Screen-Space Ambient Occlusion in a Deffered Pipeline

Roel Deckers*
930830-T150

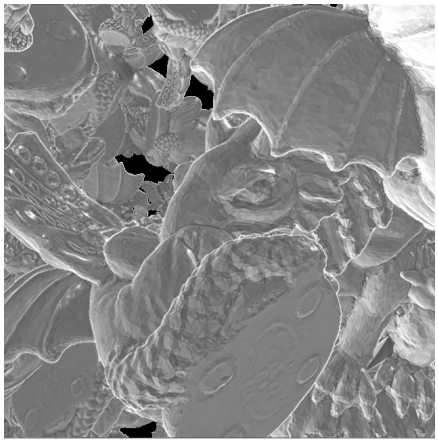

Figure 1: *short-range SSAO*

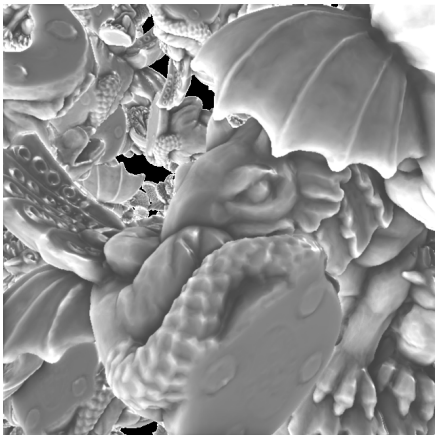

Figure 2: *medium range SSAO*

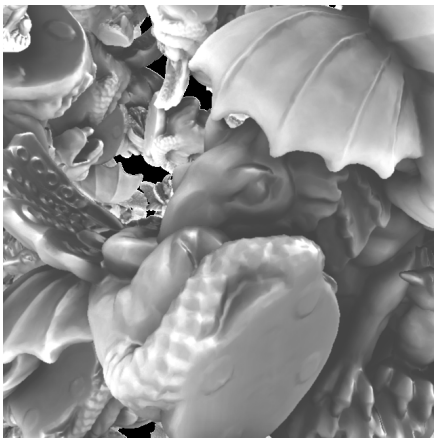

Figure 3: *long range SSAO*

1 introduction

Screen-space ambient occlusion is a method first pioneered by Cry-Tek in the game Crysis for adding approximate soft-shadows to objects in real-time using only screen-space information. The method works by stochastically shooting vectors from each fragment, if that ray hits inside a part of the geometry[1] this increase the occlusion factor of the origin fragment. Several rays are combined to form an estimate of the total occlusion. This effect is not photorealistic but adds depth perception to otherwise flat images.

Since its inception there have been many changes made to original algorithm but the key concept remains the same. However there have also been new algorithms developed which fill the same niche but work in a differen fashion such as HBAO(+). The biggest difference we see today compared to the original implementation in Crysis is the use of normal information. Many video-games use what is known as deffered rendering these days and in doing so have screen-space normals and positions readilly availible at the relevant shader stages. Historically, SSAO sampled rays in a sphere around the original fragment. This lead to flat surfaces like walls to be occluded by 50%. With the normal information readilly availible nowadays rays are often sampled in a normal-oriented hemisphere.

In both implementations, in order to get the best results one should randomly rotate the rays for neighbouring fragments to increase the entropy and quality of the sampling. A popular choice is to tile a 4×4 matrix over the framebuffer containing a random axis of rotation. This introduces new artifacts though which require a blur-pass to be added.

2 Parameters

There are many parameters one can tweak in order to achieve the desired effect using SSAO. The most major ones are the amount of rays shot and the (maximum) length of the rays shot. The sample count greatly effects both performance and visual quality. Common numbers used are 16, 32 and 64. The radius is key to achieving a realistic effect, as can be shown by the leader image.

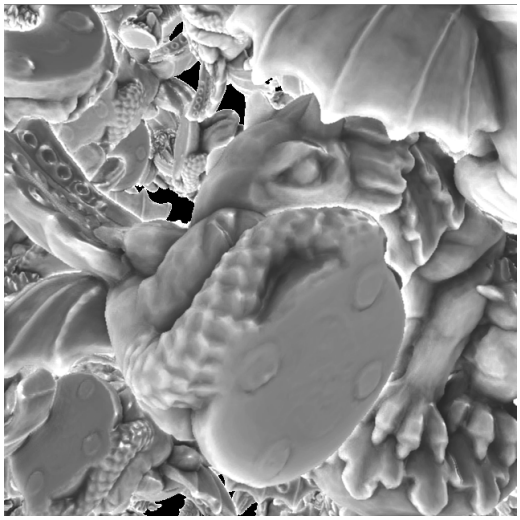

Figure 4: *Range check version 1*

Another important parameter is the choice of how to handle range filtering, that is the technique of correcting for large differences in distance/depth. We have implemented two techniques for dealing with this, and as can be seen from figure 4 and figure 5 this can have a large effect on the result.

Other tweakable parameters are bias and power, which are respectively a scalar offset and a expontial power applied to the final occlusion, this can help increase contrast and the size of shadows. This is of course a non-physicall, but popular, parameter.

*e-mail:roel@codingcat.nl

[1] to be precise: when the fragment depth at the end of the ray is overlaps the depth of the ray+depth of the origin fragment.

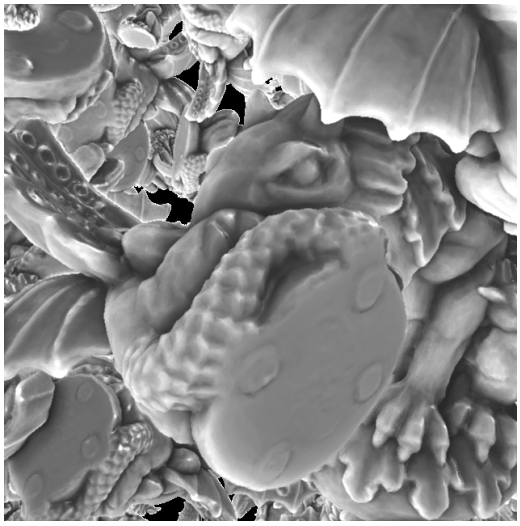

Figure 5: *Range check version 2*

3 Implementation

Our implementation is based on the hemisphere approximation described by John Chapman[Chapman]. However, we directlly sample the view-space coordinates from the G-buffer unlike his implementation which uses a depth-buffer to recompute the positions. We use a 4×4 buffer sampled using fragment coordinates for our random rotations and for our blurring we simply sample the 4 closest fragments for a given output fragment in our shading stage and average them, this allows us to blur and shade in a single step but the visual quality is not the best. Better resulst can be obtained with different blurring techiques[Games].

We have also included a deffered shading pipeline which can render up to 128 lights, as setting up a G-Buffer was a prerequisite to this implementation. An example of this lighting is shown in figure 6.

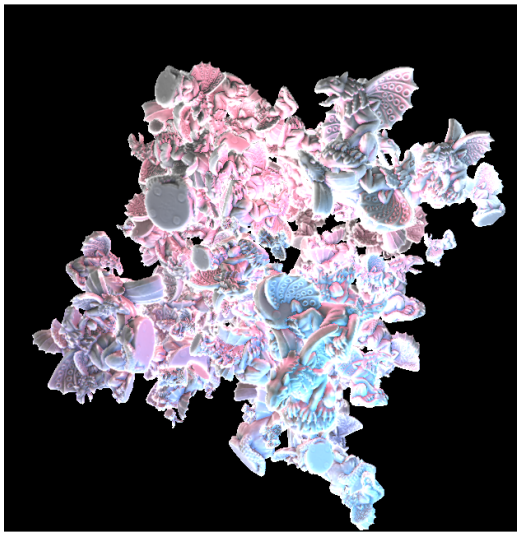

Figure 6: *Shiny!*

References

CHAPMAN, J. Ssao tutorial. http://john-chapman-graphics.blogspot.se/2013/01/ssao-tutorial.html.

GAMES, F. Tech feature: Ssao and temporal blur. http://frictionalgames.blogspot.se/2014/01/tech-feature-ssao-and-temporal-blur.html.