PROJECT INF584:
SSDO APPROXIMATION FOR GLOBAL ILLUMINATION

POUR LA PATRIE LES SCIENCES ET LA GLOIRE

24 mars 2023

Vannvatthana NORNG
X2020

l'X
ÉCOLE POLYTECHNIQUE
IP PARIS

1
INTRODUCTION

It is still an unsolved problem to achieve the global illumination in real-time calculation for large and dynamic scene, only through approximation can be achieved. AO (ambient occlusion) is an approximation method often used in films and games, which gives a really good effect but still haves the problems of decouple visibility and illumination. In this project, I would like to implement the Screen Space Directional-Occlusion (SSDO) from the paper **Approximating Dynamic Global Illumination in Image Space** by T.Ritschel, T.Grosch and HP.Seidel in 2009. This method is an extension development of the method Screen Space Ambient Occlusion that helps solve the problems that AO and SSAO have.

1.1 Screen Space Directional Occlusion technique

Since SSDO is the extension and improvement of SSAO and the classic ambient occlusion, I would like to talk about these two methods first. The method ambient occlusion (AO) is used to calculate how exposed it is to the light source for each point of the scene. The ambient occlusion of a surface point is computed by casting rays sampling the hemisphere over the sampling point (figure 1). The point is partially occluded if there are some rays intersect with the other part of geometry of the scene. While it is simple to implement and offers a better perception of the 3D shape of objects, this computation method requires and expensive computational cost. For the usage in real-time application, Crytek developed and published a technique called screen space ambient occlusion (2007). Like the name suggests, this technique approximates the ambient occlusion by taking into account of per-pixel depth of the screen space rather than using the scene geometry. Hence, SSAO is more efficient to be applied in video games and other interaction applications that have a big amount of dynamic geometry.

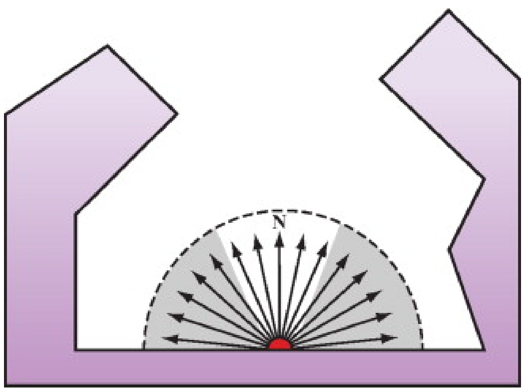

Figure 1 – Hemispherical visibility sampling of ambient occlusion (N is normal vector for the orientation of the hemisphere)

While SSAO is a much faster approximation than the classic ambient occlusion, it is hard to correctly smooth out the noise without getting the discontinuities of depth in some places, and is also local and view-dependent. Therefore, screen space directional occlusion method is introduced in 2009 to improve the quality of SSAO's illumination to be more realistic while costing only a little more computational time.

SSDO uses the information that is already computed during the process of SSAO approximation, but instead of approximate the ambient occlusion effect, it computes the directional occlusion and takes into the indirect bounce of the lighting.

So, SSDO starts by computing the geometry buffer (or G-buffer) to store the position, normal and depth values of every pixel of the screen space. Then for each pixel, a number of samples are randomly generated within the hemisphere oriented by the normal vector. Then we consider the two rendering passes : direct light and the indirect bounce. For the direct light pass, the point P of the pixel is only illuminated from the direction of sample points that are not occluded. For the indirect bounce, a small patch is placed on the surface for each occluding sample point, the direct light from the previous pass is used as the sender radiance (See figure 2 for visual explanation). Finally, each of these two rendering passes is proceeded with a bluring pass to remove the noise.

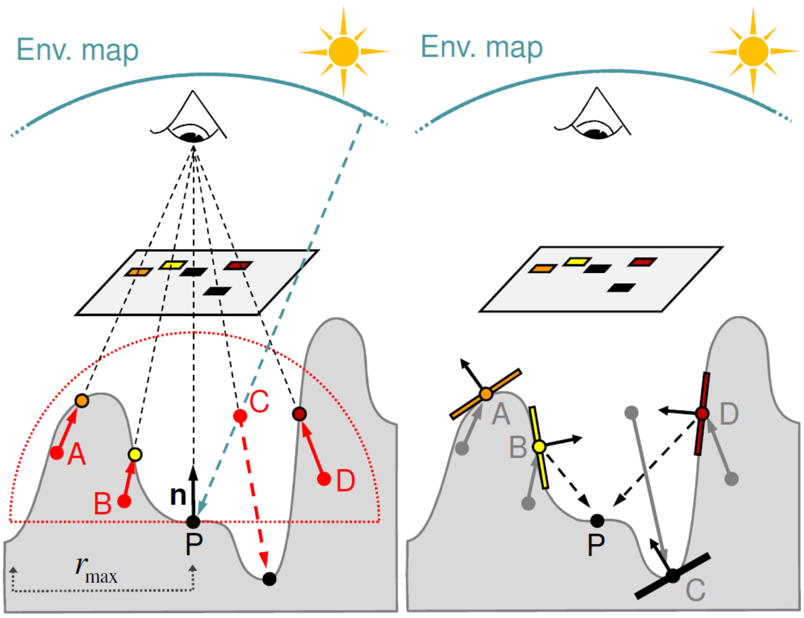

FIGURE 2 – Left : direct lighting with directional occlusion, Right : indirect bounce

Since this is a screen-space approximation, the limitation of this method is that not every blocker or indirect light source is visible. There are two approaches suggested in the article in order to overcome this problem : *Depth peeling* and *additional camera*.

2
IMPLEMENTATION

The codes used are based on the tutorial from the website LearnOpenGL. For this implementation, I have implemented the two rendering pass of the SSDO method : direct light with directional occlusion and one indirect bounce. However, the method of depth peeling and additional camera which are the improvement approaches, are not presented here.

The scene consists of an object model (for demonstration, a cartoon model is used) surrounded by a cube map, with a point light source for the illumination of the whole scene. The implementation have 6 rendering passes : geometry pass, the two passes of SSDO method mentioned earlier, the bluring passes corresponding to the SSDO rendering passes, and finally the lighting. Each pass is associated with a fragment shader file that computes and store its corresponding data values. There are also two vertex shader files, one for the geometry pass and another one is for the rest. The rendering process is in a sequence from one pass to another.

— It starts from the geometry pass where the model is loaded and the G-buffer stores position, depth, normal, and albedo values in screen-space (per-fragment).The cube box (environmental map) is also render in this stage.
— Then the SSDO direct light pass computes the directional occlusion by using the data from G-buffers from the previous pass, and the color texture of the cube box is used as the incoming radiance (L_{in}) in this computation. This pass is followed by its bluring pass in order to remove the noise. The similar process is done for the indirect bounce pass and its bluring pass for computing the indirect bounce lighting and remove the noises.
— Finally, the lighting pass computes the global illumination, and combine it with the output from direct light blur pass and the indirect bounce blur pass in order to create a SSDO effect in the scene. The specular term of the global illumination uses the Blinn-phong model which is a simple model.

3
RESULTS

A cartoon kid model is used for this demonstration. The results from figure 3 show different view angles of the cartoon model with the light source positioned on the left-hand side of the cartoon which gives a good result.

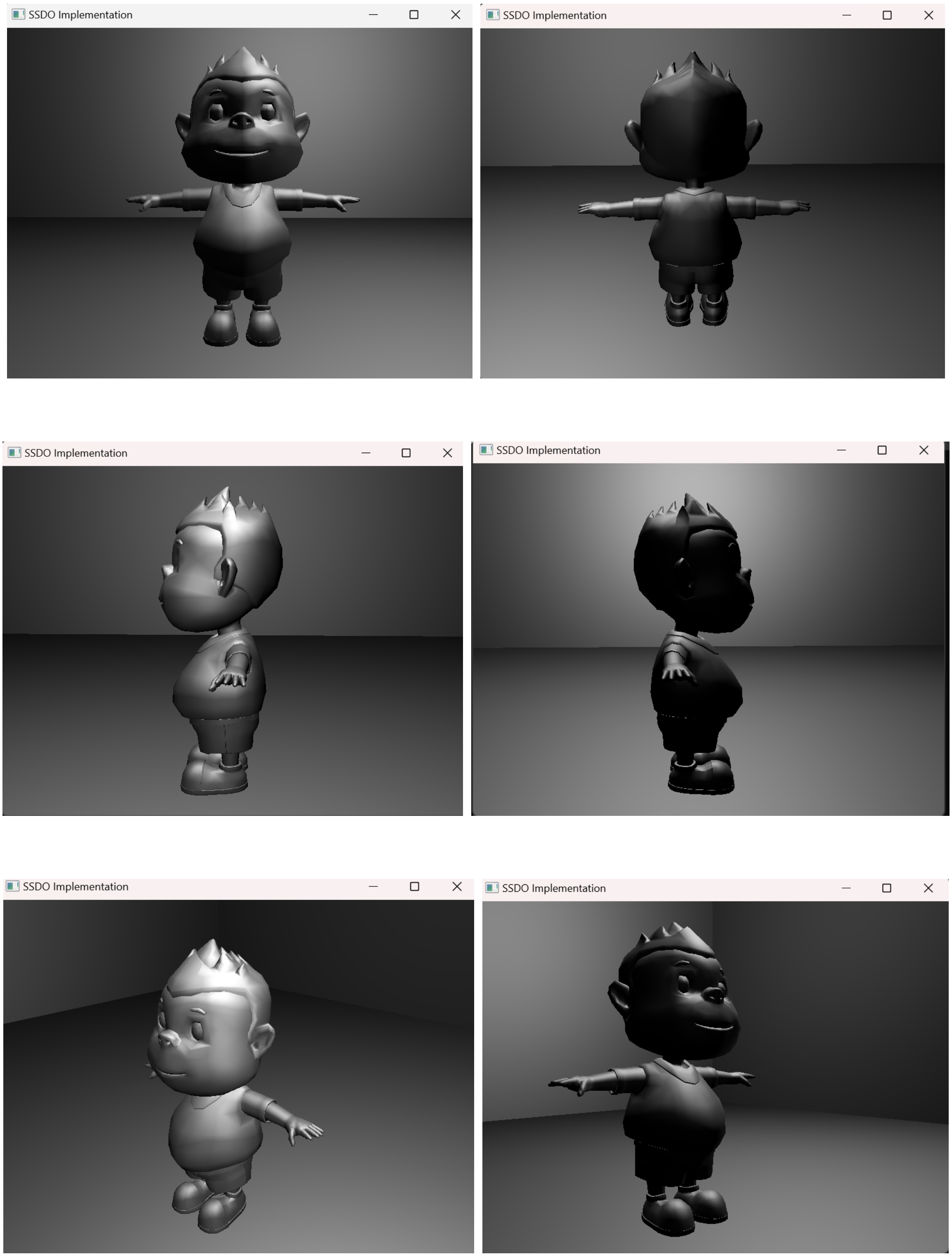

FIGURE 3 – Result of SSDO implementation from different view angles

4 CONCLUSION

In conclusion, the technique of screen space directional occlusion is successfully implemented with taking into account directional occlusion to the direct light and the indirect light bounce. Although the result shown is acceptable, it could be improved more, for example, in terms of computation so that it takes less time to compute the effects and display the results. Also, the depth peeling and the additional camera approach could be implemented to improve the overall visual of the scene.

RÉFÉRENCES

[1] Approximating Dynamic Global Illumination in Image Space, by Ritschel, Grosch, and Seidel [2009]

[2] LearnOpenGL tutorials

[3] Ambient Occlusion explained from ScienceDirect