Eurographics Symposium on Rendering (2008)
Steve Marschner and Michael Wimmer (Guest Editors)

Fast Soft Self-Shadowing on Dynamic Height Fields

John Snyder[1] and Derek Nowrouzezahrai[2]

[1]Microsoft Research
[2]Dynamic Graphics Project, University of Toronto

Abstract

We present a new, real-time method for rendering soft shadows from large light sources or lighting environments on dynamic height fields. The method first computes a horizon map for a set of azimuthal directions. To reduce sampling, we compute a multi-resolution pyramid on the height field. Coarser pyramid levels are indexed as the distance from caster to receiver increases. For every receiver point and every azimuthal direction, a smooth function of blocking angle in terms of log distance is reconstructed from a height difference sample at each pyramid level. This function's maximum approximates the horizon angle. We then sum visibility at each receiver point over wedges determined by successive pairs of horizon angles. Each wedge represents a linear transition in blocking angle over its azimuthal extent. It is precomputed in the order-4 spherical harmonic (SH) basis, for a canonical azimuthal origin and fixed extent, resulting in a 2D table. The SH triple product of 16D vectors representing lighting, total visibility, and diffuse reflectance then yields the soft-shadowed result. Two types of light sources are considered; both are distant and low-frequency. Environmental lights require visibility sampling around the complete 360° azimuth, while key lights sample visibility within a partial swath. Restricting the swath concentrates samples where the light comes from (e.g. 3 azimuthal directions vs. 16-32 for a full swath) and obtains sharper shadows. Our GPU implementation handles height fields up to 1024×1024 in real-time. The computation is simple, local, and parallel, with performance independent of geometric content.

Categories and Subject Descriptors: I.3.7 [Computer Graphics]: Color, shading, shadowing, and texture

1. Introduction

Soft shadows from large light sources provide critical cues for perceiving shape. Without shadows, geometric relationships such as how high off the ground a character jumps, how tall terrain peaks are, or whether a given feature indents or protrudes, become ambiguous. Hard shadows from point lights have other problems. Unlike soft shadows whose penumbras lighten gradually as the shadow caster recedes from the receiver, they obscure unlit areas completely. They also introduce visually prominent, often confusing edges.

Several methods exist for soft shadow rendering. Ray tracing shoots many shadow rays from each receiver point to query whether an object intervenes before the ray hits the sky or other light source. An alternative is to accumulate over many hard shadow renderings of the scene, by distributing a set of points over the area light source [SKVW*92]. In either case, large lights require the sampling of many hundreds up to thousands of different directions. This is prohibitively expensive for applications involving dynamic geometry or large datasets.

We present a new method specialized to height fields for generating soft shadows. Visibility is computed from scratch every frame, allowing interactive changes to both geometry and lighting. Direct applications include visualization of scalar, bivariate functions/data and realistic rendering of terrains for flight simulators and outdoor 3D games. Our method uses simple operations and local memory accesses that map well to the GPU, and obtains high-quality soft shadows in real-time (Figure 1).

Our work has two contributions. First, we show how to efficiently compute the height field's self-visibility in terms of a horizon map [Max88]. We do this using a multi-resolution pyramid which increasingly prefilters the height geometry as the distance from receiver to caster increases. Though multi-

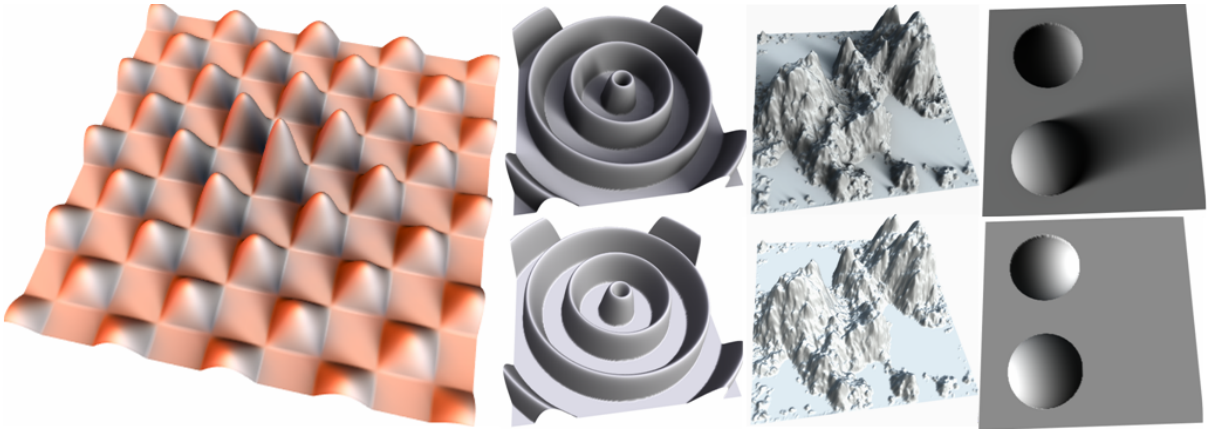


Figure 1: *Soft shadow results. We achieve frame rates of about 57Hz (including geometry generation, pyramid creation, horizon map extraction, and shadowed shading) for these 256×256 dynamic height fields on an Nvidia 8800 Ultra GPU, from both environmental and key light sources. Shadowed (top) and unshadowed (bottom) images are compared in the right three columns.*

resolution pyramids have long been used in computer graphics and image processing [Bur81, Wil83], our application of them to soft shadow rendering, based on the *multi-scale directional derivative* (Equation 2), is new. We also show how using pyramids with more levels than the standard power-of-2 improves shadowing quality (Figure 3). Second, we show how to efficiently generate soft shadows given the horizon map. Each consecutive pair of horizon angles is converted to a *visibility wedge* represented using order-4 spherical harmonics (SH), and a total visibility vector accumulated over all wedges. When the lighting is strongly directional, we also show how restricting the visibility computation to a partial swath obtains sharper shadows (Figure 6).

2. Previous Work

Self-visibility (or equivalently, self-shadowing) on a continuous height field can be represented by the maximum elevation angle it attains as a function of location and azimuthal direction. In other words, what angle does the horizon make at each spot as one gradually turns around? *Horizon maps* [Max88, SC00] apply this observation over a discretized set of directions. They were originally precomputed and used for rendering hard shadows on bump mapped surfaces. We introduce a new way to approximate them on-the-fly and use them to render soft shadows, both in real-time. Computational geometry techniques also exist for computing horizon maps [Ste98], but are not suitable for fast evaluation.

Ambient occlusion (AO) [Bun05, KL05, SA07, DBS08] produces maximally soft shadows based on the average visibility direction and its total subtended angle. Our method generalizes ambient occlusion by computing true low-frequency blocker visibility (in terms of 16D spherical harmonic vectors) rather than using a simple cone model. This "casts" our shadows much more clearly in response to directional lighting (see [RWS*06] for image comparisons). Two AO methods [SA07, DBS08] are based on screen-space depths and so do not account for shadows cast by geometry not visible from the viewpoint. [DBS08] is similar to our method in that it is based on horizon maps and employs a piecewise linear approximation to the horizon angle as a function of azimuthal angle. It uses ray marching on the fine-resolution depth buffer and thus requires extensive sampling to prevent aliasing, a problem we address using prefiltering with a multi-resolution pyramid.

Two related methods fit an ellipse [HDKS00] or circular aperture [OS07] over the height field as a preprocess. We produce more accurate soft shadows in terms of low-frequency visibility instead of simple cone-based models and compute the representation on-the-fly to allow dynamic height fields.

Precomputed radiance transfer (PRT) [SKS02] generates real-time soft shadows and allows real-time manipulation of area lighting, represented using low-order SH. Unfortunately, it requires costly precomputation involving ray tracing over each static object and so is unsuitable for dynamic geometry or immediate feedback. Spherical harmonic exponentiation (SHEXP) [RWS*06, SGNS07] allows soft shadows on dynamic geometry, but is specialized for articulated characters, which are approximated as a set of a few spheres. The sphere approximation is not appropriate for self-shadowing on detailed height fields. Moreover, it is generated by an expensive precomputation on the character's rest pose and relies on "skinning" the articulated skeleton to animate the fitted spheres.

Another family of methods generate "soft" shadows from small area light sources, based on the shadow buffer [Wil78]. See [HLHS03] for an extensive survey. Early work, called percentage-closer filtering [RSC87], softens a hard shadow by filtering the result of many shadow buffer depth tests. Variance shadow maps [DL06] reduce this cost by storing a

depth mean and variance per shadow buffer pixel. These and other related methods essentially blur a hard shadow based on an area source centered around a single direction. They are inappropriate for large light sources or lighting environments and do not accurately simulate penumbras even for small sources. They also suffer from the well-known self-shadowing problem due to discrepancies between depths sampled from the camera and from the light.

Shadow volumes can also be extended to produce soft shadows [AMA02, AAM], but are likewise based on small sources and make several heuristic assumptions (e.g., a blocker's silhouette is fixed for all receiver points). Unlike shadow buffer approaches, they exhibit performance that is sensitive to geometric content and can be quite poor. Our method is based on horizon maps rather than shadow buffers or shadow volumes. That is, we compute self-visibility over the height field independently of how it is lit. This avoids self-shadowing problems and allows large light sources, but still can be computed with fast and predictable performance using our multi-resolution approach.

Approximate soft shadows are generated by height difference sampling in [Tat06], given a single light direction. Shadows are based on a heuristic model which softens what is essentially a hard shadow based on the azimuthal distance to the blocker. We compute soft shadows from true low-frequency light sources, employing a height field pyramid to prevent aliasing.

Our method applies multi-resolution methods that have been used in the similar problem of terrain rendering [LH04, TIS08]. The problem of soft shadow rendering is not explicitly addressed by these techniques.

3. Summary of Basic Ideas

To compute shadows, we find the maximum angle attained by the horizon along each *azimuthal direction*, or 2D direction in the height field's plane. This requires sampling the height difference between a *receiver* point and many *caster* points at different distances away along the azimuthal direction. Without any prefiltering or *a priori* knowledge of the geometry, a fixed sampling rate per unit distance is needed. Many samples are taken far from the receiver, slowing the computation. The problem gets worse when sampling over a 2D swath rather than a single lighting direction.

We reduce the need for so many samples by computing a *multi-resolution pyramid* over the height field. Height difference at a given distance is sampled at an appropriate pyramid level. Far from the receiver point, height differences are prefiltered by sampling them at a coarse level. Closer to the receiver point, differences at finer pyramid levels provide more detail that is sampled more densely. Dividing height difference by the azimuthal distance yields the tangent of the blocking angle. The computation amounts to a *multi-scale directional derivative*.

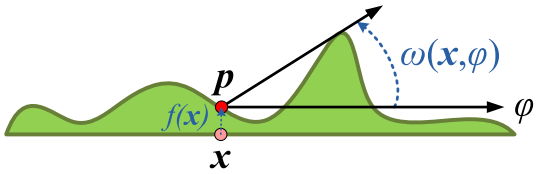


Figure 2: *Horizon angle at receiver point* $\boldsymbol{x}$ *as a function of azimuthal direction* φ.

For each azimuthal direction and each receiver point, we compute the blocking angle at each pyramid level, reconstruct a smooth function of log distance from these samples using cubic b-spline interpolation, and find its maximum. Finally, we convert a series of horizon elevation angles, sampled at various azimuthal directions, to a visibility vector using the spherical harmonic (SH) basis.

We obtain sharper shadows if we know *a priori* that the light is not coming from all around the height field, but instead is azimuthally restricted to a *partial swath* (Figure 5, right). Computing visibility over the complete horizon in a low-order SH basis blurs in information outside the swath which over-softens the shadows. Using a partial swath ensures only visibility inside the swath affects shading. It is as though the light were truly restricted to the swath (and exactly 0 everywhere outside it), even though functions bandlimited to low-order SH are not capable of this.

Two observations are important for obtaining smooth soft shadows. First, height field pyramids need not be limited to the standard power of 2 type; pyramids with more gradual resolution transitions generate smoother shadows (see Figure 3). Second, smooth reconstruction is necessary when sampling height differences to avoid artifacts in the shadows (see Figure 9). Although such reconstruction is not natively supported on current GPUs, it can be efficiently evaluated using four bilinear texture map accesses [SH05].

4. Multi-scale Self-Visibility

A *height field* is a scalar function of two variables, $\boldsymbol{x} = (x, y)$, and denoted $f(\boldsymbol{x}) = f(x, y)$. It is represented as a tabulated grid of heights with sampling resolution $n_x \times n_y = n \times (\alpha n)$, where α is the aspect ratio. For simplicity, we will assume square height fields with sample spacing denoted $\Delta = 1/n$.

The height field determines a 3D surface $z = f(\boldsymbol{x})$ containing 3D points $\boldsymbol{p} = (x, y, f(x, y))$, which we would like to shade with soft shadows. To do this, at every point $\boldsymbol{x}$ we need to compute the visibility in all azimuthal directions, parameterized by $(\cos\varphi, \sin\varphi)$, $\varphi \in [0, 2\pi]$.

Visibility is represented as the maximum elevation angle the horizon reaches in the azimuthal direction φ as seen from $\boldsymbol{p}$, called the *horizon angle*. At elevation angles below the horizon angle, the direction is blocked by the height field itself and cannot "see the sky" (see Figure 2). The horizon

angle is denoted $\omega(\boldsymbol{x},\varphi)$ and defined via

$$\max_{t\in(0,\infty)} \tan^{-1}\left(\frac{f(x+t\cos\varphi, y+t\sin\varphi)-f(x,y)}{t}\right) \quad (1)$$

where t represents azimuthal distance along the direction φ away from the receiver point $\boldsymbol{x}$. Note that the max and inverse tangent functions can be interchanged, since inverse tangent is monotonic.

The horizon angle is 0 if the height field does not occlude itself at all (i.e., one sees the entire sky all the way down to "sea level") and varies up to $\pi/2$ as the height field increasingly occludes itself. We assume $\omega(\boldsymbol{x},\varphi) \geq 0$. This is true for height fields of bounded height but unbounded extent, and roughly true for relatively flat height fields. It can be seen that (1) contains an expression very like a numerical directional derivative of f, which forms the tangent of the blocking angle as a function of distance t. This observation prompts our prefiltering approximation.

To approximate (1), we use a *multi-resolution pyramid*. Each *pyramid level* (or *scale*) is denoted $f_i(x,y), i \in \{0,1,\cdots,N-1\}$, with corresponding sample spacing $\Delta_i = b^{-i}$. (This sample spacing assumes the entire extent of the height field is normalized to unit length at each pyramid level.) The base $b = 2^{\frac{1}{k}}$ controls sampling reduction between levels in terms of a *level step* k. So $k=1$ is a standard power-of-2 pyramid while a larger k yields a finer transition between levels, having k levels for each power of 2 reduction in sample spacing. The pyramid is constructed on the original tabulated height field data by repeated decimation, as will be detailed later. $N = \lceil k \ln n / \ln 2 \rceil + 1$ is the total number of pyramid levels. Memory usage for this pyramid as a factor over the original's is given by $\gamma(k) = 1/(1-2^{-\frac{2}{k}})$. We typically use $k=4$, resulting in an expansion factor of $\gamma \approx 3.4$. Figure 3 shows how k affects shadow sampling.

Visibility can now be approximated by computing a *multi-scale directional derivative* on this pyramid, defined in terms of scale i, direction φ, and distance d_i:

$$D(f_i,\boldsymbol{x},\varphi,d_i) = \frac{f_i(x+d_i\cos\varphi, y+d_i\sin\varphi)-f_i(x,y)}{d_i} \quad (2)$$

where $d_i = l\Delta_i$. The directional derivative can be computed by subtracting two nearby evaluations of the function f_i, and multiplying by the precomputed constant $1/d_i$.

We fix the distance relative to sample spacing at each pyramid level; e.g. $l=1$. The directional derivative determines the tangent of the horizon angle (at scale i, in direction φ, and at distance l relative to the sample spacing Δ_i) which shadows light arriving along φ. We therefore define the *horizon angle at scale i* as

$$\omega_i(\boldsymbol{x},\varphi,d_i) = \tan^{-1}\left(D(f_i,\boldsymbol{x},\varphi,d_i)\right). \quad (3)$$

Inverse tangent is computed at each pyramid level because it is better to interpolate in the space of angles than in the space of tangents. Tangent is an unbounded function, and interpolating it directly overemphasizes shadowing.

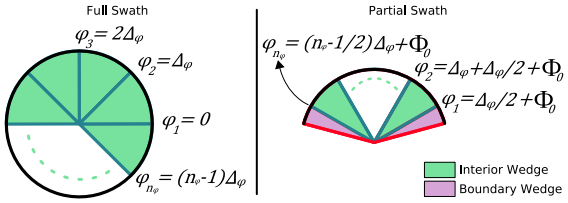


Figure 5: *Azimuthal sampling for complete and partial visibility swaths. Direction samples are drawn as blue lines.*

We use (1D) cubic b-spline interpolation to convert these blocking angle samples ω_i at each pyramid level i to a continuous function $\omega(\tau,\boldsymbol{x},\varphi)$ of scale space, τ, via:

$$\text{b-spline}\left(\tau, \{\omega_0(\boldsymbol{x},\varphi,d_0),\cdots,\omega_{N-1}(\boldsymbol{x},\varphi,d_{N-1})\}\right). \quad (4)$$

The parameter τ represents negative log distance away from the receiver point; i.e., $\tau = -\log_b t$. B-spline interpolation takes each sequence of 4 consecutive values, $(\omega_{i-1},\omega_i,\omega_{i+1},\omega_{i+2})$ and an interpolant $u = \tau - \lfloor\tau\rfloor \in [0,1]$ representing blending distance between the values at $i = \lfloor\tau\rfloor$ and $i+1$. The result is $\beta_0(u)\,\omega_{i-1} + \beta_1(u)\,\omega_i + \beta_2(u)\,\omega_{i+1} + \beta_3(u)\,\omega_{i+2}$ where the b-spline basis functions $\beta_i(u)$ are defined as $\beta_0(u) = 1/6\,(1-u)^3$, $\beta_1(u) = 1/6\,(3u^3-6u^2+4)$, $\beta_2(u) = 1/6\,(-3u^3+3u^2+3u+1)$, and $\beta_3(u) = 1/6\,(u^3)$.

Finally, we find the max of this function over τ:

$$\omega(\boldsymbol{x},\varphi) \approx \max_{\tau\in[0,N-1]} \omega(\tau,\boldsymbol{x},\varphi) \quad (5)$$

to yield an approximation of the horizon angle. We constrain it to be non-negative by taking a final max of (5) with 0.

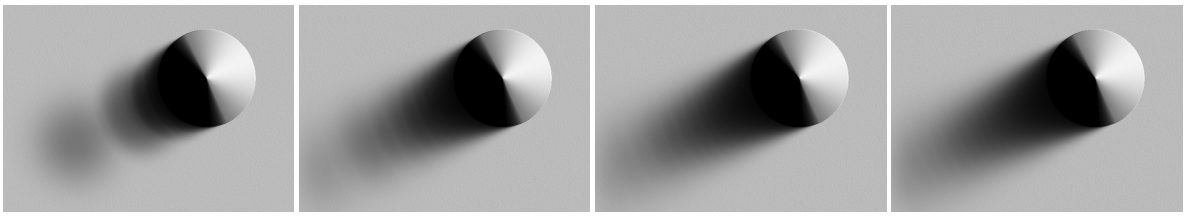
Additional control is provided by a *level offset*, o, which biases each level i evaluated in (4) by adding ko; i.e. by evaluating $f_{[i+ko]}$ instead of f_i. The offset index is clamped so as not to exceed the a maximum level index $N-1$ via

$$[i+ko] = \max(\lceil i+ko\rceil, N-1).$$

Only the pyramid level accessed is biased; the distance d_i in the directional derivative is preserved. Increasing o results in sharper shadows at the cost of increased sampling requirements (Figure 4). At large o, we end up sampling exclusively from the original, unfiltered height field.

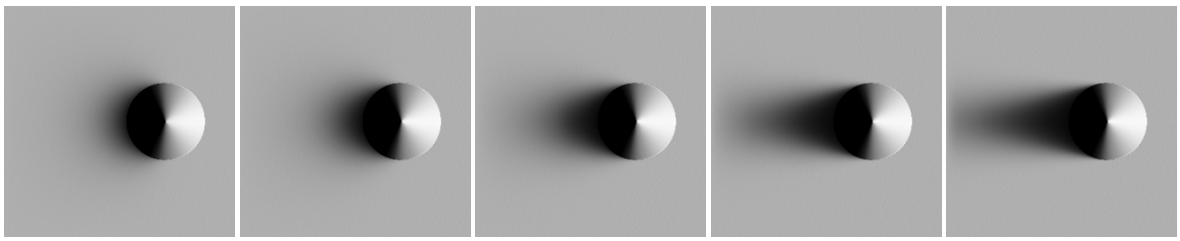
When evaluating ω_i, the first term in the numerator of (2) samples at locations off the height field grid. Smooth reconstruction (e.g. bicubic b-spline interpolation) is necessary to avoid shadow artifacts (see Figure 9). We will show in Section 6 how to achieve essentially the same result at less cost, by first reconstructing full-resolution pyramid levels using bicubic b-spline interpolation and then accessing them with standard bilinear interpolation.

So far we have considered self-shadowing relative to a single azimuthal direction, φ. An area light subtends a continuous range of such directions, called an *azimuthal swath*



(a) k=1 (b) k=2 (c) k=3 (d) k=4

Figure 3: *Effect of level step k on soft shadows. Increasing k eliminates aliasing and makes the shadow smoother. Power-of-2 pyramids do not always suffice: 4 levels per power-of-2 step (k=4) provides a smooth result in (d).*



(a) o=0 (b) o=1 (c) o=2 (d) o=3 (e) o=4

Figure 4: *Effect of level offset o on soft shadows. Increasing o results in sharper shadows. In this example, k=4.*

and denoted $\varphi \in \Phi = [\Phi_0, \Phi_1]$. Refer to Figure 5. The swath's extent is denoted $\Delta_\Phi = \Phi_1 - \Phi_0$. The light's azimuthal extent is determined by bounding the set of lighting directions projected to the height field plane. If the light is strongly directional (comes from one side of the height field only), we can sample within a *partial swath*. A broader light requires a larger partial swath. Lights positioned directly over the height field or surrounding it on all sides require the *complete swath* $\Phi^* = [0, 2\pi]$. Using the smallest swath necessary for the lighting setup samples visibility only where needed and makes the shadows sharper.

We then discretize Φ into a set of n_φ equally spaced *azimuthal direction samples* $\{\varphi_1, \varphi_2, \cdots, \varphi_{n_\varphi}\}$, where $\varphi_{i+1} - \varphi_i = \Delta_\varphi = \Delta_\Phi / n_\phi$. At each point $\boldsymbol{x}$ and for each direction sample φ_i, we evaluate the horizon angle using (5).

5. Shading with Spherical Harmonics

We represent the space of directions emanating from $\boldsymbol{p}$ via unit vectors $\boldsymbol{s}$ parameterized by azimuthal angle $\phi \in [0, 2\pi]$ and elevation angle $\theta \in [-\pi/2, +\pi/2]$:

$$\boldsymbol{s}(\phi, \theta) = (\cos\phi\cos\theta, \sin\phi\cos\theta, \sin\theta).$$

The elevation angle is 0 for a direction skirting the height field plane, and $\pi/2$ if it points straight up, out of this plane.

At a given receiver point $\boldsymbol{x}$, we obtain a sequence of horizon angles $\omega(x, \varphi_i)$ over the azimuthal direction samples φ_i. Each consecutive pair of azimuthal directions $(\varphi_i, \varphi_{i+1})$ determines a continuous *visibility wedge* as seen from x with corresponding blocker angles (ω_i, ω_{i+1}) (see Figure 7). To reconstruct a continuous function, $\omega(\varphi)$, we simply assume

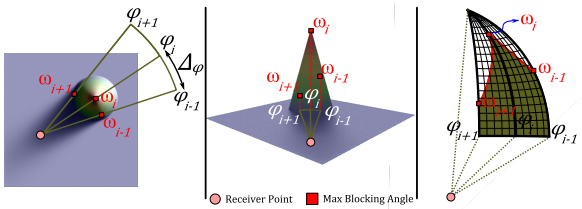


Figure 7: *Visibility wedge example. The azimuthal direction samples are at φ_{i-1}, φ_i, and φ_{i+1} with corresponding horizon angles ω_{i-1}, ω_i, and ω_{i+1}. On the right, the visibility function is linearly interpolated between the discrete samples and projected to the sphere of directions to define two interior visibility wedges.*

that the horizon angle varies linearly from ω_i to ω_{i+1} as a function of ϕ between φ_i and φ_{i+1}, yielding

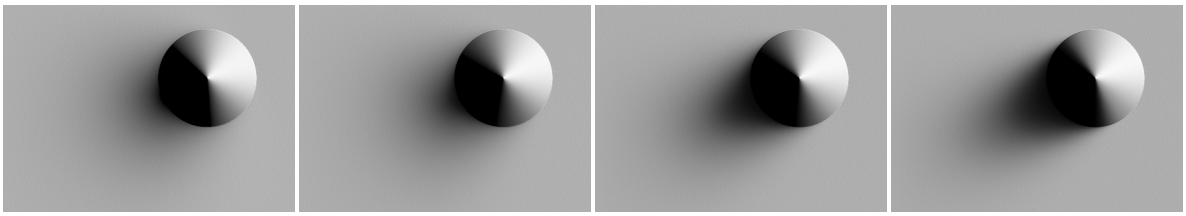
$$\omega(\phi) = \omega_i + \frac{\phi - \varphi_i}{\Delta_\varphi}(\omega_{i+1} - \omega_i). \tag{6}$$

The corresponding visibility function is given by

$$v(\phi, \theta) = \begin{cases} 0, & \text{if } \phi \in [\varphi_i, \varphi_{i+1}] \text{ and } \theta \le \omega(\phi) \\ 1, & \text{if } \phi \in [\varphi_i, \varphi_{i+1}] \text{ and } \theta > \omega(\phi) \\ 0, & \text{if } \phi \notin [\varphi_i, \varphi_{i+1}] \end{cases} \tag{7}$$

So v indicates whether a ray emanating from $\boldsymbol{p}$ in the direction $\boldsymbol{s}(\phi, \theta)$ misses the height field or not, and so is able to see the sky in that direction.

By canonically repositioning each wedge so that $\varphi_i = 0$, and assuming a fixed Δ_φ, we can precompute a low-order spherical harmonic (SH) projection of the wedge function, in terms of the two horizon angles (ω_i, ω_{i+1}). Then at run-



(a) Δ_Φ=90° (b) Δ_Φ=45° (c) Δ_Φ=22.5° (d) Δ_Φ=11.25°

Figure 6: *Effect of azimuthal swath size Δ_Φ on soft shadows. This example fixes k=4, o=2, n_ϕ=32, and lighting as the "peakiest" representable with SH order 4, normalized to produce unit maximum unshadowed shading. Restricting the swath size obtains sharper shadows, which converge to a given sharpness limit that depends on the lighting.*

time, we align the canonical wedge vector with its actual azimuthal origin φ_i, using a fast z-rotation in the SH basis [KSS02]. Wedge information is stored as a 2D table (256×256) which produces an order-4 (16D) SH visibility vector. Denote this table as $\boldsymbol{v}(\omega_a, \omega_b)$ and its corresponding visibility function as $v(\boldsymbol{s}; \omega_a, \omega_b)$. The wedge table is computed by numerically integrating v against the SH basis functions over the sphere $\boldsymbol{s} \in \boldsymbol{S}$; i.e.,

$$\boldsymbol{v}_i(\omega_a, \omega_b) = \int_{\boldsymbol{S}} v(\boldsymbol{s}; \omega_a, \omega_b)\, \boldsymbol{y}_i(\boldsymbol{s})\, ds \tag{8}$$

where $\boldsymbol{y}_i$ represents the i-th component of the SH basis functions and v is defined in (7).

Interior wedges (drawn in green in Figure 5) are determined by a consecutive pair of horizon angles. For partial swaths, we also tabulate a starting and ending *boundary wedge* (drawn in pink in Figure 5), of azimuthal extent $\Delta_\varphi/2$ as a 1D function of a single horizon angle, which interpolates to a fully blocked angle at the edge of the swath. Each wedge is evaluated in terms of its horizon angle(s), rotated, and then accumulated to yield the total SH visibility vector, $\boldsymbol{V}$. Further details and code are available in [SN08].

Assuming a Lambertian receiver with unit normal vector $\boldsymbol{N}_x$ at $\boldsymbol{x}$, shading is the SH triple product of the clamped cosine hemisphere around the normal vector, $\boldsymbol{H}(\boldsymbol{N}_x)$, the visibility vector, $\boldsymbol{V}_x$, and the (constant) SH lighting vector, $\boldsymbol{L}$. It can be computed by evaluating the binary SH product $\boldsymbol{T}_x = \boldsymbol{H}(\boldsymbol{N}_x) * \boldsymbol{V}_x$, called the *transfer vector*, followed by its 16D dot product with the lighting, $\boldsymbol{T}_x \cdot \boldsymbol{L}$. This formulation allows shading with a different lighting vector by recomputing a simple dot product. Alternatively, if the light is fixed we can compute the SH triple product directly. Fast evaluation of these operations is discussed in [Sny06].

To compute $\boldsymbol{H}(\boldsymbol{N}_x)$, representing Lambertian reflectance of the receiver point, we use the zonal harmonic (ZH) rotation rule [SLS05]. Zonal harmonics are the subset of the SH basis functions which are circularly symmetric around z. ZH coefficients for the clamped cosine function can be found analytically [RH01], yielding the 4D vector constant $\{0.886227, 1.02333, 0.495416, 0\}$. Rotating this ZH vector to align with the direction $\boldsymbol{N}_x$ then yields $\boldsymbol{H}(\boldsymbol{N}_x)$.

For small azimuthal swaths, our visibility wedge method actually obtains sharper shadows than can ordinarily be realized with low-order SH. For example, the right columns of the bottom row of Figure 4 show shadows from a cone that are much sharper than can be realized with complete swath visibility at SH order 4. In this case, the azimuthal wedge spanned 20°. Figure 6 compares different values of Δ_Φ.

6. GPU Implementation

Our implementation is built on DirectX10. We build the height pyramid on-the-fly on the GPU given the highest resolution height field f_{N-1}. Successive bilinear decimations are applied fine-to-coarse to pre-filter the data, followed by bicubic b-spline interpolation back up to the original resolution using the method in [SH05]. The computation thus takes $2N$ shader passes. This creates an "image array" representing the height field pyramid which is subsequently accessed using bilinear interpolation to evaluate the directional derivative in (2). By reconstructing the smoothed result once in a preliminary pass, we avoid doing it for every azimuthal direction. Note that we could avoid resampling coarse pyramid levels all the way back to the original resolution by using multiple image arrays, but have not done so in our implementation.

We also compute normals $\boldsymbol{N}_x$ at each point on the height field. The horizon map, total visibility vector $\boldsymbol{V}_x$, and diffuse shading are then computed in a single shader pass that straightforwardly implements Sections 4 and 5 (see code in [SN08]). Shading is calculated at all n^2 grid samples on the height field and used to texture map the tessellated geometry, containing $2(n-1)^2$ triangles.

To evaluate the multi-scale derivative in (2), we use constant arrays to store the precomputed azimuthal distances per pyramid level and their reciprocals, d_i and $1/d_i$, and the azimuthal directions, $(\cos\varphi_i, \sin\varphi_i)$. To compute the horizon angle at scale i in (3), we use a fast approximate inverse tangent requiring only a single divide [Has53] (see code in [SN08]). To compute Equation (5), we do a brute force evaluation of the b-spline at two sample points between every pair of pyramid levels, and then take the max. More precisely, we sample the b-spline at every knot point ($\tau = i$) as well as halfway between each knot pair ($\tau = (i+1)/2$). The

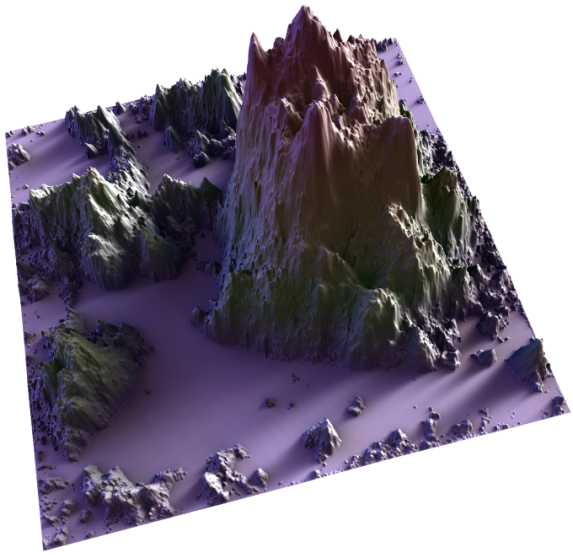


Figure 8: *Soft shadows on a fractal terrain (1024×1024 height field) with added color map. Frame rate is 2.5Hz.*

corresponding b-spline weights are $\{1/6, 2/3, 1/6, 0\}$ and $\{1/48, 23/48, 23/48, 1/48\}$.

We use two different types of light sources, $\boldsymbol{L}$, to shade the height field. *Environmental light* comes from all around the height field and uses a complete azimuthal swath to provide a very soft shadow. A *key light* represents a smaller, more directional source and uses a partial azimuthal swath to provide sharper shadows. Both are represented as 16D SH vectors and are derived from HDR light probes [Deb98] or analytic circular sources. When sampling visibility, we use many (n_φ=16-32) azimuthal directions for environmental lights and fewer (n_φ=3) for key lights.

A separate shading pass is computed for each type of light. Multiple key lights must be computed in separate passes unless they share an identical azimuthal swath. Because of the smoothness of shadows from environmental lighting, another speed-up is to avoid b-spline interpolation entirely and instead compute the max only over pyramid levels. Avoiding interpolation also lets us compute inverse tangent just once after taking the max instead of N times at each pyramid level. The more expensive visibility computation in Equations (4) and (5) is still performed for key lights.

7. Results

Our method is based on a few simple parameters: the level step k (which given the height field resolution n determines the number of pyramid levels N), the level offset o, the swath size Δ_Φ, and the number of direction samples in the swath n_φ. The effect of these parameters is demonstrated using a simple cone geometry having unit height: k in Figure 3, o in Figure 4, and Δ_Φ in Figure 6. Different reconstruction methods are compared in Figure 9, which shows that smooth interpolation is critical for shadowing quality. Our multi-scale method is compared with ground truth in Figure 10, showing that error is controllable and that shadows are made softer but without other visual artifacts as the pyramid is used more aggressively (by decreasing k and o) to increasingly prefilter height differences as a function of distance.

n	N	fps (key only)	fps (key+env)
256 (Fig. 1)	33	165	57.2
512 (Fig. 11)	37	34.5	11.5
1024 (Fig. 8)	41	7.05	2.48

Table 1: *Performance statistics for different height field resolutions, $n \times n$, with k=4. We use $n_\varphi = 3$ samples for key lights and $n_\varphi = 16$ for lighting environments. Computation time includes everything except creation of the fractal height field, which is done on the CPU. Horizon map extraction and shading is done at each of the n^2 height field grid locations.*

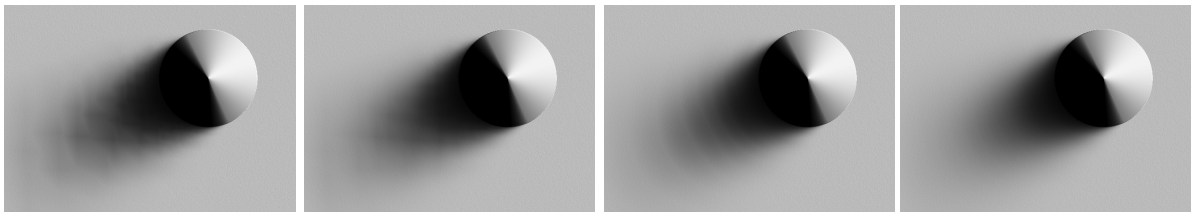
Performance results are summarized in Table 1, on an Intel Xeon 3.2GHz CPU with 2GB of memory and an Nvidia 8800 Ultra graphics card. Timings use 8× multisampling at the highest quality setting in a 640×480 image window. Since horizon map extraction and soft-shadowed shading is computed at height field grid samples rather than pixels, performance is relatively insensitive to image size, only to height field resolution. Performance slows with increasing k (N) but is realtively insensitive to o. Computing the wedge tables (2D for interior wedge, 1D for boundary wedge) using (8) takes less than a second on the CPU for 256 elevation angle samples.

We obtain a good trade-off between shadowing quality and performance using k=4. Figure 11 compares results applying the different lighting types (key and environmental) to 512×512 height fields. For visualization purposes, it is effective to combine a key light to cast a strong shadow with an environment light to brighten dark areas. Environmental shading still exhibits realistic soft shadows: terrain depressions are darkened relative to peaks. Figure 8 shows a fractal terrain which adds color mapped as a function of height to the soft shadows.

The accompanying video shows interaction with dynamic height fields (256×256) in real-time. It was recorded with n_φ=32 for environmental lighting and achieves a frame rate of about 36Hz for combined key+env lighting.

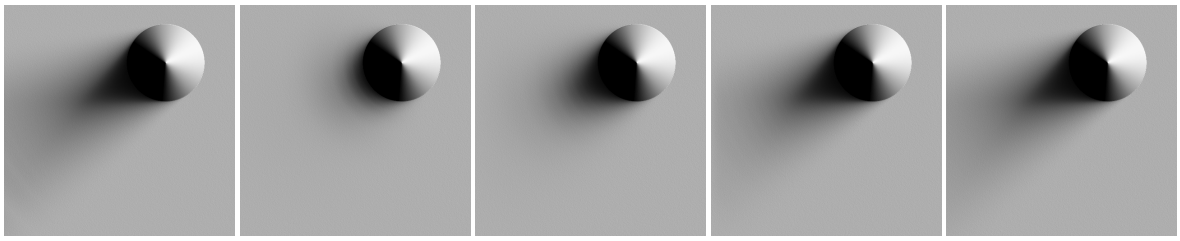
8. Conclusion

Sampling height differences from a multi-resolution pyramid provides a simple and robust way to prefilter the geometry and substantially reduces the number of samples needed to extract a horizon map. The resulting horizon map can be converted to spherical harmonic visibility wedges to efficiently render soft shadows. We map both computations to the GPU, allowing shading from smooth, distant light



(a) linear (1D), bilinear (2D) (b) b-spline (1D), bilinear (2D) (c) linear (1D), b-spline (2D) (d) b-spline (1D & 2D)

Figure 9: *Effect of reconstruction method on soft shadows. Interpolation is invoked in 1D to reconstruct a continuous function over scale space in (4), and in 2D when evaluating the multi-scale derivative in (2). Using smooth (C^2) functions like b-splines for both provides the smooth, natural-looking shadow shown in (d).*



(a) ground truth (b) k=1, o=1 (N=10) (c) k=2, o=2 (N=19) (d) k=4, o=4 (N=37) (e) k=8, o=8 (N=73)

Figure 10: *Comparison with ground truth, n=512. As we increase k and o simultaneously, we obtain a more exact approximation of the height field geometry. The resulting soft shadows converge to the ground truth result in (a), which was generated by applying (1) on the fine-resolution height field using a large number of samples (145) distributed linearly with distance.*

sources on height fields where the lighting and geometry can change in real-time.

In future work, a simple extension is higher-order interpolation of the horizon angle as a function of azimuthal angle. Subsampling the SH visibility vector (at less than one sample per height field grid location) would probably speed up the algorithm at little loss of quality. These ideas could also be extended to more general geometry such as triangle meshes and geometry maps, perhaps by treating them locally as height fields, or by doing the shadowing in screen space as in [SA07, DBS08]. Our method could also be combined with one such as spherical harmonic exponentiation [RWS*06, SGNS07] to cast soft shadows from other geometry onto the height field. Multiplying the SH visibility vector from the moving shadow casters by the self-shadowing we have computed for the height field, $\boldsymbol{V}_x$, yields the combined shadowing effect at each receiver point. Finally, our multi-resolution approach could be extended to include inter-reflections as well as direct shadowing.

References

[AAM] ASSARSSON U., AKENINE-MÖLLER T.: A geomtery-based soft shadow volume algorithm using graphics hardware. *ACM Transactions on Graphics 22*, 3, 511–520.

[AMA02] AKENINE-MÖLLER T., ASSARSSON U.: Approximate soft shadows on arbitrary surfaces usin penumbra wedges. *13th Eurographics Workshop on Rendering* (2002), 297–305.

[Bun05] BUNNELL M.: Dynamic ambient occlusion and indirect lighting. In *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation*. Addison-Weseley Professional, 2005, pp. 223–233.

[Bur81] BURT P.: Fast filter transforms for image processing. *Computer Graphics and Image Processing 16* (1981), 2–51.

[DBS08] DIMITROV R., BAVOIL L., SAINZ M.: Horizon-split ambient occlusion. In *SI3D '08: Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games* (2008).

[Deb98] DEBEVEC P.: Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. 189–198.

[DL06] DONNELLY W., LAURITZEN A.: Variance shadow maps. In *I3D '06: Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games* (2006), pp. 161–165.

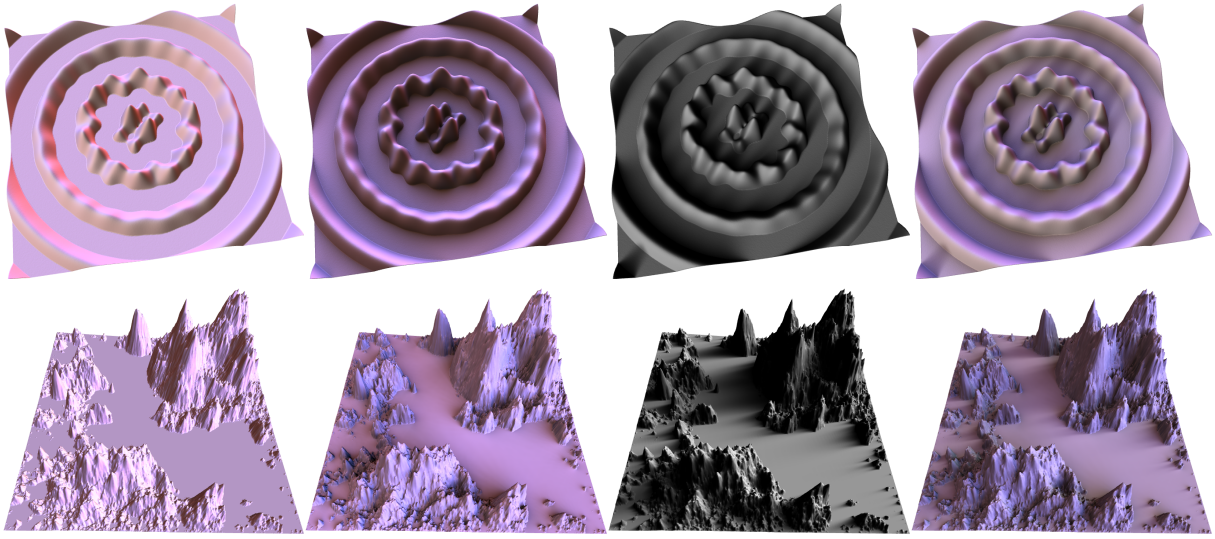
[Has53] HASTINGS C.:. *Approximation Theory, Note 143, Math. Tables Aids. Comp 6 68* (1953).

[HDKS00] HEIDRICH W., DAUBERT K., KAUTZ J., SEIDEL H.: Illuminating micro geometry based on precomputed visibility. In *Proceedings of ACM SIGGRAPH 2000* (2000), pp. 455–464.

[HLHS03] HASENFRATZ J., LAPIERRE M., HOLZSCHUCH N., SILLION F.: A survey of real-time soft shadow algorithms. *Computer Graphics Forum 22*, 4 (2003), 753–774.

[KL05] KONTKANEN J., LAINE S.: Ambient occlusion fields. In *Proc. of Symposium on Interactive 3D Graphics, SI3D* (2005), pp. 41–48.

[KSS02] KAUTZ J., SLOAN P., SNYDER J.: Fast, arbitrary BRDF shading for low-frequency lighting using spherical harmonics. In



unshadowed | env only (15.8Hz) | key only (34.6Hz) | key+env (11.5Hz)

Figure 11: *Results with 512×512 height fields, comparing environmental and key lighting types.*

Proc. of the 13th Eurographics Workshop on Rendering (2002), pp. 291–296.

[LH04] LOSASSO F., HOPPE H.: Geometry clipmaps: terrain rendering using nested regular grids. vol. 23, pp. 769–776.

[Max88] MAX N.: Horizon mapping: shadows for bump-mapped surfaces. *The Visual Computer 4*, 2 (1988), 109–117.

[OS07] OAT C., SANDER P.: Ambient aperture lighting. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D)* (May 2007), pp. 61–64.

[RH01] RAMAMOORTHI R., HANRAHAN P.: An efficient representation for irradiance environment maps. In *Proc. of ACM SIGGRAPH* (2001), pp. 497–500.

[RSC87] REEVES W., SALESIN D., COOK R.: Rendering antialiased shadows with depth maps. vol. 21, pp. 283–291.

[RWS*06] REN Z., WANG R., SNYDER J., ZHOU K., LIU X., SUN B., SLOAN P., BOA H., PENG Q., GUO B.: Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. *ACM TOG 25*, 3 (2006), 977–986.

[SA07] SHANMUGAM P., ARIKAN O.: Hardware accelerated ambient occlusion techniques on GPUs. In *Proc. ACM Symposium on Interactive 3D Graphics and Games* (2007), pp. 73–80.

[SC00] SLOAN P., COHEN M.: Interactive horizon mapping. In *Eurographics Workshop on Rendering Techniques* (June 2000), pp. 281–286.

[SGNS07] SLOAN P., GOVINDARAJU N., NOWROUZEZAHRAI D., SNYDER J.: Image-based proxy accumulation for real-time soft global illumination. In *Pacific Graphics* (2007), pp. 97–105.

[SH05] SIGG C., HADWIGER M.: Fast third-order texture filtering. In *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation, Edited by M. Pharr*. Addison-Wesley, 2005.

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

[SKVW*92] SEGAL M., KOROBKIN C., VAN WIDENFELT R., FORAN J., HAEBERLI P.: Fast shadows and lighting effects using texture mapping. In *Proc. of SIGGRAPH* (1992), pp. 249–252.

[SLS05] SLOAN P., LUNA B., SNYDER J.: Local, deformable precomputed radiance transfer. *ACM TOG 24*, 3 (2005), 1216–1224.

[SN08] SNYDER J., NOWROUZEZAHRAI D.: *Fast Soft Shadows on Height Fields via Multi-Resolution Image Processing*. Tech. Rep. MSR-TR-2008-77, Microsoft Corporation, May 2008.

[Sny06] SNYDER J.: *Code Generation and Factoring for Fast Evaluation of Low-order Spherical Harmonic Products and Squares*. Tech. Rep. MSR-TR-2006-53, Microsoft Corporation, May 2006.

[Ste98] STEWART J.: Fast horizon computation at all points of a terrain with visibility and shading applications. *IEEE Transactions on Visualization and Computer Graphics 4*, 1 (March 1998), 82–93.

[Tat06] TATARCHUK N.: Practical parallax occlusion mapping with approximate soft shadows for detailed surface rendering. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses* (2006), pp. 81–112.

[TIS08] TEVS A., IRHKE I., SEIDEL H.: Maximum mipmaps for fast, accurate, and scalable dynamic height field rendering. In *Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games* (2008), pp. 183–190.

[Wil78] WILLIAMS L.: Casting curved shadows on curved surfaces. In *Proceedings of ACM SIGGRAPH 78* (1978), pp. 270–274.

[Wil83] WILLIAMS L.: Pyramidal parametrics. In *Proceedings of ACM SIGGRAPH 83* (1983), pp. 1–11.