

Combining light animation with obscurances for glossy environments

Àlex Méndez-Feliu
amendez@ima.udg.es

Mateu Sbert
mateu@ima.udg.es

Institut d'Informàtica i Aplicacions - Universitat de Girona
Campus Montilivi E17071 - Girona (SPAIN)
Tel. (+34) 972 41 84 19 Fax. (+34) 972 41 82 59

Obscurances is a powerful technique that approximates indirect global illumination in a much faster way than classic methods as radiosity or path tracing. In this method, the local environment of a point or a patch is sampled to estimate the amount of occlusion that surrounds it. Fine effects as color reflection can be added and visually pleasant rendering of corners and soft shadows are easily computed. Combined with ray tracing techniques, non-diffuse and specular effects can be added to the surfaces, obtaining thus beautiful and realistic rendered images. Since direct and indirect illumination are decoupled, obscurances are very useful to render animation frames for moving light sources, as they have to be computed only for the first frame and then reused. Ambient light intensity can also be decoupled from obscurances and estimated for every frame, resulting in beautiful color-changing effects. Light animation help in light design and to convey shape and features of the objects in the scene.

Keywords: Computer Graphics, Three-Dimensional Graphics and realism, Raytracing, Rendering, Light animation

Introduction

Obscurances [1, 2] is an ambient term technique where occlusions are taken into account, although only a local environment is queried. The amount of occlusion will determinate the diffuse effects. This technique, introduced in videogames as a cheap alternative to radiosity, has been improved with color bleeding effects and allows a real-time update of the diffuse-like illumination when objects move in the scene [3]. A simplified version of obscurances is used in well known commercial renderers such as *Pixar's Photorealistic RenderMan* under the name of *ambient occlusion* [4, 5, 6].

Global illumination effects are very costly to simulate. Sophisticated techniques like bidirectional path tracing [7, 8], Metropolis lighting [9] and photon map [10] have allowed to reduce significantly the cost of obtaining a noiseless image but they are still computationally very expensive.

Obscurances, although it is a not a global illumination technique, is able to obtain at a low cost high quality realistic looking images that simulate a global illumination solution. Direct illumination and glossy effects are computed in the usual way [11], and diffuse effects are computed using the obscurances techniques. The overall effect is a nice looking realistic image.

The technique presented in this paper applies obscurances to rendering of animation frames with moving light sources. We take advantage of one of the properties of the obscurances, the decoupling of direct and indirect illumination. Thus, we just need to compute obscurances once in the first frame and reuse them in the rest of the animation. A different ambient light intensity is estimated for every frame, resulting in beautiful color-changing effects.

This paper is organized in the following way. Obscurances technique is explained in detail in the next section. It includes the obscurances basic idea, how color bleeding can be added, and the combination of obscurances with ray-tracing to render images of glossy scenes. In the next section the use of obscurances in combination with light animation is explained. Then, the use of different ambient intensities depending on the light animation is introduced. Results and final animations are analyzed and finally, in the last section we present our conclusions and give some directions for future research.

Obscurances

In [1, 2] the obscurances illumination model was defined. Obscurances take account of secondary diffuse illumination, being totally decoupled from direct illumination. Indirect illumination for point P is defined as:

$$I(P) = \frac{1}{\pi} R(P) I_A \int_{\omega \in \Omega} \rho(d(P, \omega)) \cos \theta d\omega \quad (1)$$

where

- $\rho(d(P, \omega))$: function with values between 0 and 1, and giving the magnitude of ambient light incoming from direction ω
- $d(P, \omega)$: distance from P to the first intersected point in direction ω
- θ : angle between direction ω and the normal at P
- I_A : ambient light intensity
- $R(P)$: Reflectivity at P
- $1/\pi$ is the normalization factor such that if $\rho(\cdot) = 1$ over the whole hemisphere Ω then $I(P)$ is $R \times I_A$

Direct illumination is added to (1) to obtain the final illumination at the point. Function $\rho(\cdot)$ increases with d . Its shape is given in figure 1. A maximum distance for interaction, d_{max} is defined, so that when $d \geq d_{max}$ then $\rho(d) = 1$. This means that we only take into account a d_{max} -neighborhood of P . Obscurance of P is then defined as:

$$W(P) = \frac{1}{\pi} \int_{\omega \in \Omega} \rho(d(P, \omega)) \cos \theta d\omega \quad (2)$$

Clearly $0 \leq W(P) \leq 1$. Obscurance for a patch is defined as the average of obscurances for all points in it. An obscurance value of 1 means that the patch is totally open (not occluded with neighbour polygons), while a value of 0 means that it is totally closed (or occluded with neighbor polygons).

Ambient light in (1) is computed with the formula:

$$I_A = \frac{R_{ave}}{1 - R_{ave}} \frac{\sum_{i=1}^n A_i E_i}{A_T} \quad (3)$$

where

$$R_{ave} = \frac{\sum_{i=1}^n A_i R_i}{A_T} \quad (4)$$

and A_i , E_i , R_i are the area, emissivity and reflectivity of patch i , A_T the sum of the areas, and n the number of patches in the scene. The ambient term considered here corresponds to the secondary illumination only, as direct illumination is computed apart.

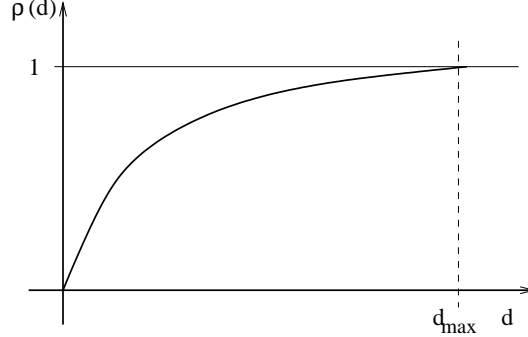


Figure 1: Shape of $\rho(d)$ function.

Real-time Obscurances with color bleeding

Computation of obscurances is very fast as it has only to consider a neighborhood of a point (or a patch). In [3] a Monte Carlo implementation with real-time update of moving objects was presented. Also in [3] color bleeding was incorporated. To do it, the reflectivity of the point Q seen from P in direction ω is incorporated to the obscurances equation:

$$W(P) = \frac{1}{\pi} \int_{\omega \in \Omega} R(Q) \rho(d(P, \omega)) \cos \theta d\omega \quad (5)$$

When no surface is hit at a distance less than d_{max} in direction ω the obscurance takes the value of R_{ave} . We can consider then that the average reflectivity factor is included in the obscurances, thus the ambient light computation has to be modified by taking out the average reflectivity factor in (3):

$$I_A = \frac{1}{1 - R_{ave}} \frac{\sum_{i=1}^n A_i E_i}{A_T} \quad (6)$$

Extension to glossy environments: Ray-traced obscurances

The extension of the technique to deal with general environments is done in the context of ray-tracing [12]. In the formulas (4,5,6) the diffuse reflectivity is substituted by the mean albedo.

The algorithm is as follows. A ray is traced in the usual way from the view point through a pixel, hitting a point P of some surface in the scene. Then from P , three kind of rays can be traced:

- **Shadow rays:** N_{dl} direct illumination rays are traced to the light sources from the hit points. A random point of a random light source of the scene is selected stochastically and a shadow ray is traced from the hit point to the selected point. If both points see each other, contribution of the light source to the hit point is computed. Figure 2.c shows this contribution.
- **Obscurance rays:** N_{obs} rays are stochastically traced with a $\cos \theta$ distribution to compute the obscurance of point P , this is, we solve by Monte Carlo the integral in (5). In this way we compute an estimator of indirect light for this point

by multiplying the obscurance term by the diffuse reflectivity at the point and the ambient intensity. Figure 2.a shows the obscurances contribution, that multiplied by the reflectivity color of the objects (fig. 2.b) results in the indirect light contribution (fig. 2.d).

- **Specular rays:** When the hit point corresponds to a glossy or specular surface a path is followed according to usual path-tracing. This is shown in fig. 2.e.

The final illumination is then given by adding the direct lighting, specular effect and diffuse effect represented by the obscurances multiplied by ambient light. Following the kitchen example, we finally get the image in fig. 2.f.

The resulting images are compared to path tracing (PT) in fig. 3, where much more noise can be appreciated in PT than in obscurances, even when PT images take almost 10 times more to compute than the obscurances ones.

Animation of light sources

Instead of computing every frame from scratch, we can take advantage of the fact that neither the obscurances nor the hit points have to be recomputed. This is possible if the camera and all objects in the scene are fixed, and only lights move from frame to frame. Thus we compute first the obscurances and store the hit points and incoming directions of the eye rays. Then, for each frame direct illumination and specular effects are computed using shadow and specular rays using the stored hit points and directions. This has the side effect of increasing frame-to-frame coherence, eliminating flickering.

Back to fig. 2, images (a) and (b) are the same for all frames. As we will see in next section, to compute indirect light in image (d), ambient light factor can be estimated independently for every frame. Thus, only direct light (c), specular effects (e), and an ambient light factor, have to be recomputed when light sources move or change intensity with respect to the next frame.

Discussion on the ambient intensity and average reflectivity terms

Equation (3) describes how ambient intensity is computed, as a draft approximation to what actual ambient intensity could be. It uses equation (4) for average reflectivity. In both equations total area of the scene is taken into account. It means that we make the assumption that the light energy gets distributed around the scene and illuminates with the same intensity not only every object but every part of every object, and that secondary lighting occur in the same way. When light emitters are positioned in the center of the scene and illuminate most of it, the ambient intensity given by equation (3) is a good estimation.

But in most scenes these assumptions do not apply. Normally, a noticeable part of the surfaces of the scene are hidden to the light, i. e., totally occluded by other objects, for example, the objects inside a cupboard, the inner part of an opaque vase and everything it could contain, etc. As we can see, normally, more than half the surfaces of the scene should not be taken into account in that *total area* term. On the other hand, if light sources illuminate with an important portion of the total power directly to colored objects, previously computed average reflectivity from the scene would be biased with respect to the actual one.

To solve this problem, I_A and R_{ave} can be more accurately estimated by shooting a few rays from the light sources beforehand, and follow their paths while computing statistically the parameters we need. In this way ambient intensity, average reflectivity and the area reached by the light are easily computed. We will compute ambient intensity for every frame, and use it to multiply the obscurances factor and diffuse color to obtain indirect lighting. If light sources power changes among frames or light movement causes changes in occlusion conditions, ambient intensity will also change and result in beautiful effects in the animation. In fig. 4 these effects can be appreciated.



(a) Obscuration image.



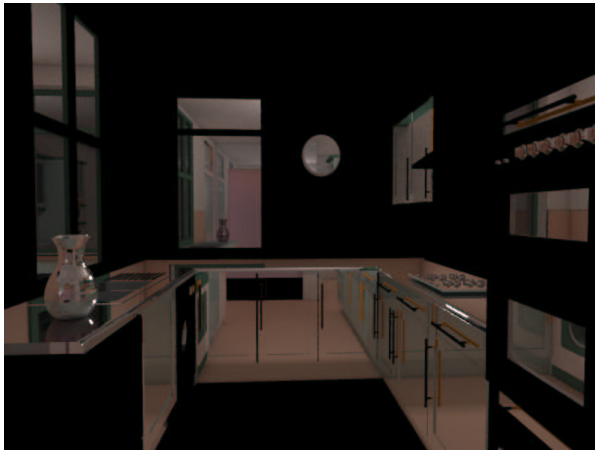
(b) Diffuse reflectivity.



(c) Direct light.



(d)=(a)x(b) Indirect light.



(e) Specular surfaces.



(f)=(c)+(d)+(e) Final image.

Figure 2: These six images show the contribution that each kind of rays of our rendering algorithm give to the final image and how it is computed. The first image (a) shows our main contribution: the obscuration. The second (b) image shows the diffuse color of each object in the scene. By multiplying both images and the ambient light we get indirect light image (d). Direct light is shown in (c). Adding the contribution of the specular surfaces (e) to (c) and (d) we get the final image (f). All the images of the kitchen scene shown here have a resolution of 800x600 pixels.

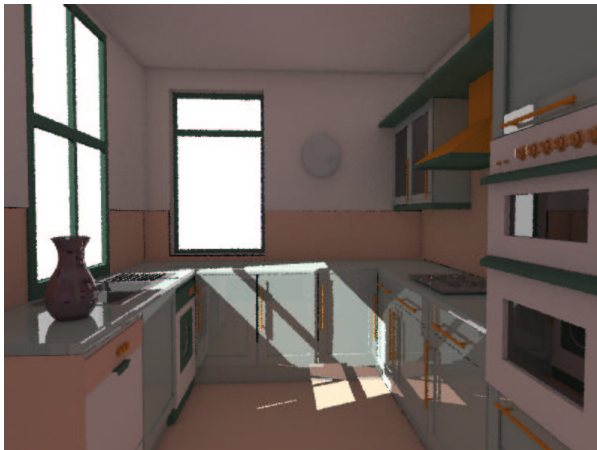


(a.) Path tracing. 6510 sec. rendering.



(b.) Ray traced obscurances. 655 sec. rendering.

Figure 3: The kitchen with daylight illumination compared with path tracing results. Image resolution is 800 x 600 pixels.

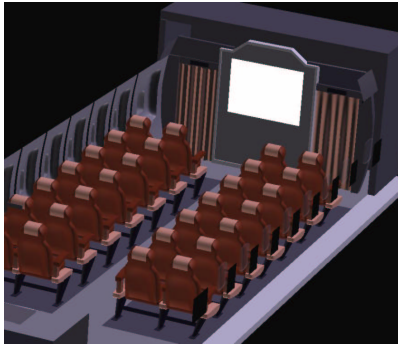


(a) Greenish ambient term.

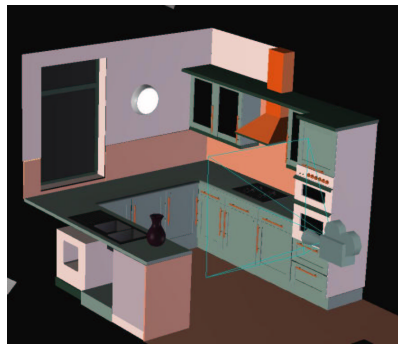


(b) Reddish ambient term.

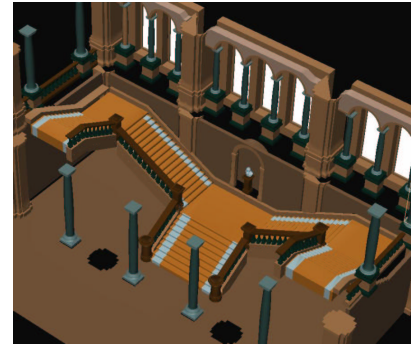
Figure 4: These two images show the difference in ambient intensity for indirect illumination when two different directional light sources are applied. Image (a) shows greenish ambient intensity and in (b) ambient intensity is reddish.



(a) Aircraft model.



(b) Kitchen model.



(c) Stairs model.

Figure 5: The three different models used.

Results

The ray-traced obscurances with light animation have been implemented in the SIR system [13]. Three types of light sources have been implemented: point, directional and area light sources, but only point and directional have been animated, because area light sources are always associated to an object of the scene, and moving objects are not treated yet in this context. Point and directional lights are animated using classic keyframes.

Three movies can be downloaded from http://ima.udg.es/~amendez/TIC2001/gal_obsanim.html, where three different models are used to show our results. One represents the inner part of an aircraft, including seats, windows, a screen, etc. It is composed of 184687 triangles featuring 461 objects and it is shown in fig. 5a. The corresponding movie presents a 2 seconds animation (48 frames) where the point light runs along the aircraft cabin from the camera position to the tail. Obscurances computation take 462 seconds, and 484 seconds the rest of the animation, what makes an average of 10.1 seconds per frame (19.7 seconds per frame if we take account of obscurances).

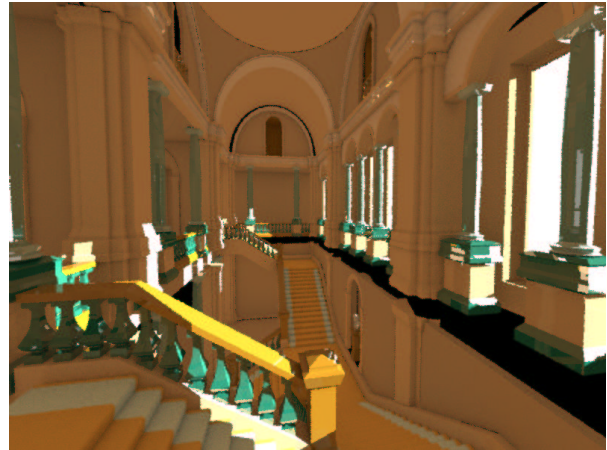
The kitchen scene is composed of 28937 triangles featuring 203 objects (fig. 5b). The movie presents a 10 seconds animation (240 frames) in which directional light follows a direction from the windows given by the two polar angles of the corresponding keyframe, thus simulating the movement of the sunlight. Different ambient term intensities for each frame are clearly appreciated. On the one hand, we get a brighter image when more light gets into the room, which we take directly proportional to the cosinus of the angle between the window and the light ray. On the other hand the ambient intensity color is directly related to the colors of the objects onto which the light rays hit. Obscurances computation take 264 seconds, and 1960 seconds the rest of the animation, what makes an average of 8.16 seconds per frame (9.27 seconds per frame if we take account of obscurances).

The third movie follows the same idea than the previous one. The model represents a set of stairs, banisters, columns, etc., that could be the main entrance of a castle or a theater. It has 121032 triangles featuring 580 objects and is shown in fig. 5c. In the animation we see the directional light from the seven windows moving in a sun-like way. It takes 144 frames (6 seconds). Obscurances computation take 290 seconds, and 1490 seconds the rest of the animation, what makes an average of 10.35 seconds per frame (12.36 seconds per frame if we take account of obscurances).

All images have a resolution of 800×600 pixels and have been rendered on a Pentium 4 1.6 Ghz with 1 Gb RAM. We have used in the obscurances computation 40 rays per pixel.



(a) Aircraft point light animation.



(b) Stairs directional light animation.

Figure 6: One frame from each animation of the aircraft and the stairs. The aircraft animation presents point light animation. The stairs movie shows the sun-like movement of directional lights.

Conclusions and future work

We have presented in this paper the combination of the obscurances technique with a moving light source. Obscurances are a fast technique that used with ray tracing allows to obtain realistic looking images. It is partly non physically realistic, as the indirect part of the illumination is computed with deep improvement over the ambient illumination. This technique is much faster than ray-traced global illumination techniques, such as path tracing, and resulting images present much less noise. Since direct and indirect illumination are decoupled, animation of source lights can be added with no extra computation time for obscurances. For indirect illumination, only the ambient light term is recomputed for every frame by shooting a few rays from the light sources, and follow their paths while computing statistically the parameters we need. In this way ambient intensity, average reflectivity or the area reached by the light can be easily computed.

As future work we plan to explore the advantages of obscurances with other kinds of animation. In particular, moving the camera in a walkthrough over the scene, and taking advantage of the slowly varying obscurances over large coherent regions, recomputation of indirect lighting can also be sped up. We can use a method similar to Ward and Heckbert's *irradiance gradients* [14] for obscurance gradients. Also for moving objects, and due to the locality of the obscurance computation, animation can be computed faster. Acceleration for graphics hardware will also be considered as in [6].

The technique presented here can be used as a fast editing tool or as the final image. Light animation can help in light design and to convey shape and features of the objects in the scene.

Acknowledgements

This project has been funded with grant number TIC2001-2416-C03 from the Spanish government.

The kitchen, aircraft and stairs models are courtesy of LightWork Design Ltd. Thanks to professor László Neumann for his colorful ideas.

References

- [1] Zhukov, S., Iones, A., Kronin, G.: “An Ambient Light Illumination Model”, *Rendering Techniques '98*, Springer-Wien, New York (*Proc. of Eurographics Rendering Workshop '98* - Vienna, Austria), pp. 45-55
- [2] Iones, A., Krupkin A., Sbert M. and Zhukov, S., “Fast realistic lighting for videogames”. *IEEE Computer Graphics & Applications*, may-june 2003.
- [3] Méndez, À., Sbert, M., Catà, J., “Real-time Obscurances with Color Bleeding”, *Proceedings of the Spring Conference on Computer Graphics 2003*, Budmerice, Slovak Republic, 2003.
- [4] Landis, H., “Production-ready global illumination”, *SIGGRAPH 2002 course notes #16* (RenderMan in Production), pages 87-102, ACM, July 2002.
- [5] Christensen, P. H., “Global illumination and all that”, *SIGGRAPH 2003 course notes #9* (RenderMan: Theory and Practice), pages 31-72, ACM, July 2003.
- [6] Pharr, M., Green, S., “Ambient Occlusion”, *GPU Gems*, Addison-Wesley, 2004
- [7] Lafortune, E. P., Willems, Y. D., “Bi-directional Path Tracing”, *Compugraphics '93 Conference Proceedings*, pp.145–153, Alvor, Portugal
- [8] Veach, E., Guibas, L. J., “Bidirectional estimators for light transport”, *SIGGRAPH 95 Conference Proceedings*, pp.419–428,1995.
- [9] Veach, E., Guibas, L. J., “Metropolis Light Transport”, *SIGGRAPH 97 Conference Proceedings*, pp.65–76, 1997.
- [10] Jensen, H. W., Christensen, N. J.: “Photon Maps in Bidirectional Monte Carlo Ray Tracing of Complex Objects”, *Computers and Graphics* 19(2), pp, 215-224, 1995
- [11] Cook, R. L., Porter, T., and Carpenter, L., “Distributed ray tracing”. *Computer Graphics*, 18(4):165-174, July 1984. ACM Siggraph '84 Conference Proceedings.
- [12] Méndez, À., Sbert, M. and Neumann, L., “Obscurances for Ray-tracing”, *EUROGRAPHICS 2003 Poster Presentation*, Granada, Spain.
- [13] Martín, I., Pérez, F., Pueyo, X., “The SIR rendering architecture”, *Computers & Graphics*, 22(5):601-609, 1998.
- [14] Ward, G., Heckbert, P., “Irradiance gradients” in *Proceedings of the Third Eurographics Workshop on Rendering*, Bristol, UK, pp. 85–98, May 1992.