

Increased Photorealism for Interactive Architectural Walkthroughs

Rui Bastos, Kenneth Hoff, William Wynn, and Anselmo Lastra

University of North Carolina at Chapel Hill

ABSTRACT

This paper presents a new method for interactive rendering of globally illuminated static scenes. Global illumination is decomposed into view-independent (diffuse) and view-dependent (non-diffuse) components. The two are recombined during rendering using a hybrid geometry- and image-based approach along with multi-pass blending techniques. This approach allows the preprocessing of both components and the fast rendering of globally illuminated scenes.

The view-independent component uses a traditional precomputed geometry-based radiosity solution that is rendered using standard graphics hardware. The view-dependent component is decomposed into “what is reflected” (radiance with depth) and “how it is reflected” (BRDF), and precomputed and rendered using image-based approaches. Radiance is stored as images with depth, and rendered using perspective reprojection; the BRDF is decomposed into an integration of incoming radiance and a directional modulation. The radiance integration term is approximated by convolving the reflected image with precomputed kernel textures based on material properties. The directional modulation is stored as a reflectance modulation texture based on material properties and is rendered using sphere-mapping during a blending pass.

Keywords: interactive walkthroughs, global illumination, glossy, BRDF, image warping

1. INTRODUCTION

Architectural walkthrough systems are expected to give convincingly realistic interactive visualizations of complex virtual environments [5]. These systems are often used in virtual prototyping of building designs, stage and set lighting design, and architectural design reviews where the demands for greater realism and higher frame-rates are always increasing. In order to provide a convincing simulation, we must attempt to mimic the model’s real-world counterpart by accurately representing the visual complexity of the scene while maintaining a smooth interactive frame-rate (greater than 20 frames per second). To capture the intricate geometry and complex lighting effects of real scenes, we require a detailed geometric model and an accurate lighting simulation. However, to maintain interactive frame-rates, the accuracy of one or both must often be compromised.

Previous work in interactive architectural walkthroughs focused on the problem of accurate geometric modeling and interactive

rendering. Focus was placed on the problem of quickly rendering a complex model, rather than on photorealism. The primary goal was reducing the number of graphics primitives rendered per frame without noticeably degrading image quality.

Accurate modeling of global illumination has been in general prohibitively expensive for interactive walkthroughs; consequently, most of the previous work placed little emphasis on an accurate or complete illumination simulation. Global illumination was generally decomposed into view-independent and view-dependent components with only a small subset of each actually being simulated.

View-independent illumination accounts for all effects of light that are not dependent on the viewer’s position. This illumination depends only on the configuration of the geometry and the lights, and so may be precomputed and stored as a dense mesh with per-vertex colors or as texture-maps that can be rendered using traditional shading hardware [6][18].

View-dependent illumination accounts for specular and glossy directionally-biased lighting. Diffuse reflection is an equal scattering in all directions based on an incoming light direction; specular reflection bounces the incoming light in a strictly narrow band of reflected directions (perfect specularity means there is only one reflected direction). Glossy reflection refers to the continuum between perfectly diffuse and perfectly specular reflections. Glossy surfaces have a directional bias that is defined by the bi-directional reflectance distribution function (BRDF). These effects depend on the viewpoint and are difficult to precompute and render accurately.

2. CONTRIBUTIONS

Our approach focuses on interactive architectural visualization that accounts for view-dependent glossy illumination. Unlike previous global illumination research, we emphasize structuring the model database to allow fast rendering at interactive rates with view-dependent illumination from arbitrary viewpoints. In addition to purely specular view-dependent effects, we properly capture imperfect reflection scattering from glossy surfaces.

We present a hybrid geometry- and image-based approach that offers the following:

- Decomposition of a glossy reflected image into incoming radiance (“what is reflected”) and a BRDF (“how it is reflected”) – this allows precomputation of the reflection visibility and material properties.
- An image-based approach to handle interactive reflections – a set of images augmented with depth is precomputed for each reflector and is then reprojected into any reflected view in constant time (independent of scene complexity).
- Decomposition of BRDF into a convolution kernel and a directional modulation reflectance texture-map based on the material properties – this allows graphics hardware to efficiently use general BRDFs at rendering time for simulation of glossy view-dependent effects.
- Rendering reflected images at an appropriate resolution depending on the level of scattering of a glossy surface – this avoids multi-pass high-resolution image accumulation.

CB# 3175, Sitterson Hall; Chapel Hill, NC 27599-3175 USA
{bastos | hoff | wynn | lastra}@cs.unc.edu
<http://www.cs.unc.edu/~{bastos | hoff | wynn | lastra}>

3. RELATED WORK

The Phong model [17] is the standard for approximating specular highlights. This model works well for certain shiny materials; however, it suffers from drawbacks. One is that indirect lighting and visibility are not considered. In addition, it is normally evaluated in hardware on a per-vertex basis. This results in visible artifacts, and hence the number of primitives must be increased to effectively capture the highlights.

Walter *et al.* [21] presented an approach that fits “virtual lights” to non-diffuse objects. A view-independent non-diffuse global illumination solution is computed and used for fitting. The scene is then rendered by Gouraud-shading the Lambertian view-independent global illumination and by Phong-lighting the specular objects using the fitted virtual lights. Phong lighting is used as a set of appearance basis functions, instead of a lighting model. The superposition of all contributions from virtual lights approximates the global illumination of the non-diffuse objects, including view-independent content and specular highlights. While effective for low-gloss surfaces, mirror-like and highly glossy reflections require excessive directional information.

Stürzlinger and Bastos [19] presented a splatting-based alternative for rendering global illumination solutions; their approach reconstructs both view-independent and view-dependent components without decoupling them. A particle tracing method is used to emit power-carrying particles from the light sources and to track them through the environment until they are probabilistically absorbed. All the particle hit-points are stored along with their incoming directions and corresponding surfaces, and each is associated with a kernel texture (Gaussian) centered at the hit-point. At rendering time, each kernel texture is intensity-scaled according to the surface’s BRDF and the current viewpoint. The overlap of all the kernel textures on a surface reconstructs view-independent and view-dependent components of the global illumination on the surface. Similar to Walter *et al.*, this approach also requires excessive directional information for highly glossy and mirror-like surfaces.

Environment-mapping is another method for providing fast view-dependent illumination. This technique efficiently accounts for mirror-like reflections by precomputing the incoming radiance for a reflector into a world-projection texture-map [4][11]. At runtime, this texture is indexed using the reflected view vector for each vertex and then mapped onto the reflector surface. This technique is often implemented in hardware in the form of sphere-mapping [15], making it nearly as fast as regular texture-mapping. Environment-mapping also has significant drawbacks. No motion parallax effects can be observed and the viewer is assumed to be infinitely far away from the infinitely small reflector. These drawbacks can be overcome by computing the reflected vector per pixel [20] and by warping the texels [3].

Multi-pass methods are another alternative for rendering non-diffuse environments [8]. Planar reflections are computed by re-rendering the entire scene for all the mirrored viewpoints. Glossy reflection is approximated by using per-pixel fog effects and by accumulating several specular images in a stochastic multi-sampling manner. Although the results can be impressive, the computational time limits its application to small environments.

General global illumination research properly accounts for the view-dependent components of illumination [6][18]. Most

research in this area focuses on creating an accurate single image in a reasonable amount of time. Emphasis is placed on high quality, measurable accuracy, and numerical robustness of the methods. Multiple frames/second is not usually an objective.

Recently, Lischinski and Rappoport [13] presented an image-based technique for rendering non-diffuse synthetic scenes based on layered depth images. They capture both view-independent and view-dependent appearance as images and recombine these two components to render images from any viewpoint; the results outperform ray tracing with similar quality for shading and reflections. The main disadvantages of their approach are the size of the required data structures, and the rendering time, which is dependent on the depth complexity of the scene.

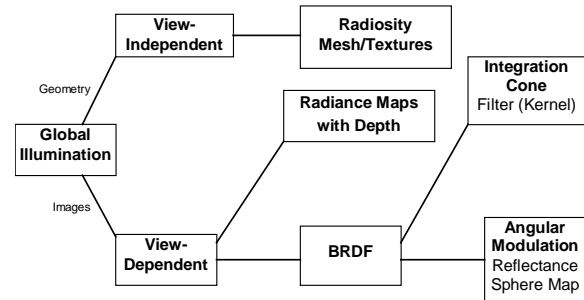


Figure 1 Our decomposition of global illumination.

4. OVERVIEW

As did previous researchers, we decompose the global illumination into view-independent diffuse and view-dependent glossy components. We store and render the diffuse scene as a geometric database with an associated precomputed radiosity solution, using either dense meshing and per-vertex color or a set of radiosity textures [2], both of which can easily be rendered on today's typical graphics hardware.

Unlike previous approaches to interactive photorealistic rendering, we decompose the view-dependent component into a visibility step and a material-properties step. The overall illumination for a reflective surface is determined by the incoming radiance and the reflectance distribution properties of the material. The incoming radiance represents what is visible from any point on a reflective surface, and the reflectance distribution represents how the reflection is blurred and modulated across the surface of the reflector. We can think of the incoming radiance as representing “what a reflector sees”, while the BRDF represents “how it is seen”; this allows the simulation of imperfect glossy reflections across arbitrary reflective surfaces. The incoming radiance is sampled as a set of images augmented with depth, which we refer to as *radiance maps*. These radiance maps are precomputed by sampling over the hemisphere of possible reflected view directions. At runtime, a subset of radiance maps closest to the current reflected viewpoint is reprojected, in constant time, onto the surface of the reflector. The BRDF is decomposed into an integration of incoming radiance and a directional modulation. The incoming radiance integration is approximated by convolving reflected images with material-based texture kernels. The directional modulation is approximated by a reflectance texture-map that blends in the reflected image, in a view-dependent manner, using sphere-mapping.

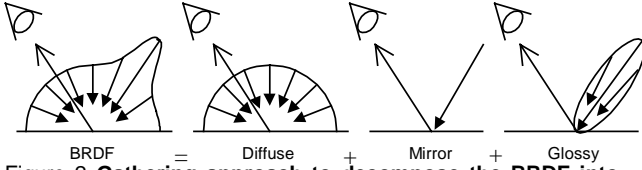


Figure 2 **Gathering approach to decompose the BRDF into qualitative components.**

One approach to computing the incoming radiance for reflective surfaces is to use geometry [8][15], which requires a rendering pass using the fully detailed geometric database for each reflected viewpoint. Glossy effects are then usually obtained by accumulation of jittered images. By using an image-based radiance sampling, we obtain the ability to render at an appropriate resolution that depends on the BRDF scattering. Mirror-like surfaces require high-resolution images and narrow convolution kernels, while nearly diffuse surfaces require low-resolution images and wide convolution kernels. This is a material property, so the radiance samples can be precomputed and stored at a minimum resolution. In short, to control a material’s light scattering effect, we render at a resolution and use a convolution kernel chosen appropriately for the reflective material.

We restrict our focus to glossy, first-order, planar reflections; however, we later discuss how to extend this technique to curved surfaces and recursive reflections. We emphasize a decomposition of the global illumination that allows a hybrid geometry and image-based approach, which efficiently accounts for glossy reflections. Due to the light scattering of glossy reflections, higher-order reflections often contribute very little to the final scene’s illumination.

5. DECOMPOSITION OF ILLUMINATION

Our rendering method approximates global illumination by combining three different lighting components. Each component is determined by the material properties of the surfaces to be rendered. We understand the BRDF from the observer’s viewpoint – as a light-gathering approach (see Figure 2). It describes how the material spreads light of an incoming solid angle into a single outgoing direction, the viewer’s direction (a single ray connecting the viewer to the point spreading the incoming light). We decompose the general BRDF into the sum of three qualitatively different components (see Figure 2): diffuse (Lambertian) reflection, mirror (ideal specular) reflection, and glossy (directionally diffuse) reflection [6].

For simplicity, we consider that a pixel receives light from the scene only along a single direction. This represents the evaluation of the rendering equation at the intersection point, *i.e.*, the radiance leaving the primitive in the direction of the viewer. We restrict our discussion to the reflection of light, but algorithmic similarities allow our techniques to be applied for transmission of light as well.

In the general case, the total outgoing radiance, $L_o(x, \omega_o)$, leaving a point x in an outgoing direction ω_o , depends on an emission term and a reflection term. The emission term represents the emitted radiance, $L_e(x, \omega_o)$, emitted from point x in direction ω_o . The reflection term is the integration of the incoming radiance, $L_i(x, \omega_i)$, over the hemisphere, Ω , covering the surface at point x , weighted by the BRDF, $\rho_{bd}(x, \omega_o, \omega_i)$, at that point and at outgoing direction ω_o . Mathematically, this defines the rendering equation:

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_{\Omega} \rho_{bd}(x, \omega_o, \omega_i) L_i(x, \omega_i) \cos \theta d\omega_i \quad (1)$$

where θ is the angle between the normal to the surface at x and the incoming direction ω_i . This equation can be split into the sum

of three components—based on the material properties—as presented above. Our rendering approach approximates the rendering equation as the sum of the following components.

For the ideally diffuse component, the BRDF is a constant and can be factored out of the integral in equation (1). This component reduces to evaluating the radiosity equation [6]. For the mirror-like component, the BRDF is a delta function and the integrand is non-null only when the incoming direction ω_i is exactly equal to the outgoing direction ω_o . This component reduces to evaluating the incoming radiance $L_i(x, \omega_o)$ weighted by the BRDF at the reflection point x . The non-ideal (non-diffuse and non-mirror) surface properties require integration of the contribution of radiance reflected from all the incoming directions weighted by the corresponding BRDF at that particular point.

The next section describes how our method handles the computation and the rapid rendering of these view-independent and the view-dependent components of the rendering equation.

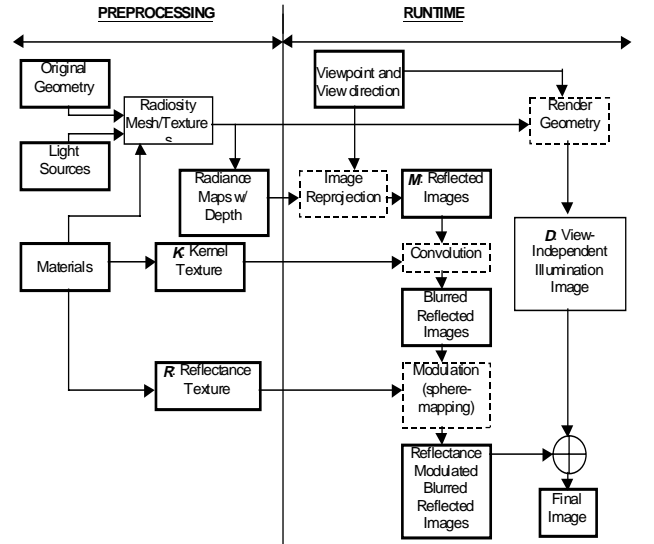


Figure 3 **Flow of data and operations for rapid rendering of globally illuminated scenes. Data is represented with solid rectangles and operations are represented with dashed rectangles.**

6. RENDERING PIPELINE

Our hybrid method for rapid rendering of globally illuminated scenes combines traditional geometry-based rendering with image-based rendering to approximate the rendering equation (1), by summing the three components described in section 5. Our multi-pass, per-pixel, shading equation is expressed as the sum of a view-independent term and a view-dependent term:

$$L_o = k_d D + k_s (M * K) R, \quad k_d + k_s \leq 1 \quad (2)$$

where L_o is the outgoing radiance from the intersected point in the direction of the viewer, D is the ideally diffuse component obtained from a rendered geometry-based radiosity solution, M is the mirror-like component (reflected image), K is the kernel texture extracted from the material’s BRDF, R is the reflectance modulation texture also deduced from the material’s BRDF, k_d and k_s are the magnitudes of the diffuse and specular components, and “*” represents the convolution operation. Note that M , K , and R are image-based quantities. Additions and multiplications are performed using blending functions in hardware. Note also that the mirror component is a special case of the view-dependent term with $K=I$ and $R=I$.

Figure 1 shows how we split global illumination into geometry and image-based components and how we process each independent component. Figure 3 is a diagram presenting the flow of data and operations used to compute images with our system by evaluating shading equation (2). We split the rendering pipeline into preprocessing and runtime phases. The preprocessing phase is composed of the following sub-phases:

Given geometry and information about materials and light sources, a radiosity solution is computed for the scene. This radiosity solution represents the ideally diffuse component D of equation (2).

Given the radiosity solution for the scene, the reflectors are sampled to capture view-independent incoming radiance. This incoming radiance is stored as a set of radiance maps with depth and represents the visibility information of the view-dependent component M .

Given the material information of each reflector, a kernel texture K and a reflectance modulation sphere-map R are computed based on the reflector's BRDF. These textures are used to post-process the reflected images for the corresponding reflectors.

The runtime rendering phase progresses as follows:

```
RenderGloballyIlluminatedScene( ViewPose )
  foreach visible reflector
    Create stencil region by rendering reflector as seen from ViewPose.
    Compute a reflected view pose, RefViewPose
    Select appropriate set of cameras using RefViewPose.
    Reproject selected radiance maps to RefViewPose // M
    Convolve framebuffer with kernel K. // M*K
    Enable sphere mapping
    Modulate framebuffer region with modulation texture R
      by drawing the reflector as seen from ViewPose.
    Disable sphere mapping //  $k_s(M*K)R$ 
    Sum diffuse component by rendering the reflector from ViewPose
      //  $k_dD+k_s(M*K)R$ 
    Disable stencilling.
  foreach visible ideally diffuse object
    Render the object from ViewPose.
```

The color plate illustrates this algorithm, by showing the results in the frame buffer for the runtime phase using a simple model. The next sections describe in more detail how we handle the main operations in the rendering pipeline.

7. VIEW-INDEPENDENT ILLUMINATION

In our approach, given the original geometry of the scene and light source information, a phase precomputes the view-independent global illumination solution for the scene. A traditional finite-element (radiosity) approach [6][18] is used to solve equation (1) for the view-independent case. The output is a dense per-vertex color mesh approximating the view-independent global illumination of the scene. Large surfaces in the scene are finely subdivided to capture lighting information. We use *Lightscape Technologies* software to compute our radiosity-illuminated models.

Given the limit in number of primitives per frame that can be rendered by graphics engines, walkthrough applications usually try to reduce mesh complexity (and lighting information) by using simplification techniques [9]. Instead of computing a dense radiosity solution and then decimating it, we improve rendering performance by replacing the fine meshing with the original larger surfaces painted with radiosity textures [2].

8. VIEW-DEPENDENT ILLUMINATION

We split the view-dependent property of reflected light into two independent sequential steps. First, we determine *what* is reflected by a surface, and then process that reflected image to take into account *how* it is reflected. The first step deals purely with a visibility problem: we compute what is visible from every point on a surface – this lends itself very well to preprocessing. The

second step deals with the evaluation of the BRDF at the reflector's surface: we compute how the material properties blur and modulate the mirror-like reflected image; this is a view-dependent operation and, as such, needs to be computed every frame.

8.1. What Is Reflected

Several approaches have been used to render mirror-like reflections, but usually they are slow or inaccurate. We will review some of these approaches and introduce our image-based technique. We limit our discussion to forward mapping approaches for reflection—approaches that map objects from 3D space onto the reflective surface. We do not review ray tracing, an inverse mapping approach.

8.1.1. Related work in mirror-like reflections

Planar mirrors are the simplest specular surfaces to handle. Given the viewpoint and the scene, one renders from the mirrored viewpoint and direction, and blends this image onto the reflector [8][15]. Although this technique is simple and attractive for polygonal models, it is computationally intensive. Every mirror polygon requires re-rendering the complete scene each frame.

Ofek and Rappoport [16] presented a technique for generating reflections for curved surfaces based on creating virtual (reflected) objects with respect to the reflectors. For each reflector in the scene, potentially reflected objects in the scene are reflected by geometry-warping their vertices into the virtual counterparts. The virtual objects are then rendered and the resulting image is blended with the main image (of the unreflected scene), as it is done for multi-pass rendering. This approach both warps and renders all the potentially reflected objects every frame—operations that may require handling a large part of the geometry database.

8.1.2. Related work in image-based rendering

Recently, several image-based methods have emerged as possible tools for producing views of an environment in time that is independent of scene complexity. In general, image-based approaches can be described in terms of the plenoptic function—a description of all the radiance that is perceived from a given position and direction in space, over all wavelengths, over all possible times.

McMillan and Bishop [14] formalized the problem statement of image-based rendering in terms of sampling and reconstruction of the plenoptic function, and proposed a set of warping equations that describe the optical flow of plenoptic samples; however, their approach does not address view-dependent illumination.

Levoy *et al.* [12] and Gortler *et al.* [10] proposed similar approaches to image-based rendering in which a very dense sampling of the plenoptic function is performed a priori and then fast table lookup is performed at run-time. Whereas both approaches produce a good reconstruction of the plenoptic function for a wide range of views, they have very large memory requirements.

8.1.3 Radiance maps with depth

Our approach for computing mirror-like reflections is based on the notion that an image is a sampling of the plenoptic function at a position in space over a range of viewing angles. Like McMillan [14], we extend the plenoptic function sampling by preserving depth along each sampled direction. We call such an image a radiance map. Each pixel in a radiance map stores the outgoing

radiance and location of a point in 3D screen space – a point on the surface intersected by the corresponding viewing ray.

Since the amount of computation required to reproject an image is proportional to the image size, the time required to render a mirrored view of the scene is constant in the resolution of the radiance map regardless of the geometrical complexity of the scene. This is in contrast to geometry-based multi-pass rendering.

The use of image-based techniques to accelerate the rendering of architectural walkthroughs is not new. In particular, Aliaga *et al.* [1] use images with disparity to simplify geometry when viewed through a portal (door or window). The incoming radiance and corresponding disparity are sampled for a number of views along an arc in the horizontal plane behind the portal, as a preprocess. During runtime, image selection and McMillan-style image-warping are used to render final views through portals.

We extend Aliaga’s sampling to create viewpoints on the hemisphere behind each planar reflector, instead of on a horizontal arc [3]. For each reflective object, we precompute a collection of radiance maps with depth for points behind the reflector’s surface (Figure 4(left)). These radiance maps represent what is seen from the reflector’s surface over a range of angles and positions. As we deal with radiosity illuminated models, the radiance stored by the maps is a view-independent quantity. As a result, reprojection of image samples from one screen space to another can be used to determine the radiance arriving at a new viewing position.

We split our approach for creating and rendering radiance maps with depth into four distinct steps:

- sampling of the extended plenoptic function,
- selection,
- reprojection, and
- reconstruction.

Note that the first step is part of the preprocessing phase, while the last three are performed in the runtime phase. The result of this image-based pass is a perspective-correct reconstructed view of the environment as seen from a mirrored viewpoint.

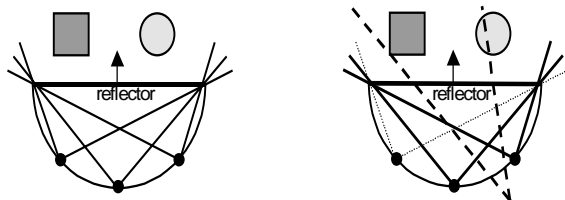


Figure 4 Left image shows radiance map sampling on the hemisphere surrounding the reflector. Three view frustums from behind the reflector are shown. Right image shows a new reflected view (dashed) and the two “best fit” sample cameras (solid) used for rendering.

Sampling of the plenoptic function

Our sampling process happens in three hierarchical stages, in order to finely sample the visibility function as seen from the reflector. First, we uniformly sample the planar reflector at a given planar resolution. Each of these points on the plane defines a reflector element on the glossy object. We then create a hemisphere centered at each of these points. The reflector element is a quadrilateral that inscribes the base of the hemisphere. Finally, we select points on the surface of the hemisphere at evenly sampled arc-lengths. For each point on the hemisphere we create a radiance map. The (possibly off-axis) view-frustums for radiance maps are created using one point on the hemisphere as the center-of-projection and the vertices of the reflector element

as the near plane. Figure 4(left) shows how we choose the location of the radiance maps for a single reflector element.

Selection

At runtime, we mirror the viewpoint and viewing direction about each reflection plane. The selection of radiance maps is performed as illustrated in Figure 4(right). By examining the viewing angle formed by the mirrored viewpoint and the center of the reflection plane, we quickly determine which cameras have the highest radiance resolution in the reflected view direction. This is a simple selection process implemented as a search for the minimal angle. We have also implemented a distance-based selection technique that evaluates the distance from the reference of the radiance maps to the reflected camera and selects the closest one(s).

Reprojection

We can project the pixels of the radiance maps to world space by using the inverse of the transformation matrix used to compute the radiance map. Then we can transform these points into a new camera space and render a new image of the sampled scene. McMillan [14] presented this process as an image warp that projects pixels from a reference image into a destination image. We prefer to treat this process as a reprojection of 3D points, in order to exploit hardware rendering of 3D points.

Reconstruction

Once the radiance maps have been projected to the new camera pose, an image needs to be reconstructed from the scattered points on the screen. Two situations can arise: the magnification or the minification of the picture elements (3D points).

For the case of magnification, we use a simple zero-order approach by just increasing the point size of the projected point on the screen; this is analogous to *splatting* [22] of a box kernel aligned with the screen. Although simple to implement, this approach may lead to artifacts due to noise introduced in the reflected images.

An estimate of the anisotropic projected point size, based on projected areas, is given by:

$$PointSize_{x,y} = \frac{1}{Step_{x,y}} = \frac{DistanceScale * Angular_{x,y}}{Window_{x,y}} \quad (3)$$

where $Window_{x,y}$ is a resolution scaling factor from the reference to the desired image in x or y ($Window_{x,y} = Ref_{x,y} / Des_{x,y}$), $DistanceScale$ is a distance scaling factor ($DistanceScale = Ref_{depth} / Des_{depth}$), and $Angular_{x,y}$ is an angular scaling factor based on the dot product of the desired view-direction and the reference X and Y axes.

The same expression above can be used for minification of the radiance maps – when the projected area of the radiance map is smaller than its original area. In this case, we use step size estimation ($Step_{x,y}$) to skip points when reprojecting the radiance map.

The estimation of point size on a per-point basis clearly generates the best results; however, it is a bottleneck. To keep the computation reasonable, we organize points of the radiance maps into a quadtree. Nodes of the quadtree contain only points in a given range of depths or have a minimum number of points. For each node, we compute a representative point as the average of all the points inside that node; this representative point is used to estimate the point size for all the points inside that node, as presented by equation (3). This hierarchy also permits view-frustum culling of sub-regions of the radiance maps.

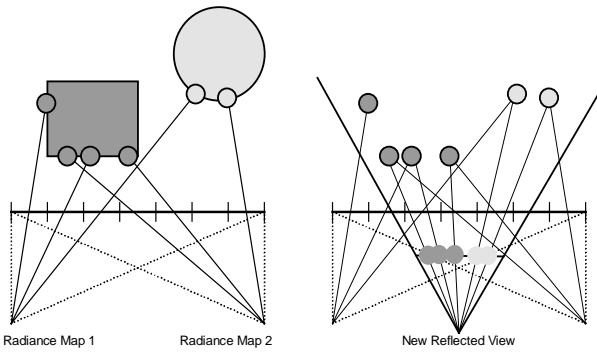


Figure 5 Left image shows radiance maps stored as images augmented with depth. Right image shows radiance maps reprojected into a new view.

8.2 How It Is Reflected

Once we know *what* is reflected for a particular reflector, we need to process this mirror-like reflected image to take into account *how* it is reflected, by evaluating the material’s BRDF. We take into account glossiness and view-dependent reflectance modulation by applying image processing techniques to the reflected image.

This decomposition of the BRDF into two components allows writing the BRDF as the product of an integration cone term and a directional amplitude function. Given the low frequency of both functions, we represent them using texture maps—a kernel texture and a reflectance modulation texture.

We can, for example, approximate the microfacets-based Cook-Torrance reflectance model [7]. We use the view-dependent term of equation (2) as Cook and Torrance’s specular component:

$$R_S = \{DG\} \left\{ \frac{F}{\pi(N \cdot L)(N \cdot V)} \right\} \quad (4)$$

where F is the Fresnel term, D is the slope distribution function, G is the geometrical attenuation term, and N , L , and V represent the normal, light, and view vector. D and G represent how light is spread locally on the surface due to microfacets on a neighborhood, and thus influences solid angle integration and our kernel texture. In terms of our technique, we sample DG to precompute the kernel texture to be used for the convolution step. F represents how light is reflected from each smooth microfacet, influencing reflectance modulation locally. We sample F divided by the denominator of equation (4) to precompute the sphere-maps for given materials.

8.2.1 Solid angle integration

The reflected images obtained with the approaches presented in section 8.1 represent a perfect mirror reflection—a single incoming direction contributes to a particular outgoing direction (any pixel in the reflected image). However, for glossy reflections, we need to consider that more than a single incoming direction over the hemisphere makes a significant contribution to the radiance in a particular outgoing direction. A solid angle over the hemisphere needs to be sampled to achieve a reasonable approximation of the outgoing radiance of equation (1). The size of this solid angle varies according to reflectance properties of the material – the blur of the reflected images – which defines an integration cone. The more diffusing the material, the larger the cone angle.

We implement glossiness by performing a 2D-convolution with a space-invariant kernel over the reflected image. The blur

produced by the convolution approximates the integration cone or incoming solid angle. This approximation assumes coherence exists across the reflected image and we disregard visibility events that may occur in the region surrounding a pixel. Each pixel of the reflected image, after the convolution, is the result of a weighted-average of the surrounding pixels. The weights for neighboring pixels are given by the kernel texture. The size of the kernel texture controls the support of the kernel (the size of the cone) and derives from the glossiness of the material. For a reflectance model such as Cook and Torrance’s, a Gaussian or a Beckman distribution is sampled in order to create the kernel.

Even though the convolution step is not aimed directly at the reconstruction stage of section 8.1, it helps obtain smoother reflected images.

8.2.2 Reflectance modulation

Besides the size of the solid angle to be sampled, the reflectance properties of a material also dictate the angular dependence of the amplitude of the reflected image. An ideal mirror and an ideal diffuse material each have constant amplitude for any reflected direction, whereas a surface such as paper or varnished wood exhibits angular variation of the amplitude; at grazing angles they exhibit a high specular component that is quite small at normal incidence/reflection.

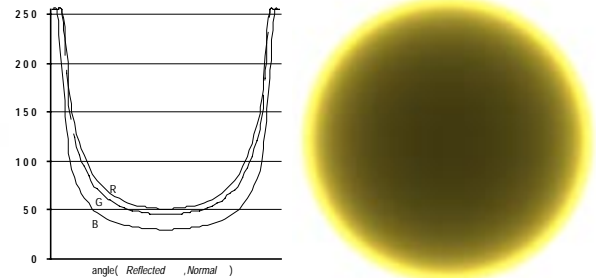


Figure 6 Reflectance sphere map for gold obtained by sampling the Fresnel equation for $RGB=(0.63, 0.56, 0.37)$. The graph shows RGB components along the middle line of the sphere map (along the angle between the reflected vector and the sphere normal). Note the color shift predicted by the Fresnel equation on both the texture and the graph.

We implement this view-dependent modulation of the reflected image with a rendering pass over the reflector. Once the reflected and blurred image is in the frame buffer, we enable multiplicative blending and render the reflector using its reflectance sphere map to modulate that image. The sphere-mapping technique evaluates the reflected vector at the vertices of the reflector and uses this information to derive the texture coordinates and index into the sphere reflectance map [15] — a texture lookup in hardware.

The sphere map itself is a square texture where texels outside an inscribed disk are never accessed and texels inside the disk contain useful values. Reflected vectors parallel to the surface normal map to the center of the disk and reflected vectors orthogonal to the surface normal map to the outer ring in the disk. We precompute our reflectance maps based on this mapping and on the material of the reflectors. Figure 6 shows a reflectance sphere map for a low-gloss material. Notice the low reflectance at orthogonal incidence/reflection (center of the disk) and the high reflectance at grazing angle (in the ring). The sphere maps are precomputed by sampling the material’s BRDF. For each pixel in the sphere-map texture, compute the reflected vector and evaluate the BRDF of the material for those conditions.

9. RESULTS

Our system is implemented in C++ using the OpenGL and GLUT libraries. The performance tests were run on a Silicon Graphics Onyx2 workstation using a single 250 MHz R10000 processor and an Infinite Reality 2 graphics pipe with four raster managers. We tested our system on two data sets: one consists of a simple textured 16,000-triangle model with up to 11 planar reflectors; the second is a large radiosity-illuminated and textured 140,000-triangle house model with five planar reflectors. (See color plates.)

The diagram in the color plates shows a sequence of images in which portions of our per-pixel shading equation have been evaluated for all the visible reflective surfaces. The kernel and reflectance modulation textures are also shown. The scene is composed of a pyramid and a teapot, both of which are inside of a textured cube. The pyramid has a specular component with copper material properties and the remainder of the scene is ideally diffuse. Three of the five non-diffuse reflectors are visible in the images. The upper left image combines the mirror-like reflected-images generated for the reflectors. This image shows the result of only computing the M component of our shading equation. Geometry-based reflections were used, in this case, to stress that our shading approach is independent of model representation (geometry versus images). The next image, labeled $M*K$, shows the result of the convolution operation used to perform solid angle integration of incoming radiance. The texture kernel K is the DG term of the Cook-Torrance reflectance model for copper. The image labeled $k_s(M*K)R$ shows the frame-buffer after the sphere-map reflectance modulation. R shows the directional reflectance modulation texture and corresponds to the fresnel term divided by the denominator of equation (4) for copper. The image labeled D shows the view-independent illumination. The bottom image, labeled $k_s(M*K)R + k_dD$, shows the final result of summing the view-dependent and view-independent components. Note the difference in modulation on the three faces of the pyramid, due to their distinct orientations with respect to the viewer. Also observe the tapering and color shift of the view-dependent component on the base of the pyramid.

To analyze the performance of our system, we used a house model. Images were rendered at 720x486 and performance data were collected by playing a pre-recorded path (944 frames) through the model. Table 1 contains a description of the radiance maps for each of the five reflectors in the scene.

	Name	# of hemispheres	Radiance maps per hemisphere	Radiance map resolution	Memory (Mbytes)
1	Piano Top	1	23	128x128	2.54
2	Living Floor	12	6	128x128	7.98
3	Door Mirrors	11	17	128x128	20.69
4	Music Floor	8	7	128x128	6.21
5	Bench Top	1	67	32x32	0.57

Table 1 Data for the five reflectors in the house model.

The graph (in the color plates) shows the performance for our image-based approach versus the geometry-based approach. It illustrates the cost of increasing the number of radiance maps reprojected per reflector. Notice that the image-based approach with one or two images was twice as fast as the geometry-based one. Note that increasing the number of selected radiance maps improves the quality of reflected images.

We have also analyzed the performance of the individual operations involved in our runtime rendering. We ran a series of tests on the house model using our image-based approach and selecting two radiance maps for each hemisphere. Allowing the number of reflectors to vary, we found that time spent rendering

the view-independent component of the scene was invariant with respect to the number of reflectors. We also found that the convolution and modulation operations combined for no more than 3% of the total rendering time. This suggests that the two operations do not dominate in the evaluation of our shading equation. It is important to note that the convolution operation is a screen space operation; consequently, the cost is dependent on the screen-space projected area of each reflector and not on model complexity.

In all these test runs with the house model, the image-based approach outperformed the geometry-based one, but this performance was not free. The image-based approach incurs additional memory overhead. The house model required 50 megabytes of storage of which 20% was for geometry-related data and 80% was for radiance maps.

Color plates (a) and (b) compare image quality of geometry-based reflections with our image-based approach. The image-based approach exhibits loss of detail and artifacts in some regions of the image. This is unfortunate for mirror-like reflections, but reasonable for glossy reflections. The primary causes for these artifacts are due to disocclusions and noise introduced by the point-based reconstruction.

Our BRDF decomposition for simulating glossy surfaces convincingly approximates the view dependent reflectance for glossy surfaces. Color plates (b) and (c) show the results on the music room of the house model. In (b) the view-direction is parallel to the floor and the piano top, while in (c) the viewer is looking down. Notice the increase in specular reflectance at grazing angles and the appropriately decreasing view-dependent component at higher angles.

Clearly, the blur obtained with convolution approximates the roughness of the surface at small neighborhoods. Unfortunately, this assumes that reflected points are at a constant distance from the reflector. A more accurate implementation would perform convolution in layers, so that points closer to the reflector would be less blurred than points farther away.

The use of hardware sphere-mapping to approximate the view-dependent reflectance function is very effective. Since we only use the sphere-map for directional information, we do not suffer from the occlusion and motion parallax problems inherent in the original technique, as described in section 3. However, our technique requires high resolution on a region (ring) of the sphere-map where the original sphere-mapping offers low resolution. Though we reduced this problem by uniformly increasing the resolution of the whole map and the tessellation of the reflectors, a better solution would be to flip the sphere-mapping indexing scheme in the hardware. Grazing angles would index texels in the center of the sphere-map, where there is higher resolution than in the ring. For a more clever hardware implementation of our technique, we could realize that sphere-mapping is under-utilized by our application. Isotropic BRDFs need just a one-dimensional texture-map indexed by the radius of the sphere-mapping indexing scheme.

10. CONCLUSIONS

In this paper we presented an approach to obtain glossy reflections that runs in constant-time (per reflector) independent of the scene complexity. In addition to mirror-like reflections, we efficiently account for the scattering effect of glossy reflections by decomposing the illumination into components that can be precomputed and quickly rendered using a multi-pass image-based approach. Our approach for glossy reflections is not tightly

coupled with the image-based reflection that we present; it can be used with any other model representation.

Our approach runs in constant time, like sphere-mapping, and preserves motion parallax, like geometric reflections. Unfortunately, mirror-like reflections in our approach require precomputing and storing a large quantity of data.

We have obtained constant time rendering per reflector, but the computation and storage increases as the number of reflective surfaces increase; however, if the constant time to perspective-project a radiance map is less than the time to render the geometry, we obtain greater performance. We have found that the sampling strategy and the selection approach are crucial in achieving improved reconstruction of the reflected images.

We effectively obtain view-dependent glossy effects, while incurring significantly less computational overhead than previous techniques, especially as the complexity of the scene increases.

11. FUTURE WORK

As in the geometry-based planar reflections, we can approximate curved surface reflections by using a piecewise-linear, polygonal tessellation of the surface; each facet is treated as a planar reflector. As the tessellation gets finer, we obtain more accurate results. A benefit of our approach is that we can tailor the reflector sampling density and resolution to be appropriate for the many small reflectors; furthermore, the scattering effect helps to mask the discontinuities in the curved reflection.

Recursive reflections can be handled exactly like in multi-pass geometry-based planar reflections [8] — by recursively calling the reflection routine for all reflectors visible from the reflected viewpoints. However, we have two major differences from normal geometric perfectly specular reflections: we do not require much recursion depth for non-diffuse reflectors, and we only require a constant time rendering of the reflected views. Previous approaches simulate these effects by redisplaying the entire scene geometry. In addition, the scattering effect is frequently simulated by multiple passes over the geometry in a jitter-and-accumulate strategy.

A more promising approach to the solid angle integration technique presented in section 8.2.1 would use splatting of the 3D points onto the surface of the reflector — a feed-forward reconstruction approach (for each point in the object, splat its contribution to all the pixels in the reflected image). This idea combines the reconstruction step of section 8.1.3 with the solid angle integration of section 8.2.1.

Our hybrid approach can also be extended to account for glossy transmission (translucency) and other effects, including depth-of-field and motion blur.

12. ACKNOWLEDGEMENTS

We would like to thank Fred Brooks, Mary Whitton, Wolfgang Stuerzlinger and the reviewers for their helpful comments. The Brooks' house model is the result of several years of work by the UNC-CH Walkthrough team.

This work was partially supported by NIH/National Center for Research Resources number RR02170, Silicon Graphics, CNPq/Brazil, PRAXIS XXI, NSF grant number MIP-9612643 and DARPA order E278. INTEL generously donated equipment used in this research.

REFERENCES

- [1] D. Aliaga and A. Lastra. 3D Image Warping in Architectural Walkthroughs. *IEEE Visualization '97*, pp. 355-362.
- [2] R. Bastos, M. Goslin, and H. Zhang. Efficient Rendering of Radiosity using Texture and Bicubic Interpolation. In *ACM Symposium on Interactive 3D Graphics*, Providence, RI, 1997.
- [3] R. Bastos, W. Stürzlinger. Forward Mapped Planar Mirror Reflections. University of North Carolina at Chapel Hill, Technical Report TR98-026, April 1998.
- [4] J. Blinn and M. Newell. Textures and Reflection in Computer Generated Images. *Communication of the ACM*. Vol. 19, no. 10 (1976), pp. 542—547.
- [5] F. Brooks. Walkthrough: A dynamic graphics system for simulating virtual buildings. In *ACM Symposium on Interactive 3D Graphics*, Chapel Hill, NC, 1986.
- [6] M. Cohen and J. Wallace. Radiosity and Realistic Image Synthesis. Academic Press, Boston, 1993.
- [7] R. Cook and K. Torrance. A Reflectance Model for Computer Graphics. *SIGGRAPH 81*, pp.244—253.
- [8] P. Diefenbach. *Pipeline Rendering: Interaction and Realism Through Hardware-Based Multi-Pass Rendering*. University of Pennsylvania, Department of Computer Science, Ph.D. dissertation, 1996.
- [9] M. Garland and P. Heckbert. Surface Simplification using Quadratic Error Bounds. *SIGGRAPH 97*, pp. 209—216.
- [10] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen. The Lumigraph. *Proceedings of SIGGRAPH, 1996*, pp. 43—54.
- [11] N. Greene. Environment mapping and other applications of world projections. In *IEEE CG&A 6*(11), Nov 1986, pp. 21—29.
- [12] M. Levoy and P. Hanrahan. Light Field Rendering. *Proceedings of SIGGRAPH, 1996*, pp. 31--42.
- [13] D. Lischinski and A. Rappoport. Image-Based Rendering for Non-Diffuse Synthetic Scenes. In *Rendering Techniques '98, Proceedings of the Eurographics Workshop on Rendering*, Eurographics, 1998.
- [14] L. McMillan and G. Bishop. Plenoptic Modeling: An Image-Based Rendering System. *SIGGRAPH 95*, pp. 39--46.
- [15] T. McReynolds, D. Blythe, B. Grantham, S. Nelson. Advanced Graphics Programming Techniques Using OpenGL. SIGGRAPH 98 Conference Course Notes #17.
- [16] E. Ofek and A. Rappoport. Interactive Reflections on Curved Objects. *Proceedings of ACM Siggraph*, 1998.
- [17] B. T. Phong. Illumination for computer generated images. In *Communication of the ACM*. 18(6): 311—317, June 1975.
- [18] F. Sillion and C. Puech. *Radiosity and Global Illumination*. Morgan Kaufmann. San Francisco, 1994.
- [19] W. Stürzlinger and R. Bastos. Interactive Rendering of Globally Illuminated Scenes. In *Rendering Techniques '97, Proceedings of the Eurographics Workshop on Rendering*, Eurographics, 1997.
- [20] D. Voorhies and J. Foran. Reflection Vector Shading Hardware. *SIGGRAPH 94*, pp. 163—166.
- [21] B. Walter *et al.* Fitting Virtual Lights For Non-Diffuse Walkthroughs. *SIGGRAPH 97*, pp. 45-48.
- [22] L. Westover. Splatting: A Feed-Forward Volume Rendering Algorithm, Ph.D. Dissertation, University of North Carolina, 1991.

Images of the music room computed interactively by evaluating shading equation (2) at 720x486 pixels. Images (a) and (b) compare geometry- and image-based glossy reflections. Images (b) and (c) demonstrate the view-dependent behavior of the glossy floor and piano top. Observe the high reflectance at grazing angles, image (b), and the low reflectance at high angles, image (c).



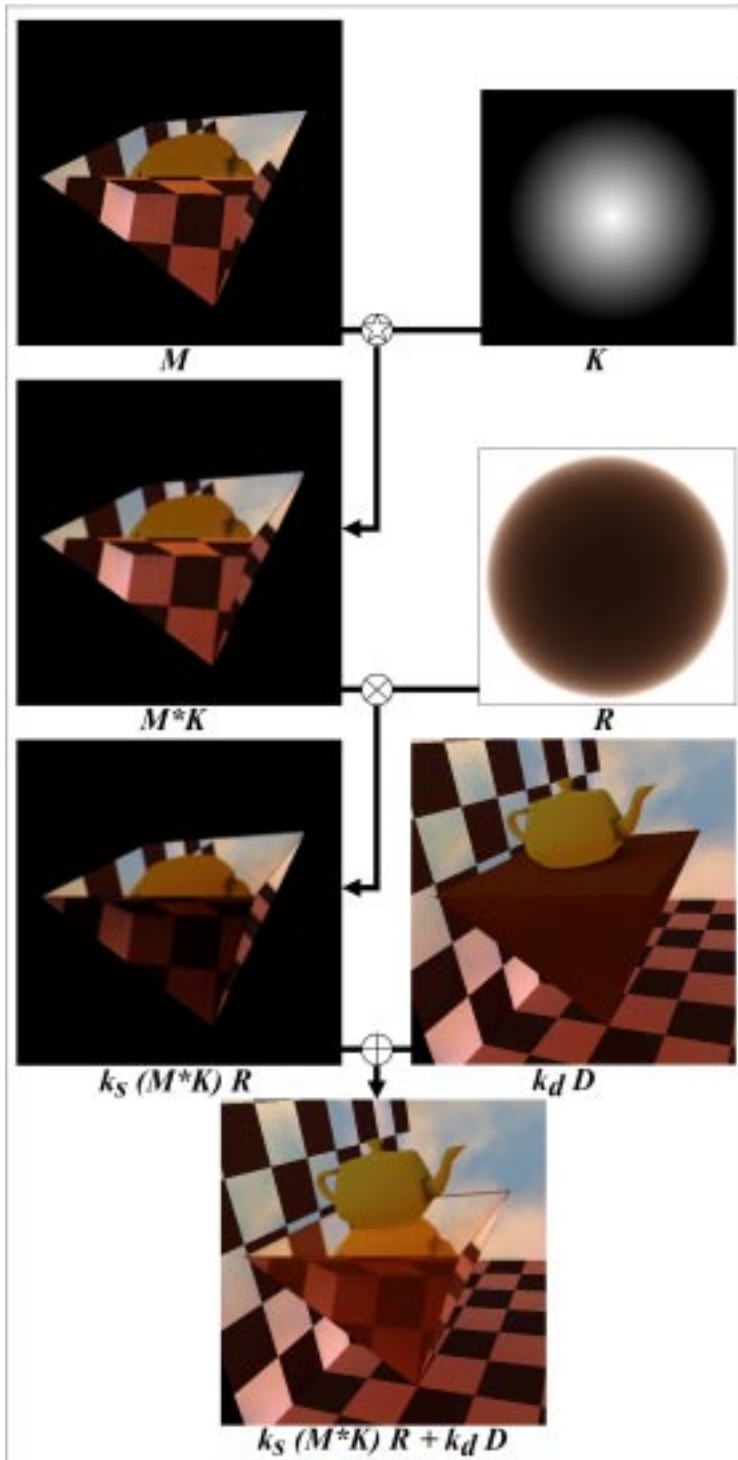
(a)



(b)



(c)



Runtime per-pixel evaluation of the shading equation.

