Approximate Ambient Occlusion For Trees

Kyle Hegeman
Stony Brook University

Simon Premože
Stony Brook University

Michael Ashikhmin
Stony Brook University

George Drettakis
REVES/INRIA Sophia-Antipolis

Abstract

Natural scenes contain large amounts of geometry, such as hundreds of thousands or even millions of tree leaves and grass blades. Subtle lighting effects present in such environments usually include a significant amount of occlusion effects and lighting variation. These effects are important for realistic renderings of such natural environments; however, plausible lighting and full global illumination computation come at prohibitive costs especially for interactive viewing. As a solution to this problem, we present a simple approximation to integrated visibility over a hemisphere (ambient occlusion) that allows interactive rendering of complex and dynamic scenes. Based on a set of simple assumptions, we show that our method allows the rendering of plausible variation in lighting at modest additional computation and little or no precomputation, for complex and dynamic scenes.

1 Introduction

Synthetic natural scenes contain very large amounts of geometry. As an example, a field can contain several trees, each with hundreds of thousands of leaves, and millions of grass blades. Although it is now possible to render animated versions of such scenes using current graphics hardware, the lighting models used are usually very basic, typically involving point or directional hardware light sources, and simple local illumination.

We are interested in the illumination of such scenes from the skydome. Since such environments usually include a significant amount of mutual occlusion, for example between leaves or grass blades, subtle lighting effects are very visible. Evidently, a true global illumination solution (e.g., using path tracing or photon mapping [Dutre et al. 2003]) would correctly simulate such effects. The prohibitive computational costs of such approaches, especially in the case of scenes with moving objects, precludes their usage in an interactive setting.

One approximation to full global illumination which has become quite popular is ambient occlusion [Christensen 2002; Landis 2002], which we present in more detail below. In a nutshell, for every pixel we can compute how much of the skydome is visible, and modulate illumination by this percentage. This corresponds to outdoors illumination on a cloudy day. This approach is easy to understand, and thus has apparently become popular with artists in the production of films or video games. Most ambient occlusion based methods, however, require an expensive preprocessing step. For example, ray-casting in a large number of directions towards the sky for each visible point is frequently used. As a result, this approach cannot be used for interactive viewing of dynamic scenes.

In this paper we propose a simple and efficient approximation to ambient occlusion, which is effective in producing the desired visual effect and allows interactive viewing of complex and dynamic natural outdoor scenes. We emphasize that our main goal is not physical accuracy, but a plausible and convincing effect at acceptable computational cost. Such a goal is particularly appropriate for games applications, where visual quality is important, even if the image is not completely physically accurate.

We develop our approximation based on a statistical argument in a simple geometric configuration, which is similar to tree leaves contained in a sphere, but with several simplifying assumptions. We will show that for such an environment, the approximation is close to the true solution. For more complex configurations our approximation can vary, sometimes significantly, from the true solution. Nonetheless, our method preserves a convincing and compelling visual appearance.

We argue that our approach provides a way to give an interactive "ambient occlusion effect" for complex scenes of grass and trees, including moving objects, such as grass moving in the wind. We believe that our technique can be easily integrated into any interactive application, such as games or VR systems (e.g., driving simulators).

2 Background and previous work

If a virtual camera is looking in direction $\vec{\omega}_e$, the reflected radiance from the scene at point $\mathbf{x}$ (with surface normal $\vec{n}$) due to illumination from a skydome is

$$L(\mathbf{x},\vec{\omega}_e) \;=\; \int_{\vec{\omega}\in\Omega} L(\mathbf{x},\vec{\omega}) f_r(\mathbf{x},\vec{\omega}_e,\vec{\omega}) V_{\mathbf{x}}(\vec{\omega})(\vec{\omega}\cdot\vec{n}) d\omega \qquad (1)$$

where f_r is the BRDF of the surface point, and $V_{\mathbf{x}}(\vec{\omega})$ is the visibility function at the point $\mathbf{x}$ in the direction $\vec{\omega}$, which is 1 if the point sees the sky in this direction and 0 otherwise. If the BRDF f_r is Lambertian and we ignore occlusion by objects in the scene ($V_{\mathbf{x}} = 1$), the above equation can be simplified to

$$L(\mathbf{x},\vec{n}) \approx f_r(\mathbf{x}) \int_{\vec{\omega}\in\Omega} L(\vec{\omega})(\vec{\omega}\cdot\vec{n}) d\omega \qquad (2)$$

with the radiance $L(\mathbf{x},\vec{\omega})$ is now becoming just a function of the surface normal $\vec{n}$ for a given point $\mathbf{x}$. Shading a point with surface normal $\vec{n}$ by illumination from an environment map using this equation involves a simple texture lookup if the above integral has been precomputed and stored in a texture map. This approach has been used in recent work [Kautz et al. 2000], but the above approximation does not take shadows into account.

Two earlier ideas of accessibility maps and obscurances are related to *ambient occlusion* (which we define more precisely below). In the work of [Miller 1994], "accessibility" is computed based on local surface properties such as curvature. This results in an approximation of occlusion due to the local features of an object, giving aesthetically pleasing renderings which greatly enhanced contrast. Obscurance shading [Zhukov et al. 1998] can be seen as a precursor to ambient occlusion. Here visibility is approximated as a function of distance; the result is visually convincing, but does not offer any guarantee of accuracy.

Ambient occlusion was introduced by [Christensen 2002] and [Landis 2002] and incorporates effects of shadowing in Eq. 2. The ambient occlusion approximation is as follows:

$$L(\mathbf{x},\vec{n}) = \left(\frac{1}{2\pi}\int_{\vec{\omega}\in\Omega} V_{\mathbf{x}}(\vec{\omega}) d\omega\right)\left(f_r(\mathbf{x})\int_{\vec{\omega}\in\Omega} L(\vec{\omega})(\vec{\omega}\cdot\vec{n})\right) \qquad (3)$$

The first integral is *ambient visibility* which can be precomputed and stored in another texture map. Landis [Landis 2002] also computes the average direction (or least occluded direction) which is

used in the environment map lookup instead of the true surface normal. A hardware version of this was presented in [Pharr 2004].

As we shall see, our approximation uses some specific assumptions concerning trees and grass. The work of [Reeves and Blau 1985] can be seen as a precursor to our work. In their approach, ambient light on tree leaves is attenuated by an approximation of leaf self shadowing, which is based on the depth of the point being shaded inside the tree. The intuition used is that points deep in the tree are darker by an exponential factor of the depth. Despite some similarity to our approach, this paper does not attempt to explicitly approximate ambient occlusion, and as we shall see (Figure 12), gives significantly less convincing results than our method.

There have been several methods which improve the speed of obscurance or ambient occlusion calculations. [Mendez et al. 2003] recompute obscurance values in the context of radiosity-like secondary diffuse illumination computations; the method requires subdivided environments and could not handle the type of complexity we address here. [Sattler et al. 2004] compute visibility at vertices by sampling the environment and utilizing temporal coherence. The approach uses hardware occlusion queries in a vertex-based approach, which would probably not provide an acceptable quality/speed tradeoff for the very complex scenes of vegetation and trees we target.

Precomputed radiance transfer [Sloan et al. 2002] is also related to ambient visibility, but computes similar quantities at a high precomputation and storage cost. This approach generally results in a more accurate solution than the approaches cited previously.

[Kautz et al. 2004] use a hemispherical rasterizer to compute a 2D occlusion mask for visibility to achieve interactive rendering rates. [Bunnell 2005] presented an adhoc method for computing ambient occlusion and one-bounce global illumination for dynamic objects. As with the other approaches cited above, these two solutions can handle relatively small dynamic models, but are not adapted to our large outdoor scenes. Finally, [Kontkanen and Laine 2005] use a precomputation step to compute occlusion which allows real-time viewing. The precomputation step is in the order of minutes for relatively small models.

3 Visibility approximation

As mentioned above, we do not necessarily seek a physically accurate solution for ambient occlusion, but only one which is *visually plausible*. For the very geometrically complex scenes we consider, the resources of even the most powerful rendering systems are already stretched just to push through enough geometry with even the most basic lighting model. We thus require very simple additional computational overhead, and we preclude precomputation since we want to treat dynamic scenes and avoid extra storage.

Most of the difficulty in computing ambient occlusion is due to its *global* nature - even objects arbitrarily far away can potentially change sky dome visibility from a given point. Our solution is to apply an aggressive approximation to the problem and replace global visibility computation with a purely *local* expression.

During a typical shading computation, the only geometric information readily available for a given point to be shaded is its position in space $\mathbf{x}$ and normal direction $\vec{n}$. Ambient visibility V will therefore be written as a function of only these variables: $V = V(\mathbf{x},\vec{n})$. Since we cannot compute V for a specific scene, we take a statistical approach, i.e., we consider the value of V averaged over many instances of similar scenes, which we write $\langle V \rangle$.

We develop our approximation on a simple environment consisting of a single large sphere filled with small occluders according to some distribution, which is a reasonable approximation for many types of trees. We can create multiple instances of such an environment, with constant sphere radius and position, and in each of them compute visibility $V_i(\mathbf{x},\vec{n})$ at a given point $\mathbf{x}$ with normal $\vec{n}$.

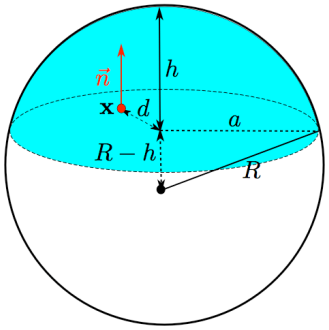

Figure 1: Spherical cap geometry as defined by a point $\mathbf{x}$ and normal $\vec{n}$. The approximate visibility is determined by spherical cap volume V_{cap}, density of occluders ρ and off-center distance d.

This quantity will, of course, be different for each sphere created. However, the average visibility over this ensemble will converge to the value $V(\mathbf{x},\vec{n}) = \langle V_i(\mathbf{x},\vec{n}) \rangle$ we are interested in. Since averages over an ensemble and over distribution are equivalent, one could in principle derive $V(\mathbf{x},\vec{n})$ if the shape and all parameters of occluder distribution are known. However, obtaining useful simple closed form expressions is unlikely even for the simplest distributions such as uniform.

In reality, the occluder distribution inside a canopy of a tree is, of course, much more complex than uniform. To obtain a practical solution, we propose to estimate $V(\mathbf{x},\vec{n})$ for a large, near-spherical approximation of a tree as follows.

We first enclose the complete tree inside a spherical shell and obtain its position and radius. This is the only precomputed information we use and is just a simple spherical bounding box computation. At runtime, for a visible point $\mathbf{x}$ to be shaded, a plane in world space is defined by the point and the normal vector (see Figure 1). The volume of the sphere segment (V_{cap}) in the direction of the surface normal is then computed as:

$$V_{cap} = \frac{1}{3}\pi h^2(3R-h) \tag{4}$$

where h is the height of the segment and R is the sphere radius. For a uniform occluder distribution, this volume is proportional to the total number of occluders affecting the given point. A point closer to the sphere boundary should be less occluded than a point on the same plane but closer to the sphere center. An additional factor f accounts for such off-center effects:

$$f = \frac{h^2}{(h^2+d^2)} \tag{5}$$

where d is off-center distance in the plane. f is the (squared) cosine of the "off-center angle" for the given point. The visibility approximation we use is then:

$$V = \exp(-\alpha\rho f V_{cap}) \tag{6}$$

where α is a constant which can be adjusted by the user and ρ is occluder density.

For a point at the center of a disk cutting through the sphere ($f = 1$), ambient visibility simply gives the average probability (over directions $\vec{\omega}$) of finding no occluders between the current point $\mathbf{x}$ and a point outside the tree. Equation 6 then corresponds to the probability of finding zero occluders in a Poisson random process with intensity proportional to the number of occluders affecting the current point. A more detailed discussion of visibility in random fields can be found in the mathematical literature [Zacks 1994; Hall 1988].

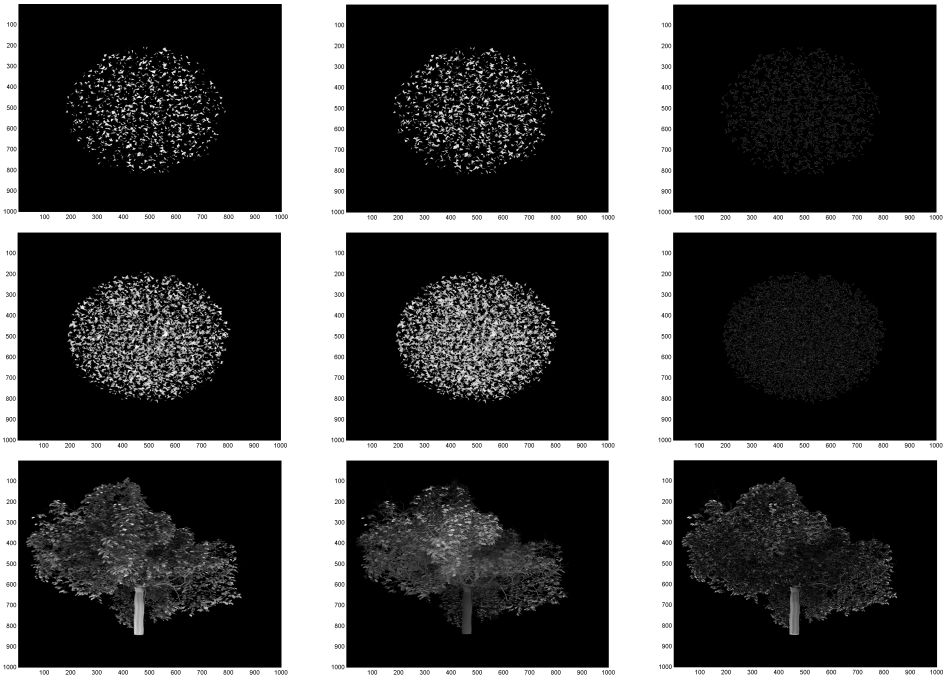

Figure 2: Comparisons between true occlusion and our approximation. Left: true image Middle: our approximation Right: absolute difference

In the first two rows of Figure 3, we show comparisons of two spheres with uniform distributions of small triangles. The leftmost column is the true ambient occlusion, the middle column is the approximation described above and the righmost column is the absolute difference. As we can see, for the specific case of uniform distributions, the approximation works well. Evidently, as can be seen in the last row, the error for the case of a more general object (an apple tree), is much larger, but the overall appearance trends are still well preserved.

As underlined by this last example, the reasoning above simply provides some intuition behind the formulas we have chosen, which work well in practice, and is not intended as a rigorous derivation. Clearly other forms are possible, but both visual and numerical results show that our approach works well when we are close to our assumptions. For the general case, it delivers visually plausible results at the cost we can afford, despite potentially high numerical error.

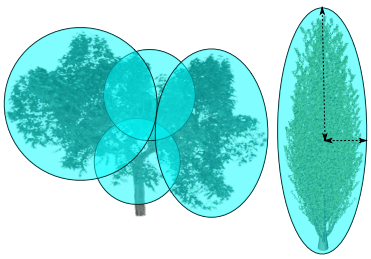

Figure 3: A tree can be approximated by multiple spherical shells (left) or by ellipsoid (right).

Not all trees can be approximated well by a spherical shell. For some an ellipsoid is a much better choice (Figure 3, right). In such cases, the same runtime computation is performed but only after corresponding scaling along the axes is applied, bringing the ellipsoidal shell into a sphere. This gives a new point position and cutting plane orientation. More involved approximations are briefly discussed later (Section 5).

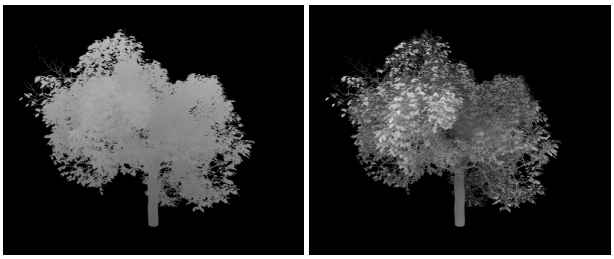

Figure 4: Comparison between Reeves and Blau method (left) and our method (right). Reeves and Blau parameters are ad hoc and do not correspond to parameters in our model. We show the best image we were able to produce with the Reeves and Blau method.

A comparison to the Reeves and Blau technique [Reeves and Blau 1985] for a single tree is shown in Figure 4. As we can see, our method has significantly richer visual detail, compared to the somewhat smooth solution provided by their approximation.

Interaction with Objects. The model presented so far computes occlusion only for a single object in isolation. A tree can also affect visibility for objects such as ground or nearby moving objects. The simplest approximation to computing ambient occlusion is to find the solid angle of large occluders (trees in our case).

The solid angle subtended by the sphere of radius r at a point at distance d away from the center of the sphere is approximately

$$A_{SA} = \pi(r/d)^2 \tag{7}$$

The ambient occlusion due to large occluders is then

$$V = 1.0 - (A_{SA}/2\pi) \tag{8}$$

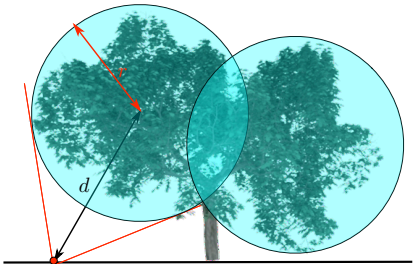

Figure 5: Tree ground interaction. A large occluders affects visibility on the ground. The distance to the occluder and the solid angle it covers are used to compute approximate occlusion.

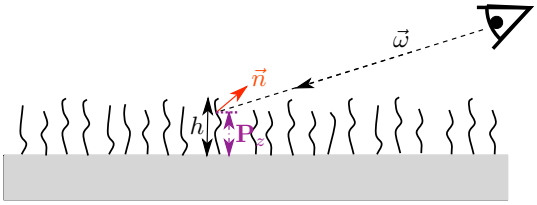

Figure 6: Approximate visibility computation for a grass field. The visibility is affected by the height of the hit point and orientation.

A large occluder can be represented by several spheres of various sizes (see Figure 5). In our experience with outdoor scenes, a single sphere is often enough to convey the effects of large occluders on nearby objects. More accurate inter-object shadowing due to distant illumination can be computed using other methods, see [Mei et al. 2004; Kontkanen and Laine 2005].

Visibility approximation for grass. In the case of a grass field (see Figure 6), integrated visibility is given as follows:

$$V = \frac{P_z}{2h}(1+\cos\theta) \tag{9}$$

where P_z the height of the hit point on the grass blade and h is the total height.

4 Examples

The visibility approximation from the previous section has been implemented in both a software-based renderer (raytracer) and graphics hardware. We show some examples of our test system, running on an Intel Xeon 2.4GHz PC with a 256MB NVIDIA GeForce 6800 Ultra graphics card. All the images in the paper are rendered using the 8xS antialiasing mode. The density and size of occluders ($\alpha\rho$ product) have been chosen by hand for all examples; this is a simple once-per-object operation. In the future, a simple one-time preprocessing step could compute these two parameters automatically and provide the user with an initial guess for further fine-tuning.

We used the NVShaderPerf tool from NVidia to analyze the theoretical performance of our fragment shaders. This tool computes the pixel throughput of the fragment pipeline measured in megapixels per second (MP/s). We evaluated each of our three shaders: our approximation, Reeves and Blau, and simple ambient. For a 6800 Ultra card, the resulting throughput numbers are 460 MP/s, 580 MP/s, and 2300 MP/s respectively.

Figure 7 compares results using our ambient occlusion approximation with that obtained with ambient illumination only. A single sphere was used to approximate the shape of the tree. At 512x512 image size, we can render a single 550k triangle tree mesh at 23 frames per second using our approximation. By comparison, ambient runs at 31 fps and the Reeves and Blau approximation at 25 fps. These numbers correspond well to the expected performance ratio as computed by NVShaderPerf. Figure 3 (bottom row) shows the difference between the approximate ambient occlusion and a ground truth occlusion computed using a standard raytracer (128 directions over a hemisphere) for this scene.

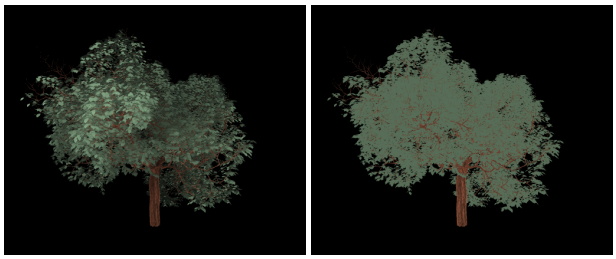

Figure 7: Comparison between true approximated ambient occlusion (left) and ambient only approximation.

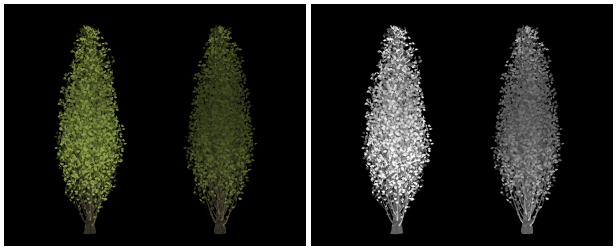

Figure 8: "Ellipsoidal" tree. This tree is not well approximated by a single sphere or multiple shells. The tree on the left uses an ellipsoid while the tree on the right uses a spherical shell approximation.

The effectiveness of the ellipsoidal approximation for some trees is illustrated by Figure 8. The difference between using a single spherical shell and a single ellipsoid is prominent. An example of applying the approximation for the interaction of the three with the ground plane is shown in Figure 9. A grass field rendered using the approximation described in Equation (9) is shown in Figure 10.

Figure 12 shows two vantage points (image resolution 800x500) of a complex outdoors scene containing grass and a several different types of plants and trees. This scene contains over 22 million vertices and 12 million triangles and was generated by an ecosystem modeling program [Deussen et al. 1998]. Our current OpenGL-based implementation only performs very basic optimizations. Due to the geometric complexity of this scene, the bottleneck in the graphics pipeline is not in the fragment stage. For this reason, the rendering speed is about 5 frames per second for all three fragment shaders we compare. Performance could be improved significantly by using an LOD scheme such as [Deussen et al. 2002] or one of many other acceleration techniques. Our approach is independent from, and can be used together with, any such method. In addition, any increase in future graphics hardware performance will be directly reflected on the frames rates cited here.

The middle row of Figure 12 shows that the Reeves and Blau [Reeves and Blau 1985] approach gives a pleasing, smooth image, which is appropriate as a replacement to an ambient term. However, the visual result of our approximation (top row) is much closer to what one would expect from an ambient occlusion solution, see Figure 11. In particular, there is a noticable gain in the visual quality of the trees.

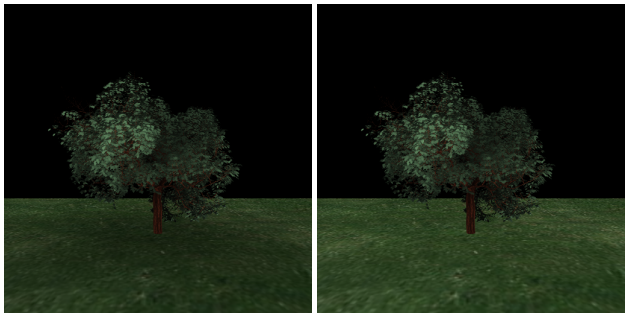

Figure 9: Interaction between the tree and the ground. Using our approximation (left) and no interaction (right).

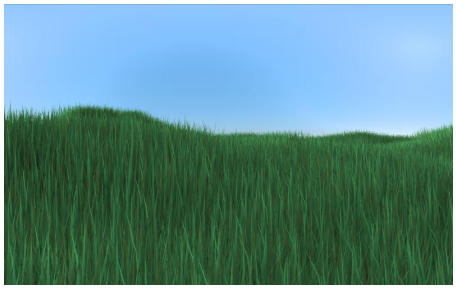

Figure 10: Grass field rendered using our approximation.

5 Conclusion

Our goal at the outset was to provide a simple approximation to ambient occlusion, allowing plausible and compelling skydome-like lighting for complex and dynamic outdoors scenes containing trees and grass at interactive rates.

To achieve this goal, we developed an approximation based on assumptions which hold in a very simple configuration. We have illustrated that our approximation gives quite accurate results for these simple configurations, and that it provides plausible visual results for the more general case, even though it can result in large numerical error.

Our method requires virtually no precomputation, and is sufficiently fast to interactively render scenes with moving geometry. In the images and the accompanying video, we show that we can walk through complex and dynamic outdoors scenes at interactive rates.

Given our self-imposed constraints we clearly needed a very aggressive approximation approach such as that presented here. Even though we illustrate that the heuristic chosen does have some justification in the simple case, for the general case we have no guarantee on error. Our argument is that the resulting visualizations are nonetheless compelling, and sufficient for example in the context of interactive environments such as games, since the user is not sensitive to the *type* of error which our approximation incurs. In the results presented here, we only provide subjective evidence of this claim.

In future work, an even better approximation of tree shape is possible by combining several, usually relatively few, smaller spherical (or ellipsoidal) shells (Figure 3, left). For simplicity, we can consider a point to belong to exactly one such shell (that with the closest center) even if geometrically it lies in several shells simultaneously.

Creating the set of shells and localizing a point to a specific one does require some additional precomputation and runtime processing. In fact, a complete hierarchy of such shells of different sizes on multiple levels can be built with any point clustering method. If the performance of the system allows it, visibility can also be computed on several levels and the results combined.

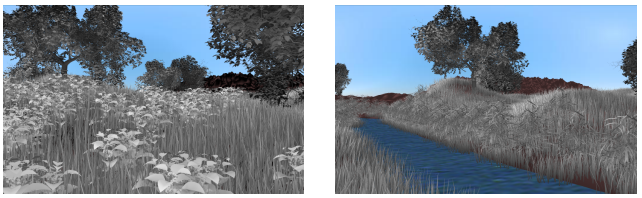

Figure 11: Our ambient occlusion approximation for complex ecosystem scene.

Finally, we would also like to conduct a user study to determine the level of "believability" of such approximations: we would present users with a walkthrough of the exact solution and that of the approximation, and ask them to rate "perceived realism". We are fairly confident that for a non-expert user, the difference will be low.

Acknowledgements

The ecosystem scene was generated using XFrog modelling software (www.greenworks.de). The last author's participation was funded in part by the French Ministry of Research ACI MD project ŠHOW.

References

BUNNELL, M. 2005. *GPU Gems 2*. Addison-Wesley, ch. Dynamic Ambient Occlusion and Indirect Lighting.

CHRISTENSEN, P., 2002. Ambient occlusion, image-based illumination and global illumination. Photorealistic RenderMan Application Notes.

DEUSSEN, O., HANRAHAN, P. M., LINTERMANN, B., MECH, R., PHARR, M., AND PRUSINKIEWICZ, P. 1998. Realistic modeling and rendering of plant ecosystems. In *Proceedings of SIGGRAPH 98*, 275–286.

DEUSSEN, O., COLDITZ, C., STAMMINGER, M., AND DRETTAKIS, G. 2002. Interactive visualization of complex plant ecosystems. In *Proceedings of the IEEE Visualization Conference*.

DUTRE, P., BEKAERT, P., AND BALA, K. 2003. *Advanced Global Illumination*. AK Peters.

HALL, P. 1988. *Introduction to the Theory of Coverage Processes*. Wiley, NY.

KAUTZ, J., VÁZQUEZ, P.-P., HEIDRICH, W., AND SEIDEL, H.-P. 2000. A unified approach to prefiltered environment maps. In *Rendering Techniques 2000: 11th Eurographics Workshop on Rendering*, 185–196.

KAUTZ, J., LEHTINEN, J., AND AILA, T. 2004. Hemispherical rasterization for self-shadowing of dynamic objects. In *Rendering Techniques 2004: 15th Eurographics Workshop on Rendering*, 179–184.

KONTKANEN, J., AND LAINE, S. 2005. Ambient occlusion fields. In *ACM Siggraph Symposium on Interactive 3D Graphics and Games*.

LANDIS, H., 2002. Ambient occlusion. Course 16: RenderMan in Production. ACM SIGGRAPH 2002 Course Notes.

MEI, C., SHI, J., AND WU, F. 2004. Rendering with spherical radiance transport maps. *Computer Graphics Forum 23*, 3 (Sept.), 281–290.

MENDEZ, A., SBERT, M., AND CAT, J. 2003. Real-time obscurances with color bleeding. In *Proceedings of the 19th Spring Conference on Computer Graphics*.

MILLER, G. 1994. Efficient algorithms for local and global accessibility shading. In *Proceedings of SIGGRAPH 94*, 319–326.

PHARR, M. 2004. *GPU Gems*. Addison-Wesley, ch. Ambient Occlusion.

REEVES, W. T., AND BLAU, R. 1985. Approximate and probabilistic algorithms for shading and rendering structured particle systems. In *Computer Graphics (Proceedings of SIGGRAPH 85)*, 313–322.

SATTLER, M., SARLETTE, R., ZACHMANN, G., AND KLEIN, R. 2004. Hardware-accelerated ambient occlusion computation. In *Vision, Modeling, and Visualization 2004*, 331–338.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics 21*, 3 (July), 527–536.

ZACKS, S. 1994. *Stochastic Visibility in Random Fields*, vol. 95 of *Lecture Notes in Statistics*. Springer-Verlag.

ZHUKOV, S., IONES, A., AND KRONIN, G. 1998. An ambient light illumination model. In *Eurographics Rendering Workshop 1998*, 45–56.

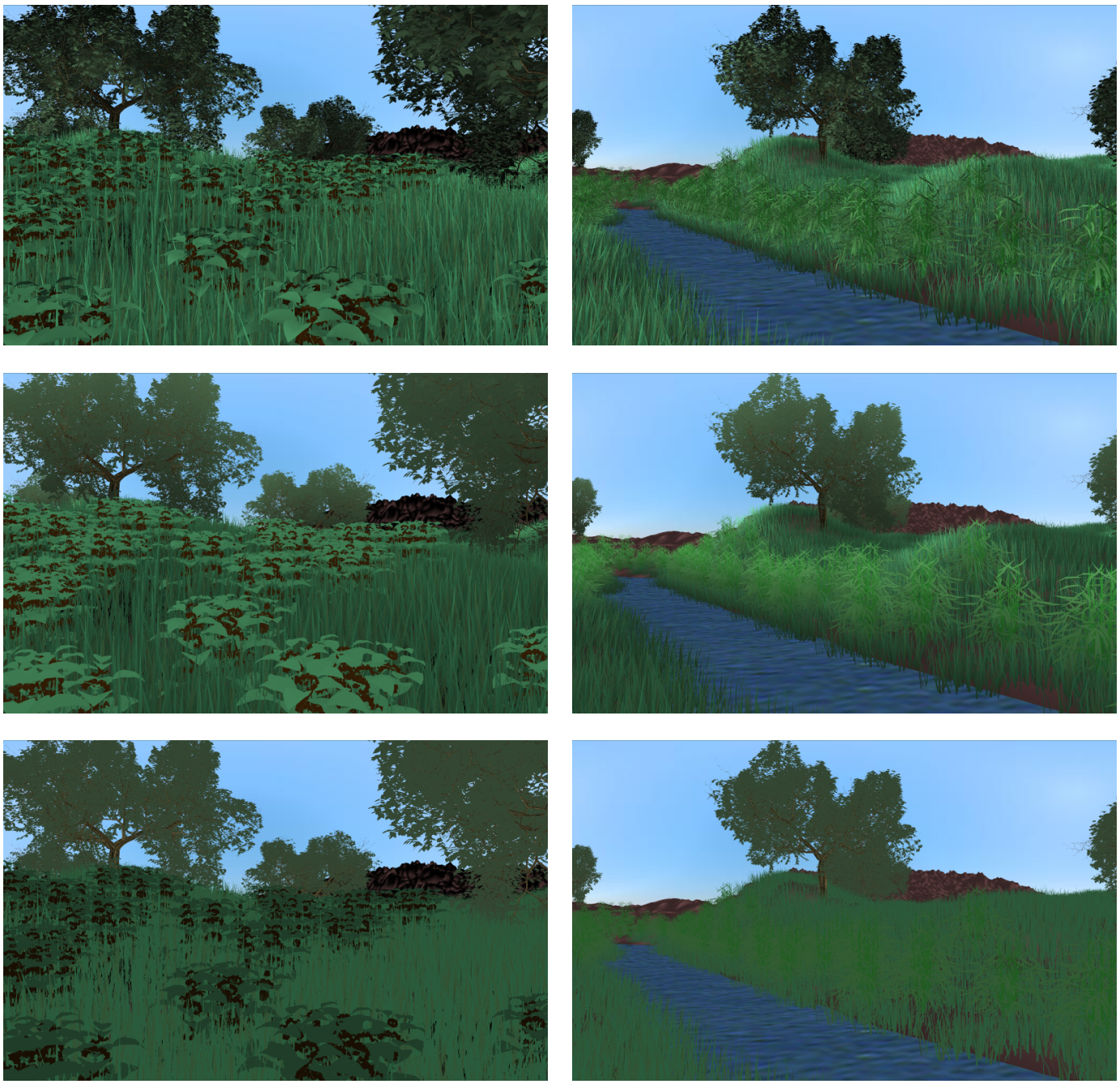

Figure 12: Two views of an ecosystem rendered with our ambient occlusion (top row), Reeves Blau approximation (center row) and ambient term only (bottom row). Note the improvement in quality in the trees with our approximation.