Screen Space Ambient Occlusion

Louis Bavoil
lbavoil@nvidia.com

Miguel Sainz
msainz@nvidia.com

September 2008

Document Change History

Version	Date	Responsible	Reason for Change
1.0	August 13, 2008	Louis Bavoil	Initial release
1.1	September 1, 2008	Louis Bavoil	Added note about ShaderX 7

This technique will appear in the upcoming book ShaderX 7, edited by Wolfgang Engel. For more information, please visit www.shaderx7.com.

Overview

Ambient occlusion is a lighting model that approximates the amount of light reaching a point on a diffuse surface based on its directly visible occluders. It is commonly used to add a global illumination look to rendered images. This ambient occlusion algorithm operates on the depth buffer of the scene being rendered and associated per-pixel normals. For multiple directions defined in image space around the current pixel, the algorithm samples the depth buffer, computes a horizon angle for each sample, and integrates the ambient occlusion for the current direction based on the tangent plane, the horizon angle and a weighting function. This whitepaper includes a brief introduction to the algorithm and its parameters. For more details, please refer to [Bavoil and Sainz 08a] and [Bavoil and Sainz 08b].

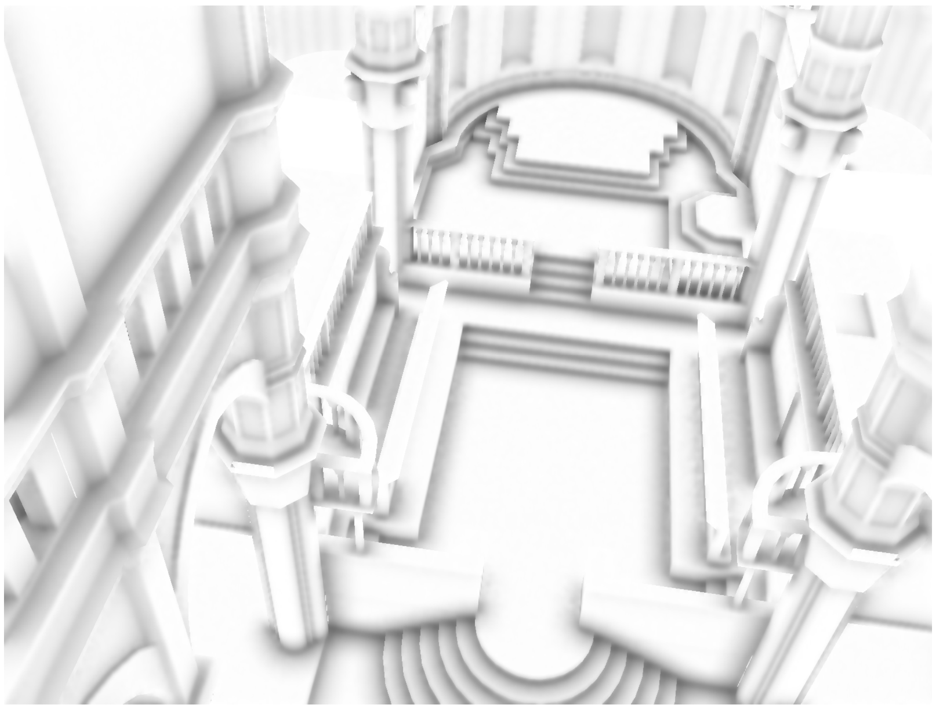

Figure 1. Screen Space Ambient Occlusion (SSAO) using our horizon-based algorithm with per-pixel normals. Image rendered in 1600x1200 at 91 fps on GeForce GTX 280, with half-resolution AO, 8x8 depth samples per AO pixel, and a 15x15 cross bilateral filter.

NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050
www.nvidia.com

Introduction

Ambient occlusion can be defined as the soft shadow generated by a sphere light of uniform intensity surrounding the scene. For outdoor scenes, it is sometimes referred to as sky light, in which case the sphere light encompasses the whole scene geometry at infinity, like a white environment map [Landis 02] [Christensen 03]. For general scenes, ambient occlusion is defined as the integral of the occlusion contributed from inside a hemisphere Ω of a given radius R, centered at the current surface point P and oriented towards the normal **n** at **P** (Figure 2), as described in [Akenine-Moller et al. 08].

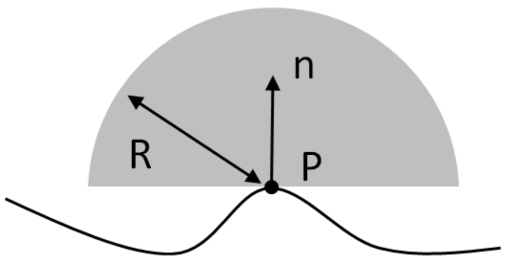

Figure 2. Hemisphere Ω around a surface point P.

Surface elements inside Ω cast ambient occlusion to point P, and surface elements outside Ω are ignored. To avoid discontinuities in the rendered ambient occlusion, we weight the contributions of the surface elements S by an attenuation function W(r /R), where r is the distance between S and P. This attenuation function is referred to as "falloff" in Mental Ray and Gelato [Gritz 06].

Our Approach

Similarly to [Mittring 07], [Shanmugam and Orikan 07] and [Fox and Compton 08], we use the depth buffer of the scene being rendered as an approximation of the scene geometry. We look at the depth buffer as a heightfield surface. Given this heightfield, one way to render ambient occlusion is to trace rays in image space through the normal-oriented hemisphere Ω at each surface point P. We refer to this approach as ray marching. Although the heightfield is limited to one depth layer inside the view frustum, screen space ambient occlusion tends to produce perceptually realistic results. However, tracing rays in 3D is wasteful since the input data is a 2D heightfield.

Instead of tracing rays in eye space (3D) and projecting them back to image-space (2D), we trace rays directly in 2D and approximate the final ambient occlusion from these 2D rays. We do this by using spherical coordinates with the zenith axis oriented parallel to the Z axis in eye space. Our algorithm is based on the concept of horizon like in horizon mapping [Max 86] and horizon-split ambient occlusion [Dimitrov et al 08]. A key difference with [Dimitrov et al 08] is that our algorithm steps in image space, not in eye space. For each direction θ around P in the heightfield, we compute incremental horizon angles as we step forward along a line, and integrate the ambient occlusion based on the horizon angles, the angle with the tangent plane, and the attenuation function W. For more details on the algorithm, please refer to [Bavoil and Sainz 08a] and [Bavoil and Sainz 08b].

Running the Sample

The sample presents to the user a set of controls on the right side of the window. The top section of these controls is common to the two ambient occlusion techniques included in this sample. The user can select:

- MSAA level: This allows changing the number of samples per pixel from 1 to 16. The "8x MSAA" mode uses 4 color samples per pixel and 8 coverage samples (CSAA). The "8xQ MSAA" mode uses 8 color samples per pixel and 8 coverage samples. The "16x MSAA" mode uses 4 color samples and 16 coverage samples, and the "16xQ MSAA" mode 8 color samples and 16 coverage samples.
- Model selection: This allows selecting from different available models to evaluate the quality of the AO approach in different scenarios.
- Technique: This drop box allows switching between the horizon-based ambient occlusion engine (default), and the ray-marching engine. The R key reloads all the shaders.

Additionally, the following global controls are presented to the user as checkboxes:

- When the "Half-Res AO" check box is enabled, the ambient occlusion is computed with a half-resolution viewport, and the blur passes are still performed in full resolution.
- When "Show Diffuse" is checked, shaded green colors are computed during the geometry pass, otherwise a uniform white color is used.
- The "Show AO" check box toggles on and off the computation of the ambient occlusion.

The user can control the view of the scene by using the mouse or the keyboard. The left mouse button rotates the camera and the middle button (wheel) zooms in and out. The A/D, W/S and Q/E keys translate the model along the X, Y and Z axis respectively.

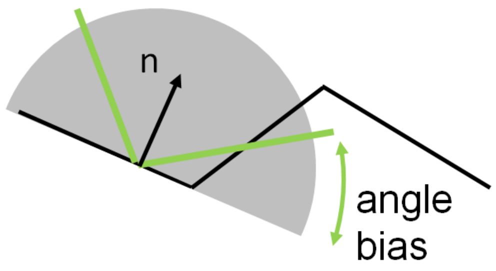

Figure 3. A low-tessellated surface (side view). Occluders below the angle bias from the tangent plane are ignored.

Horizon-Based Ambient Occlusion

The horizon-based engine has the following parameters:

- **Radius scale.** This is a scale factor for the radius R. The radius is the distance outside which occluders are ignored, as shown on Figure 2.
- **Angle bias.** For low-tessellated geometry, occlusion variations tend to appear at creases and ridges, which betray the underlying tessellation. To remove these artifacts, we use an angle bias parameter which restricts the hemisphere, as shown on Figure 3.
- **Number of directions.** This is the number of randomly-rotated 2D directions in image space distributed around the current pixel. The higher this parameter, the lower is the noise in the ambient occlusion.
- **Number of steps.** This is the maximum number samples per direction. The actual number depends on the projected size of the sphere of radius R around the current surface point.
- **Attenuation.** This scale factor W_0 is applied to the per-sample attenuation function. The occlusion contribution of a given sample is attenuated by $W_0 * W(r/ R)$ where $W(x) = 1 – x^2$.
- **Contrast.** This value allows to scales up the ambient occlusion values.

The following 3 modes are available in the GUI:

- **Low-quality mode.** This mode does not use normals. The ambient occlusion is integrated inside the whole sphere of radius R around P and the occlusion contribution from half of the sphere is subtracted to cancel out the occlusion below the normal.
- **Normal-quality mode.** This mode uses the per-pixel normal **n** and integrates the ambient occlusion inside the hemisphere of radius R oriented towards **n**.
- **High-quality mode.** This mode uses normals as well, but computes horizon angles more accurately than the normal-quality mode. This mode removes rare false shadowing artifacts and reduces banding artifacts which appear when using small step sizes (small radii and/or large number of steps per direction).

Ray-Marched Ambient Occlusion

The radius of influence, attenuation factor and contrast have the same behaviors as in the horizon-based technique. The following parameters are different:

- **Number of directions.** This is the number of directions distributed in eye-space around the normal.
- **Number of rays/dir.** For each direction around the normal, multiple rays are ray-marched, each with a different elevation angle in spherical coordinates.
- **Number of steps.** This is the number of samples per ray. The step size is defined by subdividing the radius R into Ns segments of uniform lengths.

Cross Bilateral Filter

A cross bilateral filter [Eisemann and Durand 04] [Petschnigg et al. 04] is a depth-dependent Gaussian blur which diffuses the ambient occlusion in image space, while avoiding blurring across edges. Our implementation has the following parameters:

- **Blur size.** The kernel width, in pixels. Note that we blur separately in the X and Y dimensions, although bilateral filters are not theoretically separable.
- **Blur sharpness.** This parameter controls the depth-dependent weight of the bilateral filter, to avoid bleeding across edges. A zero sharpness is a pure Gaussian blur. Increasing the blur sharpness removes bleeding by using lower weights for samples with large depth delta from the current pixel.
- **Edge threshold.** When MSAA is enabled, we perform an edge detection pass over the R32_FLOAT texture and we perform a supersampled blur at the edge pixels only, to reduce halo artifacts. This parameter controls the edge threshold for the Sobel filter.

Performance

The performance of the technique is a function of the resolution of the final image, the use of the "half-resolution AO" option, the blur size and the MSAA mode which makes the blur more expensive. In addition, the performance is also slightly view dependent since it uses dynamic branching in its inner loop, as well as a variable number of steps per pixel.

When using half-resolution AO, it is possible to double the number of directions, increasing the visual quality without compromising the performance. Below are frames per second for a given view of the Sibenik cathedral similar to Figure 1, rendered in 1600x1200 on GeForce GTX 280 for 6x6 and 8x8, 12x8, 16x8 depth samples per pixel. (The first number indicates the number of directions and the second number indicates the number of steps per direction.)

1600x1200 4xAA	6x6	8x8	8x8 NoBlur
Full-Res AO	31	24	28
Half-Res AO	69	62	109

1600x1200 NoAA	6x6	8x8	8x8 NoBlur
Full-Res AO	41	29	31
Half-Res AO	135	112	158

1600x1200 NoAA	12x6	16x8	16x8 NoBlur
Full-Res AO	28	20	21
Half-Res AO	105	79	99

Code Overview

The code is organized with one C++ class and shader (.fx) file per rendering relevant pass:

- **Scene3D.** Draws the scene geometry and writes shaded colors, eye-space depths and eye-space normals with Multiple Render Targets with respective formats R8G8B8A8_UNORM, R32_FLOAT and R8G8B8A8_SNORM. To make sure that the normals are orthogonal to the underlying geometry, the normals are computed from the eye-space position in the pixel shader, not using interpolated per-vertex normals.
- **ResolveEngine.** If MultiSample AntiAliasing (MSAA) is enabled, resolve the MSAA colors using ResolveSubresource and resolve the MSAA depths and normals using a fullscreen shader pass which outputs an arbitrary sample per pixel.
- **HorizonBasedAOEngine.** Takes as input the non-multisample R32_FLOAT depth texture the non-multisample R8G8B8A8_SNORM texture, and outputs an ambient occlusion value per pixel into a R8_UNORM texture.
- **RayMarchingAOEngine**. A brute-force ray-marching algorithm which can be used as a reference solution to compare the horizon-based ambient occlusion with.
- **BilateralBlurEngine.** Blurs the ambient occlusion texture using a separated cross bilateral filter. It blurs the texture in the X direction first, writing to an intermediate R8_UNORM texture, and blurs this intermediate texture in the Y direction, writing to the original ambient occlusion texture. These two blur passes are sourcing the R32_FLOAT eye-space depth texture to avoid bleeding across edges.

Integration Notes

The technique takes as input a depth texture and optionally a normal texture. To generate these, we recommend using Multiple Render Targets (MRT) for the draw calls where you draw opaque objects. Alternatively, if you have are using depth pre-pass for all your opaque geometry, you can also generate the MRT depths and normals in the depth pre-pass. The ambient occlusion fullscreen pass should be inserted you render the opaque geometry, and before you render blended objects such as semi-transparent particles. We recommend doing the ambient occlusion pass in half-resolution and the blur passes in full resolution.

References

[Akenine-Moller et al. 08] Tomas Akenine-Möller, Eric Haines, Naty Hoffman, "Real-Time Rendering - Third Edition", 2008.

[Bavoil and Sainz 08a] Louis Bavoil, Miguel Sainz, "Image-Space Horizon-Based Ambient Occlusion", SIGGRAPH 2008, Talk Program, August 2008.

[Bavoil and Sainz 08b] Louis Bavoil, Miguel Sainz, "Image-Space Horizon-Based Ambient Occlusion", ShaderX7 - Advanced Rendering Techniques (to appear).

[Christensen 03] Christensen, P. H. 2003. "Global illumination and all that". In ACM SIGGRAPH Course 9.

[Dimitrov et al 08] Rouslan Dimitrov, Louis Bavoil, Miguel Sainz, "Horizon-Split Ambient Occlusion". In I3D '08: Symposium on Interactive 3D Graphics and Games 2008 (poster).

[Eisemann and Durand 04] Elmar Eisemann and Frédo Durand, "Flash Photography Enhancement via Intrinsic Relighting", In Proceedings of SIGGRAPH 2004.

[Fox and Compton 08] Megan Fox and Stuart Compton, "Ambient Occlusive Crease Shading", Game Developer Magazine, March 2008.

[Gritz 06] Larry Gritz, "Gelato 2.1 Technical Reference", NVIDIA, 2006.

[Landis 02] Landis, 2002. "Production-Ready Global Illumination," In ACM SIGGRAPH Course #16.

[Max 86] Nelson Max, "Horizon mapping: Shadows for bump-mapped surfaces." In Proceedings of Computer Graphics Tokyo '86 on Advanced Computer Graphics.

[Mittring 07] Martin Mittring, "Finding Next Gen: Cryengine 2." In SIGGRAPH '07: ACM SIGGRAPH 2007 courses.

[Petschnigg et al. 04] Georg Petschnigg, Richard Szeliski, Maneesh Agrawala, Michael F. Cohen, Hugues Hoppe, Kentaro Toyama: Digital photography with flash and no-flash image pairs, In Proceedings of SIGGRAPH 2004.

[Shanmugam and Orikan 07] Shanmugam, P. and Orikan O., "Hardware accelerated ambient occlusion techniques on GPUs". In I3D '07: Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games.