Hardware Accelerated Ambient Occlusion Techniques on GPUs

Perumaal Shanmugam[*]
University of Texas at Austin

Okan Arikan[†]
University of Texas at Austin

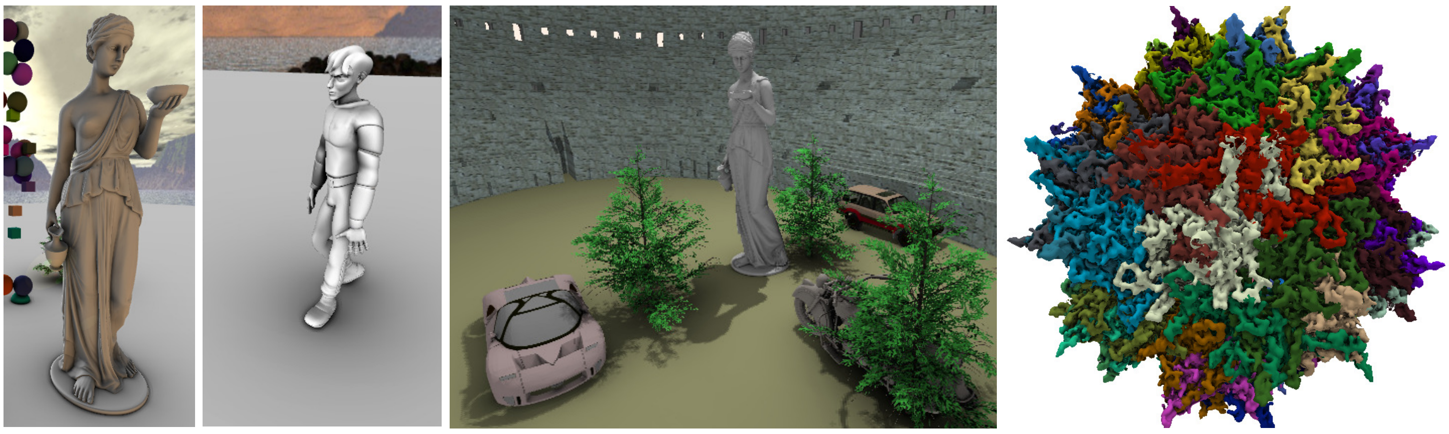


Figure 1: *These images illustrate our ambient occlusion approximation running in real-time on a modern GPU. Our method can be used for a number of applications including (a) Rendering high-detail models; this model has around 65K triangles. (b) Dynamic/deforming models (that undergo non-rigid deformations) (c) Enhancing natural objects such as trees, and in games where models such as cars are used. (d) Realistic rendering of molecular data.*

Abstract

We introduce a visually pleasant ambient occlusion approximation running on real-time graphics hardware. Our method is a multi-pass algorithm that separates the ambient occlusion problem into high-frequency, detailed ambient occlusion and low-frequency, distant ambient occlusion domains, both capable of running independently and in parallel. The high-frequency detailed approach uses an image-space method to approximate the ambient occlusion due to nearby occluders caused by high surface detail. The low-frequency approach uses the intrinsic properties of a modern GPU to greatly reduce the search area for large and distant occluders with the help of a low-detail approximated version of the occluder geometry. Our method utilizes the highly parallel, stream processors (GPUs) to perform real-time visually pleasant ambient occlusion. We show that our ambient occlusion approximation works on a wide variety of applications such as molecular data visualization, dynamic deformable animated models, highly detailed geometry. Our algorithm demonstrates scalability and is well-suited for the current and upcoming graphics hardware.

CR Categories: I.3.3 [COMPUTER GRAPHICS]: Picture/Image Generation— [I.3.7]: COMPUTER GRAPHICS—Three Dimensional Graphics and Realism

Keywords: gpu,ambient occlusion,soft shadows,real-time rendering

1 Introduction

Current real-time graphics applications such as games provide increasingly convincing realism, using a number of methods such as soft-shadows, high-dynamic range effects, post-processing effects and surreal lighting. Ambient occlusion is another method that has recently attracted real-time graphics researchers aiming to increase the level of realism without performing a complete global-illumination step. Compared to other global illumination approximations, ambient occlusion gives a unique edge by providing a significant increase in quality despite a crude and a simple approximation of global illumination. We propose a real-time multi-pass approximation for ambient occlusion. We provide different approximations to the occlusion caused by nearby surface detail (often creating the high frequency detail in ambient occlusion) and the occlusion caused by distant surfaces (often creating a smooth change in ambient occlusion). We describe how these approximations can be computed on the graphics hardware to perform ambient occlusion in real-time. Our method has the advantage of being largely independent from the scene complexity. Our method approximates the ambient occlusion for each frame independently, making it suitable for deformable and dynamic objects.

2 Related Work

Ambient occlusion has been used in films as an alternative to full-scale global illumination methods. In the photorealistic rendering context, such as films, ambient occlusion is performed by determining visibility along various directions according to (eqn. 1). However, this is computationally too expensive for real-time applications for now. [Arikan et al. 2005] is a method that uses near-field

[*]e-mail: perumaal@cs.utexas.edu
[†]e-mail: okan@cs.utexas.edu

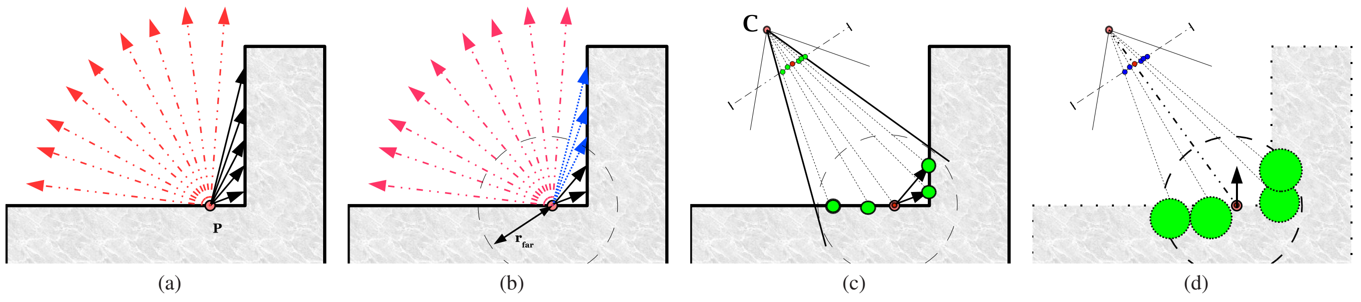


Figure 2: *Local ambient occlusion in image-space: (a) Rays emerging from a point* **P** *many of which are not occluded (marked red). (b) Rays being constrained within a distance of r_{far}, with distant occluders being neglected (blue arrows). (c) Pixels as viewed by a camera C. Neighboring pixels are obtained (marked in green). (d) We de-project these pixels back to world-space and approximate them using spherical occluders (green circles). These are the final approximated occluders around the point* **P** *that are sought in image-space. Note that the spheres are pushed back a little along the opposite direction of the normal ($-\hat{n}$) so as to prevent incorrect occlusion on flat surfaces.*

ambient occlusion for obtaining more performance but this method is not geared for real-time either.

In the real-time rendering context, there exist many approximations to ambient occlusion. Historically, ambient occlusion is referred to as "accessibility" which defines how accessible a surface is; for e.g. [Miller 1994] uses accessibility to provide a rough shading model mimicking ambient occlusion. [Bunnell 2005] uses disc-based occluders and a per-vertex ambient-occlusion term; however, it requires a huge pre-computation step with little support for dynamic objects, and requiring high-tessellation of scene geometry.

[Kontkanen and Laine 2005] partially solves the dynamic object problem; however it is limited to rigid body motion and is not suitable for deformable objects. This method suffers from a large pre-computation step and from the usage of a large number of textures.

[Ren et al. 2006] uses spherical occluders and spherical harmonics to calculate approximate ambient occlusion (or low-frequency soft shadows). This solution attacks the problem of over-occlusion and provides a simple pre-computation step while supporting deformable objects. However, this method suffers in performance when there are a large number of objects.

As all of these approaches target an approximated version of the original geometry that has far less details, they tend to miss high-frequency ambient occlusion effects that would provide a huge increase in quality. We borrow some ideas from the research mentioned above to build on our ambient occlusion solution.

There are other research papers such as [Kontkanen and Aila 2006; Hegeman et al. 2006; Wassenius 2005] targeting specific applications such as animated characters, trees, etc. Solely pre-computation based static methods such as PRT (Precomputed Radiance Transfer) [Sloan 2006; Sloan et al. 2002; James and Fatahalian 2003], baked-AO (such as [ShadeVis 2006]) are less flexible yet simpler solutions for static ambient occlusion. Ambient occlusion is also used in a wide variety of real-time applications, such as molecular rendering (QuteMol, [Tarini et al. 2006]; TexMol, [Bajaj et al. 2004]) and nature-rendering (grass-rendering) mainly to increase realism.

3 Overview

Ambient Occlusion is a fast approximation to global illumination. Ambient occlusion at a point on an object is the amount of occlusion it receives due to occluders surrounding it. The ambient occlusion A at a point **P** with a surface normal $\hat{n}$ is given by:

$$A(\mathbf{P},\hat{n}) = \frac{1}{\pi}\int_{\Omega}(V(\hat{\omega},\mathbf{P}))max(\hat{\omega}\cdot\hat{n},0)d\hat{\omega} \qquad (1)$$

where $\hat{\omega}$ represents the various directions at **P** along the unit hemisphere Ω; V is the visibility function along that direction, which is 1 if occluded and 0 otherwise. The effect of ambient occlusion can roughly be described as the effect seen in a cloudy day; here we consider the sky to be a uniform area light source resulting in soft, low-frequency non-directional shadows [Ren et al. 2006].

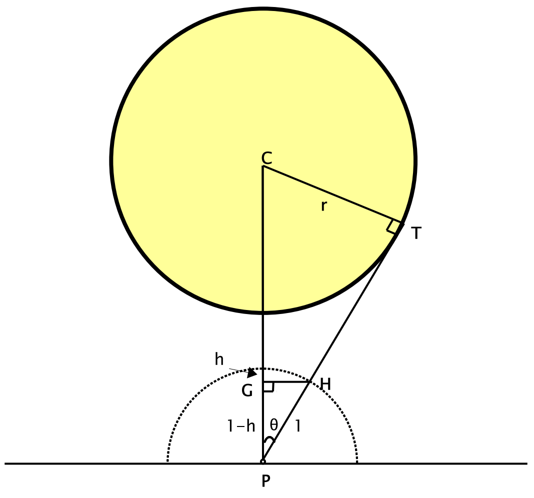


Figure 3: *Ambient occlusion due to a single sphere at a point on a plane.*

Throughout this paper, we use the notion of approximated ambient occlusion (A_ψ) due to a sphere S with center **C** and radius r (written as $<\mathbf{C},r>$) at a receiver point **P** having a normal $\hat{n}$ (fig. 3). This quantity is denoted by $A_\psi(C,r,\mathbf{P},\hat{n})$.

$$A_\psi(\mathbf{C},r,\mathbf{P},\hat{n}) = S_\Omega(\mathbf{P},\mathbf{C},r)*max(\hat{n}\,.\hat{\mathbf{PC}},0) \qquad (2)$$

where S_Ω is the surface area of the spherical cap subtended by the sphere $<\mathbf{C},r>$ on the unit hemisphere Ω at **P** and $\hat{\mathbf{PC}}$ is the unit vector from **P** to **C**.

$$
\begin{aligned}
S_\Omega(\mathbf{P}, \mathbf{C}, r) &= 2 * \pi * h * 1 \\
h &= 1 - cos(\theta) \\
\theta &= sin^{-1}(\frac{r}{|\mathbf{PC}|}) \\
S_\Omega(\mathbf{P}, \mathbf{C}, r) &= 2 * \pi * (1 - cos(sin^{-1}(\frac{r}{|\mathbf{PC}|})))
\end{aligned} \tag{3}
$$

This is an approximate quantity, and we trade physical accuracy for performance and quality.

4 Image Space Approach For High-Frequency Ambient Occlusion

In the case of complex objects having high-surface detail, we note that the effect of ambient occlusion is mostly due to the occluders that are near to the receiver point (fig. 2). In figures (5(a),5(b),5(c),5(d)) consisting of the ground-truth images, we see that as the ray-length of the ambient occlusion visibility rays is decreased, the quality decreases only slightly. This is because, high-illumination detail is due to high-surface detail or the rapid appearance and disappearance of occluders as we move along the surface. This is correlated with (eqn. 1) by observing that most of the rays hit these nearby occluders and hence cause the most ambient occlusion effect. This observation is also used in [Arikan et al. 2005; Wassenius 2005].

4.1 Intuition

We make a number of observations to use this notion of nearby occluders. First, note that we only need to perform ambient occlusion for those receiver points that are visible to the camera; that is, we only care about those pixels that are present in the $Z-buffer$. Each pixel in this $Z-buffer$ or the depth-buffer corresponds to a point in the world space; along with the normal buffer, we can obtain the position $\mathbf{P}$ and the normal $\hat{n}$ for a given camera pixel using these two buffers. We call the combination of these two buffers the $ND-buffer$ (normals/depth buffer).

Each pixel in the $ND-buffer$ corresponds to a sample of some surface in the world-space. Therefore, we can use it as an occluder to other pixels. The individual pixel by itself cannot considered to represent a point, as it has zero-volume leading to no ambient occlusion. We choose to approximately reconstruct this surface represented by a pixel using a sphere in world-space that roughly projects to a pixel on the screen. This is because we do not know what surface this pixel actually represents; also, a sphere helps avoid surface discontinuity problems due to rapid changes in the normal, since a sphere is symmetrical having no orientation. We have found that approximating pixels using disks does not work well due to surface discontinuities.

Due to the nature of 3D projection in world-space onto a 2D image plane, the occluders that are nearby in world-space to a particular receiver point $\mathbf{P}$ are also nearby to the corresponding projected pixel $\mathbf{p}$ in the $ND-buffer$ (ignoring the depth test). By nearby occluders, we mean those occluders that are within a certain distance, say r_{far}, from the given receiver point $\mathbf{P}$ (fig. 2). So by looking at nearby pixels, we can get a fairly good idea of nearby world-space occluders (fig. 4).

Note that although points that are nearby in world-space are nearby in image-space, the converse is not true. This means that we will be gathering some pixels whose corresponding world-space points are farther than r_{far} from $\mathbf{P}$. Fortunately, the ambient occlusion caused

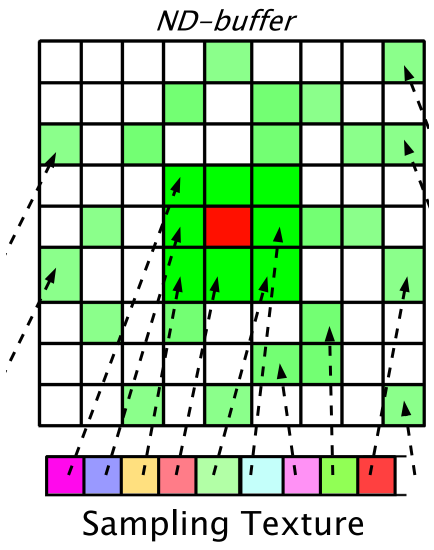


Figure 4: *This image shows a portion of the $ND-buffer$ and the neighboring samples. We gather neighboring pixel samples around the receiver pixel* $\mathbf{p}$ *(shown in red) in the $ND-buffer$. The neighboring pixels (shown in green) are sampled randomly around the given receiver pixel so as to access as many pixels possible. The number of sample pixels to be gathered is calculated using the projected distance r_{far} at* $\mathbf{p}$*; a pixel far away requires fewer number of samples than one that is nearer to the camera. The sampling texture provides the random distribution for the samples' locations.*

by these faraway points is low due to the inverse-square attenuation as a function of distance.

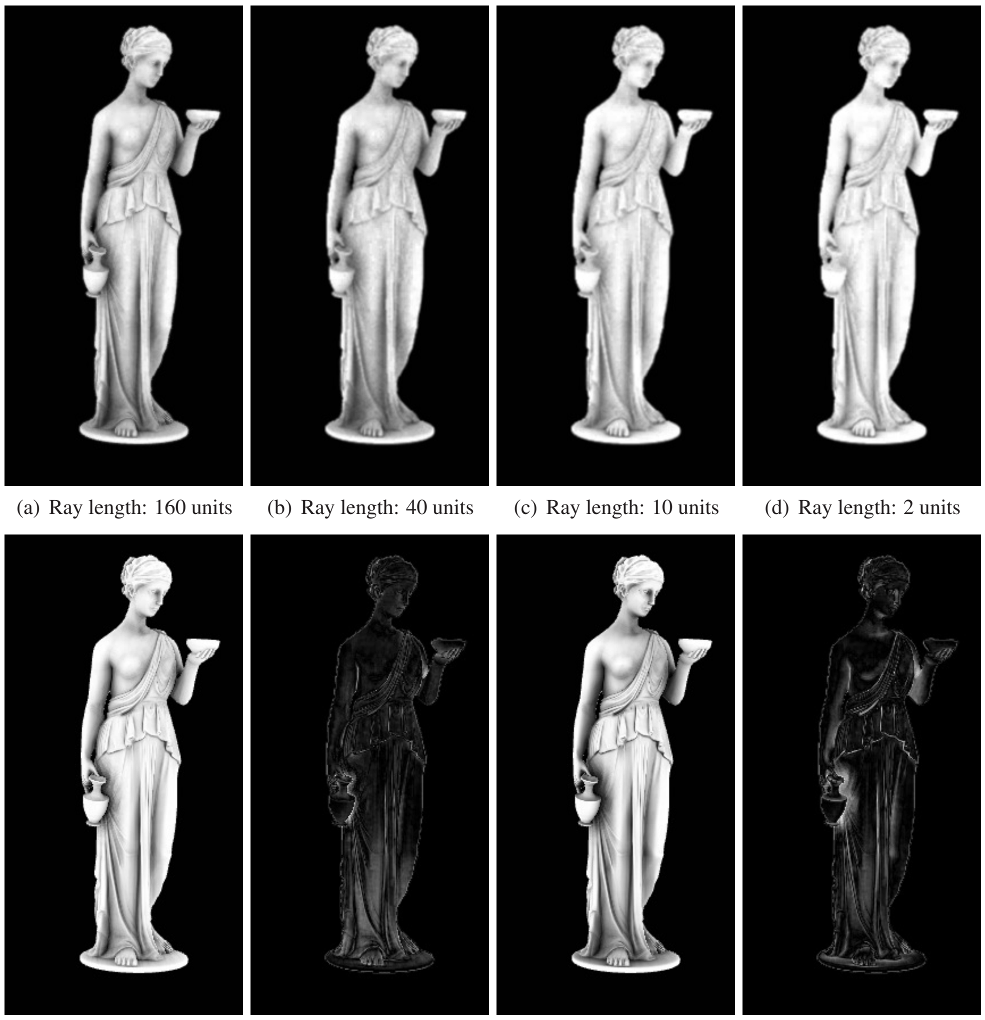
4.2 Detailed View

For a given pixel in the $ND-buffer$, we gather neighboring pixels. The number of pixels gathered is controlled by the projection of r_{far} at the pixel $\mathbf{p}$. Intuitively, we gather more neighboring pixels for a point closer to the camera than for a point lying at a further distance.

$$
A(\mathbf{P}, \hat{n}) = \sum_{|\mathbf{P}-\mathbf{Q_i}|<r_{far}} A_\psi(\mathbf{Q_i}, r_i, \mathbf{P}, \hat{n}) \tag{4}
$$

Here, we approximate the ambient occlusion at a receiver point $\mathbf{P}$ with a normal $\hat{n}$, corresponding to the given pixel $\mathbf{p}$ in the $ND-buffer$. This is due to the ambient occlusion effect A_ψ of the spheres $<\mathbf{Q_i}, r_i>$. r_i, the radii of the sphere at Q_i, is a function of the depth of the pixel $\mathbf{q_i}$. These spheres are the approximations for the neighboring pixels $\mathbf{q_i}$ to the receiver pixel $\mathbf{p}$ using the previously mentioned logic. Thus, for a given receiver pixel in the $ND-buffer$, we sum the A_ψ contributions due to each of the neighboring pixels' spherical approximation.

As we have essentially reduced the approximation to an image-space operation, we draw a full-screen quad to invoke the fragment processor across all the pixels in the $ND-buffer$. For each such pixel, we calculate the sum due to various A_ψ according to (eqn. 4) and output the approximated ambient occlusion value. We can also control the way the neighboring pixels are looked-up in the $ND-buffer$. We can gather pixels inside, say, a simple square area centered around the given pixel or we can sum only a random subset of these pixels due to the limited bandwidth (fig. 4).


(a) Ray length: 160 units (b) Ray length: 40 units (c) Ray length: 10 units (d) Ray length: 2 units

(e) Image space (f) Difference: $(e)-(a)$ (g) Combined Approach (h) Difference: $(g)-(a)$

Figure 5: *Difference images. (a) through (d) show the ground-truth images, ray-traced using 128 ambient occlusion visibility rays with the corresponding ray lengths. (e) shows the image-space approach, which renders at around 36 fps. (f) shows the difference image between (e) and (a). Note that the large differences are concentrated at areas receiving occlusion from distant surfaces, such as the leg due to the pedestal. (g) shows the combined approach. (h) shows the difference between (g) and (a). The large differences are due to few over-occlusion artifacts. Although both (g) and (a) look visually pleasing, note that we only want to mimic the visual quality and not the physical accuracy of actual ambient occlusion. This combined approach runs at around 34 fps. All measurements are made using a nVidia GeForce 8800 GTX card. Some errors along the edges are due to the differences in anti-aliasing.*

We note that the first layer, viz. the $ND-buffer$, does not provide a good enough approximation. Due to the depth test, the neighborhood of a world-space point may not be present for the corresponding pixel in the $ND-buffer$, because they may have been occluded by surfaces closer to the camera. One could remedy this problem by using depth-peeling ([Everitt 2001]) to get more layers and applying the algorithm to each such layer. Practically only the first two layers provide enough information to approximate the ambient occlusion well. Extracting more layers is expensive and impractical, and works well only for certain special cases such as an occluder inside another occluder that cannot be seen by the two layers. Hence we limit ourselves to just the two layers.

4.3 Advantages and Limitations

Some of the salient points with regards to existing ambient occlusion approximations such as [Kontkanen and Laine 2005], [Bunnell 2005] are:

- No precomputation is necessary; fitting this algorithm for animation of complex models is simple and fast.
- Easily plugged into existing approaches, with the only necessity being the use of deferred shading.
- The constraints are reduced to a single factor - memory bandwidth. The number of neighboring pixels accessed governs the memory bandwidth, and also the quality and performance.
- Comparing with a "baked" ambient occlusion approach, our method offers real-time detailed ambient occlusion that may

otherwise be difficult to compute using such a baked-AO approach due to differing resolutions in the baked textures and the underlying geometry.

- The performance of this approach does not depend on the polygon count to a large extent; instead, it is directly related to the number of pixels shaded in the *ND-buffers*. This is a significant advantage over existing approaches.

The following are some of the limitations of this approach:

- The memory bandwidth poses a problem with older hardware, as the quality of the approach directly depends on the bandwidth. One can try to reduce this by using fewer samples per pixel albeit at the cost of less ambient occlusion effect.
- Artifacts may appear on surfaces where the occluders are nearly parallel to the direction of view or when the occluders are at a large distance from the camera since the sampling pattern may miss such occluders. This can be compensated by distributing more samples along the direction of such an occluder, at increased branching costs.
- For the algorithm to be visually pleasing we need high-polygon count geometry. This may also be complimented by the current trend of modern GPUs that scale well to high-detail geometry.

We intentionally ignore large and distant occluders in this approach. We can use the methods described in [Kontkanen and Laine 2005], [Ren et al. 2006], [Kontkanen and Aila 2006] or [Bunnell 2005] for this purpose. However we solve this problem using a different and a faster approach.

5 Distant-occluder Approach For Low-Frequency Ambient Occlusion

In this approach, we are not concerned with the high surface details or the complexity of the object. Hence, we find a crude and simple approximation for the underlying surface geometry (fig. 6). There have been extensive research in finding approximations of surface geometry using discs ([Bunnell 2005]), spheres ([Wang et al. 2006; Hegeman et al. 2006]), etc. We have chosen a spherical approximation because there exist a number of extensive studies ([Wang et al. 2006; Bradshaw and O'Sullivan 2002]) for such an approximation. Moreover a sphere has just four degrees of freedom, unlike a disc which has six; a sphere is also symmetrical and has no orientation.

5.1 Intuition

We first analyze the ambient occlusion effect due to a sphere at a receiver point. For a given sphere, $<\mathbf{C}, r>$, we observe that the maximum ambient occlusion due to it at a point $\mathbf{P}$ decreases rapidly with the distance. From (eqn. 1), we can intuitively see that as the distance increases the surface area subtended by the sphere $<\mathbf{C}, r>$ on the unit hemisphere Ω at $\mathbf{P}$ decreases (fig. 8). In order to maximize the ambient occlusion at $\mathbf{P}$, all else remaining constant, the normal $\hat{n}$ at $\mathbf{P}$ should point directly at the center of the spherical occluder, $<\mathbf{C}, r>$, i.e. $\hat{n} = \hat{\mathbf{PC}}$.

Beyond a certain distance, say $d_{far} = |\mathbf{PC}|$, the ambient occlusion value $A_\psi(\mathbf{C}, r, \mathbf{P}, \hat{n})$ due to $<\mathbf{C}, r>$ falls to a very small ε. We use this notion of a negligible ambient occlusion value to limit the ambient occlusion influence due to an occluder within a certain distance. This is the significant difference between other methods such as [Kontkanen and Laine 2005], [Ren et al. 2006], [Kontkanen and Aila 2006], [Bunnell 2005] and ours. We avoid the problem of iterating through every possible occluder for all pixels in the $ND-buffer$ by providing a limit to such occlusion. This approximation also is justified by the observation that we choose ε to be a very low value that cannot be visually distinguished from zero ambient occlussion.

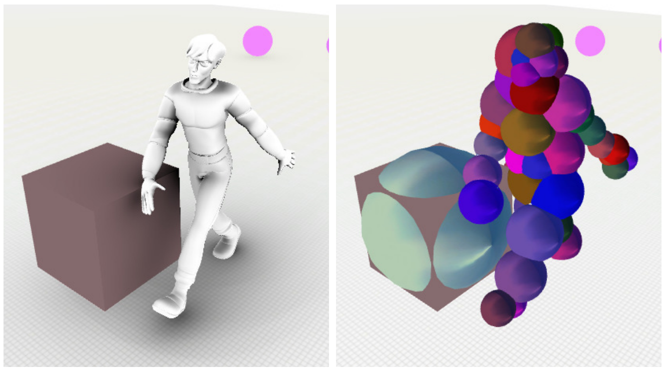


Figure 6: *The left image shows the distant occluder stage in isolation. Shadows and the image-space stage have been turned off to illustrate the nature of the distant occluder approach. The image on the right shows the spheres for the model that are being recalculated per frame. The rest pose provides us the initial position of these spheres due to Lloyd clustering, and we use this to reposition the spheres based on the new vertex positions. This recomputation is simple and fast.*

This ε value (chosen by the user) allows us to provide an approximation as well as an extent of the influence of the sphere's ambient occlusion:

$$\varepsilon = A_\psi(\mathbf{C}, r, \mathbf{P}, \hat{n})$$
$$\varepsilon = 2\pi * (1 - cos(sin^{-1}(\frac{r}{|\mathbf{PC}|}))) * max(\hat{n} . \hat{\mathbf{PC}}, 0) \text{ from } (\text{ eqn. } 2)$$
$$\varepsilon = 2\pi * (1 - cos(sin^{-1}(\frac{r}{d_{far}}))) \text{ (as } \hat{n} = \hat{\mathbf{PC}} \text{ and } d_{far} = |\mathbf{PC}|)$$
$$\text{Hence, } d_{far} = r * \frac{1}{sin(cos^{-1}(1 - \frac{\varepsilon}{2\pi}))} \tag{5}$$

Once we choose the constant ε, finding d_{far} for a given radius r amounts to a constant scaling of r. Intuitively, d_{far} represents the limit of the ambient occlusion effect of any sphere for a given radius; beyond d_{far}, the ambient occlusion 'influence' of this sphere is below ε.

5.2 Detailed View

We can project d_{far} onto the image plane (i.e., the $ND-buffer$) to obtain the projected distance, d'_{far} (in pixels). Due to the nature of perspective projection, we observe that for any two points that are at most d_{far} apart, their projected distance on the $ND-buffer$ will not be farther than d'_{far}. Thus, by covering all the pixels in the $ND-buffer$ that are within a distance of d'_{far} from $\mathbf{C}'$ (where $\mathbf{C}'$ is the projected pixel of the center of the sphere, $\mathbf{C}$), we ensure that we cover all pixels that receive at least ε ambient occlusion due to $<\mathbf{C}, r>$.

Developing this idea further, we can now derive an image-space operation. For a given sphere, $<\mathbf{C}, r>$, we first calculate d_{far} based on its radius. We then invoke the fragment shader at all pixels q_i, that are at most d'_{far} apart from $\mathbf{C}'$ in the image-plane. In order for this to happen, we draw a billboard invoking the fragment shader on all such pixels.

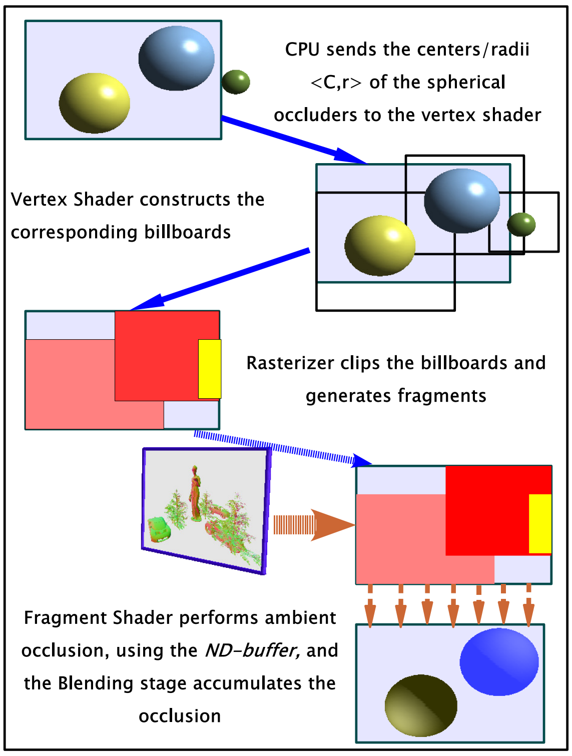

CPU sends the centers/radii <C,r> of the spherical occluders to the vertex shader
Vertex Shader constructs the corresponding billboards
Rasterizer clips the billboards and generates fragments
Fragment Shader performs ambient occlusion, using the *ND-buffer,* and the Blending stage accumulates the occlusion

Figure 7: *Distant-occluder stage.*

Intuitively, we 'splat' the ambient occlusion effect due to a sphere $< \mathbf{C}, r >$ onto points that are capable of being influenced at least ε ambient occlusion. Although the use of billboards for a myriad of purposes is common ([Dachsbacher and Stamminger 2006; Botsch et al. 2005]), targeting ambient occlusion is new. At every such invoked pixel, we obtain $(\mathbf{P}, \hat{n})$ using the $ND-buffer$ and calculate the value of $A_\psi(\mathbf{C}, r, \mathbf{P}, \hat{n})$. We then use the GPU's blending functionality to additively blend these A_ψ values (fig. 7). Thus, we can push a large number of such billboards for a correspondingly large number of spheres and use the additive commutativity to ensure that the ambient occlusion values are properly accumulated in accumulation buffer. As we target only larger and distant occluders, we call this approach the distant-occluder approach.

5.3 Issues and Advantages

Since we use direct accumulation of the ambient occlusion effect on to a buffer, *over-occlusion* artifacts may occur. This happens when there are too many occluders influencing a particular point; that is, when there are a number of occluders in a particular direction, we take into account all of them, producing over-occlusion artifacts. This is the major issue with our distant-occluder approach. A discussion regarding the use of multiplicative-blending and multi-pass approaches for reducing this over-occlusion problem is provided in [Bunnell 2005; Kontkanen and Laine 2005].

An approximation we make is the fact that even large, distant occluders have an extent or a limit to their ambient occlusion influence. In the case of a closed room, every point inside the room is occluded by some occluder, and hence the room should appear completely black. The same is true for any scenario composed of a large number of small occluders at a large distance from a receiver

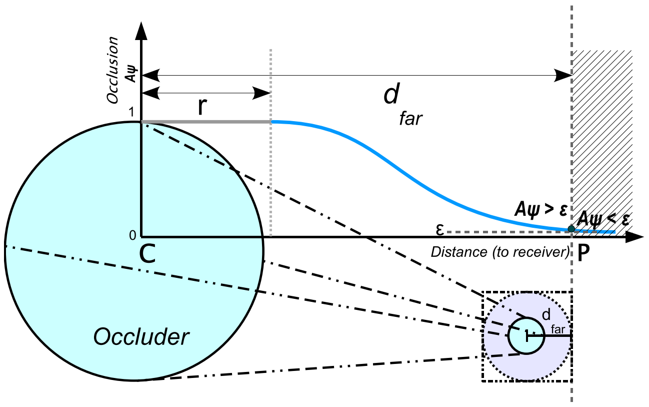


Figure 8: *This 2D visualization of occlusion explains the fall-off parameter ε as the receiver's distance (x-axis) from a spherical occluder varies. The inset illustrates the sphere along with the billboard corresponding to the distance d_{far} from the center of the sphere.*

point. In our case, it does not happen so, as we cut off even distant occluders at points where they contribute less than ε occlusion. However in reality, one would limit the length of the ambient occlusion visibility rays even in a photo-realistic rendering context to avoid a completely dark effect in a closed room.

One major advantage of the distant-occluder approach as compared to [Ren et al. 2006; Bunnell 2005; Hegeman et al. 2006; Kontkanen and Laine 2005] is that we use the various GPU units to their advantage. We achieve frame-rates in excess of 80 fps on a nVidia GeForce 8800GTX for over 6,000 spheres (using the distant-occluder approach alone); this is because, regardless of the number of spheres present in the scene, the rasterizer clips the billboards to the view-frustum and generates the targeted fragments. The vertex processor produces the billboards' positions; the fragment processor performs the calculation of the A_ψ values and the blending units accumulate these values (fig. 7). Hence, our approach is highly parallel, and can sustain a large number of spheres. In the case of deforming and dynamic characters, we make use of the correspondence between the spheres and the underlying geometry to trivially recalculate the positions/radii of the spheres (similar to [Ren et al. 2006]); because of the symmetry and absence of orientation, we are only concerned with their positions/radii of the spheres.

6 Combined Ambient Occlusion

Using the two algorithms mentioned above, we obtain two buffers each containing the ambient occlusion approximation values for every pixel in the $ND-buffer$. We combine these two buffers additively. After obtaining the original color buffer we blend it with the combined ambient occlusion value by multiplication $(Color * (1 - Occlusion))$ (fig. 9). Although we try to separate the near and far occluders using these two approaches, there might be an overlap between thee two; we try to avoid this possible overlap by setting $A_\psi(\mathbf{C}, r, \mathbf{P}, \hat{n}) = 1$, if $\mathbf{P}$ is within the sphere $< \mathbf{C}, r >$ in the case of the distant occluder approach.

The combination is compared against ground-truth images obtained by ray-tracing 128 ambient occlusion rays with various constrained ray-lengths. (figures (5(a), 5(b), 5(c), 5(d))). The difference images (fig.(5(f), 5(h))) show that our approximations result in visually pleasing yet physically inaccurate results that works well for real-time scenarios.

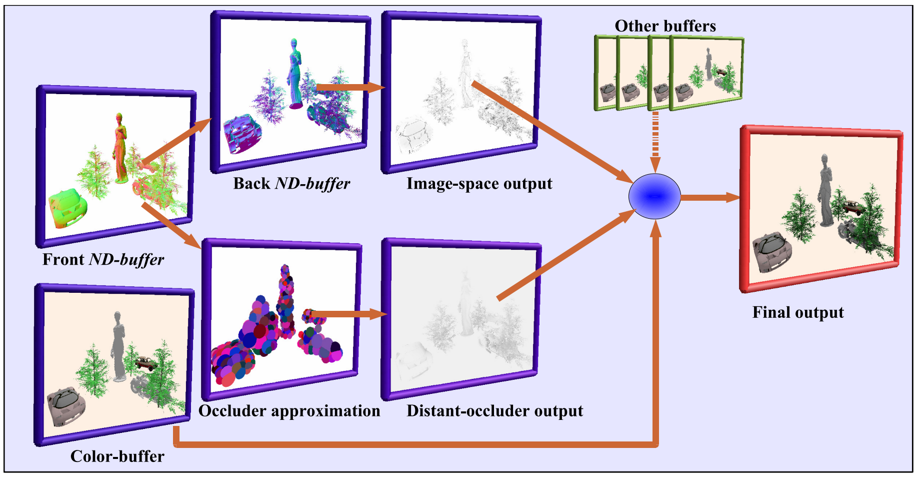


Figure 9: *Pipeline: This flow-diagram explains the various stages in our combined ambient occlusion algorithm. We start with the ND-buffer for the front-faces, along with the color-buffer. The color-buffer is required for deferred shading. The back-faces' ND-buffer is then created and supplied to the image-space approach. Similarly, we prepare the spheres' billboards for the distant-occluder approach. Finally, we combine these stages including shadow-buffers, color-buffers, and other buffers.*

7 Results

We have tested our algorithm on nVidia GeForce 6800, 7800 GT, 7950 GX2 and an 8800 GTX (table (1)). There is a clear break in performance between the nVidia 7 series and the 8 series.

Graphics card	Frame-rate
GeForce 6800	3-5 fps
GeForce 7800 GT	3-8 fps
GeForce 7950 GX2	3-13 fps
GeForce 8800 GTX	17-30 fps

Table 1: *Approximate frame-rate ranges for our approach on various graphics cards running at 1024x768. The scene consists of around a million polygons as shown in (fig. 1(c)).*

The results show that our algorithm is best suited for the upcoming and future hardware; we experience a worst-case frame-rate of around 17 fps in a GeForce 8800 GTX which maps to about 3 fps on a GeForce 7950 GX2. On an average, our algorithm is capable of sustaining the standard 25-30 fps for a 500,000 polygon scene with a number of deforming and dynamic characters on the newer GeForce 8800 GTX.

We make a number of approximations. In the image-space approach, we ignore distant occluders. The quality, accuracy and the performance depend on the choice of r_{far} value which directly affects the number of samples gathered in the neighborhood of a particular pixel. As such, our trade-off is quality/performance for physical accuracy. We also use only the first two layers as seen from the camera; the trade-off in this case is minor as explained previously (Section 4.2), losing only little physical accuracy and quality.

In the distant-occluder approach, we ignore the complexity of the object. We might also experience over-occlusion problems due to a number of occluders in a particular direction. As we disregard even large occluders past a certain distance, we might lose some of the the physical accuracy of ground-truth ambient occlusion (fig. 5). The same occurs when there are a significant number of small occluders at a large distance. Our trade-off is to obtain the best perceptual visual quality and performance for physical accuracy. In the combined approach, we only use two user-defined constants viz., r_{far} and ε which directly affect the quality and performance.

8 Conclusion

In conclusion, we have provided an approximation to ambient occlusion by separating near and far ambient occlusion. Our approximation demonstrates good quality and performance. Furthermore, it is suitable for dynamic, deforming objects with substantial geometric complexity. By observing the results obtained using GeForce 7 and 8 series, we believe that our algorithm is well-adapted for the current and upcoming hardware.

9 Acknowledgments

This research was partially funded by the Undergraduate Research Opportunities Program (Summer 2006) and an Undergraduate Research Fellowship (Fall 2006). We would like to thank Gregory S Johnson and the Texas Advanced Computing Center for supporting our research and providing various resources; Vinay K Siddavanahalli of Computational Visualization Center for providing the molecular datasets; Kun Zhou of Microsoft Research (Asia) for the animation datasets; Sony Computer Entertainment America for their motion capture datasets; nVidia Corporation for providing the GeForce 8800 GTX card; Meghna Gopalakrishnan, Gregory S Johnson, Paul A Navrátil, and the anonymous reviewers for their valuable suggestions and comments.

References

ARIKAN, O., FORSYTH, D. A., AND O'BRIEN, J. F. 2005. Fast and Detailed Approximate Global Illumination by Irradiance Decomposition. In Proceedings of ACM SIGGRAPH 2005, ACM Press, Volume 24, Issue 3.

BAJAJ, C., DJEU, P., SIDDAVANAHALLI, V., AND THANE, A. 2004. TexMol: Interactive Visual Exploration of Large Flexible Multi-component Molecular Complexes. In Proceedings of IEEE Conference on Visualization, 243–250.

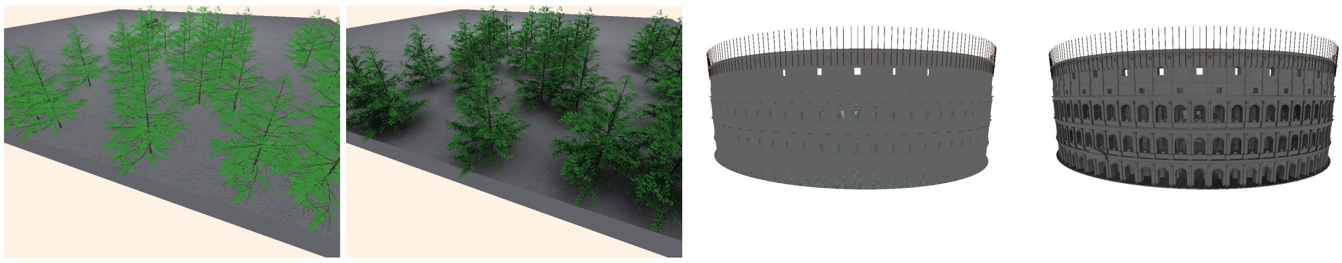


Figure 10: *(a) and (c) show the effect without any ambient occlusion. (b) and (d) show the effect when ambient occlusion using our method is applied to this scene. These images are obtained in real-time on a modern GPU.*

BOTSCH, M., HORNUNG, A., ZWICKER, M., AND KOBBELT, L. 2005. High-quality surface splatting on todays GPUs. Proceedings of Symposium on Point-Based Graphics, 17–24.

BRADSHAW, G., AND O'SULLIVAN, C., 2002. Sphere-tree construction using dynamic medial axis approximation.

BUNNELL, M. 2005. Dynamic Ambient Occlusion and Indirect Lighting. GPU Gem2, NVidia Corporation, 223–234.

DACHSBACHER, C., AND STAMMINGER, M. 2006. Splatting indirect illumination. In SI3D '06: Proceedings of the 2006 symposium on Interactive 3D graphics and games, ACM Press, New York, NY, USA, 93–100.

EVERITT, C. 2001. Interactive order-independent transparency. White paper, NVidia 2, 6, 7.

HEGEMAN, K., PREMOZE, S., ASHIKHMIN, M., AND DRETTAKIS, G. 2006. Approximate ambient occlusion for trees. In Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, C. Sequin and M. Olano, Eds., ACM SIGGRAPH.

JAMES, D. L., AND FATAHALIAN, K. 2003. Precomputing interactive dynamic deformable scenes. ACM Transactions on Graphics 22, 3, 879–887.

KONTKANEN, J., AND AILA, T. 2006. Ambient occlusion for animated characters. In Rendering Techniques 2006 (Eurographics Symposium on Rendering), T. A.-M. Wolfgang Heidrich, Ed., Eurographics.

KONTKANEN, J., AND LAINE, S. 2005. Ambient Occlusion Fields. In Proceedings of ACM SIGGRAPH 2005 Symposium on Interactive 3D Graphics and Games, ACM Press, 41–48.

MILLER, G. 1994. Efficient algorithms for local and global accessibility shading. Proceedings of the 21st annual conference on Computer graphics and interactive techniques, 319–326.

REN, Z., WANG, R., SNYDER, J., ZHOU, K., LIU, X., SUN, B., SLOAN, P.-P., BAO, H., PENG, Q., AND GUO, B. 2006. Real-time soft shadows in dynamic scenes using spherical harmonic exponentiation. ACM Trans. Graph. 25, 3, Volume 25, Issue 3, 977–986.

SHADEVIS. 2006. MeshLab http://meshlab.sourceforge.net.

SLOAN, P., KAUTZ, J., AND SNYDER, J., 2002. Precomputed Radiance Transfer for Real-time rendering in Dynamic, Low-frequency Lighting Environments.

SLOAN, P.-P. 2006. Normal Mapping for Precomputed Radiance Transfer. In Proceedings of ACM 2006 Symposium in Interactive 3D Graphics and Games.

TARINI, M., CIGNONI, P., AND MONTANI, C. 2006. Ambient occlusion and edge cueing to enhance real time molecular visualization. IEEE Transaction on Visualization and Computer Graphics 12, 6 (sep/oct).

WANG, R., ZHOU, K., SNYDER, J., LIU, X., BAO, H., PENG, Q., AND GUO, B. 2006. Variational sphere set approximation for solid objects. Submitted to Pacific Graphics.

WASSENIUS, C. 2005. Accelerated Ambient Occlusion Using Spatial Subdivision Structures.