# Poisson Disk Ray-Marched Ambient Occlusion

Gaël Sourimant Pascal Gautron Jean-Eudes Marvie
Technicolor Research & Innovation

Figure 1: Our ray-marched SSAO provides high-quality results even with few samples. *Dragon*, from left to right: 8, 13 and 24 samples (bottom: detailed view). It compares well to AO solutions. *Temple*, left: ray-traced AO (512 samples) ; right: our method (16 samples).

Ambient occlusion (AO) [Pharr and Green 2004] provides an approximation to global illumination. It can be computed in real-time if restricted to screen-space [Mittring 2007; Loos and Sloan 2010], at the cost of an approximation of solutions computed in geometric space, thus introducing a loss of quality.

We present a method for computing screen-space ambient occlusion (SSAO) at interactive rates, while preserving the original formulation of the standard ray-traced ambient occlusion. Based on ray-marching on graphics hardware, our method achieves interactive frame rates with results consistent to ray-traced solutions, even with few samples. Our SSAO computation is easily integrable into post-production and previsualization workflows, or in computer games.

## Ray-marching Poisson sampling directions

A classical method for estimating SSAO relies on Monte-Carlo sampling of the rendered depth map, and on the evaluation of the amount of occlusion around a given point. Contrary to most SSAO techniques, we aim at finding occluders in a geometrically plausible way. The whole method is summarized in Algorithm 1.

**Algorithm 1** Poisson-based Ray-marched SSAO

1. Store normal and eye-space positions in a first render pass
2. **for all** fragments **do**
3. Define a 2D rotation matrix applied to the sampling kernel
4. **for all** Poisson disk samples $\mathbf{s}_i$ **do**
5. Rotate the sampling direction $\omega_i$ in the eye $XY$ plane
6. Update $\omega_i$ as its projection on the $Z$-oriented hemisphere
7. Align $\omega_i$ to the normal-oriented hemisphere
8. March along $\omega_i$ to evaluate its contribution to the AO term
9. **end for**
10. Compute the final AO term as the normalized sum of the contributions from each $\mathbf{s}_i$
11. **end for**

For each pixel, we define a set of sampling directions $\omega_i$ from Poisson-Disk samples. These directions are settled in the normal-oriented hemisphere instead of the image-oriented one to avoid erroneous self-occlusions. The search for potential occluders is performed by ray-marching the pixel neighborhood in a depth map (Figure 2). The direction $\omega_i$ is uniformly stepped with an increasing distance from the fragment. Given the eye-space position of each step offset $\mathbf{s}_{\omega_j}$, we retrieve the corresponding scene position $\mathbf{s}_{t_j}$ *via* a projection into the image space and a texture lookup. If $\mathbf{s}_{\omega_j}$ is closer to the camera than $\mathbf{s}_{t_j}$, no occlusion is detected and the step distance is increased. Otherwise, an occlusion is detected and the marching process is aborted. This method does not provide any information about the occluder distance $\Delta_o$. We thus derive this information by assuming that the object surface is piecewise linear between two sampled offsets. In case of non occlusion, we store the value $\delta_j$ as the eye-space distance between $\mathbf{s}_{\omega_j}$ and $\mathbf{s}_{t_j}$. If

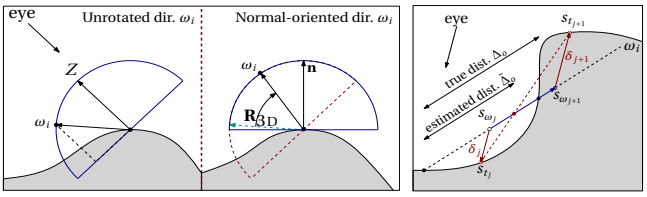


Figure 2: Left: projection of sampling directions in the normal-oriented hemisphere. Right: estimation of the occluder distance $\tilde{\Delta}_o$ using piecewise linear interpolation.

an occlusion is detected at the next step, the approximated distance $\tilde{\Delta}_o$ to the occluder is linearly interpolated as:

$$\tilde{\Delta}_o = \|\mathbf{s}_{\omega_j}\| + \frac{\delta_j \left(\|\mathbf{s}_{\omega_{j+1}}\| - \|\mathbf{s}_{\omega_j}\|\right)}{\delta_j + \delta_{j+1}} \quad (1)$$

Finally, the occluder distance is added to the current occlusion contribution and scaled by the number of samples. As in [Filion and McNaughton 2008], we allow the user to raise the final value to a power exponent $\alpha$ to modulate the contrast of the ambient occlusion map:

$$\mathcal{A}(\mathbf{p}) = \left(\frac{1}{N}\sum_i V_{\mathbf{p},\omega_i}\tilde{\Delta}_{o_i}\right)^{\alpha} \quad (2)$$

## Results

| | Mittring | Filion | Our Method | Ray-traced |
|---|---|---|---|---|
| *Temple* | 15ms / 0.65 | 38ms / 0.72 | 70ms / 0.85 | 351000ms |
| *Sibenik* | 14ms / 0.73 | 40ms / 0.77 | 77ms / 0.87 | 238000ms |
| *Back Streets* | 13ms / 0.68 | 16ms / 0.76 | 47ms / 0.85 | 220000ms |
| *Mediteranean* | 14ms / 0.67 | 16ms / 0.72 | 59ms / 0.86 | 238000ms |

The *Dragon* on Figure 1 illustrates the influence of the number of steps on the final AO map. As this number increases, the algorithm better captures high frequencies of local occlusions. However, since AO remains a low frequency phenomenon, there is no need to use a high number of steps to produce a physically plausible result.We compared our results against two common methods of the literature using 16 samples. The table above reports both the rendering times for several test scenes, together with a structural similarity measure (SSIM, the closer to 1 the better) of obtained results against reference ray-traced AO using 512 samples. Our SSAO computation provides results more consistent with reference ray-traced AO solutions, with only a small computation overhead compared to existing methods.Our solution can then be used for previsualization in post-production workflows, and even for post-production speed-up.

## References

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

LOOS, B. J., AND SLOAN, P.-P. 2010. Volumetric obscurance. In ACM *Symposium on Interactive* 3D *graphics and games*.

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

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