The Alchemy Screen-Space Ambient Obscurance Algorithm

Morgan McGuire*
NVIDIA & Williams College

Brian Osman
Vicarious Visions

Michael Bukowski
Vicarious Visions

Padraic Hennessy
Vicarious Visions

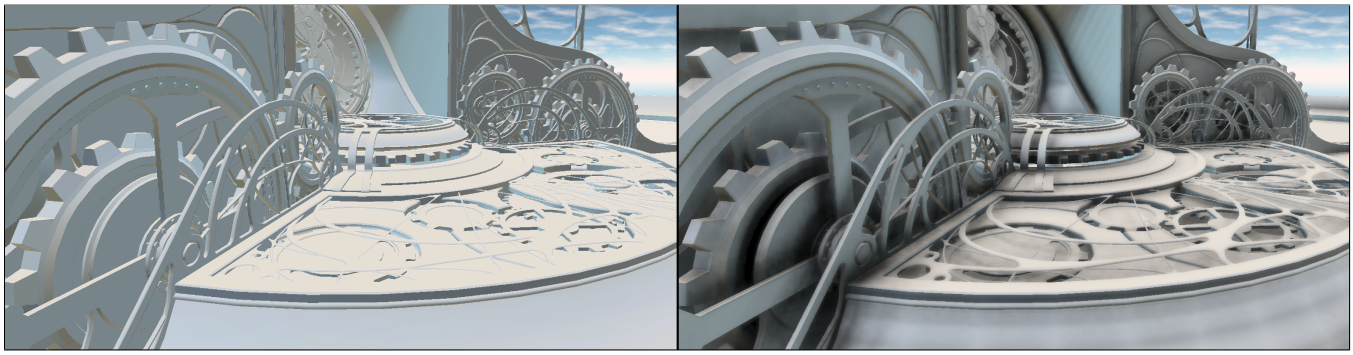

Figure 1: left: Environment lighting. right) Modulated by Alchemy ambient obscurance, computed from 12 samples per pixel at 1280×720 in 3 ms on GeForce 580. The algorithm is easy to tune, robust, and captures darkening at many scales and orientations.

Abstract

Ambient obscurance (AO) produces perceptually important illumination effects such as darkened corners, cracks, and wrinkles; proximity darkening; and contact shadows. We present the AO algorithm from the Alchemy engine used at Vicarious Visions in commercial games. It is based on a new derivation of screen-space obscurance for robustness, and the insight that a falloff function can cancel terms in a visibility integral to favor efficient operations. Alchemy creates contact shadows that conform to surfaces, captures obscurance from geometry of varying scale, and provides four intuitive appearance parameters: world-space radius and bias, and aesthetic intensity and contrast.

The algorithm estimates obscurance at a pixel from sample points read from depth and normal buffers. It processes dynamic scenes at HD 720p resolution in about 4.5 ms on Xbox 360 and 3 ms on NVIDIA GeForce 580.

CR Categories: I.3.3 [Picture/Image Generation]: Display Algorithms; I.3.7 [Three-Dimensional Graphics and Realism]: Color, shading, shadowing, and texture

Keywords: ambient occlusion, ambient obscurance, screen space

1 Introduction

Indirect illumination is a significant factor in realistic lighting. Every game approximates indirect light sparsely over large distances, either via precomputation (e.g., [Larsson 2010]; environment maps and constant ambient are the classic examples) or dynamic generation (e.g., [Kaplanyan and Dachsbacher 2010; Martin and Einarsson 2010]). Those sparse methods miss occlusion on the small, sub-meter scale. *Ambient obscurance* (AO) is an illumination term that corrects the indirect light by scaling it proportional each point's visible *obscurance* on that scale. A point that is locally obscured from most directions should receive little indirect illumination from distant objects, while highly accessible points receive most indirect illumination. Obscurance is visually important for object definition, to provide a sense of scale, and as a spatial cue through contact shadows and darkening on concave surfaces. It is also computationally intense to estimate directly from scene geometry–any point may be obscured from any direction. This is why screen space approximations, which are independent of the number of polygons, have proven very attractive in practice.

This paper presents the screen space AO algorithm we developed for a specific *Guitar Hero* title and subsequently generalized and integrated into the cross-platform Alchemy game engine. Figure 1 demonstrates its visual impact. The left image shows a scene with environment lighting only. The image on the right modulates that lighting by Alchemy AO, which resolves the fine details and spatial relationships between objects. The algorithm follows from three insights: Derive a robust estimator from the rendering equation; provide temporal coherence by making the estimator efficient enough to evaluate many times per pixel; and achieve that efficiency by shaping the falloff function to cancel expensive operations. Alchemy addresses the drawbacks of previous screen-space AO methods, none of which satisfy all of the following requirements:

1. **Robust**: Conform obscurance to affected surfaces (e.g., no shadows "floating in air" near silhouettes), limit viewer dependence of intensity, and maximize temporal coherence.
2. **Multiscale**: Capture phenomena at multiple scales: shadowed deep pits, corner darkening, contact shadows, wrinkles.
3. **Artist-control**: Provide intuitive parameters with large sweet-spots and predictable quality.
4. **Scalable**: Compute in 3-5 ms, from Xbox 360 to Windows Direct3D 11 hardware by varying quality.

Like all screen-space methods, its limitations are sample variance (addressed by edge-aware filtering) and under-obscurance due to unseen occluders behind the depth buffer surface and outside the field of view. Rendering a guard band about the viewport can reduce the latter. We attribute the visual fidelity and robustness of

*e-mail: morgan@cs.williams.edu,{bosman,mbukowski,phennessy}@vvisions.com

Alchemy AO compared to phenomenological methods to its radiometric derivation, however it is not physically correct. The value of the algorithm is that it makes practical concessions to artistic and performance goals.

1.1 Related Work

Screen space Luft et al. [2006] observed that placing blurry black lines at depth discontinuities in screen space by unsharp masking increases visual contrast similarly to ambient occlusion. A line of efficient screen-space AO methods rapidly followed, some published and some orally presented based on production use. These developed efficient GPU sampling [Shanmugam and Arikan 2007; Mittring 2007; Kajalin 2009] and reconstruction [Filion and McNaughton 2008] methods, increasingly physically-based derivations [Bavoil and Sainz 2009a; Szirmay-Kalos et al. 2010; Loos and Sloan 2010], and multipass methods for addressing both performance and quality issues [Ritschel et al. 2009; Bavoil and Sainz 2009b]. Our technique builds on all of this work but revisits the core derivation, which we focus on in section 2. We report experimental results comparing Alchemy to the two most recent methods in section 3. Briefly, Volumetric Obscurance [Loos and Sloan 2010; Ownby et al. 2010] estimates occlusion as proportional to the volumetric occlusion of a point, which tends to produce black halos. Alchemy uses projected solid-angle occlusion, like Horizon-Based AO [Bavoil and Sainz 2009a], but uses a more efficient sampling scheme and avoids over-darkening by conservatively assuming that unseen space is empty, like VO does. The low-level sampling techniques of all methods are based on earlier work done at Crytek [Mittring 2007; Kajalin 2009] and we incorporate filtering and aesthetic falloff methods from StarCraft II [Filion and McNaughton 2008].

Geometric Computing AO from geometry avoids the viewer-dependent errors from screen-space approximations, at the cost of performance for complex scenes.. We briefly describe geometric methods of interest for real-time rendering, i.e., within a factor of $20\times$ the performance of screen-space methods. See *Real Time Rendering* [Akenine-Möller et al. 2008] for a broad survey.

Oat and Sander [2007] precomputed an accessible cone about each texel on a static mesh, which they then applied to dynamic illumination. Reinbothe et al. [2009] dynamically voxelized polygonal scenes and sampled occlusion in voxel space. Bunnell [2005] computed occlusion on a GPU in multiple passes with per-vertex proxy geometry.

Ambient Occlusion Fields [Kontkanen and Laine 2005] are precomputed voxel occlusion values relative to a static mesh, which can be composed between moving rigid objects to estimate AO. McGuire [2010] and Laine and Karras [2010] extended this idea to dynamic meshes.

2 Algorithm

We begin by summarizing the key terms and equations governing indirect illumination. Previous screen space methods approximate these equations (or their resultant phenomena) in ways that are extremely efficient but can introduce significant error. We instead retain the full model and observe that the falloff function customarily chosen for aesthetics can also be selected to eliminate expensive operations. With this, we are able to retain the full radiometric model until the screen-space sampling step. That process leads to a relatively robust solution. We augment the screen-space solution with parameters for managing buffer precision and artistic goals in production use.

```
1. Render an early−Z pass
2. Render the camera−space normal and position buffers
3. Render the ambient obscurance buffer
4. [Render other effects]
5. Cross−bilateral filter obscurance
6. Composite obscurance, texture, and lighting in a forward pass
```

Listing 1: Rendering system context for obscurance.

Listing 1 describes the AO algorithm in the context of a larger multi-pass rendering system. Our algorithm comprises obscurance pass 3 and filtering pass 5. Each of these may be implemented either by rendering a full-screen rectangle with a shader that reads from the position and normal buffers, or as a forward pass over the scene geometry.

Environment light, shadows, and direct illumination are computed in pass 6, for forward lighting, or in step 4 for deferred shading. The final compositing pass uses multisample antialiasing (MSAA) to apply screen-resolution obscurance with subpixel edge precision. It gathers obscurance from five pixels in a cross pattern and cross-bilateral filters them based on camera-space proximity to the surface being shaded. A deferred shading pass may be substituted for pass 6 at the loss of antialiasing.

2.1 Background

Let the *distance-limited radiance function* $L_x(C, \hat{\omega})$ denote radiance incident at point C from a unit direction $\hat{\omega}$, *due to points within the ball* of radius x about C. Let the *visibility function* $V(C, P)$ be 1 if there is no intersection between the open-ended line segment $\overline{CP}$ and the scene, and 0 otherwise. The net incident radiance function is the sum of *near field* radiance and *far field* radiance modulated by visibility [Arikan et al. 2005; McGuire 2010] for some given radius r,

$$L_\infty(C, \hat{\omega}) = L_r(C, \hat{\omega}) + V(C, C + r\hat{\omega})L_\infty(C + r\hat{\omega}, \hat{\omega}). \quad (1)$$

This decomposition is useful for combining estimates of near and far field illumination from different algorithms. It is common in real-time rendering to choose radius r less than one meter and separate the far-field $V \cdot L_\infty$ term into shadowed *direct illumination* and *ambient-occluded environment lighting* terms. It is also common to ignore small-scale global illumination by assuming $L_r \approx 0$, although recent screen-space gathering methods offer more accurate alternatives [Ritschel et al. 2009; Soler et al. 2010].

The *ambient occlusion* (see [Landis 2002]) of surface point C with normal $\hat{n}$ is

$$AO^*_r(C, \hat{n}) = 1 - A^*_r(C, \hat{n}), \quad (2)$$

where $0 \leq A^*_r(C, \hat{n}) \leq 1$ is the fractional visibility of the far field at C [Cook and Torrance 1982]. Function A is visibility weighted by projected area over the positive hemisphere,

$$A^*_r(C, \hat{n}) = \frac{1}{\pi} \int_\Omega V(C, C + r\hat{\omega})\, \hat{\omega} \cdot \hat{n}\, d\hat{\omega}. \quad (3)$$

Note that $(\hat{\omega} \cdot \hat{n})$ in eqn 3 accounts for the solid angle of *occluders* projected onto the tangent plane at C. For example, this decreases the impact of occluders only block incident light at shallow angles, which is radiometrically correct. Specifically, this is *not* the projected area factor that will later be applied to indirect illumination, so this does not "double count" that factor.

Shadows account for the visibility factor in direct illumination. obscurance is an approximation of shadowing for indirect illumination that modulates environment light. To transition smoothly between

the near- and far-field illumination models, ambient occlusion is often extended to *ambient obscurance* (AO) [Zhukov et al. 1998] using a *falloff function* that reduces the influence of occlusion with distance. The analogous obscured function is

$$A_r(C,\hat{n}) = 1 - \int_\Omega [1 - V(C + \hat{\omega}\min(t(C,\hat{\omega}) + \epsilon, r))] \cdot g(t) \cdot (\hat{n} \cdot \hat{\omega})\, d\hat{\omega} \tag{4}$$

where t is the distance from C to the first scene point encountered by a ray cast in direction $\hat{\omega}$ and $0 \le g(x) \le 1$ is the aesthetically-selected falloff function, which commonly obeys $g(x) = 0|_{x \ge r}$ and $g(0) = 1$. For the remainder of this paper we omit the parameters of the A function, which are always r, C, and $\hat{n}$.

2.2 Derivation of the obscurance Estimator

We now derive a robust estimator of obscurance with intuitive parameters for aesthetic falloff. Our particular falloff function leads to an extremely efficient implementation.

Let $\Gamma \subseteq \Omega$ be the subset of the hemisphere for which there exists some distance $0 < t(C,\hat{\omega}) < r$ at which the ray from C in direction $\hat{\omega} \in \Gamma$ intersects the scene. That is, Γ is the set of all near-field occluders projected onto the unit sphere. Because Γ covers all near occluders, the visibility term from eq. 4 is always zero on it:

$$V(C, \hat{\omega} \min(t(C,\hat{\omega}) + \epsilon, r)) = 0 \,\big|_{\hat{\omega} \in \Gamma}. \tag{5}$$

Changing the integration domain in eq. 4 to Γ gives

$$A = 1 - \int_\Gamma g(t(C,\hat{\omega}))\ \hat{\omega} \cdot \hat{n}\, d\hat{\omega}. \tag{6}$$

We choose the falloff function $g(t) = u \cdot t \cdot \max(u,t)^{-2}$, which resembles the shifted hyperbola that Filion and McNaughton [2008] reported artistically desirable in *StarCraft II* (figure 2) *and* leads to a remarkable simplification under our geometric construct. Substitute into eq. 6:

$$A = 1 - \int_\Gamma \frac{u t(C,\hat{\omega})}{\max(u, t(C,\hat{\omega}))^2} \hat{\omega} \cdot \hat{n}\, d\hat{\omega}. \tag{7}$$

Rewrite this in terms of the vector $\vec{v}(\hat{\omega}) = \hat{\omega} t(C,\hat{\omega})$ from C to the occluder (P in figure 3). Because $\hat{\omega} = \vec{v}(\hat{\omega})/||\vec{v}(\hat{\omega})||$ and $t(C,\hat{\omega}) = ||\vec{v}||$,

$$A = 1 - u \int_\Gamma \frac{\vec{v}(\hat{\omega}) \cdot \hat{n}}{\max(u^2, \vec{v}(\hat{\omega}) \cdot \vec{v}(\hat{\omega}))}\, d\hat{\omega}. \tag{8}$$

Numerically estimate this integral by sampling a set $\{\hat{\omega}_i\}$ of s directions uniformly at random, giving:

$$A \approx 1 - \frac{2\pi u}{s} \sum_{i=1}^{s} \frac{\max(0, \vec{v_i} \cdot \hat{n}) \cdot H(r - ||\vec{v_i}||)}{\max(u^2, \vec{v_i} \cdot \vec{v_i})}. \tag{9}$$

where $H(x)$ is the Heaviside step function.

2.3 Screen-Space Approximation

For practical application, we evaluate our radiometrically-based estimator from eq. 9 in screen space by sampling a depth or position buffer, introduce concessions to finite precision, and extend the result with aesthetic controls for artists.

We choose each sample in the same manner as Loos and Sloan [2010]: consider the disk of radius r and center C that is parallel to the image plane, select a screen-space point Q' uniformly at random on its projection, and then read a depth or position buffer to find the camera-space scene point $P = (x_P, y_P, z(Q'))$ on that ray. Figure 3 summarizes the process. This selection biases the samples because a ball about C projects to an ellipse, not a disk. Like Loos and Sloan, we ignore this source of error because it is relatively small for a typical field of view up to 70 degrees.

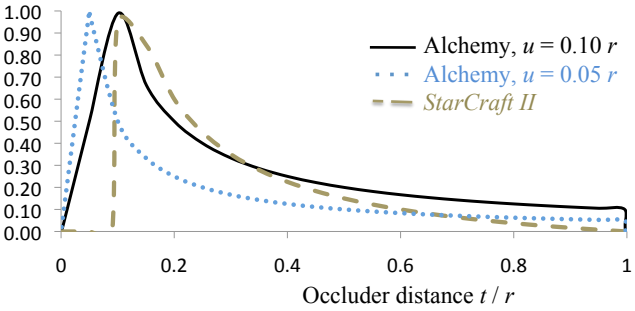

Figure 2: Plot of falloff $g(t)$ vs. t for Alchemy and StarCraft II AO.

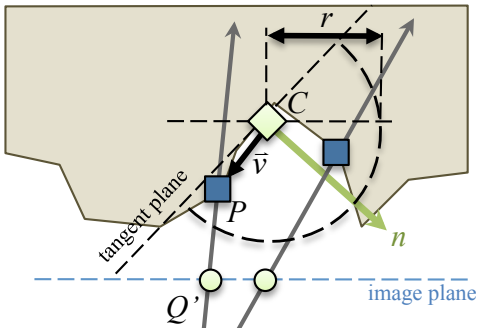

Figure 3: Geometric construction for Alchemy AO.

We augment the solution with expressive parameters for intensity scale (σ) and contrast (k) to allow intentional over- and under-occlusion, and nominally scale intensity by $1/\pi$ so that $\sigma = 1.0$ and $k = 1.0$ produce a useful baseline.

In practice, the falloff constant u in eq. 9 serves only to avoid division by zero and interferes with setting σ independently, so we eliminate it from the numerator and replace $\max(u^2, \vec{v_i} \cdot \vec{v_i})$ in the denominator with $\vec{v_i} \cdot \vec{v_i} + \epsilon$. Choose ϵ large enough to prevent underflow and small enough that corners do not become white from light leaks. We found that values within two orders of magnitude of $\epsilon = 0.0001$ produced indistinguishable results.

Bias distance β is the ambient-obscurance analog of shadow map bias: increase β to reduce self-shadowing (figure 4), decrease β if light leaks into corners. We scale it by z (which is negative) to efficiently compensate for the dot products becoming increasingly sensitive to error in $\hat{n}$ at distant points. Our final equation is:

$$\boxed{A \approx \max\left(0, 1 - \frac{2\sigma}{s} \cdot \sum_{i=1}^{s} \frac{\max(0, \vec{v_i} \cdot \hat{n} + z_C \beta)}{\vec{v_i} \cdot \vec{v_i} + \epsilon}\right)^k} \tag{10}$$

The parameters are straightforward to adjust because they have intuitive interpretations and reasonable values work for a large range of scenes. Obscurance has a maximum world-space radius of r and does not occur between objects closer than β. From there, one tunes for appearance. Images in this paper use $r = 0.5\text{m}$, $\sigma = 1$, $k = 1$, and $\beta = 10^{-4}\text{m}$ unless specified (*N.B.*, art directors for our games prefer to overdarken with higher intensity and contrast than this).

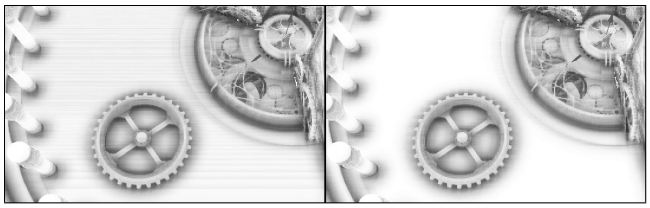

Figure 4: Top view with 8-bit normals and 16-bit positions. The radius of the large gear is about 0.8 m. Left: Self-occlusion with $\beta = 0$ m. Right: Corrected with $\beta = 10^{-4}$ m.

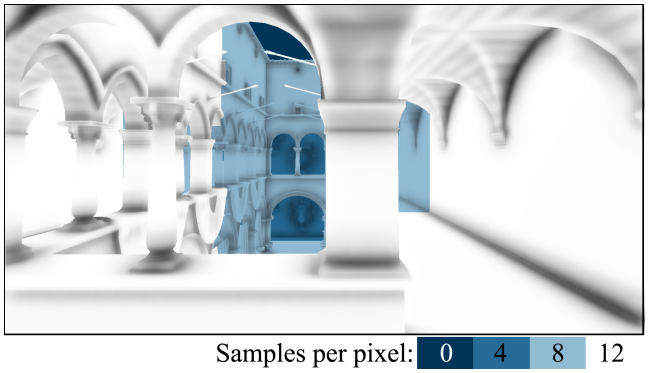

Figure 5: Diminishing sample count with distance. (The horizontal banding arises from interpolated vertex normals under coarse tessellation–if undesired, it can be eliminated by finer tessellation, face normals, or per-surface σ values.)

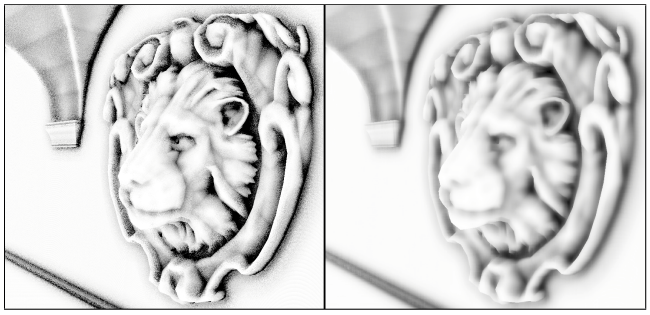

Figure 6: left: Raw obscurance from 12 samples per pixel; right: after two 13-tap 1D cross-bilateral filter passes.

3 Results

We report results on six scenes with diverse geometry:

Industrial, a giant machine with occlusion at many scales.
Knight character interacting with an environment.
Rocks and Grass nature scene. [Baraka 2010]
Sponza architectural benchmark with long sight-lines. [Meinl 2009]
Orc character head with many organic curves. [3D Hannibal 2008]
Dragon model scanned by Stanford (10 cm tall). [Stanford 1996]

The first two are from our studio to demonstrate application to production game assets. The others are publicly available.

Accompanying this paper is a set of video results rendered on a Windows PC and on Xbox 360 to demonstrate temporal coherence under camera and character motion. For the PC version, we show the guard band region on-screen to reveal the loss of obscurance within that region. One would normally render a slightly larger image and crop during blur; this adds about 0.2 ms to the render time on GeForce 580.

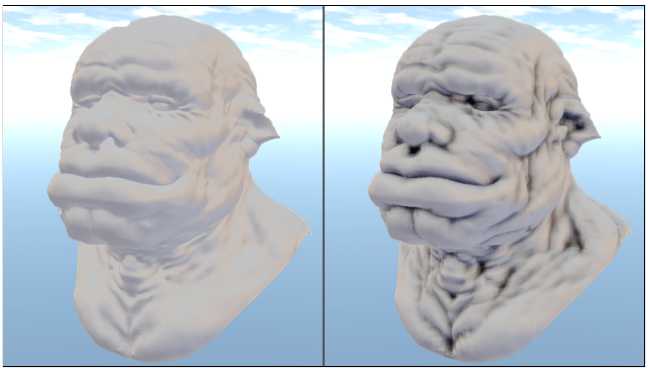

Figure 7: left: Environment-lit character; right: Alchemy AO enhances folds and wrinkles. $r = 0.2$

Figure 1 shows obscurance in Industrial at varying scales and orientations. This scene's sharp concave and convex corners, many layers, and curves cover many important cases; figure 7 also demonstrates obscurance for the organic shapes of the Orc model.

The darkening between vertices on the large platform disk in the lower right of Industrial is undesirable but consistent with the model. Those are locations where the interpolated vertex normal indicates a tangent plane that passes beneath the nearby vertices. This can be reduced by computing AO from face- instead of interpolated-normals, or by increasing the tessellation.

Figure 8 shows that increasing radius r while holding the number of samples constant increases the range of obscurance phenomena at the cost of sharpness. The farther-back gears gradually become darker as the radius grows to capture the influence of higher layers. Large radii are visually desirable but incur two costs. The first is that more samples are required to reduce variance. The second is that performance falls as r increases because samples near C correspond to position-buffer locations likely to be in cache, while points distant in screen space are likely not cached. All previous screen-space ambient occlusion algorithms have this property and it is widely known in the industry, although rarely reported in the literature.

Figure 9 shows the corresponding render times on GeForce 580 with 12 samples per pixel. The probability of a cache hit is theoretically proportional to r^{-2}, but we observe a more favorable, nearly-linear slowdown. We attribute this to the interaction of a small sample count compared to cache line size, and a very flat parabola. For $r > 1.5$m in this scene all fetches are cache misses, so time plateaus.

The banded LOD and increase in performance when the screen-space projection of r is small mean that the performance of Alchemy AO *increases* when viewing a deep scene, or with a wide field of view. This is precisely the case where other parts of a rendering system tend to lose performance, such as particle systems and rasterization itself, because more objects and a larger depth range are visible. This property helps balance the total frame rendering time.

Figure 10 compares Volumetric Obscurance [Loos and Sloan 2010], the most recent previous work, to Alchemy AO. The top two rows show the Industrial scene and the bottom row shows the Sponza scene. We tuned both algorithms for comparable performance: on

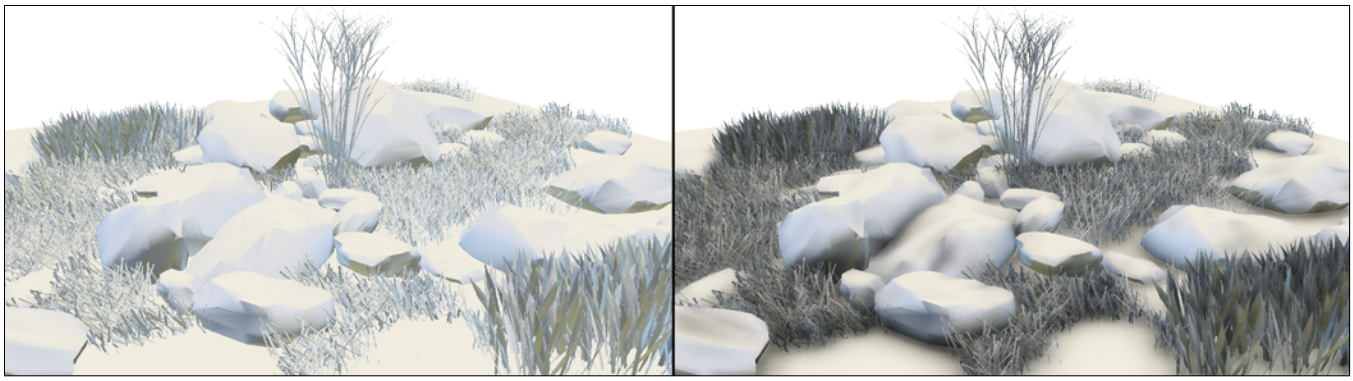

Figure 12: left: Environment-lit untextured Rocks and Grass scene; right: With Alchemy AO.

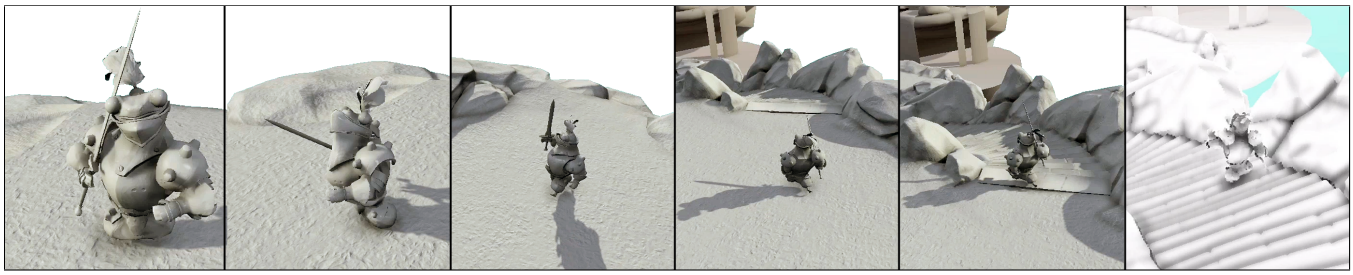

Figure 13: Frames of the dynamic 2 m tall Frog Knight character and moving camera in the Knight scene on Xbox 360, with all lighting terms and no materials. The rightmost frame shows the obscurance buffer. $r = 0.76$ m

	Xbox 360 *~D3D9*	**GeForce 460 SE** *D3D9*	**GeForce 580 GTX** *D3D11 / GL4*		
Samples/band	3	4	4	3	2
Maximum bands	1	1	3	3	3
Registers	6 vec4	15 scalar	15 scalar	15 scalar	15 scalar
32-bit words/pix IO	28	28	40	31	22
Accessibility time	**2.3 ms**	**0.6 ms**	**2.0 ms**	**1.5 ms**	**1.0 ms**
Filter pixel width	11	11	13	13	9
Filter time	**2.0 ms**	**1.6 ms**	**1.0 ms**	**1.0 ms**	**0.7 ms**

Table 1: Sample counts to hold worst-case performance constant by varying quality for low, mid, and high-end systems.

a GeForce 460 SE GPU, Volumetric Obscurance computes obscurance in 0.7 ms and Alchemy runs in 0.6 ms. Volumetric Obscurance runs at two resolutions, taking six samples per pixel at each resolution. Alchemy uses four samples per pixel in this test; it receives fewer samples because it reads normals and positions instead of just depth values. Volumetric Obscurance *can* run at half performance with normals to better capture fine detail, but that is redundant with multiple resolutions.

The Alchemy AO on the right of figure 10, exhibits desirable properties. The top row shows that it conforms to surfaces (b) instead of shadows "floating in air" near silhouettes (a). This is important for properly planted pillars and feet. The second row demonstrates that it is fast and robust enough to operate over a large radius, so occlusion appears from a continuous range of geometric frequencies. The third row combines these challenging cases over a continuous distance range.

Figure 11 compares Bavoil's and Sainz's [2009a] implementation of Horizon-based AO (HBAO) to Alchemy AO for the Dragon scene on GeForce 580. HBAO gives smoother results (a). Alchemy captures more fine details (b) and more consistent intensity (c) as the view changes, although both screen-space effects are limited by view dependence. Note the intense dark shadows under HBAO. The more conservative Alchemy result arises from adopting Loos's and Sloan's [2010] assumption that unseen space is empty. The Alchemy result exhibits higher temporal coherence under camera motion because it operates at screen resolution (see video); HBAO upsamples substantially and takes 22 ms if run at full resolution to avoid aliasing.

Figure 12 shows a natural scene with dense clumps of thin grass blades and large, smooth rocks. The appearance of foliage strongly depends on ambient occlusion. Alchemy's high sampling resolution compared to other screen space methods makes it well suited to capturing the interaction of many thin objects.

Figure 13 shows frames from the sequence captured on Xbox 360 in our result video. The animated character deforms under animation and establishes and breaks contact with the ground during the sequence. Unlike methods that process AO at very low resolution and upsample significantly, Alchemy samples multiple times at each pixel and distributes samples over space and angle, so it provides high spatial resolution and a good estimate that are robust to object and camera motion.

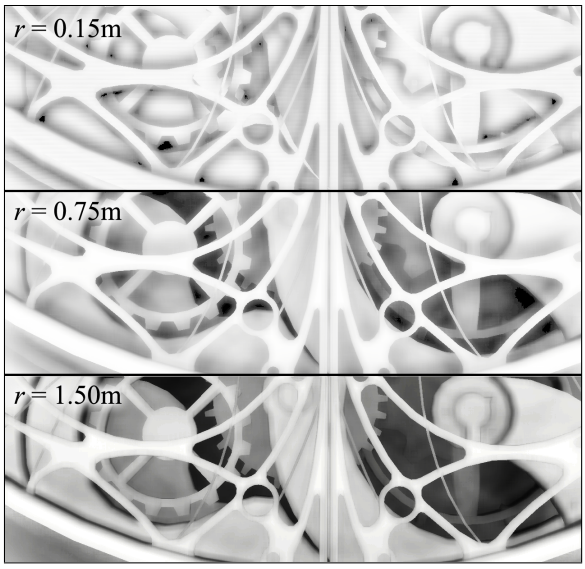

Figure 8: Varying obscurance radius r in the Industrial scene.

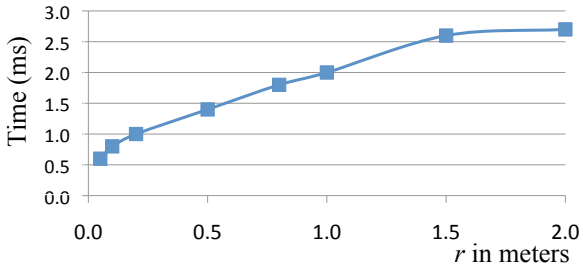

Figure 9: Increasing radius increases the number of texture cache misses and thus the render time.

Table 1 compares the algorithm across platforms for the view of Sponza in figure 10, which shows a range of distances but no sky (since sky pixels are inexpensive). We target 2-4 ms for the total cost of computing and filtering obscurance. Our algorithm is bandwidth-limited, so to maintain approximately constant performance we reduce the number of samples per pixel on lower-end hardware, at the cost of higher noise on those platforms. RGB8 costs one four-byte word and RGB16F position or DEPTH32 both cost two words per fetch due to memory layout, so our total bandwidth cost is higher than the number of usable bytes that we read. In the context of a game, one can exploit this to amortize the cost of filtering over four effects. For example, one could include soft shadows, indirect illumination, or noisy stochastic particles in the unused three channels of an obscurance buffer for simultaneous blurring. Microsoft's Pix tool reports that the Xbox 360 implementation maintains 90% ALU utilization.

4 Discussion

The Alchemy AO algorithm meets a production need for very fast and plausible ambient occlusion with intuitive aesthetic parameters. It combines the most effective elements of techniques previously proven in an industry context by developers at Crytek, Avalanche, and Blizzard and extends them with a new derivation for increased performance and robustness. The algorithm is stable under a large range of parameters and can naturally leverage the performance ability of a specific target GPU by adjusting the sample count (i.e., variance) and sample radius.

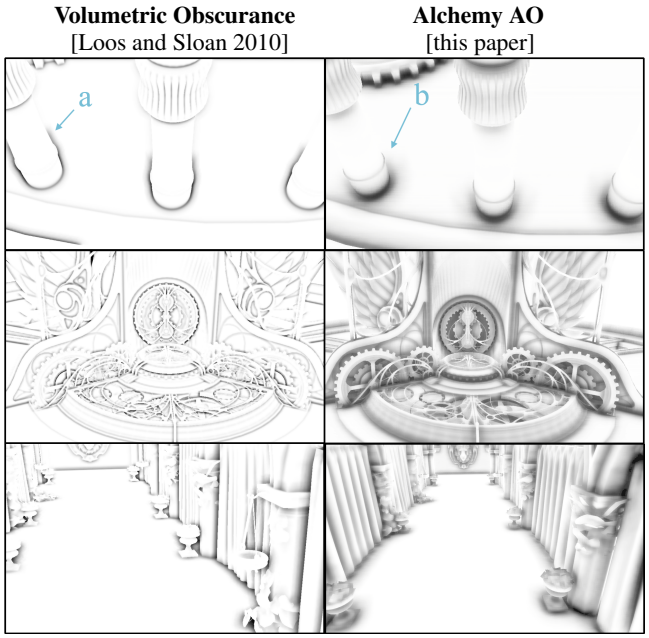

Figure 10: Quality comparison to the most recent previous work at comparable performance. One aspect we especially sought to improve is that where the pillars in the top row intersect the floor a) Volumetric Obscurance produces occlusion "floating in air" about the silhouette; b) Alchemy AO conforms to the ground orientation.

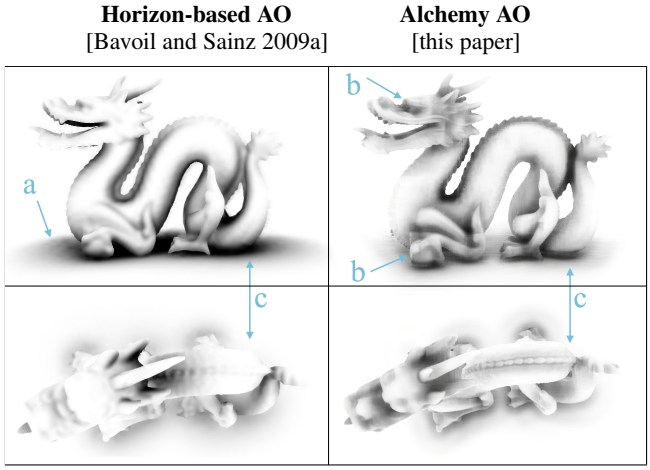

Figure 11: Dragon top and side views cropped from 1280×720. Left: HBAO with all optimizations enabled (8.5 ms). Right: Alchemy AO (2.6 ms) with $r = 0.1$.

We anticipate a future in which GPU performance increases and geometric methods for light transport become standard for real-time rendering. However, we also anticipate resolution and geometric complexity increasing. It is possible that general light-transport methods will never be practical for capturing the small-scale occlusion modeled by AO–indeed, even *offline* rendering systems seek efficient explicit AO approximations (e.g., [Bunnell 2005]). So we expect that Alchemy AO and methods like it may be useful for some time to come.

The most direct future work is to estimate local indirect illumina-

tion by sampling a color buffer as well as occlusion at each of the samples taken by the Alchemy algorithm. Local screen-space specular reflections work surprisingly well in many games, and there is active work in screen-space diffuse and glossy reflection. The interaction of multiple surfaces along the light transport path makes these physically similar to the AO problem and simulating indirect light presents comparable performance and robustness challenges. So an algorithm like Alchemy's is a natural candidate for extension. However, it remains open whether AO algorithms are generally extensible to viable estimators of indirect light, in part because the human perception of aliasing and bias in indirect light may be different than that in shadows.

Acknowledgements

We thank Naty Hoffman (Activision Studio Central) for checking the equations carefully and supporting the publication of algorithms used in games published by Activision, artists Nathan Hawkinson, Justin Wharton, Kevin Dobler, and Brent Gibson who created the game scenes used in this paper, and the entire team at Vicarious Visions.

References

3D HANNIBAL, 2008. Orc head. http://turbosquid.com/3d-models/orc-head-3d-3ds/420140.

AKENINE-MÖLLER, T., HAINES, E., AND HOFFMAN, N. 2008. *Real-Time Rendering 3rd Edition*. A. K. Peters, Ltd., Natick, MA, USA.

ARIKAN, O., FORSYTH, D. A., AND O'BRIEN, J. F. 2005. Fast and detailed approximate global illumination by irradiance decomposition. *ACM Trans. Graph. 24*, 3, 1108–1114.

BARAKA, 2010. Rock 02, May. http://turbosquid.com/FullPreview/Index.cfm/ID/528532.

BAVOIL, L., AND SAINZ, M. 2009. Multi-layer dual-resolution screen-space ambient occlusion. In *ShaderX*^{7}, W. Engel, Ed.

BAVOIL, L., AND SAINZ, M. 2009. Multi-layer dual-resolution screen-space ambient occlusion. In *SIGGRAPH 2009: Talks*, ACM, 1–1.

BUNNELL, M. 2005. *Dynamic Ambient Occlusion and Indirect Lighting*. NVIDIA Corporation.

COOK, R. L., AND TORRANCE, K. E. 1982. A reflectance model for computer graphics. *ACM Trans. Graph. 1* (January), 7–24.

FILION, D., AND MCNAUGHTON, R. 2008. Effects & techniques. In *SIGGRAPH 2008 courses*, ACM, 133–164.

IONES, A., KRUPKIN, A., SBERT, M., AND ZHUKOV, S. 2003. Fast, realistic lighting for video games. *IEEE Comput. Graph. Appl. 23*, 3, 54–64.

KAJALIN, V. 2009. Screen space ambient occlusion. In *ShaderX*^{7}, W. Engel, Ed. Charles River Media, March.

KAPLANYAN, A., AND DACHSBACHER, C. 2010. Cascaded light propagation volumes for real-time indirect illumination. In *Proceedings of I3D 2010*, ACM, I3D '10, 99–107.

KAPLANYAN, A. 2010. CryENGINE 3: Reaching the speed of light. In *SIGGRAPH 2010 courses*, ACM.

KONTKANEN, J., AND LAINE, S. 2005. Ambient occlusion fields. In *Proceedings I3D 2005*, ACM Press, 41–48.

LAINE, S., AND KARRAS, T. 2010. Two methods for fast ray-cast ambient occlusion. *Computer Graphics Forum (Proc. Eurographics Symposium on Rendering 2010) 29*, 4.

LANDIS, H. 2002. Production ready global illumination. In *SIGGRAPH 2002 courses*, 331–338.

LARSSON, D. 2010. Pre-computing lighting in games. In *SIGGRAPH 2010 courses*, ACM. in Global illumination across industries, http://cgg.mff.cuni.cz/ jaroslav/gicourse2010/.

LOOS, B. J., AND SLOAN, P.-P. 2010. Volumetric obscurance. In *Proceedings of I3D 2010*, ACM, 151–156.

LUFT, T., COLDITZ, C., AND DEUSSEN, O. 2006. Image enhancement by unsharp masking the depth buffer. *ACM Transactions on Graphics 25*, 3 (jul), 1206–1213.

MARTIN, S., AND EINARSSON, P. 2010. A real-time radiosity architecture for video games. In *SIGGRAPH 2010 courses*, ACM.

MCGUIRE, M. 2010. Ambient occlusion volumes. In *Proceedings of High Performance Graphics 2010*.

MEINL, F., 2009. Crytek sponza. Refinement of Marco Dabrovic's model. `http://crytek.com/cryengine/cryengine3/downloads`.

MILLER, G. 1994. Efficient algorithms for local and global accessibility shading. In *SIGGRAPH 1994*, ACM, 319–326.

MITTRING, M. 2007. Finding next gen: Cryengine 2. In *SIGGRAPH 2007 courses*, ACM, 97–121.

NVIDIA, 2009. Enabling ambient occlusion in games. Online *GeForce Guide*, viewed on April 3, 2011. http://geforce.com/#/Optimize/Guides/ambient-occlusion.

OAT, C., AND SANDER, P. V. 2007. Ambient aperture lighting. In *Proceedings of I3D 2007*, ACM, 61–64.

OWNBY, J.-P., HALL, R., AND HALL, C. 2010. Rendering techniques in Toy Story 3. In *SIGGRAPH 2010 courses*, ACM.

REINBOTHE, C., BOUBEKEUR, T., AND ALEXA, M. 2009. Hybrid ambient occlusion. *EUROGRAPHICS 2009Areaspapers*.

RITSCHEL, T., GROSCH, T., AND SEIDEL, H.-P. 2009. Approximating dynamic global illumination in image space. In *Proceedings of I3D 2009*, ACM, 75–82.

SHANMUGAM, P., AND ARIKAN, O. 2007. Hardware accelerated ambient occlusion techniques on GPUs. In *Proceedings of I3D 2007*, ACM, 73–80.

SOLER, C., HOEL, O., AND ROCHET, F. 2010. A deferred shading pipeline for real-time indirect illumination. In *SIGGRAPH 2010 talks*, ACM.

STANFORD, 1996. Dragon reconstructed from cyberware scan. http://graphics.stanford.edu/data/3Dscanrep/.

STEWART, A. J., AND LANGER, M. S. 1997. Towards accurate recovery of shape from shading under diffuse lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, 1020–1025.

SZIRMAY-KALOS, L., UMENHOFFER, T., TÓTH, B., SZÉCSI, L., AND SBERT, M. 2010. Volumetric ambient occlusion for real-time rendering and games. *IEEE CG&A 30*, 1, 70–79.

ZHUKOV, S., IONES, A., AND KRONIN, G. 1998. An ambient light illumination model. In *Rendering Techniques '98*, Springer-Verlag Wien New York, G. Drettakis and N. Max, Eds., Eurographics, 45–56.