Horizon-Split Ambient Occlusion

Rouslan Dimitrov Louis Bavoil Miguel Sainz

NVIDIA Corporation

Figure 1: (a) Horizon occlusion contribution (b) Normal occlusion contribution (c) Final occlusion.

1 Introduction

Ambient occlusion is a technique that computes the amount of light reaching a point on a diffuse surface based on its directly visible occluders. It gives perceptual clues of depth, curvature, and spatial proximity and thus is important for realistic rendering. The ambient occlusion illumination at a surface point $\mathbf{P}$ with surface normal $\vec{n}$ can be defined as

$$A = \frac{1}{\pi}\int_{\Omega}(1 - V(\vec{\omega}))(\vec{n}\cdot\vec{\omega})d\omega \quad (1)$$

where V is the light visibility function over the normal-oriented unit hemisphere Ω, which returns 1 if a ray starting from $\mathbf{P}$ in direction $\vec{\omega}$ intersects an occluder and 0 otherwise.

Like [Mittring 2007] and [Shanmugam and Arikan 2007], we compute the ambient occlusion as a postprocessing pass based on a depth buffer from the eye's point of view. This approach requires no scene-dependent precomputations.

2 Horizon-Split Ambient Occlusion

Similarly to horizon mapping [Max 1986; Sloan and Cohen 2000], our approach splits the unit hemisphere into two parts by a horizon line defined by the angle $h(\theta)$ (see Figure 2). Rays that would normally be traced below the horizon are known to intersect an occluder so the intersection test for these rays can be skipped.

The ambient occlusion illumination A can be expressed in terms of the occlusion as:

$$A = 1 - \frac{1}{\pi}\int_{\theta=0}^{2\pi}\left(T(\theta) + \int_{z=sin(h(\theta))}^{1} V(\vec{\omega})(\vec{n}\cdot\vec{\omega})d\omega\right)d\theta \quad (2)$$

where $T(\theta) = sin(h(\theta))^2$ is the horizon occlusion in direction θ. Thus for a given θ, the horizon occlusion term does not require any sampling of the hemisphere. This enables focusing the sampling to a subset of the hemisphere, while still capturing the occlusion contribution of the non-sampled part. Figure 1 shows the contribution of the horizon and normal components.

3 Our Image-Space Algorithm

Our algorithm takes as input per-pixel linear depths and eye-space normals. For each pixel, we compute an eye-space position of its corresponding surface point $\mathbf{P}$ and we integrate the ambient occlusion terms from Equation 2. We pick N_d random directions θ distributed around the normal $\vec{n}$ at point $\mathbf{P}$. For each angle θ, the horizon angle $h(\theta)$ is computed by sampling the depth buffer. We start with a ray $\mathbf{h} = (\mathbf{P}, \vec{\mathbf{t}})$ where $\vec{t}$ is the tangent at $\mathbf{P}$ in the direction θ,

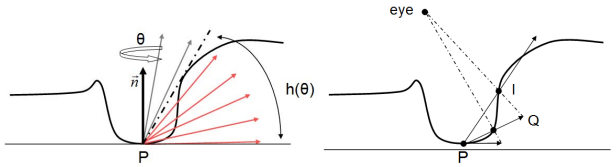

Figure 2: (a) The red rays are below the horizon so they are known to be occluded. The gray rays may or may not be occluded and need to be traced. (b) Finding the horizon angle $h(\theta)$ seen from $\mathbf{P}$ within a certain radius of influence R around $\mathbf{P}$.

and we incrementally extend the length of this ray N_s times. At each step, the end point $\mathbf{Q}$ of the ray $\mathbf{h}$ is projected in screen space and its depth is compared with the depth of the corresponding point in the depth buffer. If the later is closer ($\mathbf{Q}$ is below the surface seen from the eye), $||\mathbf{P}-\mathbf{Q}|| < R$, and point $\mathbf{Q}$ does not cross over the normal, then $\mathbf{h}$ is deflected towards the intersection point $\mathbf{I}$ and its direction is normalized and rescaled. (See Figure 2). Then, the horizon occlusion $T(\theta)$ is computed and additional normal rays are optionally traced in image space in the part of the hemisphere above $h(\theta)$ to complete the sampling of the hemisphere. The algorithm is applied to every surface point $\mathbf{P}$ visible on the screen. Figure 3 shows a comparison of our algorithm with object-space ray tracing.

Figure 3: (a) Our algorithm with $N_d = 10$ directions, $N_s = 6$ steps per direction, $N_n = 2$ normals rays, (b) $N_d = 13$, $N_s = 8$, $N_n = 4$, (c) ray tracing (Mental Ray).

References

MAX, N. L. 1986. Horizon mapping: Shadows for bump-mapped surfaces. In *Proceedings of Computer Graphics Tokyo '86 on Advanced Computer Graphics*.

MITTRING, M. 2007. Finding next gen: Cryengine 2. In *SIGGRAPH '07: ACM SIGGRAPH 2007 courses*.

SHANMUGAM, P., AND ARIKAN, O. 2007. Hardware accelerated ambient occlusion techniques on gpus. In *I3D '07: Proceedings of the 2007 symposium on Interactive 3D graphics and games*.

SLOAN, P.-P. J., AND COHEN, M. F. 2000. Interactive horizon mapping. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*.

I3D 2008, Redwood City, California, February 15–17, 2008.
ACM 978-1-59593-983-8/08/0002