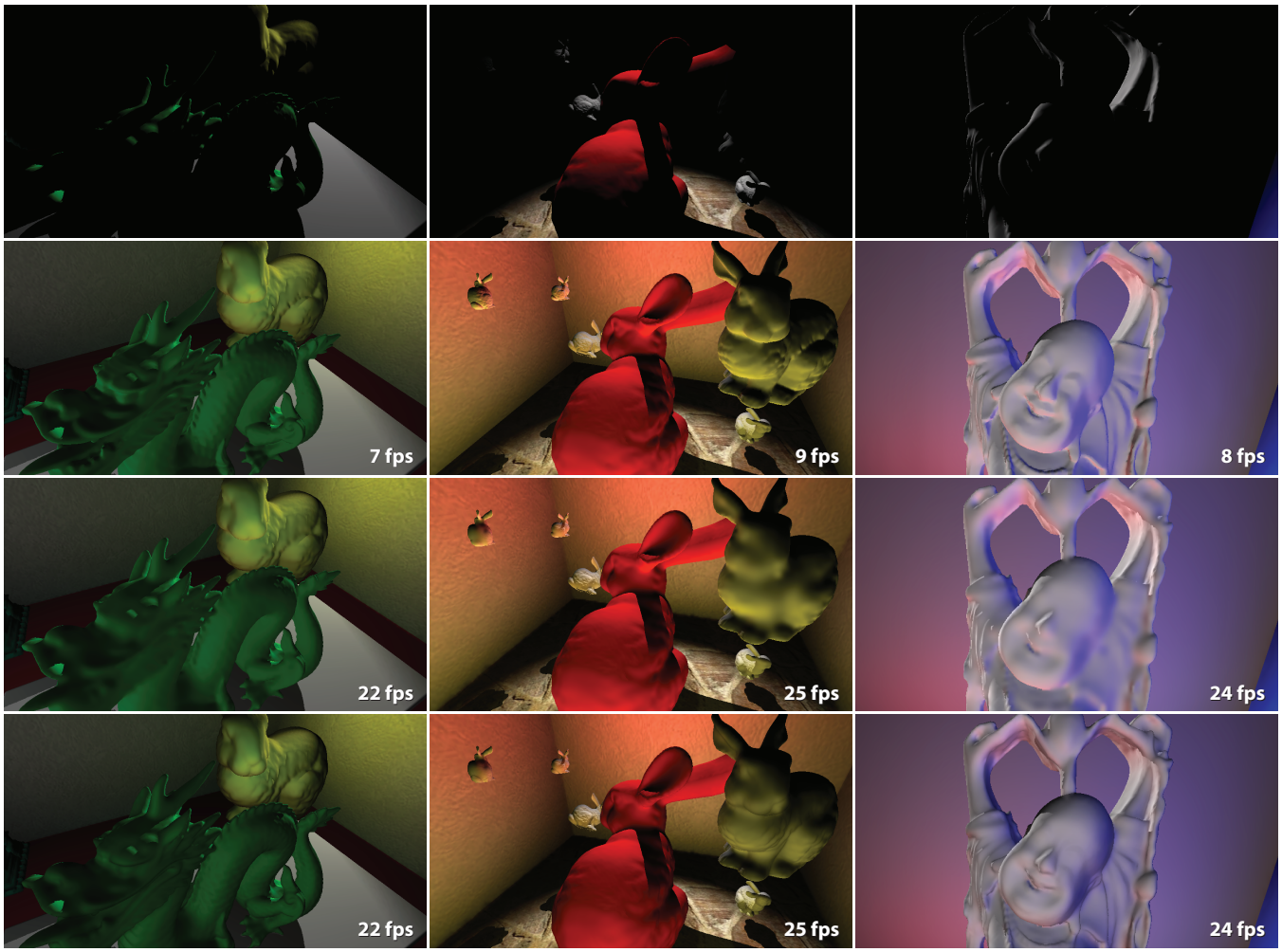

Figure 9: *Example scenes using our technique, with indirect illumination rendered at* 1024^2*. Top row: direct illumination only. Second row: indirect illumination generated with normal threshold* 0.2*. Third row: using normal threshold* 1.5*. Fourth row: using normal threshold* 1.5*, and replacing surface detail using the method described in Section 4.2 (with* $\alpha = 0.5$*).*

maximum refinement level of 1024^2. The second row is rendered with a low normal threshold, and the third and fourth rows with a high threshold. The fourth row illustrates the effects of reintroducing detail using the method described in Section 4.2.

Figure 8 explores performance on the dragon scene, showing variations due to the number of VPLs and refinement passes. With no refinement, indirect illumination is rendered at 16^2 resolution. Each refinement doubles the illumination resolution (up to 1024^2 after 6 passes). In our scenes 16^2 to 30^2 VPLs gave smooth results under animation, without popping artifacts. We regularly sample the reflective shadow maps to choose our virtual lights; a more sophisticated VPL sampling scheme may reduce the number required.

6 Conclusion

This paper introduced a novel multiresolution splatting technique, and described its application to the rendering of indirect illumination with a reflective shadow map. Our method reduces the cost of rendering indirect illumination by rendering each piece of it at the lowest possible resolution. This allows indirect illumination to be rendered at high resolutions at interactive rates, without artificially restricting each VPL's ability to contribute to the entire scene.

Several interesting avenues of future work exist. The fact that each subsplat is rendered and refined separately allows interesting possibilities: each subsplat could be rendered with only the appropriate set of VPLs, for example, reducing the total number of subsplats that needed to be rendered and thereby increasing performance. Additionally, our method might benefit from more intelligent traversal metrics that would offer finer control over subsplat refinement, perhaps detecting convergence rather than simply iterating a set number of times. Finally, we believe that our basic multiresolution splatting approach is applicable to any number of additional rendering problems, such as caustics or shadows, that do not require all areas of an image to be rendered at high resolution.

References

BUNNELL, M. 2005. *GPU Gems 2*. Addison-Wesley, ch. Dynamic Ambient Occlusion and Indirect Lighting, 223–233.

CARR, N. A., HOBEROCK, J., CRANE, K., AND HART, J. C. 2006. Fast gpu ray tracing of dynamic meshes using geometry images. In *Proceedings of Graphics Interface*, 203–209.

COHEN, M. F., AND WALLACE, J. R. 1993. *Radiosity and Realistic Image Synthesis*. Academic Press Professional.

DACHSBACHER, C., AND STAMMINGER, M. 2005. Reflective shadow maps. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, 203–231.

DACHSBACHER, C., AND STAMMINGER, M. 2006. Splatting indirect illumination. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, 93–100.

DACHSBACHER, C., STAMMINGER, M., DRETTAKIS, G., AND DURAND, F. 2007. Implicit visibility and antiradiance for interactive global illumination. *ACM Transactions on Graphics 26*, 3, 61.

DONG, Z., KAUTZ, J., THEOBALT, C., AND SEIDEL, H.-P. 2007. Interactive global illumination using implicit visibility. In *Proceedings of Pacific Graphics*, 77–86.

GAUTRON, P., KŘIVÁNEK, J., BOUATOUCH, K., AND PATTANAIK, S. N. 2005. Radiance cache splatting: A GPU-friendly global illumination algorithm. In *Proceedings of the Eurographics Symposium on Rendering*, 55–64.

GUENNEBAUD, G., BARTHE, L., AND PAULIN, M. 2006. Real-time soft shadow mapping by backprojection. In *Eurographics Symposium on Rendering (EGSR)*, 227–234.

HANRAHAN, P., SALZMAN, D., AND AUPPERLE, L. 1991. A rapid hierarchical radiosity algorithm. In *Proceedings of SIGGRAPH*, 197–206.

HERZOG, R., HAVRAN, V., KINUWAKI, S., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2007. Global illumination using photon ray splatting. *Computer Graphics Forum 26*, 3, 503–513.

IWASAKI, K., DOBASHI, Y., YOSHIMOTO, F., AND NISHITA, T. 2007. Precomputed radiance transfer for dynamic scenes taking into account light interreflection. In *Proceedings of the Eurographics Symposium on Rendering*, 35–44.

KAUTZ, J., LEHTINEN, J., AND AILA, T. 2004. Hemispherical rasterization for self-shadowing of dynamic objects. In *Proceedings of the Eurographics Symposium on Rendering*, 179–184.

KELLER, A. 1997. Instant radiosity. In *Proceedings of SIGGRAPH*, 49–56.

KONTKANEN, J., AND LAINE, S. 2005. Ambient occlusion fields. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, 41–48.

KRISTENSEN, A. W., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Precomputed local radiance transfer for real-time lighting design. *ACM Transactions on Graphics 24*, 3, 1208–1215.

LAINE, S., SARANSAARI, H., KONTKANEN, J., LEHTINEN, J., AND AILA, T. 2007. Incremental instant radiosity for real-time indirect illumination. In *Proceedings of Eurographics Symposium on Rendering*, xx–yy.

LAUR, D., AND HANRAHAN, P. 1991. Hierarchical splatting: a progressive refinement algorithm for volume rendering. In *Proceedings of SIGGRAPH*, 285–288.

MALMER, M., MALMER, F., ASSARSSON, U., AND HOLZSCHUCH, N. 2007. Fast precomputed ambient occlusion for proximity shadows. *Journal of Graphics Tools 12*, 2, 59–71.

REN, Z., WANG, R., SNYDER, J., ZHOU, K., LIU, X., SUN, B., SLOAN, P.-P., BAO, H., PENG, Q., AND GUO, B. 2006. Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. *ACM Transactions on Graphics 25*, 3, 977–986.

RITSCHEL, T., GROSCH, T., KAUTZ, J., AND MUELLER, S. 2007. Interactive illumination with coherent shadow maps. In *Proceedings of the Eurographics Symposium on Rendering*.

RUSINKIEWICZ, S., AND LEVOY, M. 2000. Qsplat: A multiresolution point rendering system of large meshes. In *Proceedings of SIGGRAPH*, 343–352.

SAITO, T., AND TAKAHASHI, T. 1990. Comprehensible rendering of 3-d shapes. In *Proceedings of SIGGRAPH*, 197–206.

SAMET, H. 1990. The design and analysis of spatial data structures. Addison-Wesley.

SHAH, M., KONTTINEN, J., AND PATTANAIK, S. 2007. Caustics mapping: An image-space technique for real-time caustics. *IEEE Transactions on Visualization and Computer Graphics 13*, 2, 272–280.

SHANMUGAM, P., AND ARIKAN, O. 2007. Hardware accelerated ambient occlusion techniques on gpus. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, 73–80.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics 21*, 3, 527–536.

SLOAN, P.-P., GOVINDARAJU, N., NOWROUZEZAHRAI, D., AND SNYDER, J. 2007. Image-based proxy accumulation for real-time soft global illumination. In *Proceedings of Pacific Graphics*, 97–105.

TABELLION, E., AND LAMORLETTE, A. 2004. An approximate global illumination system for computer generated films. *ACM Transactions on Graphics 23*, 3, 469–476.

TEVS, A., IHRKE, I., AND SEIDEL, H.-P. 2008. Maximum mipmaps for fast, accurate, and scalable dynamic height field rendering. In *Proceedings of the Symposium on Interactive 3D graphics and games*, 183–190.

TOLE, P., PELLACINI, F., WALTER, B., AND GREENBERG, D. 2002. Interactive global illumination in dynamic scenes. In *Proceedings of SIGGRAPH*, 537–546.

WYMAN, C., AND DACHSBACHER, C. 2008. Reducing noise in image-space caustics with variable-sized splatting. *Journal of Graphics Tools 13*, 1, 1–17.

WYMAN, C. 2008. Hierarchical caustic maps. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, 163–171.

ZHOU, K., HU, Y., LIN, S., GUO, B., AND SHUM, H.-Y. 2005. Precomputed shadow fields for dynamic scenes. *ACM Trans. Graph. 24*, 3, 1196–1201.

ZHUKOV, S., IONES, A., AND KRONIN, G. 1998. An ambient light illumination model. In *Proceedings of the Eurographics Rendering Workshop*, 44–45.