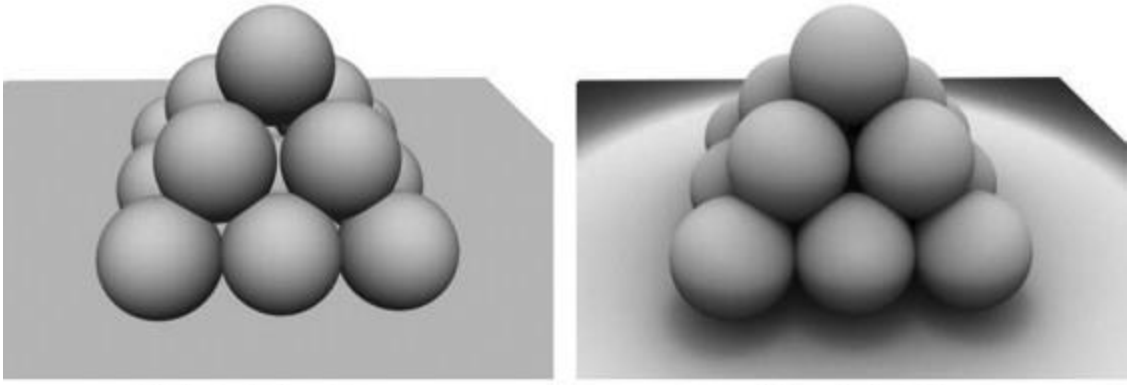
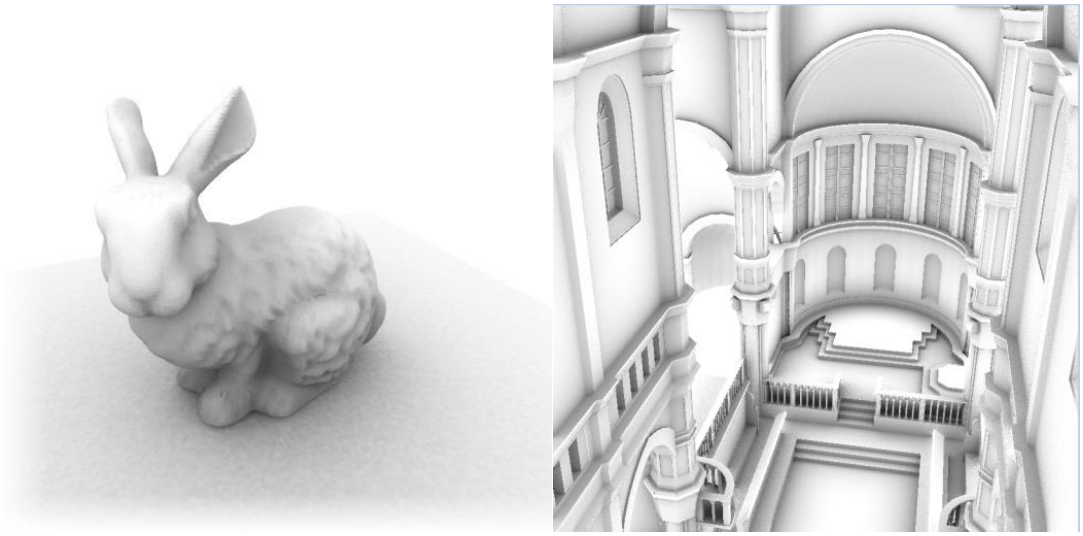
CS535: Assignment #3 – Screen Space Ambient Occlusion

Out: October 5th, 2024
Back/Due: October 25th, 2024

Objective:
This objective of this assignment is to implement a parameterized screen-space ambient occlusion (SSAO) renderer. As explained in class, SSAO yields an approximation of global illumination – it is not quantitatively correct but it is qualitatively reasonable, and once implemented can be used for just about any content.



Without ambient occlusion | With ambient occlusion



Two SSAO examples.

There are numerous SSAO algorithms and implementations. They vary in performance and quality of imagery. The basic idea is to sample a set of positions "around each fragment" and determine an occlusion factor to use as a multiplicative factor in diffuse/specular shading, for example. More details have been explained in class and OpenGL implementation details are in https://learnopengl.com/Advanced-Lighting/SSAO.

You will use OpenGL with FreeGLUT (or similar) and you can extend your base template. This program will use CPU and GPU programming.

Specifics:

(0) (40%) Occlusion Factor: around each fragment (e.g., surface point/pixel on the screen with depth), you will construct a hemisphere of positions slightly in front of the surface point and use them to compute an occlusion factor:
   a. (20%) The number of these samples N and their "distance range" from the surface point M are parameters. For example, $N = 16$ and $D =$ "a few pixels". Using N and D, you can sample and randomly compute such points. One way to do this more efficiently is to define a small random texture with noise values and use that to generate these sample points deterministically but without having to actually generate them at random for each pixel and for each frame.
   b. (20%) Given these samples, you can generate offset vectors from the surface point and then project said samples back to image plane and compute a z value. This z value can be compared to the neighboring z values to determine how many of the samples are occluded. This is the occlusion factor f (e.g., if all samples are occluded, $f = 1$).

(1) (20%) Given the occlusion factor, you can use it to modulate (i.e., multiply) the diffuse (or specular) color of the object (e.g., multiply by $(1 - f)$), giving it an ambient occlusion color feel. You do not need to explicitly worry about shadow computation in this assignment. However, SSAO will give you an illusion of computing shadows.

(2) (20%) Similar to "toon shading", define at least one additional shading profile for the occlusion factor. The profile shall be non-linear (e.g., $f' = kf^m$, where f' is clipped to [0,1]) so the effect of partial occlusion is altered (e.g., the darkness resulting from SSAO for $m = 2$ is less strong; if $m = 0.5$ then it is more strong/dark; k is a simple linear scale ($k > 0$).

(3) (20%) You should develop a simple GUI to be able to change the core rendering parameters (e.g., N, D, k, m). The GUI and program shall produce "nice" SSAO using at least the provided 3D polygonal models, or others you might choose.

(4) EXTRA CREDIT (10%)
   a. Changing the program to render several objects on a ground plane would be nice, for example.
   b. To achieve a smoother SSAO, the rendering image could be slightly "blurred". This would also enable using less sample points per pixel.

Turn-in:

To give in the assignment, please use Brightspace. Give in a zip file with your complete project (project files, source code, and precompiled executable, but no debug output). The assignment is due BEFORE class on the due date. It is your responsibility to make sure the assignment is delivered/dated before it is due.

Don't wait until the last moment to hand in the assignment! For grading, the program will be run with no command line parameters and the code will be inspected. If the program does not compile, zero points will be given.

If you have more questions, please ask!