Efficient Methods for Ambient Lighting

Tamás Umenhoffer
TU Budapest

Balázs Tóth
TU Budapest

László Szirmay-Kalos
TU Budapest

Abstract

This paper presents a model and algorithms for the reflection of the ambient light. Simplifying the rendering equation we derive an ambient transfer function that expresses the response of the surface and its neighborhood to ambient lighting, taking into account multiple reflection effects. The ambient transfer function is built on the obscurances of the point. If we make assumptions that the material properties are locally homogenous and incorporate a real-time obscurances algorithms, then the proposed ambient transfer can also be evaluated in real-time. Our model is physically based and thus can not only provide better results than empirical ambient occlusion techniques at the same cost, but also reveals where tradeoffs can be found between accuracy and efficiency.

1 Introduction

In computer graphics, materials are usually defined by their BRDFs that describe the optical response to point or directional illumination. However, in real life the indirect illumination caused by multiple reflections on distant surfaces acts more like a sky light source with roughly uniform intensity in different directions.

The optical response to unoccluded homogeneous illumination is the *albedo*, which can be obtained from the BRDF by integration. If occlusions can also happen, we should consider that the illumination cannot take effect in directions where the source is occluded by the same surface or other objects. Occlusion factors can be precomputed for static scenes or approximated on-line as proposed by real-time ambient occlusion methods. These methods play the role of shadow algorithms for ambient lighting. Both classical shadow mapping used for point and directional lights and ambient occlusion overdo their role in the sense that they completely eliminate lighting at surfaces that are not directly visible from the sources, making these surfaces darker than they should be. In case of point or directional lights, to compensate this darkening in an efficient way, the classical ambient lighting model is used. To extend the compensation for ambient occlusion as well, its definition should be generalized and made more similar to global illumination approaches.

According to the intuition that the albedo is the integrated BRDF and the ambient occlusion is the integrated visibility factor of shadowing algorithms, we can follow the line of the research on BRDF modeling and visibility computation. However, it is worth also considering techniques that directly attack the ambient lighting.

This paper focuses on the fast computation of the reflection of the ambient light. We shall assume that the primary source of illumination in the scene is a homogeneous sky light source of intensity L^a. Our approach is approximate but is physically based. That is, we derive our model from the rendering equation applying a series of approximations. This derivation has not only theoretical interest, but also shows the intimate relationship of the approximate and the physical models and opens new directions for other compromises between accuracy and the cost of evaluation. In this respect, our model is the first that finds a bridge between obscurances or ambient occlusion and the rendering equation.

2 Previous work

The oldest ambient lighting model assumes that ambient light L^a is constant at all points and directions, and a point reflects $k_a L^a$ intensity. Since this model ignores the geometry of the scene, the resulting images are plain and do not have a 3D appearance. A physically correct approach would be the solution of the rendering equation that can take into account all factors missing in the classical ambient lighting model. However, this approach is too expensive computationally when dynamic scenes need to be rendered in real-time.

Instead of working with the rendering equation, local approaches examine only a neighborhood of the shaded point. The *obscurances method* [Zhukov et al. 1998; Iones et al. 2003] computes just how "open" the scene is in the neighborhood of a point, and scales the ambient light accordingly. The method called *ambient occlusion* [Hayden 2002; Pharr and Green 2004; Kontkanen and Aila 2006] also approximates the solid angle and the average direction where the neighborhood is open, thus we can use environment map illumination instead of the ambient light. In the *spectral obscurances method* the average spectral reflectivities of the neighborhood and the whole scene are also taken into account, thus even *color bleeding* effects can be cheaply simulated [Mendez et al. 2005; Bunnel 2005].

The ambient occlusion and obscurances are usually off-line methods, which store the result in a texture. Applying simplifications, however, real-time generation is also possible. Since ambient occlusion is the "integrated local visibility", real-time ambient occlusion methods rely on scene representations where the visibility can be easily determined. These scene representations include the approximation of surfaces by disks [Bunnel 2005; Hoberock and Jia 2007] or spheres [Shanmugam and Arikan 2007]. Alternatively, a cube map or a depth map [Luft et al. 2006; Mittring 2007; Sainz 2008] rendered from the camera can also be considered as a sampled representation of the scene. Since these maps are already in the texture memory of the GPU, a fragment shader program can check the visibility for many directions. The method called *screen-space ambient occlusion* [Mittring 2007] took the difference of the depth values. Depth differences were obtained in tangent space and an efficient importance sampling scheme has been developed for screen-space ambient occlusion in [Tóth et al. 2009]. *Horizon split ambient occlusion* [Sainz 2008] generated and evaluated a horizon map [Max 1988] on the fly. A very recent work combined ambient occlusion with the one-bounce direct lighting and addressed the artifacts of screen-space ambient occlusion methods by rendering multiple depth layers or images [Ritschel et al. 2009].

Obscurances and ambient occlusion describe the openness or accessibility of a point with the size of that part of the directional hemisphere, which corresponds to occluded directions. Other definitions are also feasible, which can result in simpler and more efficient evaluations. For example, the occlusion can also be defined by that volume of a tangent sphere which is in the open region [Szirmay-Kalos et al. 2009].

Obscurances and ambient occlusion became popular also in rendering volumetric models [Rezk-Salama 2007; Hernell et al. 2007; Diaz et al. 2008; Ruiz et al. 2008; Ropinski et al. 2008].

Notation	Meaning
$\vec{x}$	surface point where the radiance is computed
$\vec{y}$	point on the occluder surface
d	distance between $\vec{x}$ and $\vec{y}$
R	maximum distance of considered occlusion
$\vec{\omega}$	direction from $\vec{x}$ to $\vec{y}$
r	parameter of the ray of origin $\vec{x}$ and direction $\vec{\omega}$
θ	angle between the surface normal at $\vec{x}$ and $\vec{\omega}$
Ω	illumination hemisphere
$L^r(\vec{x})$	reflected radiance
$L^{in}(\vec{x},\vec{\omega})$	incident radiance of point $\vec{x}$ from direction ω
L^a	ambient light intensity
$f_r(\vec{x})$	diffuse BRDF
$a(\vec{x})$	albedo
$O(\vec{x})$	obscurances
$W(\vec{x})$	ambient transfer function
$\mu(d)$	fuzzy membership of occluders at distance d
$\varepsilon(d)$	step function which is 1 if $d > 0$ and 0 otherwise

Table 1: Notations of the paper.

3 A general ambient illumination model

For the sake of simplicity, let us assume that the surfaces are diffuse. According to the rendering equation, the *reflected radiance* L^r in point $\vec{x}$ can be obtained as:

$$L^r(\vec{x}) = \int_\Omega L^{in}(\vec{x},\vec{\omega}) f_r(\vec{x}) \cos^+\theta \, d\omega, \qquad (1)$$

where $f_r(\vec{x})$ is the BRDF, and $\cos^+\theta$ is a geometric term. If incident angle θ is greater than 90 degrees, — i.e. the light illuminates the "back" of the surface — then the negative cosine value should be replaced by zero, which is indicated by superscript $^+$ in $\cos^+$. Due to occlusions or multiple scattering, *incident illumination* $L^{in}(\vec{x},\vec{\omega})$ at point $\vec{x}$ and direction $\vec{\omega}$ is not necessarily equal to ambient intensity L^a.

The integrand of the rendering equation depends on three factors, the incident illumination, the material properties, and the local geometry. In order to decompose the reflected radiance integral according to these factors, we rewrite the integral in the following form:

$$L^r(\vec{x}) = a(\vec{x}) \cdot W(\vec{x}) \cdot L^a, \qquad (2)$$

where

$$a(\vec{x}) = \int_\Omega f_r(\vec{x}) \cos^+\theta \, d\omega = f_r(\vec{x})\pi$$

is the *albedo* of the surface and $W(\vec{x})$ is the *ambient transfer function* of the ambient light:

$$W(\vec{x}) = \int_\Omega \frac{L^{in}(\vec{x},\vec{\omega})}{L^a} \frac{f_r(\vec{x})}{a(\vec{x})} \cos^+\theta \, d\omega = \frac{1}{\pi}\int_\Omega \frac{L^{in}(\vec{x},\vec{\omega})}{L^a} \cos^+\theta \, d\omega. \qquad (3)$$

Note that this factor is defined separately on each wavelength. If the ambient light intensity is zero at a wavelength, then this formula includes 0/0 factors. These factors should be replaced by 0 for the correct interpretation.

The ambient transfer function defined by equation 3 needs incident radiance L^{in} from direction $\vec{\omega}$. If point $\vec{y}$ is visible from $\vec{x}$ at direction $\vec{\omega}$, and the space is not filled with participating media, then the incident radiance is equal to the exiting radiance $L^r(\vec{y})$ of point $\vec{y}$. If no surface is seen, then shaded point $\vec{x}$ is said to be *open* in this direction, and the incident radiance is L^a.

However, this does not meet our intuition and everyday experience that the effect of far surfaces is replaced by their average. This experience is due to that the real space is not empty but is filled by participating media. If the space also contains homogeneous participating media, then the radiance along a ray of direction $\vec{\omega}$ changes according to the volumetric rendering equation:

$$\frac{dL(r,\vec{\omega})}{dr} = -\sigma_t L(r,\vec{\omega}) + \sigma_s \int_{\Omega'} L^{in}(r,\vec{\omega}')P(\vec{\omega}',\vec{\omega}) d\omega',$$

where r is the ray parameter, σ_t is the *extinction coefficient* representing the probability of photon–material collisions in a unit distance, σ_s is the *scattering coefficient*, which equals to the probability that collision happened and the photon is reflected, and phase function $P(\vec{\omega}',\vec{\omega})$ is the probability density of the reflection direction.

Assuming that incident radiance L^{in} can be different from the ambient intensity only at the surfaces, the integral of the in-scattering term can be simplified:

$$\int_{\Omega'} L^{in}(r,\vec{\omega}')P(\vec{\omega}',\vec{\omega}) d\omega' = \int_{\Omega'} L^a P(\vec{\omega}',\vec{\omega}) d\omega' = L^a.$$

Thus the volumetric rendering equation gets the following form:

$$\frac{dL(r,\vec{\omega})}{dr} = -\sigma_t L(r,\vec{\omega}) + \sigma_s L^a.$$

The solution of this differential equation can be given in closed form:

$$L(r,\vec{\omega}) = e^{-\sigma_t r} L(0,\vec{\omega}) + \frac{\sigma_s}{\sigma_t} L^a \left(1 - e^{-\sigma_t r}\right).$$

As we wish to have ambient intensity everywhere in an "empty" scene, we must choose non-absorbing media, i.e. $\sigma_t = \sigma_s$. Indeed, in this case the $L(0,\vec{\omega}) = L^a$ boundary condition results in solution $L(r,\vec{\omega}) = L^a$ everywhere.

Now let us consider surfaces that provide the boundary conditions for the volumetric rendering equation. If surface point $\vec{y}$ is visible at direction $\vec{\omega}$ with reflected radiance $L^r(\vec{y})$, then the incident radiance at point $\vec{x}$ being at distance d is

$$L^{in}(\vec{x},\vec{\omega}) = e^{-\sigma_t d} L^r(\vec{y}) + L^a \left(1 - e^{-\sigma_t d}\right).$$

Note that in this equation, factor $\mu(d) = 1 - e^{-\sigma_t d}$ and its complement $1 - \mu(d) = e^{-\sigma_t d}$ express the effects of the ambient lighting and of the occluder on shaded point $\vec{x}$, respectively. This effect of the occluder diminishes with the distance. Considering that if there is no participating media, in open directions the incident radiance is the ambient intensity and in closed directions the radiance of the occluder, function μ is a *fuzzy measure* that defines how strongly direction $\vec{\omega}$ belongs to the set of open directions based on distance d of the occlusion at this direction.

The exponential function derived from the physical analogy has a significant drawback [Iones et al. 2003]. As it is non-zero for arbitrarily large distances, very distant surfaces need to be considered that otherwise have negligible effect. Thus, for practical fuzzy measures we use functions that are non-negative, monotonically increasing from zero and reach 1 at finite distance R. This allows the consideration of only those occlusions that are nearby, i.e. closer than R. The particular value of R can be set by the application developer. When we increase this value, shadows due to ambient occlusions get larger and softer. For example, Mendez et al. [Mendez et al. 2005] proposed the following function

$$\mu(d) = \sqrt{\frac{d}{R}} \quad \text{if } d < R \text{ and 1 otherwise,}$$

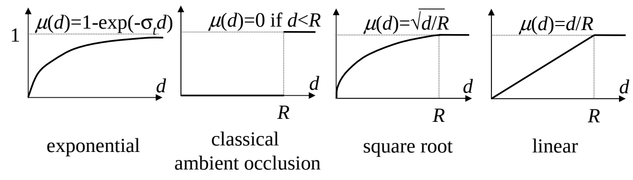

Figure 1: Example fuzzy membership functions. Classical ambient occlusion uses a non-fuzzy separation. The localized square root is a good compromise between the global exponential and the non-fuzzy separation.

but any other non-negative, monotonously increasing function can be used, which is 1 if d is sufficiently large, i.e. when d is greater than the radius R of the considered neighborhood (Figure 1).

With the fuzzy measure, the ambient transfer function can be written in the following form:

$$W(\vec{x}) = \frac{1}{\pi}\int_{\Omega} \mu(d(\vec{\omega}))\cos^{+}\theta \mathrm{d}\omega +$$

$$\frac{1}{\pi}\int_{\Omega}(1-\mu(d(\vec{\omega})))\frac{L^{r}(\vec{y}(\vec{\omega}))}{L^{a}}\cos^{+}\theta \mathrm{d}\omega. \qquad (4)$$

The first term is the *obscurances* value [Zhukov et al. 1998; Iones et al. 2003] of point $\vec{x}$:

$$O(\vec{x}) = \frac{1}{\pi}\int_{\Omega} \mu(d(\vec{\omega}))\cos^{+}\theta \mathrm{d}\omega. \qquad (5)$$

Ambient occlusion can also be obtained as a special case of obscurances setting the membership function to zero if the distance is smaller than R and 1 otherwise (Figure 1).

The second integral represents global illumination effects, i.e. additional light bounces, and contains the radiance of the occluder $L^{r}(\vec{y})$. According to equation 2, the reflected radiance at $\vec{y}$ can be expressed by the ambient transfer function as

$$L^{r}(\vec{y}) = a(\vec{y})\cdot W(\vec{y})\cdot L^{a}.$$

Thus, substituting this into equation 4 we obtain a recursive formulation for the ambient reflectivity:

$$W(\vec{x}) = O(\vec{x}) + \frac{1}{\pi}\int_{\Omega}(1-\mu(d(\vec{\omega})))a(\vec{y}(\vec{\omega}))W(\vec{y}(\vec{\omega}))\cos^{+}\theta \mathrm{d}\omega.$$

If the neighborhood is sufficiently small, then $W(\vec{y})$ is similar to $W(\vec{x})$, thus we can write:

$$W(\vec{x}) \approx O(\vec{x}) + \frac{1}{\pi}\int_{\Omega}(1-\mu(d(\vec{\omega})))a(\vec{y}(\vec{\omega}))W(\vec{x})\cos^{+}\theta \mathrm{d}\omega,$$

from which the ambient transfer function is

$$W(\vec{x}) \approx \frac{O(\vec{x})}{1-\frac{1}{\pi}\int_{\Omega}(1-\mu(d(\vec{\omega})))a(\vec{y}(\vec{\omega}))\cos^{+}\theta \mathrm{d}\omega} =$$

$$\frac{\int_{\Omega}\mu(d(\vec{\omega}))\cos^{+}\theta \mathrm{d}\omega}{\pi - \int_{\Omega}(1-\mu(d(\vec{\omega})))a(\vec{y}(\vec{\omega}))\cos^{+}\theta \mathrm{d}\omega}. \qquad (6)$$

Let us interpret this formula. If the albedo is small, then the ambient transfer function is similar to the obscurances value. However, if the albedo is close to 1, then the ambient transfer function gets also close to 1, reducing the darkening of obscurances. For example, the corners of white rooms will not be darker than the wall itself, which corresponds to our everyday observations. This phenomenon is well known by skiers. In cloudy and foggy weather, the scratches and bumps of the high albedo snow become invisible.

The ambient transfer function depends on two directional integrals, thus its efficient evaluation requires fast algorithms to estimate these integrals.

4 Computing the directional integrals

The evaluation of the directional integrals in equation 6 requires rays to be traced in many directions, which is rather costly. We transform this directional integral to replace the expensive ray tracing operation by a simple containment test, provided that neighborhood R is small enough to allow the assumption that the ray intersects the surface at most once in interval $[0,R]$. This assumption imposes restrictions on the surface curvature.

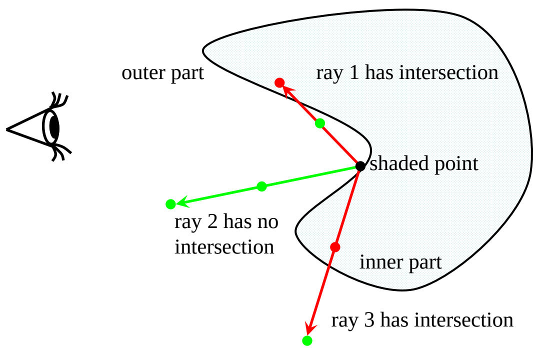

Figure 2: Replacing ray tracing by containment tests. If all test points along a ray at distance less than R from the shaded point is in the same region as the origin of the ray, then the ray has not intersected the surface. Instead of testing all points, we sample just a few points on the ray

To formalize the containment test, let us assume that the surfaces subdivide the space into an *outer part* where the camera is and to *inner parts* that cannot be reached from the camera without crossing the surface. We define *characteristic function* $\mathscr{I}_0(\vec{p})$ that is 1 if point $\vec{p}$ is an outer point and zero otherwise. The boundary between the inner and outer parts, i.e. the surfaces, belong to the inner part by definition (Figure 2).

Characteristic function $\mathscr{I}_0(\vec{p})$ is easy to define for height fields and displacement mapped surfaces (Figure 3). As proposed in the context of the screen-space ambient occlusion methods [Mittring 2007; Sainz 2008], the content of the z-buffer provides an inner–outer distinction since it can also be considered as a height field. If a point is reported to be occluded by the depth map, then surfaces separate this point from the eye, thus its characteristic value is zero. If the point would pass the depth test, then it is in the same region as the eye, so it gets value 1. In this case, the projection onto the surface would be the point that is represented by the pixel coordinates and the content of the z-buffer.

In order to evaluate the obscurances integral using containment tests, we express it as a three dimensional integral. For a point

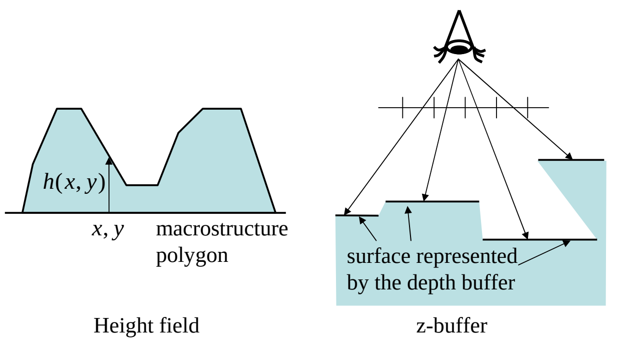

Figure 3: Two examples for the definition of the characteristic function. Outer regions where $\mathscr{I}_0 = 1$ are blank, inner regions where $\mathscr{I}_0 = 0$ are filled.

at distance d, fuzzy measure $\mu(d)$ can be found by integrating its derivative μ' from 0 to d since $\mu(0) = 0$. Then the integration domain can be extended from d to R by multiplying the integrand by a step function which replaces the derivative by zero when the distance is greater than d:

$$\mu(d) = \int_0^d \mu'(r)\mathrm{d}r = \int_0^R \mu'(r)\varepsilon(d-r)\mathrm{d}r,$$

where $\varepsilon(x)$ is the step function, which is 1 if $x \geq 0$ and zero otherwise. Similarly, for the complement of the fuzzy membership function, we can write:

$$1-\mu(d) = \int_d^R \mu'(r)\mathrm{d}r = \int_0^R \mu'(r)(1-\varepsilon(d-r))\mathrm{d}r.$$

Substituting these integrals into the ambient transfer function, we get

$$W(\vec{x}) \approx \frac{\int_0^R \int_\Omega \mu'(r)\varepsilon(d(\vec{\omega})-r)\cos^+\theta\mathrm{d}\omega\mathrm{d}r}{\pi - \int_0^R \int_\Omega \mu'(r)(1-\varepsilon(d(\vec{\omega})-r))a(\vec{y}(\vec{\omega}))\cos^+\theta\mathrm{d}\omega\mathrm{d}r}.$$

Let us consider a ray of equation $\vec{x}+\vec{\omega}r$ where shaded point $\vec{x}$ is the origin, $\vec{\omega}$ is the direction, and distance r is the ray parameter. Condition $\varepsilon(d(\vec{\omega})-r) = 1$ means that no intersection happened closer than r, which is equivalent to the condition that none of the points between the ray origin and $\vec{x}+\vec{\omega}r$ is in the inner region. Instead of checking all points on the ray, this condition is checked for a few, say m samples that are uniformly distributed between the ray origin and point $\vec{x}+\vec{\omega}r$. These uniformly distributed points are $\vec{x}+\vec{\omega}r/m, \vec{x}+\vec{\omega}r2/m, \ldots, \vec{x}+\vec{\omega}r$. As $\mathscr{I}_0(\vec{x}+\vec{\omega}ri/m)$ indicates that the ith point in the outer region, the condition that none of the points is in the inner region is indicated by the product of the point indicators. Thus we can approximate $\varepsilon(d(\vec{\omega})-r)$ as

$$\varepsilon(d(\vec{\omega})-r) \approx \mathscr{I}(\vec{x}+\vec{\omega}r) = \prod_{i=1}^{m} \mathscr{I}_0\left(\vec{x}+\vec{\omega}r\frac{i}{m}\right).$$

In practice the number of samples m along a ray is set to 1 or 2.

The albedo $a(\vec{y})$ at the hit point of the occluder surface is relevant only if occlusion happens, i.e. when $\varepsilon(d(\vec{\omega})-r) = \mathscr{I}(\vec{x}+\vec{\omega}r) =$ 0. According to the model, we would need that point $\vec{y}$ where $\mathscr{I}(\vec{x}+\vec{\omega}r)$ changes, but it is difficult to find and does not fit to the concept of replacing ray-intersection calculation by containment test. Thus we approximately replace $\vec{y}$ by the projection of that point which is first tested positively for occlusion onto the surface, which is denoted by $\tilde{a}(\vec{x}+\vec{\omega}r)$.

With the inner–outer characterization and the approximation of the albedo, the ambient transfer function is

$$W(\vec{x}) \approx \frac{\int_0^R \int_\Omega \mu'(r)\mathscr{I}(\vec{x}+\vec{\omega}r)\cos^+\theta\mathrm{d}\omega\mathrm{d}r}{\pi - \int_0^R \int_\Omega \mu'(r)(1-\mathscr{I}(\vec{x}+\vec{\omega}r))\tilde{a}(x+\vec{\omega}r)\cos^+\theta\mathrm{d}\omega\mathrm{d}r}. \tag{7}$$

4.1 Quasi-Monte Carlo integration

In order to provide the ambient transfer values in real-time, the integrals of equation 7 should be accurately estimated with just a few, carefully selected samples. We have two integrals, one in the enumerator, and the other in the denominator. Both of them are estimated using the same samples, thus their estimation errors will be correlated. As we compute their ratio, the division will well compensate the correlated error as suggested by *weighted importance sampling* [Powell and Swann 1966].

The careful sampling starts with a uniform series $(\xi_x, \xi_y, \xi_z)_i, i = 1 \ldots n$ in the unit cube, which is then transformed to mimic the integrand according to the concepts of *importance sampling*. We sampled the uniform series $(\xi_x, \xi_y, \xi_z)_i$ from the *Poisson disc distribution* due to its superior uniformness and blue-noise properties. Samples distributed with the Poisson disc distribution is obtained by iterative Lloyd-relaxation [Lloyd 1982].

The uniform series is transformed to directions and distances. Unfortunately, it is impossible to make the target density exactly proportional to the integrand, thus we aim at a sampling density that is equal to only two integrand factors

$$p(\vec{\omega}, r) = \mu'(r)\frac{\cos^+\theta}{\pi}.$$

We use the first two coordinates (ξ_x, ξ_y) to find the direction of cosine distribution:

$$\vec{\omega} = \vec{T}\cos(2\pi\xi_x)\sqrt{\xi_y} + \vec{B}\sin(2\pi\xi_x)\sqrt{\xi_y} + \vec{N}\sqrt{1-\xi_y},$$

where $\vec{T}$, $\vec{B}$, and $\vec{N}$, are the tangent, binormal, and normal vectors, respectively. Having found a cosine distributed point on the unit hemisphere, distance r is obtained with density $\mu'(r)$, transforming ξ_z as $r = \mu^{-1}(\xi_z)$.

Note that the generation of the Poisson disc distribution and the transformation of the sample points are rather expensive computationally, but in our case the number of samples n is small, and the series should be generated only once. These pre-computed values are hardwired into the program code, that is, we obtain n samples $(\vec{\omega}_i, r_i)$ with the discussed procedure, and values

$$(X_i, Y_i, Z_i) = \vec{\omega}_i r_i \tag{8}$$

are stored in a file or as constants in the program.

When starting the application, samples are passed to the fragment shader as a constant array. The fragment shader computes test points from these as

$$\vec{p}_i = \vec{x} + X_i\vec{T} + Y_i\vec{B} + Z_i\vec{N}$$

where $\vec{T}$, $\vec{B}$, and $\vec{N}$, are the tangent, binormal, and normal vectors, respectively. The test points are included into the quadrature of the ambient transfer function:

$$\boxed{W(\vec{x}) \approx \frac{\sum_{i=1}^{n} \mathscr{I}(\vec{p}_i)}{n - \sum_{i=1}^{n}(1-\mathscr{I}(\vec{p}_i))\tilde{a}(\vec{p}_i)}. \quad (9)}$$

This requires $m \cdot n$ containment tests for the generated points and averaging both in the numerator and denominator. It also makes sense to replace the albedo in this formula by the radiance stored in the pixel. This replacement would include the effect of the directional and point lights in the indirect illumination.

4.2 Noise reduction with interleaved sampling

The quasi-Monte Carlo quadrature has some error in each pixel, which depends on the particular samples used in the quadrature. If we used different quasi-random numbers in neighboring pixels, then dot noise would show up. Using the same quasi-random numbers in every pixel would make the error correlated and replace dot noise by "stripes". Unfortunately, both stripes and pixel noise are quite disturbing.

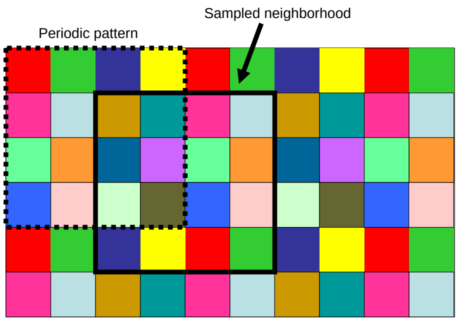

Figure 4: Interleaved sampling. In this figure colors represent quasi-random sequences. Note that an arbitrary 4×4 square includes all colors, thus all different quasi-random sequences are represented.

In order to reduce the error without taking excessive number of samples, we apply *interleaved sampling* [Keller and Heidrich 2001] that uses different sets of samples in the pixels of a 4×4 pixel pattern, and repeat the same sample structure periodically. The 16 different sample sets can be obtained from a single set by a rotation around the surface normal vector by random angle α. The rotation is executed in the fragment shader that gets 16 $(\cos\alpha, \sin\alpha)$ pairs in addition to quasi-random samples (x_i, y_i). The errors in the pixels of a 4×4 pixel pattern are uncorrelated, and can be successfully reduced by a low-pass filter of the same size. Thus, interleaved sampling using a 4×4 pixel pattern multiplies the effective sample number by 16 but has only the added cost of a box-filtering with a 4×4 pixel window (Figure 4).

Note that the assumption that low-pass filtering reduces the error is valid only if the same surface is visible in neighboring pixel. If other surfaces being at significantly different distances and in different illumination conditions occupy the neighboring pixels, then their ambient transfer function estimates should not be included into the current pixel. Thus, we also check whether or not the depth difference of the current and the neighbor pixels exceeds a given limit. If it does, then the neighbor pixel is not included in the averaging operation.

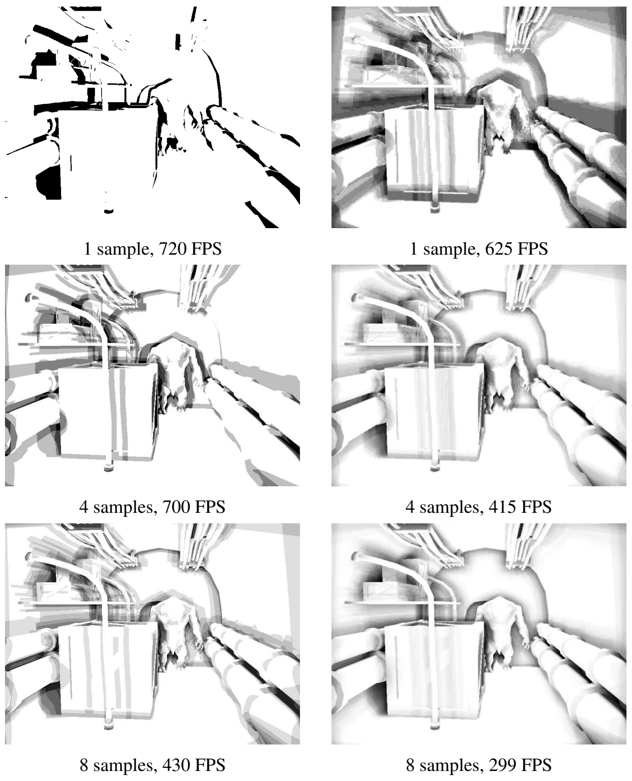

Figure 5: Comparison of the obscurance results without interleaved sampling (left) and with interleaved sampling (right).

4.3 Using the distance to the surface

The method of the previous section uses the average results of containment tests to approximate the ambient transfer integrals, that is, it examines whether a test point is in the inner or the outer part with respect to a surface. However, we have more information available that can be used to increase the accuracy of the approximation. Namely, in practical cases, not only are we able to classify a point as inner or outer, but we can also determine the distance of a test point and the surface along a given direction. In our particular methods, this direction will be parallel with axis z. When the content of the z-buffer defines the separation, the z-direction is the viewing direction in clipping space. Reading the depth value with the x, y coordinates of the test point provides the required threshold z^*. If the characteristic function is defined by height field $h(x,y)$, then $z^* = h(x,y)$ is the threshold.

A simple and straightforward application of this distance information is to eliminate false silhouette shadows showing up in depth map based methods (Figure 6). Suppose that a "foreground" object is in front of a background object and their distance is greater than R, and we inspect a pixel in which the background object is visible but its neighbors belong to the front object. When the ambient transfer value of the background is estimated, sample points in its neighborhood would quite probably fail the depth test, so the front object would darken this point, i.e. its silhouette would cast a fake shadow. However, since their distance is greater than R, this darkening should not happen. This problem can be solved by checking the difference of the stored depth value z^* and the depth z of the sample

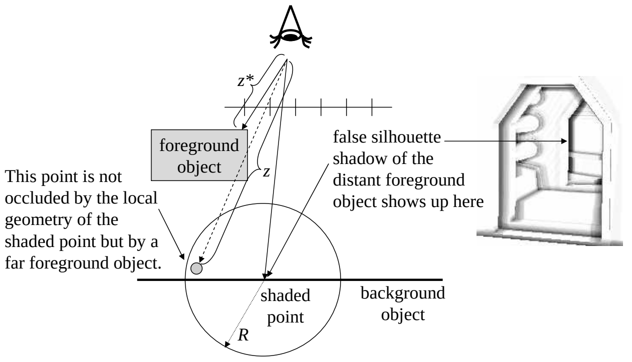
z*
foreground object
z
false silhouette shadow of the distant foreground object shows up here
This point is not occluded by the local geometry of the shaded point but by a far foreground object.
shaded point
background object
R

Figure 6: False silhouette shadow caused by a foreground object being far from the shaded surface. Note that the wall of the corridor is far from the door, so no shadow should be cast.

point, and reporting a point to be occluded only if $R > z - z^* > 0$ (Figure 4.3).

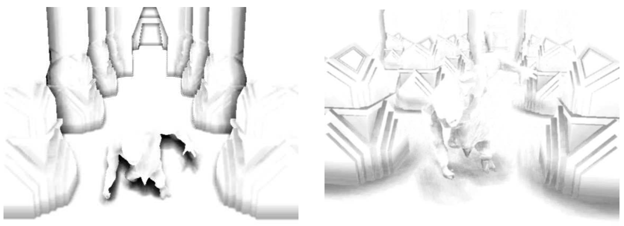

Figure 7: Images without (left) and with silhouette edge elimination (right).

5 Implementation

The run-time of the proposed ambient transfer function calculation is implemented as a post-processing pixel shader. The shader returns a float4 variable, where the "rgb" channels store the denominator of the ambient transfer function and the "a" channel stores the enumerator (equation 9). The calculation uses the depth buffer (depthMap) storing camera space z values and also the surface normal. The tangent and binormal vectors, and matrix `TBN` transforming from tangent space to camera space are reconstructed from the stored normal. The shader gets the current point's position in texture space (`wPos`) and in camera space (`cPos`). For interleaved sampling, the shader also uses pixel coordinates of the point. The random rotation is obtained from texture `noiseMap`, and is multiplied to the tangent to camera space transformation.

The offsets of equation 8 are obtained from constant array `XYZ`, and are used to shift the point in tangent space, which is then transformed to camera space (`sCPos`) and to texture space (`sSPos`), where depth `sDepth` is read from the depth buffer. Finally, equation 9 is evaluated according to the result of depth comparisons.

The following shader program uses just a single sample in each ray and does not include silhouette edge elimination, but implements interleaved sampling:

```
sampler2D depthMap;  // depth map
sampler2D noiseMap;  // for interleaved sampling
sampler2D albedoMap; // albedo of visible points

float4 AmbientTransfer(
   float2 wPos : TEXCOORD0, // texture space
   float3 cPos : TEXCOORD1, // camera space
   float4 vPos : VPOS       // screen space
   ) : COLOR {
   float depth = tex2D(depthMap, wPos).a;
   float3 T, B, N; // Determine tangent space
   N = tex2D(depthMap,wPos).xyz;
   T = normalize(cross(N,float3(0,1,0.01)));
   B = cross(N,T);
   float3x3 TBN = float3x3(T, B, N);

   // Interleaved sampling
   float3 r1 = tex2D(noiseMap, vPos.xy/4).xyz;
   float3 r3 = float3(0,0,1);
   float3 r2 = float3(r1.y, -r1.x, 0);
   TBN = mul(float3x3(r1,r2,r3), TBN);

   float4 W = float4(n, n, n, 0); // n = count
   for (int k = 0; k < n; k++) {
      // Transform samples to camera space
      float3 sCPos = cPos+mul(XYZ[k].xyz,TBN)*R;
      float4 sSPos = mul(sCPos, projTexMatrix);
      sSPos.xy = sSPos.xy / sSPos.w;
      float sDepth = tex2D(depthMap, sSPos.xy).a;
      // Compare sample depth with depth buffer
      if(sDepth >= sCPos.z - bias) W.a += 1;
      else W.rgb -= tex2D(albedoMap, sSPos.xy);
   }
   return W;
}
```

In case of interleaved sampling, this shading pass is followed by low pass filtering.

6 Results

The proposed methods have been implemented in DirectX/HLSL environment and their performance has been measured on an NVIDIA GeForce 8800 GTX GPU at 800×600 resolution.

Figure 6 compares images rendered with the proposed model to the result of only environment lighting. We used two samples for each ray. The scene consists of 415.000 polygons and is rendered at 180 FPS with constant shading and 100 FPS with the proposed ambient transfer calculation.

The proposed ambient transfer function W (equation 6) is compared to environment lighting and using obscurances O (equation 5) in Figure 7. The average albedo of the texture of the carved object is $(0.6, 0.3, 0.3)$ on the wavelengths of primary colors. Note that the unrealistic darkening of the obscurances is eliminated by the application of the new ambient transfer function.

Here we checked occlusions using the displacement map of the surface rather than the z-buffer. In fact, we can also combine the two techniques. The z-buffer based obscurances are responsible for occlusions of other objects, while the height field based obscurances handle self-shadowing. Note that this way the obscurances value can be obtained for displacement mapped surfaces even if the depth value is not modified in the fragment shader, and the accuracy problems of the z-buffer are also eliminated.

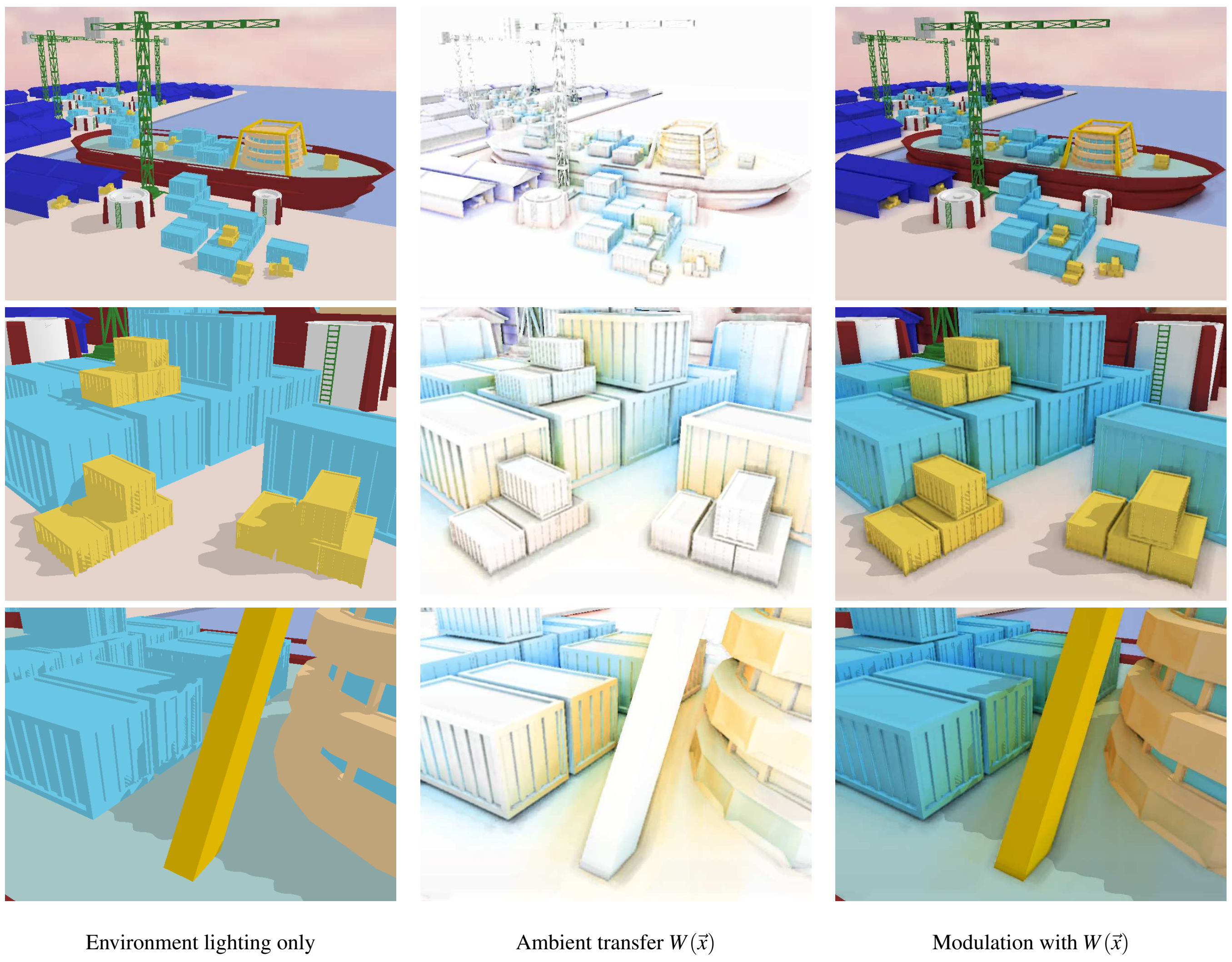
Environment lighting only
Ambient transfer $W(\vec{x})$
Modulation with $W(\vec{x})$

Figure 8: A harbor rendered with environment lighting only and with the proposed ambient transfer function (100 FPS at 800×600 resolution on an NV8800 GPU).

7 Conclusions

This paper proposed a general ambient illumination model that mimics not only local occlusions but also multiple scattering around the shaded point and thus reduces the excessive darkening of obscurances or ambient occlusion models. All these effects are summarized in the proposed ambient transfer function. We also considered a fast method for the computation of the ambient transfer function, which replaces ray-tracing by containment tests done on height fields or on the z-buffer. Finally, we discussed further improvements such as the application of interleaved sampling and the elimination of false silhouette shadows.

Acknowledgement

This work has been supported by the National Office for Research and Technology and by OTKA K-719922 (Hungary).

References

BUNNEL, M. 2005. Dynamic ambient occlusion and indirect lighting. In *GPU Gems 2*, M. Parr, Ed. Addison-Wesley, 223–233.

DIAZ, J., YELA, H., AND VÁZQUEZ, P. 2008. Vicinity occlusion maps: Enhanced depth perception of volumetric models. In *Computer Graphics International*.

HAYDEN, L. 2002. Production-ready global illumination. Tech. rep., SIGGRAPH Course notes 16. http://www.renderman.org/RMR/Books/ sig02.course16.pdf.gz.

HERNELL, F., YNNERMAN, A., AND LJUNG, P. 2007. Efficient ambient and emissive tissue illumination using local occlusion in multiresolution volume rendering. In *Eurographics/IEEE-VGTC Symposium on Volume Graphics*, 1–8.

HOBEROCK, J., AND JIA, Y. 2007. High-quality ambient occlusion. In *GPU Gems 3*, H. Nguyen, Ed. Addison-Wesley, 257–274.

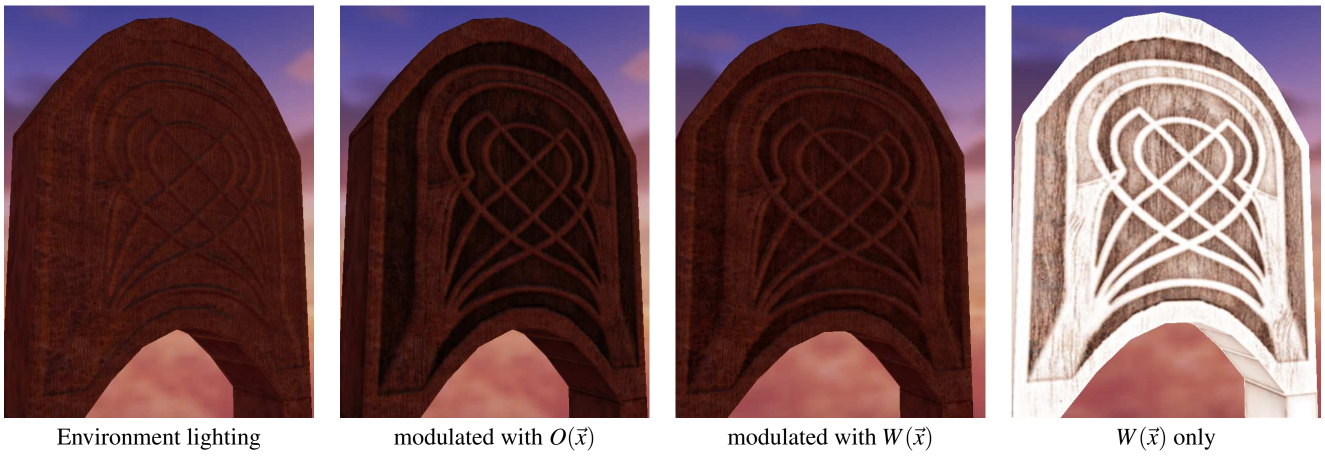

Figure 9: The effects of modulating the environment lighting with the obscurances, $O(\vec{x})$, and with the proposed ambient transfer function, $W(\vec{x})$. The ambient transfer function is also shown on the right.

IONES, A., KRUPKIN, A., SBERT, M., AND ZHUKOV, S. 2003. Fast realistic lighting for video games. *IEEE Computer Graphics and Applications 23*, 3, 54–64.

KELLER, A., AND HEIDRICH, W. 2001. Interleaved sampling. In *Rendering Techniques 2001 (Proceedings of the 12th Eurographics Workshop on Rendering)*, 269–276.

KONTKANEN, J., AND AILA, T. 2006. Ambient occlusion for animated characters. In *Proceedings of the 2006 Eurographics Symposium on Rendering*.

LLOYD, S. 1982. Least square quantization in pcm. *IEEE Transactions on Information Theory 28*, 129–137.

LUFT, T., COLDITZ, C., AND DEUSSEN, O. 2006. Image enhancement by unsharp masking the depth buffer. *ACM Trans. Graph. 25*, 3, 1206–1213.

MAX, N. 1988. Horizon mapping: Shadows for bump-mapped surfaces. *The Visual Computer 4*, 2, 109–117.

MENDEZ, A., SBERT, M., CATA, J., SUNYER, N., AND FUNTANE, S. 2005. Real-time obscurances with color bleeding. In *ShaderX 4: Advanced Rendering Techniques*, W. Engel, Ed. Charles River Media.

MITTRING, M. 2007. Finding next gen — CryEngine 2. In *Advanced Real-Time Rendering in 3D Graphics and Games Course - Siggraph 2007*. 97–121.

PHARR, M., AND GREEN, S. 2004. Ambient occlusion. In *GPU Gems*. Addison-Wesley, 279–292.

POWELL, M., AND SWANN, J. 1966. Weighted importance sampling — a Monte-Carlo technique for reducing variance. *Inst. Maths. Applics. 2*, 228–236.

REZK-SALAMA, C. 2007. Production-Ready GPU-based Monte-Carlo Volume Raycasting. Tech. rep.

RITSCHEL, T., GROSCH, T., AND SEIDEL, H.-P. 2009. Approximating dynamic global illumination in image space. In *Proceedings ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D)*.

ROPINSKI, T., MEYER-SPRADOW, J., DIEPENBROCK, S., MENSMANN, J., AND HINRICHS, K. H. 2008. Interactive volume rendering with dynamic ambient occlusion and color bleeding. *Computer Graphics Forum (Eurographics 2008) 27*, 2, 567–576.

RUIZ, M., BOADA, I., VIOLA, I., BRUCKNER, S., FEIXAS, M., AND SBERT, M. 2008. Obscurance-based volume rendering framework. In *Proceedings of Volume Graphics 2008*.

SAINZ, M. 2008. Real-time depth buffer based ambient occlusion. In *Games Developers Conference '08*.

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

SZIRMAY-KALOS, L., UMENHOFFER, T., TÓTH, B., SZÉCSI, L., AND SBERT, M. 2009. Volumetric ambient occlusion. *IEEE Computer Graphics and Applications*.

TÓTH, B., UMENHOFFER, T., AND SZIRMAY-KALOS, L. 2009. Efficient post-processing with importance sampling. In *ShaderX 7*, W. Engel, Ed. Charles River Media.

ZHUKOV, S., IONES, A., AND KRONIN, G. 1998. An ambient light illumination model. In *Proceedings of the Eurographics Rendering Workshop*, 45–56.