IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 15, 2021, accepted October 10, 2021, date of publication October 28, 2021,
date of current version November 8, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3124008

Chebyshev Ambient Occlusion

KA-HOU CHAN[1], (Member, IEEE), AND SIO-KEI IM[2], (Member, IEEE)
[1]School of Applied Sciences, Macao Polytechnic Institute, Macao, China
[2]Macao Polytechnic Institute, Macao, China

Corresponding author: Ka-Hou Chan (chankahou@ipm.edu.mo)

This work was supported by the Science and Technology Development Fund, Macau, under Grant 0001/2018/AFJ.

ABSTRACT Ambient Occlusion (AO) is a widely used shadowing technique in 3D rendering. One of the main disadvantages of using it is that it requires not only the surface depth but also the normal vector, which usually causes severe aliasing. This work introduces a novel AO algorithm by applying Chebyshev's inequality. Instead of requiring the normal vector, we process the mean (μ) of the distribution of depth values from the screen space. We can efficiently calculate the variance (σ^2) over any kernel region. Using the Chebyshev AO, we can get the upper bound on the percentage of the shaded surface that is occluded. Our proposed method can usually provide a good approximation of true occlusion and can be used as an approximate value for AO rendering. As a post-processing rending, the time complexity of Chebyshev AO is $O(2k)$ which is simple to implement on current graphics devices and can be applied to the next generation of ray tracing technology.

INDEX TERMS Ambient occlusion, Chebyshev's inequality, post-processing.

I. INTRODUCTION

Local illumination is usually used in real-time rendering, making it difficult to carry out the requirements of 3D vision. Especially for lighting and shadows, it is difficult to strike a balance between visualization and performance. The gaming industry is thriving thanks to technological advancements in hardware, processing power, and graphics. With new generation graphics devices, many gaming companies strive to deliver increasingly realistic ambient lighting. Many graphics settings can be used to make the game visually appealing and more dynamic, such as Ambient Occlusion (AO) [1]. AO is a compromise method to achieve limited global illumination. The idea is to measure how many occludes are in the effective range of the rendered surface [2]. AO rendering considers objects within a certain radius and assumes that the wall is the origin of the ambient light. The result is a non-directional shadow effect, which casts no clear shadows but will darken enclosed and obscured areas and may affect the overall tone of the rendered image. It is usually used as a post-processing effect for providing realistic simulation of occluded objects. These occluded objects will block the ambient light from different light sources and add realism to 3D scenes by simulating the distribution of light in reality (as indicated in Figure 1).

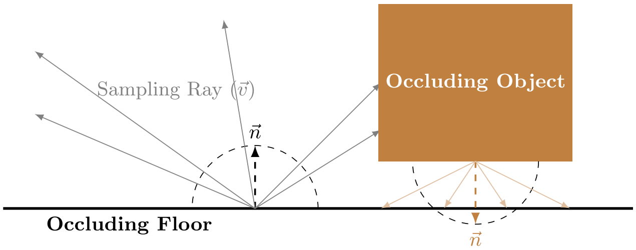


FIGURE 1. The occlusion rendering has considered the floor and a rectangular object. A hemisphere of sampling rays is emitted above each surface surround its normal vector ($\vec{n}$), and then test the amount of shadowing each point should receive.

The first introduction of AO in the real-time rendering of the game industry was in 2007 [3], to approximate the light intensity of internal parts of the visible light surface. It takes account into different factors, such as the game environment and the location of the light source.

A. RELATED WORK

However, considering that shadows are too demanding on graphics for many games with dynamic scenes, AO still needs a lot of calculations to achieve its sampling steps. To achieve this in real-time, one approach proposes to achieve the sampling of the depth buffer based on the current screen space, instead of the 3D scene space [4]. This is called Screen Space Ambient Occlusion (SSAO) [5]. As a post-processing

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Wang.

rendering, SSAO aims to save processing power. It does not consider the space and elements that produce shadows from each other, instead looking at the pixels around the elements and their depth. Rather than focusing on the entire screen, it measures the ambient occlusion in pixel depth at a local range of current screen. However, if the pixels are not measured correctly, it will bring some defects, such as noise and potentially inaccurate shadow distribution. In addition, [6] introduces the Horizon Based Ambient Occlusion (HBAO), which is an enhanced SSAO based on the horizon in the 3D scene. This approach solves the grain and noise caused by SSAO pixel depth measurement by considering the ambient and the environment instead of the depth buffer only, but it requires more processing power in the CPUs and in the GPUs. Later, [7] proposed to use a mixture of the HBAO sampling from multi-viewport screen space, which achieves a better render result but requires heavy computation. Further, [8] introduced the HBAO+ with the interleaved rendering method, which can render at a decent frame rate 2 to 3 times faster than HBAO at full resolution and overcomes flickering at one render-pass. Later, a similar technique High-Definition Ambient Occlusion (HDAO) was introduced to apply to graphics devices to compete with HBAO. HDAO increases the number of samples used when calculating the areas that should be darkened and renders at full resolution for a more accurate representation of AO. There is no difference between HBAO and HDAO from a visual analysis point of view, but they are native to their respective distributer's GPU. In terms of advanced lightening and shadows, Voxel Ambient Occlusion (VXAO) is the best approximation for realistic scenes [9]. VXAO is part of the Voxel Global Illumination (VXGI) [10] surround lighting technology, which more correctly considers direct and reflected light. Its performance depends on the total number of triangles, the size of the triangles, and the number of draw calls required to render the triangles. Post-processing involves the process of removing, filtering, and down-sampling voxels. Its performance depends on the total number of voxels generated during the voxelization process. Then cone tracking is performed in screen space, its performance depending on the shading rate and screen resolution. However, VXAO rendering requires too many render-passes, which will exceed the limit of the processor, and few graphics devices can support it in real time.

Inspired by these screen space AO methods, this article introduces a novel consideration scheme by using Chebyshev's inequality with its application results [11], [12]. Our method does not require the normal vector information about the surface of the object in the depth buffer, instead it uses the probability of the occlusive object to approximate the right brightness in the scene ambient. It is most suitable for processing those scenes with complexity and dense noise. The processing time of this render-pass is in $O\left(k^2\right)$ for each pixel, where k is the size in pixel of the local kernel. Since k is independent of the size of the scene, it can more effectively enhance the parallelism used by the GPUs while introducing dynamic rendering to make the ambient more realistic.

II. CHEBYSHEV AMBIENT OCCLUSION

The main idea of this method is to use a non-linear filter (applying Chebyshev's inequality) as post-processing to calculate the shadow intensity. In a depth buffer, the depth value information in a pixel is presented, and its neighbors can be grouped as a set of sample data for statistical analysis. These samples may initially have a completely arbitrary distribution, except for the mean and variance. We know nothing else about the pixels and want to find some data samples that interest us. In general, the probability of a pixel being a part of the shadow is often less than that of being a part of the light surface. Therefore, the concept of our proposed method is to use probability to approximate the shadow intensity.

A. CHEBYSHEV's INEQUALITY

As discussed in our previous work [12], Chebyshev's inequality is not suitable for this kind of pixel domain analysis, because this inequality usually gives a poor upper bound, since this bound gets increasingly weaker as the selected grey-value goes up it will cause the result to exceed the valid range. Therefore, we make use the variant form of Chebyshev's inequality, namely the one-sided Chebyshev's inequality [13] or Cantelli's inequality [14] as:

$$P\left(x-\mu \geq \lambda\right) \leq P_{max} \equiv \frac{\sigma^2}{\sigma^2+\lambda^2} \tag{1}$$

For any $\lambda > 0$, where x is a random depth in a pixel of samples with finite mean μ and variance σ^2. This inequality gives us a way to handle the extreme situation in which the only things we know about the samples, or the probability distribution, are the mean and the standard deviation. It implies that in any probability distribution, almost all values are close to the mean, and the chance of a value being outside that range is no more than $\frac{\sigma^2}{\sigma^2+\lambda^2}$. The advantage of Equation 1 is that it provides a stronger bound $P_{max} \in [0, 1]$. Since the distribution can be arbitrary, this reason can be applied in various general situations to obtain results in the valid range. Being more general, let $t = \lambda + \mu$ and x be a random variable drawn from a distribution with mean μ and variance σ^2, and then for $x \geq t$, Equation 1 can be rewritten to

$$P\left(x \geq t\right) \leq P_{max}\left(t\right) \equiv \frac{\sigma^2}{\sigma^2+\left(t-\mu\right)^2} \tag{2}$$

Suppose that t is the current depth to be considered about AO, then Equation 2 means there are no more than probability $P_{max}\left(t\right)$ of the samples in which the depth is equal to or more than t within its kernel range (k). Even for a scene without a normal vector, since it is just a comparison of the depth values without considering other information, the bound can still provide a good approximation of the amount of interested pixels within a given sample.

B. COMPARISON OF DEPTH PROBABILITY

Another reason why we do not choose the Chebyshev's inequality is that it only reflects the absolute difference. It can only find the probability of the sample x whose absolute difference of x is greater than λ, but cannot tell the probability of $x > \mu$ or $x < \mu$. In contrast, the one-sided Chebyshev's inequality can clearly tell us the probability that the sample is larger than t within kernel range (k). Therefore, we can use the depth value as t in Equation 2 to evaluate the states of its neighbors, and the result range is [0, 1]. Based on the discussion above, we can make use of this to find out the intensity of AO that are clustered around the mean.

For instance, consider the case of the occluded surface at depth d; suppose p is the percentage of the occluded surface, then $d_{\geq}$ and $d_{<}$ denote the depth of unoccluded and occluded surface, respectively. We have:

$$\begin{aligned} \mu = E(x) &= (1-p)\,d_{\geq} + p d_{<} \\ E\left(x^2\right) &= (1-p)\,d_{\geq}^2 + p d_{<}^2 \\ \sigma^2 &= E\left(x^2\right) - E^2(x) \\ &= (1-p)\,d_{\geq}^2 + p d_{<}^2 - \left((1-p)\,d_{\geq} + p d_{<}\right)^2 \\ &= p(1-p)\left(d_{\geq} - d_{<}\right)^2 \end{aligned}$$

Then we can find the intensity of AO according to Equation 2 as:

$$\begin{aligned} &P_{max}(d) \\ &= \frac{p(1-p)\left(d_{\geq} - d_{<}\right)^2}{p(1-p)\left(d_{\geq} - d_{<}\right)^2 + \left(d - p d_{\geq} + (1-p)\,d_{<}\right)^2} \\ &\geq \frac{p(1-p)\left(d_{\geq} - d_{<}\right)^2}{p(1-p)\left(d_{\geq} - d_{<}\right)^2 + (1-p)^2\left(d_{\geq} - d_{<}\right)^2} = p \end{aligned}$$

Therefore, by using one-sided Chebyshev's inequality, we can obtain a good approximation result: There is at least p (impact factor) effects into the AO render at the current depth d. This is also proportional to the percentage of the occluded surfaces. We can adjust the kernel range (k) to calculate the percentage in which the depth of the occluded surface is approximate. In general, it does not provide an exact value but a close approximation, thus we still use the P_{max} in rendering as an approximation to the true value p.

III. IMPLEMENTATION

We now give the implementation of the Chebyshev AO in detail, on a Nvidia GeForce GTX 1080 graphic device and OpenGL with GLSL. The proposed method requires multiple-pass rendering because it is a post-processing technology. The mean μ and variance σ^2 can be calculated by using a two-pass order to depth buffer with kernel range (k) as:

$$\begin{aligned} M_1 &= E(x) = \frac{1}{k^2}\sum_{k^2} x_i \\ M_2 &= E\left(x^2\right) = \frac{1}{k^2}\sum_{k^2} x_i^2 \end{aligned}$$

From the above, we can find the mean μ and variance σ^2 by:

$$\begin{aligned} \mu &= E(x) = M_1 \\ \sigma^2 &= E\left(x^2\right) - E^2(x) = M_2 - M_1^2 \end{aligned}$$

In fact, the mean function $E(x)$ can be achieved by two mutually perpendicular blur filters, which can reduce the time complexity from $O\left(k^2\right)$ to $O(2k)$. As a result, the variance can be interpreted as a quantitative measure of the width of the distribution. It should place a bound on how much of the distribution can be concentrated far away from the mean. Therefore, this k should be adjusted according to the current d, as $k \propto \frac{1}{d}$. Once μ and σ^2 are determined, the impact factor (p) of AO can be calculated through Equation 2, but there are some obvious defects as shown in Figure 2d.

A. ADDITIONAL BIAS FOR VARIANCE

The most obvious defect is that there are strange effects like some stain inside this bowl. This is caused by the denominator being equal to zero in Equation 2, so its result becomes an undefined situation. The depth of each pixel on this plane is the same if its surface is parallel to the viewport screen, which implies:

$$\begin{cases} \mu = d \\ \sigma^2 = 0 \end{cases} \implies P_{max}(d) = \frac{0}{0 + (d-d)^2} = \frac{0}{0} = \mathbf{Undefined}$$

To overcome this, we recommend adding a bias (ε) to the variance and the issue can be solved by:

$$\begin{cases} \mu = d \\ \sigma^2 = \varepsilon \end{cases} \implies P_{max}(d) = \frac{\varepsilon}{\varepsilon + (d-d)^2} = \frac{\varepsilon}{\varepsilon} = 1$$

This bias can make it more stable, and the impact factor is equal to one, indicating that surface is facing to the camera, and no AO effect will be applied here (see Figure 2e).

B. IGNORE UNOCCLUDED SURFACES

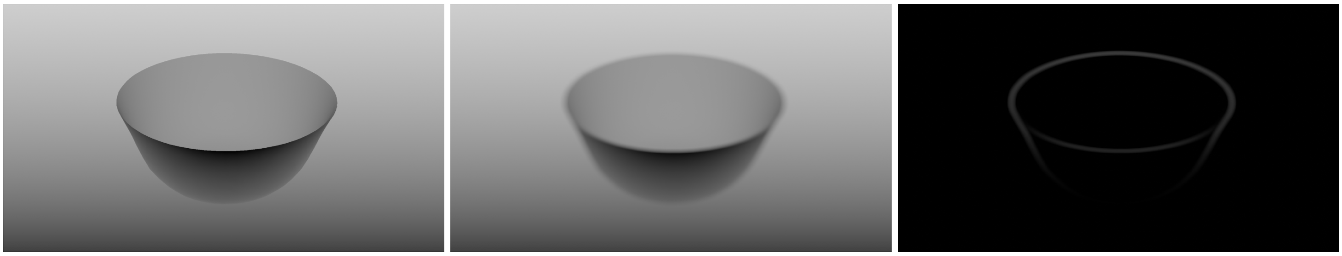
AO is only allowed to be generated on the occluded surface, but there are some abnormal shadows around the object boundary in Figure 2e. This is due to the comparison of depth $(t-\mu)^2$ in Equation 2. Because this formula only reflects the absolute difference in-depth, it does not care which one is deeper. Therefore, both the occluded surface and the unoccluded surface are considered to have an AO effect (This will be mistaken for outline rendering). One of the solutions is to modify the depth comparison as follows,

$$P_{max}(d) \equiv \frac{\left(\sigma^2 + \varepsilon\right)}{\left(\sigma^2 + \varepsilon\right) + \max(\mu - d, 0)^2} \tag{3}$$

Equation 3 corresponds to Figure 2f, the unoccluded surfaces $(d \geq \mu)$ is no longer affected by AO.

C. RANGE CHECK

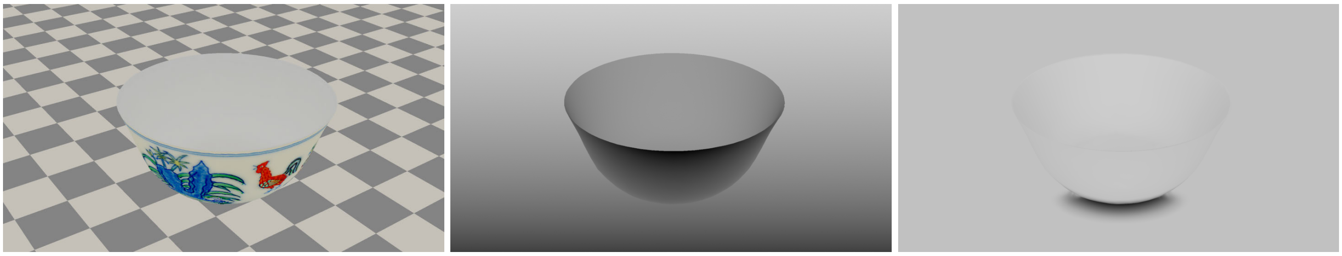
The remaining problem is to check the available depth range. It will also consider the depth value of the surface far away



(a) Screen space depth buffer. (b) Mean (μ) of depth buffer. (c) Variance (σ) of depth buffer.

(d) AO generated by Equation 2. (e) Additional bias for variance. (f) Ignore unoccluded surfaces.

FIGURE 2. **A bowl rendering with Chebyshev AO.**



(a) Texture color. (b) Depth buffer of screen. (c) (Proposed) Chebyshev AO.

(d) Color + AO. (e) Diffuse lighting. (f) Combined result.

FIGURE 3. **A simple shading pipeline for scene rendering from Figure 3a to Figure 3f.**

from the test surface whenever the ambient light occlusion is aligned near the edge of the surface of the test segment. These values will (incorrectly) contribute to the blocking factor. We can solve this problem by introducing range checking, as shown in the Figure 3c.

As illustrated in Figure 3, Figure 3a is rendered by original vertices and textures from the 3D model. Figure 3b can be obtained from Figure 3a's geometry information on the current screen coordinate; Figure 3c is produced from Figure 3b by processing the Chebyshev AO method we propose; Figure 3d is derived from the combination of Figure 3a and Figure 3c; Figure 3e has applied diffuse lighting in the scene environment; and Figure 3f is the final combined result. Furthermore, another high-quality deer model is indicated in the appendix.

IV. CONCLUSION

We have identified the unique challenges in all aspects of the AO rendering scene and the research opportunities related to it. We focus on the conversion of the one-sided Chebyshev's inequality and how it provides information for the probability of approximate shadow intensity; we also consider the

FIGURE 4. Different depth of available ranges.

FIGURE 5. Combine all rendering effects.

performance and quality trade-offs related to the goal of this algorithm, and discuss the distribution probability that reflects the Chebyshev AO scene rendering effect. We also provide the subtleties of applying appropriate solutions.

Furthermore, AO rendering is suitable for real-time and offline applications, and can also better approximate to the full global illumination. At the same time, AO can also be applied to the next generation of ray tracing technology, which can greatly reduce the computational cost of illumination sampling. Using this approach, the scene in the AO processing can not only understand the changes in depth of field but also enhance the authenticity of the scene. Therefore, efficient AO is not only suitable for the end-user application environment but also contributes to the design and development of 3D modeling.

APPENDIX

See Figures 4 and 5.

REFERENCES

[1] M. Pharr and S. Green, "Ambient occlusion," *GPU Gems*, vol. 1, pp. 279–292, Mar. 2004.
[2] J. Kontkanen and S. Laine, "Ambient occlusion fields," in *Proc. Symp. Interact. 3D Graph. Games (SI3D)*, 2005, pp. 41–48.
[3] M. Mittring, "Finding next gen: CryEngine 2," in *Proc. ACM SIGGRAPH Courses (SIGGRAPH)*, 2007, pp. 97–121.
[4] L. Bavoil and M. Sainz, "Multi-layer dual-resolution screen-space ambient occlusion," in *Proc. SIGGRAPH Talks (SIGGRAPH)*, 2009, p. 1.
[5] L. Bavoil and M. Sainz. (2008). *Screen Space Ambient Occlusion*. NVIDIA Developer Information. [Online]. Available: http://developers.nvidia.com
[6] L. Bavoil, M. Sainz, and R. Dimitrov, "Image-space horizon-based ambient occlusion," in *Proc. SIGGRAPH Talks*. New York, NY, USA: ACM, 2008, p. 22.
[7] Y. Song, F. Liu, and J. Xu, "Horizon-based screen-space ambient occlusion using mixture sampling," in *Proc. ACM SIGGRAPH ASIA Posters (SA)*, 2010, p. 1.
[8] L. Bavoil. *Hbao+*. Accessed: Oct. 28, 2021. [Online]. Available: https://www.nvidia.com/en-gb/geforce/technologies/hbao-plus/technology/
[9] (Aug. 2016). *Vxao: Voxel Ambient Occlusion*. [Online]. Available: https://developer.nvidia.com/vxao-voxel-ambient-occlusion
[10] S. Thiedemann, N. Henrich, T. Grosch, and S. Müller, "Voxel-based global illumination," in *Proc. Symp. Interact. 3D Graph. Games (I3D)*, 2011, pp. 103–110.
[11] W. Donnelly and A. Lauritzen, "Variance shadow maps," in *Proc. Symp. Interact. 3D Graph. Games (SI3D)*, 2006, pp. 161–165.
[12] K.-H. Chan, S.-K. Im, and W. Ke, "Fast binarisation with Chebyshev inequality," in *Proc. ACM Symp. Document Eng.*, Aug. 2017, pp. 113–116.
[13] B. K. Ghosh, "Probability inequalities related to Markov's theorem," *Amer. Statist.*, vol. 56, no. 3, pp. 186–190, 2002.
[14] F. P. Cantelli, "Sui confini della probabilita," in *Atti del Congresso Internazionale Dei Matematici: Bologna Del 3 al 10 de Settembre Di 1928*, 1929, pp. 47–60. [Online]. Available: https://dialnet.unirioja.es/servlet/articulo?codigo=3183299

KA-HOU CHAN (Member, IEEE) received the bachelor's degree in computer science from the Macau University of Science and Technology, in 2009, and the master's degree from the Faculty of Science and Technology, University of Macau, in 2015. He is currently pursuing the Ph.D. degree with the School of Applied Sciences, Macao Polytechnic Institute. His research interests include algorithm analysis and optimization, parallel computing, video coding, neural networks, and computer graphics.

SIO-KEI IM (Member, IEEE) received the degree in computer science and the master's degree in enterprise information system from the King's College, University of London, U.K., in 1998 and 1999, respectively, and the Ph.D. degree in electronic engineering from the Queen Mary University of London (QMUL), U.K., in 2007. He was a Lecturer with the Computing Program, Macao Polytechnic Institute (MPI), in 2001. In 2005, he became the Operations Manager of the MPI-QMUL information systems research center jointly operated by MPI and QMUL, where he carried out signal processing work. He was promoted to a Professor with the MPI, in 2015. He was a Visiting Scholar with the School of Engineering, University of California, Los Angeles (UCLA) and an Honorary Professor of The Open University of Hong Kong.

• • •