Precomputed Shadow Fields for Dynamic Scenes

Kun Zhou Yaohua Hu Stephen Lin Baining Guo Heung-Yeung Shum

Microsoft Research Asia

Abstract

We present a soft shadow technique for dynamic scenes with moving objects under the combined illumination of moving local light sources and dynamic environment maps. The main idea of our technique is to precompute for each scene entity a *shadow field* that describes the shadowing effects of the entity at points around it. The shadow field for a light source, called a *source radiance field* (SRF), records radiance from an illuminant as cube maps at sampled points in its surrounding space. For an occluder, an *object occlusion field* (OOF) conversely represents in a similar manner the occlusion of radiance by an object. A fundamental difference between shadow fields and previous shadow computation concepts is that shadow fields can be precomputed independent of scene configuration. This is critical for dynamic scenes because, at any given instant, the shadow information at any receiver point can be rapidly computed as a simple combination of SRFs and OOFs according to the current scene configuration. Applications that particularly benefit from this technique include large dynamic scenes in which many instances of an entity can share a single shadow field. Our technique enables low-frequency shadowing effects in dynamic scenes in real-time and all-frequency shadows at interactive rates.

Keywords: soft shadow, area lighting, environment lighting, video texture lighting, precomputed source radiance, precomputed visibility.

1 Introduction

Soft shadows that arise from area light sources add striking realism to a computer generated scene. Computing this lighting effect for interactive applications, however, is difficult because of the complex dependence of soft shadows on light sources and scene geometry. To make the problem tractable, earlier techniques mostly deal with small or point-like area sources. Recently, researchers have developed techniques such as precomputed radiance transfer (PRT) (e.g., [Sloan et al. 2002; Ng et al. 2003]) for the special case of distant illumination represented by environment maps. The problem of efficient shadow computation under the combined illumination of general local light sources and environment maps remains unsolved.

Computing soft shadows is even more challenging for dynamic scenes such as shown in Fig. 1, where objects, local light sources, and the environment map are all in motion. With traditional methods based on shadow maps [Williams 1978] and shadow volumes [Crow 1977], computational costs increase rapidly as the numbers of objects and light sources increase. The high run-time expense cannot be reduced by precomputation, since shadow maps and volumes are determined with respect to a specific arrangement of lights and occluders. With PRT, dynamic scenes present an even greater problem because the pre-computed transfer matrices are only valid for a fixed scene configuration, and re-computing transfer matrices for moving objects on-the-fly is prohibitively expensive.

Figure 1: *Soft shadow rendering in a dynamic scene containing moving objects and illumination from both moving local light sources and a dynamic environment map.*

In this paper we present an efficient technique for computing soft shadows in dynamic scenes. Our technique is based on a new shadow representation called a *shadow field* that describes the shadowing effects of an individual scene entity at sampled points in its surrounding space. For a local light source, its shadow field is referred to as a *source radiance field* (SRF) and consists of cube maps that record incoming light from the illuminant at surrounding sample points in an empty space. An environment map can be represented as a single spatially-independent SRF cubemap. For an object, an *object occlusion field* (OOF) conversely records the occlusion of radiance by the object as viewed from sample points around it. With shadow fields precomputed, the shadowing effect at any receiver point can be computed at run-time as a simple combination of SRFs and OOFs according to the scene configuration.

A fundamental difference between shadow fields and previous shadow computation concepts is that a shadow field represents the shadowing effects of a single scene element in an empty space, and thus can be precomputed independent of scene configuration. In contrast, shadow maps, shadow volumes, and PRT transfer matrices can only be computed after fixing the scene geometry and also the illuminants in the case of shadow maps and volumes. This property of shadow fields is critical for enabling soft shadow computation in dynamic scenes. Another important property of shadow fields is that they can be quickly combined at run-time to generate soft shadows. Specifically, for any configuration of light sources and objects in a dynamic scene, the incident radiance distribution at a given point can be determined by combining their corresponding shadow fields using simple coordinate transformations, multiplications, and additions. Finally, with shadow fields both local light sources and environment maps are represented as SRF cube maps, which makes it straightforward to handle the combined illumination of local light sources and environment maps.

Shadow fields are fairly compact as they can be compressed considerably with low error by representing cube maps in terms of spherical harmonics for applications in which low-frequency shadows are

sufficient, and using wavelets for all-frequency applications. Furthermore, soft shadows can be directly rendered from compressed SRFs and OOFs. For low-frequency shadowing, dynamic scenes containing numerous objects can be rendered in real time. For all-frequency shadows, real-time rendering can be achieved for relighting of static scenes. Dynamic all-frequency shadowing with moving, non-rotating light sources and objects can be rendered at interactive rates.

The use of shadow fields especially benefits applications such as games, which often contain many instances of a given light source or object in a scene, such as street lamps and vehicles on a road. A single shadow field can represent multiple instances of a given entity, which promotes efficient storage and processing. This shadow field technique is also particularly advantageous for user interaction with a scene, such as in lighting design which requires immediate visualization of local light source rearrangements.

The remainder of this paper is organized as follows. The following section reviews related work and highlights the contributions of shadow fields. Section 3 describes the representation and precomputation of shadow fields, and also the method for combining them to determine the incident radiance distribution at a scene point. In Section 4, we present the soft shadow rendering framework, including the basic algorithm and acceleration techniques. Results and applications are shown in Section 5, and the paper concludes with some discussion of future work in Section 6.

2 Related Work

Soft Shadow Generation: Soft shadow techniques are generally based on combining multiple shadow maps (e.g., [Heckbert and Herf 1997; Agrawala et al. 2000]) or extending shadow volumes (e.g., [Assarsson and Akenine-Moller 2003]). There also exist related methods that gain speed by computing "fake", geometrically incorrect soft shadows (e.g., [Chan and Durand 2003; Wyman and Hansen 2003]). These techniques have become increasingly efficient in recent years, but they share the common restriction that the illuminants must be point-like or small area sources. Environment maps cannot be efficiently handled.

For a dynamic scene, shadow map and volume computation escalates significantly with greater scene complexity and needs to be repeated for each frame. This rendering load cannot be alleviated by precomputation, since shadow maps and volumes are determined with respect to a specific arrangement of lights and occluders, for which there exists numerous possible configurations. In our approach, we decouple lighting and visibility by modeling the effects of illuminants and occluders individually, which allows precomputation that is independent of arrangement.

A radiosity-based approach [Drettakis and Sillion 1997] has been presented for rendering soft shadows and global illumination by identifying changes in hierarchical cluster links as an object moves. With this technique, the corresponding changes in only coarse shadows have been demonstrated at interactive rates, and it is unclear how to handle multiple moving objects that may affect the appearance of one another.

PRT for Static Scenes: PRT provides a means to efficiently render global illumination effects such as soft shadows and interreflections from an object onto itself. Most PRT algorithms facilitate evaluation of the shading integral by computing a double product between BRDF and transferred lighting that incorporates visibility and global illumination [Sloan et al. 2002; Kautz et al. 2002; Lehtinen and Kautz 2003] or between direct environment lighting and a transport function that combines visibility and BRDF [Ng et al. 2003]. Instead of employing double product integral approximations, Ng et al. [2004] propose an efficient triple product wavelet algorithm in which lighting, visibility, and reflectance properties are separately represented, allowing high resolution lighting effects with view variations. In our proposed method, we utilize this triple product formulation to combine BRDFs with multiple instances of lighting and occlusions.

PRT methods focus on transferring distant illumination from environment maps to an object. To handle dynamic local illumination, they would require lighting to be densely sampled over the object at run time, which would be extremely expensive. In a technique for reducing surface point sampling of illumination, Annen et al. [2004] approximate the incident lighting over an object by interpolating far fewer samples using gradients of spherical harmonic coefficients. Complex local illumination with moving occluders can often induce frequent, time-varying fluctuations in these gradients, which may necessitate prohibitively dense sampling for all-frequency shadow rendering in dynamic scenes.

PRT for Dynamic Scenes: The idea of sampling occlusion information around an object was first presented by Ouhyoung et al. [1996], and later for scenes with moving objects, Mei et al. [2004] efficiently rendered shadows using precomputed visibility information for each object with respect to uniformly sampled directions. By assuming parallel lighting from distant illuminants, some degree of shadow map precomputation becomes manageable for environment maps. For dynamic local light sources, however, precomputation of shadow maps nevertheless remains infeasible.

Sloan et al. [2002] present a neighborhood-transfer technique that records the soft shadows and reflections cast from an object onto surrounding points in the environment. However, it is unclear how neighborhood-transfers from multiple objects can be combined. In our technique, the overall occlusion of a point by multiple objects is well defined by simple operations on the respective OOFs.

PRT methods for deformable objects have suggested partial solutions for dynamic lighting. James and Fatahaltian [2003] compute and interpolate transfer vectors for several key frames of given animations, and render the pre-animated models under environment maps in real time. This method, however, does not generalize well to dynamic scenes that contain moving local light sources and numerous degrees of freedom. Kautz et al. [2004] propose a hemispherical rasterizer that computes the self-visibility for each vertex on the fly; however, this approach would incur heavy rendering costs for a complex scene containing many objects.

Light Field Models of Illuminants: To facilitate rendering with complex light sources, such as flashlights and shaded lamps, light field representations of these illuminants have been obtained from global illumination simulations [Heidrich et al. 1998] or by physical acquisition [Goesele et al. 2003]. The acquired illuminants are then used with photon tracing for off-line rendering. Our work is complementary with this line of work since their light field models can be resampled into SRFs for interactive rendering of shadows.

3 Shadow Fields

To deal with scenes with dynamic illumination conditions produced by light sources and objects in motion, we employ shadow fields to represent their illumination and occlusion effects. In this section, we detail the precomputation of source radiance fields and object occlusion fields, describe how these fields are combined to obtain the incident radiance distribution of soft shadows for arbitrary scene configurations, and then discuss the sampling and compression schemes used for efficient storage and rendering of shadow fields.

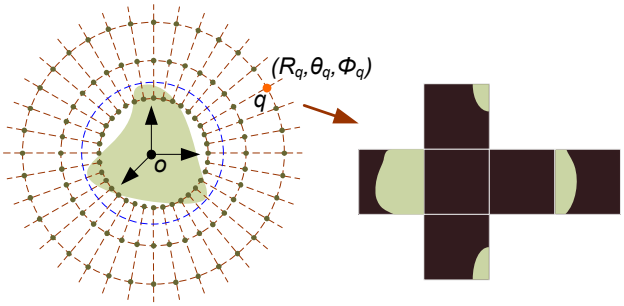

Figure 2: *Shadow field sampling. Left: Radial and angular sampling (R,θ,ϕ) of cubemap positions on concentric shells. The bounding sphere of the scene entity is shown in blue. Right: The sampled cube map (θ_{in},ϕ_{in}) of point q.*

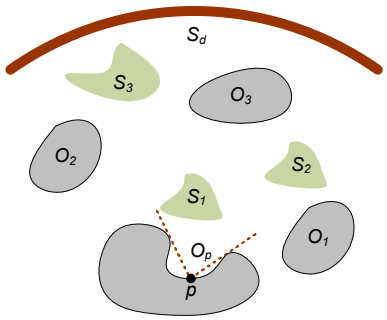

Figure 3: *Combining shadow fields to obtain the incident radiance distribution. Point p on an object has self-visibility O_p. There are three objects in the scene with OOFs O_1, O_2, O_3, and the scene illumination consists of distant lighting S_d and three local light sources S_1, S_2, and S_3. In a dynamic scene, these entities can move and change the scene structure from frame to frame.*

3.1 SRF and OOF Precomputation

The effect of a light source or occluding object can be expressed as a field in the space that surrounds it. For a local light source L_i, its radiated illumination at an arbitrary point (x,y,z) in the scene can be expressed as a 6D plenoptic function $P(x,y,z,\theta_{in},\phi_{in},t)$ [McMillan and Bishop 1995], where (θ_{in},ϕ_{in}) is the incident illumination direction and t denotes time. Reparameterizing the position (x,y,z) as polar coordinates (R_p,θ_p,ϕ_p) with respect to the local frame of the light source and removing the time dependency, a source radiance field can be represented as the 5D function $S(R_p,\theta_p,\phi_p,\theta_{in},\phi_{in})$, or equivalently as cube maps (θ_{in},ϕ_{in}) as a function of position (R_p,θ_p,ϕ_p). Occlusion fields can be expressed similarly, but representing alpha values instead of radiance intensities.

The SRF of each light source and the OOF of each local object is individually precomputed at sampled locations in its surrounding space as illustrated in Fig. 2. While optimal sampling densities could potentially be established from a detailed analysis, we instead allow the user to empirically determine for each shadow field the radial and angular sampling densities that result in an acceptable level of error.

Distant illumination represented in an environment map is handled as a special light source with a shadow field S_d that is independent of spatial location and may undergo changes such as rotations. Distant lighting is noted as being farther away from any scene point than any local entity. The self-occlusion of an object at a surface point p can similarly be expressed as a special object occlusion field O_p that is precomputed at sampled points on an object's surface and is considered the closest entity to any point on the object.

3.2 Incident Radiance Computation

With movement of light sources and objects, lighting and visibility can change rapidly within a scene and vary significantly among different scene points. For efficient rendering of soft shadow at a given point, the SRFs and OOFs of illuminants and objects must be quickly combined according to the scene configuration to solve for the incident radiance distribution.

To determine the contribution of a light source to the incident radiance distribution of a soft shadow point, the visibility of the light source from the point needs to be computed with respect to the occluding objects in the scene. While this conventionally requires tracing rays between the illuminant and the point, it can be efficiently calculated by some simple operations on precomputed shadow fields. First, the fields induced by the illuminant and scene objects are aligned to their scene positions and orientations using coordinate transformations. The distances of these scene entities from the shadow point are then sorted from near to far. To avoid ambiguity in this distance sort, we do not allow the bounding spheres of objects to enter the bounding spheres of light sources. For objects that lie closer to the point than the light source, their OOFs are multiplied to solve for their aggregate occlusion, and this product is multiplied by the SRF of the light source to yield the source radiance that arrives at the shadow point. Adding the contributions of each light source in the scene gives the final incoming radiance distribution that determines the soft shadow at the point.

For example, the incident radiance distribution at p in Fig. 3 is computed as follows. For each scene entity, its shadow field is translated and rotated according to its position and orientation in the scene. The only potential occluders of a given light source are objects that lie closer to p, determined by a distance sorting of the shadow fields. For S_1, only the object that p itself lies on can occlude it. The incident lighting from S_1 can be expressed as $S_1(p)*O_p$, where $*$ denotes a product of corresponding cubemap entries. For S_2, an additional object O_1 may occlude it. Therefore, its lighting contribution will be $S_2(p)*O_1(p)*O_p$. Since the other local light source S_3 and the environment illumination S_d are both farther from p than all the occluders in the scene, the radiance at p that arrives from S_3 and S_d are respectively $S_3(p)*O_3(p)*O_2(p)*O_1(p)*O_p$ and $S_d*O_3(p)*O_2(p)*O_1(p)*O_p$. Summing the radiance from each light source results in the incident radiance distribution at p. Instead of directly adding together the radiances from all light sources, the sum can be computed more efficiently as $(((S_d+S_3(p))*O_3(p)*O_2(p)+S_2(p))*O_1(p)+S_1(p))*O_p$ by extracting the common occluders of different lights.

Many scenes contain multiple instances of a given object (e.g., O_1 and O_2 in Fig. 3) or light source (e.g., S_1 and S_2 in Fig. 3), and a benefit of this scheme is that only a single OOF or SRF needs to be maintained for all of its instances.

3.3 Data Sampling and Compression

Shadow fields need not be sampled throughout the entire space, since shadowing effects decay with increasing distance. Sampling only within the space near the object or light source is sufficient. The radius of the largest sphere may be set according to the size of the local scene, or at a distance beyond which the entity has a negligible effect. In our implementation, we simply use for low-frequency shadows 16 concentric spheres uniformly distributed between $0.2r$ and $8r$, where r is the radius of the bounding sphere. For all-frequency shadows, denser sampling is needed, and we have found that 32 spheres ranging from $0.2r$ to $20r$ give satisfactory results. From points beyond the largest radial distance, the shadowing effects of an entity are disregarded.

Each sphere is sampled angularly at projected points from a cubemap bounded by the sphere. For low frequency effects, the sam-

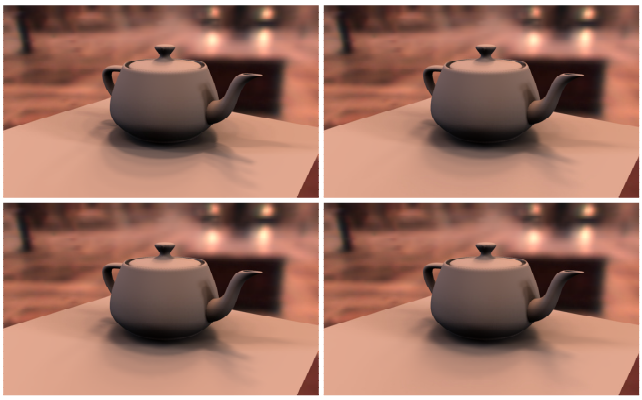

Figure 4: *OOF wavelet compression, under environment lighting from St. Peter's Basilica. Top left: reference image. Top right: OOF rendering without compression, which uses 268 wavelet coefficients on average. Bottom left: with 100 wavelet coefficients. Bottom right: with 50 wavelet coefficients.*

pling need not be dense, so we use 16 concentric spheres each angularly sampled at $6 \times 16 \times 16$ directions of a cube map for all the low-frequency examples in this paper. Since all-resolution shadow fields require a high sampling density for accurate representation, we use 32 spheres of $6 \times 32 \times 32$ samples.

At each sample point, a cube map representing the incoming light or occlusion is computed and recorded. Shadow fields at intermediate points are approximated by trilinear interpolation of the eight nearest samples. Although our implementation uses a fixed sampling density, the user may select different sampling densities for different entities in a scene. For a simple object such as a sphere, sparse sampling yields sufficient accuracy at a low storage cost, while an intricate object like a plant would require denser sampling for high fidelity.

Although the raw data for a shadow field can consume a substantial amount of space, it can be compressed considerably with low error by representing cube maps in terms of spherical harmonics for applications in which low-frequency shadows are sufficient, and using wavelets for all-frequency applications. As noted in [Sloan et al. 2002], fourth or fifth order spherical harmonics provide good shadowing results on typical meshes in smooth lighting environments. In our low-frequency implementation, we use fourth-order spherical harmonics expressed by 16 basis function coefficients, each quantized to 8 bits, to approximate shadow field cube maps. For low frequency effects represented with a $6 \times 16 \times 16$ cube map at each sampled point in the surrounding space, the data size for each OOF is 384 KB, while a 3-channel SRF occupies about 1 MB.

For all-frequency applications, we utilize $6 \times 32 \times 32$ cube maps, which have 6144 wavelet basis functions. The SRF and OOF cube maps are very sparse, with most nonzero entries aggregated in concentrated regions for typical objects and light sources. For example, the OOF of the teapot model shown in Fig. 4 has an average of 268 nonzero wavelet coefficients per sample point. With this sparsity and the area-weighted compression scheme for cube maps [Ng et al. 2003], OOFs can be adequately compressed with low error to 1%-4% of their raw data size for all examples in this paper. Differences in rendering quality at various compression rates are exhibited in Fig. 4, where the reference image is rendered using the pre-computed visibility of each vertex on the planar surface. The difference between the reference image and the uncompressed OOF rendering image demonstrates little visible error caused by OOF sampling. Since light sources tend to have simpler shapes than objects, SRFs can in general be more highly compressed, to 0.1%-1% of their original data size.

```
Rotate distant lighting S_d to align with global coordinate frame
For each scene entity that is an object do
    For each vertex p do
        B_p = 0
        T_p = TripleProduct(O_p, ρ̃)
        rotate T_p to align with global coordinate frame
        compute distance from p to each scene entity
        sort entities in order of increasing distance
        For each entity J do
            If J is a light source
                query SRF array to get its SRF S_J(p)
                rotate S_J(p) to align with global coordinate frame
                B_p += DoubleProduct(S_J(p), T_p)
            Else
                query OOF array to get its OOF O_J(p)
                rotate O_J(p) to align with global coordinate frame
                T_p = TripleProduct(O_J(p), T_p)
        B_p += DoubleProduct(S_d, T_p)
```

Figure 5: *Outline of basic algorithm for shadow field rendering.*

4 Soft Shadow Rendering

With the proposed technique for precomputing and combining shadow fields to determine the incident radiance distribution at arbitrary scene points, generation of soft shadows simply involves factoring in the BRDF. The soft shadow at a scene point can be computed according to the rendering equation, expressed as

$$\begin{aligned} B(\mathbf{x}, \omega_o) &= \int_\Omega L(\mathbf{x}, \omega_i) V(\mathbf{x}, \omega_i) \rho(\mathbf{x}, \omega_i, \omega_o) (\omega_i \cdot \mathbf{n})\, d\omega \\ &= \int_\Omega L(\mathbf{x}, \omega_i) V(\mathbf{x}, \omega_i) \tilde{\rho}(\mathbf{x}, \omega_i, \omega_o)\, d\omega, \end{aligned}$$

where B is the exitant radiance as a function of position $\mathbf{x}$ and outgoing direction ω_o, L is the lighting vector with incident direction ω_i, V is the visibility function, ρ is the BRDF, $\mathbf{n}$ denotes the surface normal, and $\tilde{\rho}$ is the product of the BRDF and a cosine term $\omega_i \cdot \mathbf{n}$. To evaluate the rendering equation using shadow fields, we present a basic algorithm and some acceleration techniques.

4.1 Basic Algorithm

In our rendering system, with each object is associated its OOF, mesh data, BRDF data, and self-visibility function for each vertex with respect to its local frame. For each light source, its SRF and rendering information are stored.

In computing the outgoing radiance distribution B_p from a point p, the effects of surrounding scene entities are processed from near to far, as outlined in Fig. 5. For greater computational efficiency, we factor in the BRDF at the beginning of the algorithm, rather than determining the incident radiance distribution and then incorporating the BRDF. The product of the self-visibility and the BRDF at p is first computed and stored in T_p, which represents the product of OOFs and the BRDF. T_p is iteratively updated by multiplying OOFs of increasing distance from p. Since each light source can be occluded only by objects located nearer than it to p, its contribution to outgoing radiance is determined by multiplying its SRF with the T_p that incorporates the appropriate set of OOFs. The contributions of each light source are finally aggregated to obtain the soft shadow.

The double product of two functions expressed in an orthonormal basis is simply the dot product of their coefficient vectors. Computing this operation has been shown to be very fast for both linear spherical harmonics [Sloan et al. 2002] and nonlinear wavelets [Ng et al. 2003]. We compute triple products using the efficient algorithm in [Ng et al. 2004], which calculates the projection G_i on an orthonormal basis Ψ_i of a product $G = E \cdot F$ as

$$G_i = \sum_j \sum_k C_{ijk} E_j F_k,$$

where $C_{ijk} = \int_{S^2} \Psi_i(\omega)\Psi_j(\omega)\Psi_k(\omega)d\omega$ are *tripling coefficients*.

In determining the soft shadows of a scene point, the computation required for each scene entity consists of a table query with interpolation, a rotation, and a double product for light sources or a triple product for objects. Since table queries with interpolation and double products are fast, computational costs principally arise from rotations and triple products. For a spherical harmonic basis, rotations can be efficiently computed using the rotation function in DirectX. For a wavelet basis, rotations are computationally expensive, so we consequently do not support rotations of local entities in all frequency applications. Rotations of a wavelet environment map, however, are efficiently supported since only a single rotation of its cube map is needed to update the environment lighting in the entire scene. To efficiently handle wavelet rotations for a local entity, wavelet projections could be precomputed at sampled rotations of a shadow field, but at a large expense in storage.

The complexity of triple products for N spherical harmonic basis functions in complex form is $O(N^{5/2})$ as noted in [Ng et al. 2004]. Since we use only low-order spherical harmonic projections in real form [Sloan et al. 2002], the number of nonzero tripling coefficients is small (25 for third order and 77 for fourth order). For a wavelet basis, we use the sublinear-time algorithm presented in [Ng et al. 2004].

4.2 Acceleration

To accelerate rendering, we employ a two-stage culling technique and utilize a lazy updating of occlusions.

Culling: Visibility culling is performed on two levels. With respect to the viewer, we cull vertices that do not contribute to screen pixels. With respect to each remaining vertex p, we cull entities that do not contribute to its radiance. Such entities exhibit at least one of the following three conditions. First, the bounding sphere of the entity is under the local horizon plane of the vertex. Second, the bounding sphere of the entity is occluded by the inscribed spheres of closer entities. Third, the entity is far enough away from p to be disregarded. This distance is set according to $\| \mathbf{p} - \mathbf{o} \| > \varepsilon \cdot r$, where $\mathbf{o}$ is the origin of the entity's coordinate frame, r is the radius of its bounding sphere, and $\varepsilon \cdot r$ is the maximum sampling radius of the shadow field. In our implementation, ε is set to 8 for low-frequency applications and 20 for all-frequency applications.

Lazy Occlusion Updating: If only light sources are moving in the scene, we can cache the visibility at each vertex with respect to each light source. In this way, as long as the occluders for a specific light source remain unchanged, the cached visibility can be reused; otherwise, the visibility is recomputed and cached. This acceleration scheme is particularly useful for interactive lighting design applications, and can typically reduce rendering costs by a factor of 20.

5 Applications and Results

We have implemented the shadow field algorithm on a dual Intel Xeon 3.0 GHz workstation. The rendering performance and total memory costs for several scenes are listed in Table 1. Since self-visibility is stored as per-vertex data, its memory cost depends on the number of vertices in the model. For example, in Fig. 8, the all-frequency self-shadow OOF of the Robot model occupies about 20 MB. For all-frequency shadow fields, precomputation takes less than two hours with a $32 \times 6 \times 32 \times 32$ sampling density and a $6 \times 32 \times 32$ cube map resolution.

For low-frequency shadowing, we present two examples. Fig. 6 displays a chess match with moving objects in the first row, and illumination from moving local light sources in the second row. With colored illuminance in different RGB channels, the visual effects of each local light source are clearly revealed. Although the scene contains 32 chess pieces, these pieces are instances of only seven different OOFs. Also, two of the local light sources are also instances that share a single SRF.

Table 1: ***Shadow Field Rendering Performance***

Scene	Frequency	Vertices	FPS	Memory
Chess	low	30K	15-18	50 MB
Robots	low	26K	12-30	15 MB
Spaceships	high	73K	0.1	280 MB
Relighting	high	70K	7-16	500 MB

In the second low-frequency example, we apply the shadow field technique to three articulated robots, which are walking and fighting on a deck in Fig. 7. One OOF is used to represent each of the robot legs, and one OOF represents the robot bodies. Although these three robots are of different sizes, an OOF can simply be rescaled according to the size of the object. Each robot carries a spot light which moves with the robot, and the bullet fired by the robot is also a local light source.

All-frequency shadows cast by two moving spaceships onto the robots are shown in Fig. 8. The memory usage of the wavelet OOFs is about 50 MB. These shadows are generated at only interactive rates because triple products for wavelet coefficients are relatively time consuming.

For all-frequency relighting applications where only light sources move, as shown in the first row of Fig. 9, a much higher rendering speed can be attained by employing the lazy occlusion updating scheme described in Section 4.2. Unlike previous shadowing techniques, shadow fields can easily handle illumination from both an environment map and local light sources, as displayed in top-left image of Fig. 9.

The explosion in the second row of Fig. 9 demonstrates illumination from a video texture. This effect is handled by expanding the SRF representation to support an additional dimension of time. In this example, the 6D SRF contains a sequence of standard 5D SRFs, one for each of six key frames in the video texture, and takes about 400 MB of memory. Each frame in the rendered video is associated with an SRF in the sequence. The use of video-based illumination brings rich visual effects to the rendering result.

6 Discussion and Conclusion

In this paper, we presented a comprehensive field-based framework for real-time soft shadow computation in dynamic scenes. Since shadow fields represent the shadowing effects of individual entities, they can be precomputed without regard to scene configuration. For a given configuration at run time, incident radiance distributions of soft shadows can be efficiently determined by simple combinations of shadow fields. An additional benefit of this representation is that multiple instances of a given entity can share a single shadow field, leading to more efficient use of computational resources.

From this shadow field technique, there exist a number of interesting directions for further investigation. First, we plan to explore different sampling schemes for shadow fields and are seeking a theoretical foundation to decide the sampling density. Currently, we decide the sampling density by experiments. Second, our current implementation does not incorporate level-of-detail (LOD) acceleration which is widely used in large scene rendering. LOD processing, for example, could be employed to reduce vertex relighting computation. For relighting objects that are far away from the viewpoint, instead of computing the radiance from each vertex, we

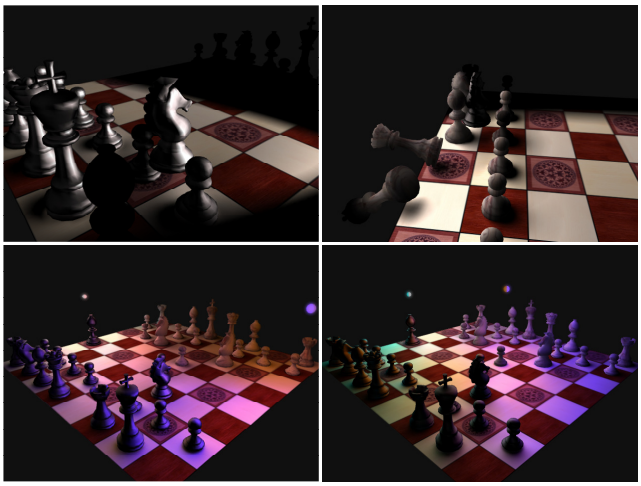

Figure 6: *A chess match. The first row shows moving objects; the second row shows moving local light sources.*

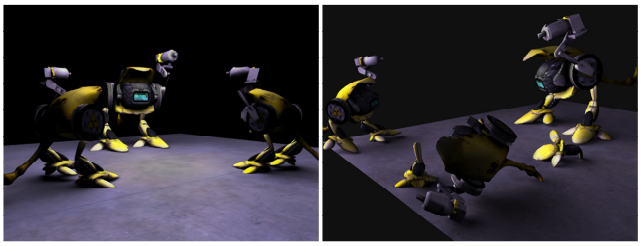

Figure 7: *Articulated robots. Left: walking; Right: fighting.*

could compute radiance from only a sparse set of vertices and interpolate their colors for other vertices. Third, we currently handle only direct lighting, and we are interested in examining methods for incorporating radiance from interreflections into our framework. Another area of future work is to capture shadow fields of actual light sources and objects, which could allow shadowing effects to be generated from light field models of these entities.

Acknowledgements

The authors would like to thank Wei Zhou for designing and animating the 3D models and for video production. Many thanks to Dr. Chunhui Mei and Prof. Jiaoying Shi for useful discussions about their work. Thanks to Paul Debevec for the light probe data. The authors are grateful to the anonymous reviewers for their helpful suggestions and comments.

References

AGRAWALA, M., RAMAMOORTHI, R., HEIRICH, A., AND MOLL, L. 2000. Efficient image-based methods for rendering soft shadows. In *SIGGRAPH '00*, 375–384.

ANNEN, T., KAUTZ, J., DURAND, F., AND SEIDEL, H.-P. 2004. Spherical harmonic gradients for mid-range illumination. In *Euro. Symp. on Rendering*, 331–336.

ASSARSSON, U., AND AKENINE-MOLLER, T. 2003. A geometry-based soft shadow volume algorithm using graphics hardware. In *SIGGRAPH '03*, 511–520.

CHAN, E., AND DURAND, F. 2003. Rendering fake soft shadows with smoothies. In *Euro. Symp. on Rendering*, 208–218.

CROW, F. C. 1977. Shadow algorithms for computer graphics. In *Computer Graphics (Proceedings of SIGGRAPH 77)*, vol. 11, 242–248.

DRETTAKIS, G., AND SILLION, F. X. 1997. Interactive update of global illumination using a line-space hierarchy. In *SIGGRAPH '97*, 57–64.

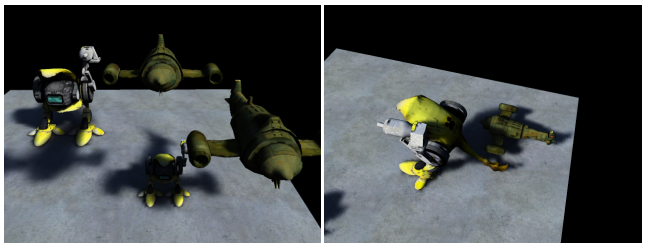

Figure 8: *All frequency shadows from moving spaceships.*

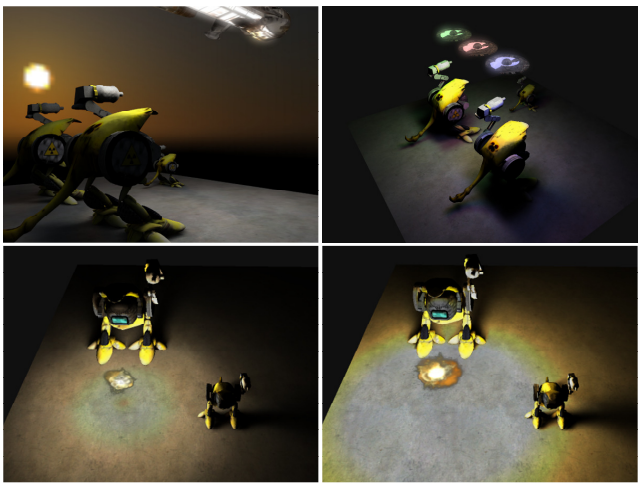

Figure 9: *Relighting and light design. Top left: relighting with both an environment map and a local light source; Top right: multiple local light sources; Bottom: a local light source represented by a video texture.*

GOESELE, M., GRANIER, X., HEIDRICH, W., AND SEIDEL, H.-P. 2003. Accurate light source acquisition and rendering. In *SIGGRAPH '03*, 621–630.

HECKBERT, P., AND HERF, M. 1997. Simulating soft shadows with graphics hardware. Tech. Rep. CMU-CS-97-104, Carnegie Mellon University.

HEIDRICH, W., KAUTZ, J., SLUSALLEK, P., AND SEIDEL, H.-P. 1998. Canned lightsources. In *Rendering Techniques '98, Eurographics*, 293–300.

JAMES, D., AND FATAHALTIAN, K. 2003. Precomputing interactive dynamic deformable scenes. In *SIGGRAPH '03*, 879–887.

KAUTZ, J., SLOAN, P., AND SNYDER, J. 2002. Fast, arbitrary brdf shading for low-frequency lighting using spherical harmonics. In *Euro. Workshop on Rendering*, 291–296.

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

LEHTINEN, J., AND KAUTZ, J. 2003. Matrix radiance tranfer. In *Symp. on Interactive 3D Graphics*, 59–64.

MCMILLAN, L., AND BISHOP, G. 1995. Plenoptic modeling: an image-based rendering system. In *SIGGRAPH '95*, 39–46.

MEI, C., SHI, J., AND WU, F. 2004. Rendering with spherical radiance transport maps. In *Eurographics '04*, 281–290.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2003. All-frequency shadows using non-linear wavelet lighting approximation. In *SIGGRAPH '03*, 376–381.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2004. Triple product wavelet integrals for all-frequency relighting. In *SIGGRAPH '04*, 477–487.

OUHYOUNG, M., CHUANG, Y.-Y., AND LIANG, R.-H. 1996. Reusable Radiosity Objects. *Computer Graphics Forum 15*, 3, 347–356.

SLOAN, P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH '02*, 527–536.

WILLIAMS, L. 1978. Casting curved shadows on curved surfaces. In *Computer Graphics (Proceedings of SIGGRAPH 78)*, vol. 12, 270–274.

WYMAN, C., AND HANSEN, C. 2003. Penumbra maps: Approximate soft shadows in real-time. In *Euro. Symp. on Rendering*, 202–207.