Final Project: Shadow Mapping

Xiaoyue Wang, Yi Li

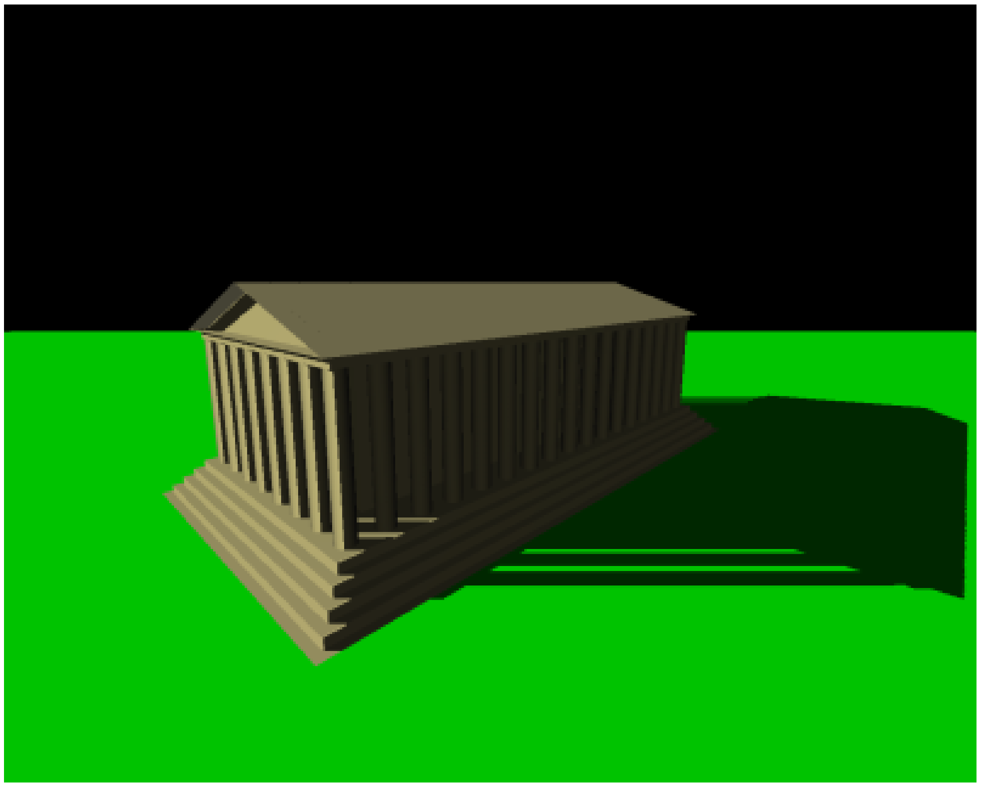
University of California, San Diego

Author Note

This report serves as a course final project for CSE 167, Fall Quarter 2022

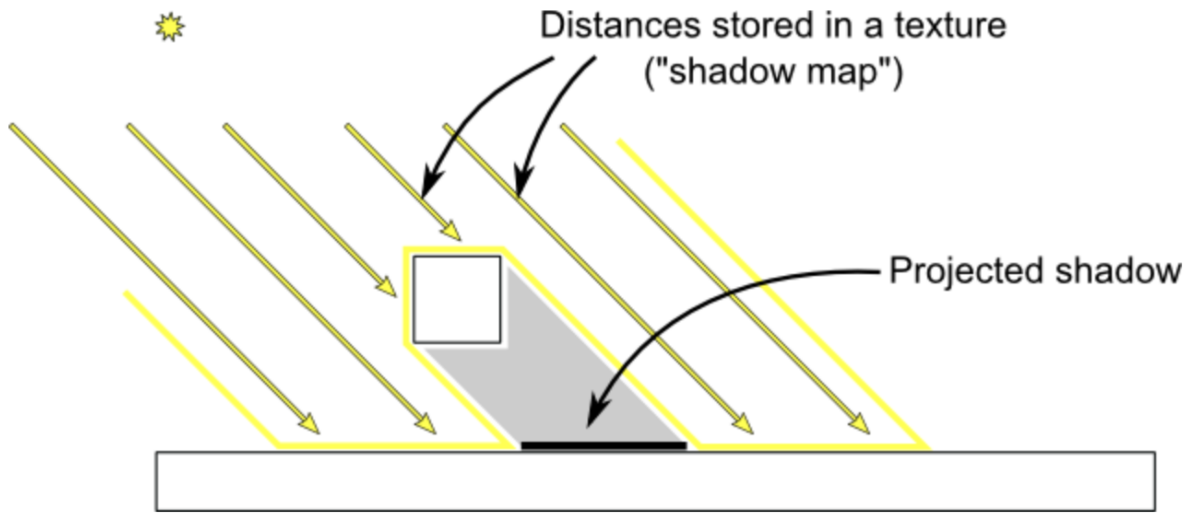
Introduction:

Throughout the development of Computer Graphic technology, we continue building powerful tools to mimic the 3D scene in our computers. OpenGL, one of the most popular libraries, contains many built-in functions which can significantly improve the efficiency of the implementation process. While rendering the 3D scene, it will be difficult to mimic the real scene without incorporating the shadows. The reason is that all scenes contain lights and the area where the object blocks lights should be considered a shadow area. To include the shadow, we apply the "Shadow Mapping" algorithm introduced by Lance Williams and use distant light as our light source.



Principle / Algorithm:

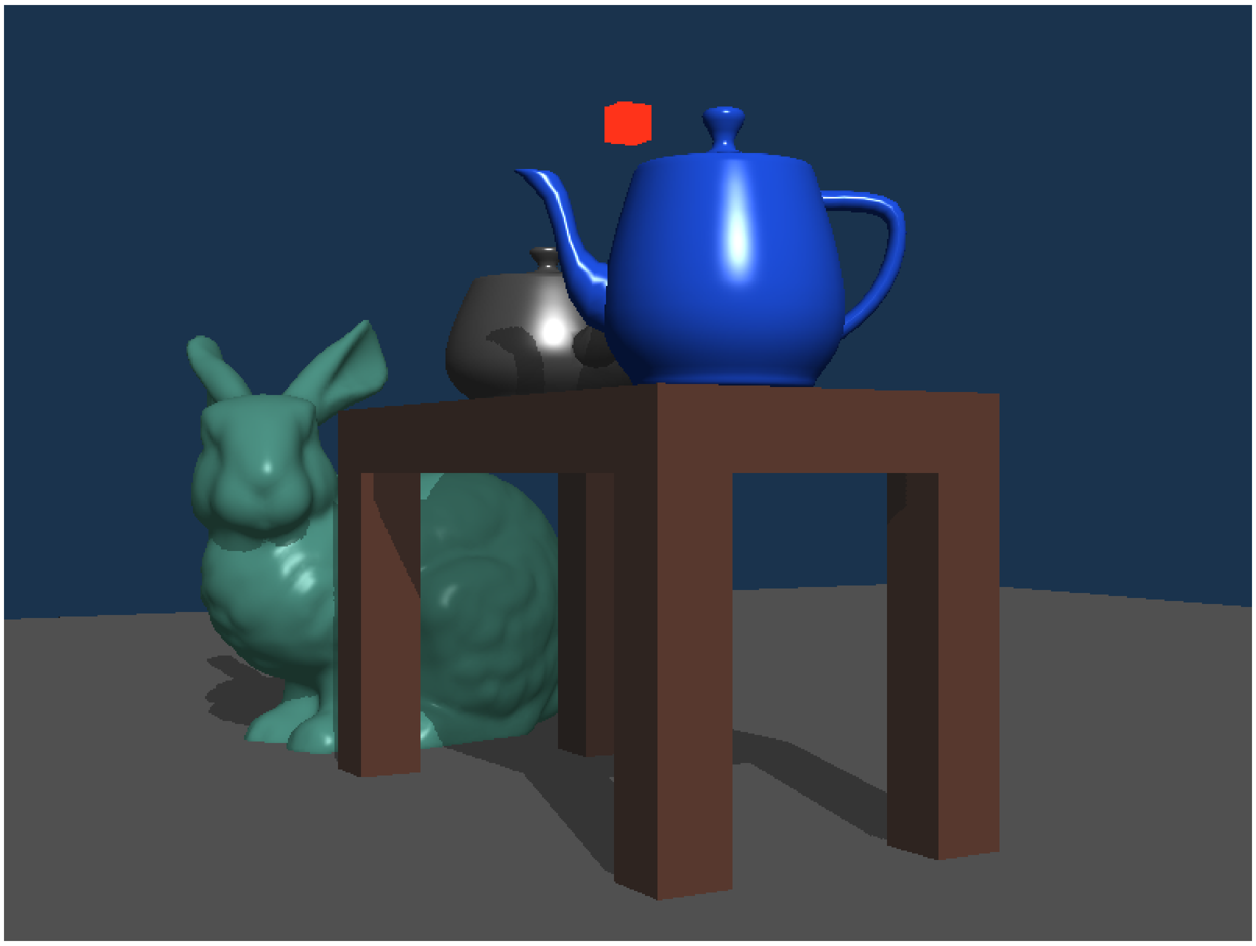
The principle behind the “Shadow Mapping” algorithm is straightforward. The algorithm contains two passes: building and using the shadow map. During the first pass, we will render the scene from the point of view of the light. Since the shadow map only requires information on the distance between the distant light and the object, we only need to compute the depth for each fragment. We will remove the object during the second pass and render the scene as usual. If the current distance is smaller than the distance stored in the shadow map at the exact location, we can conclude that the current location is in shadow.

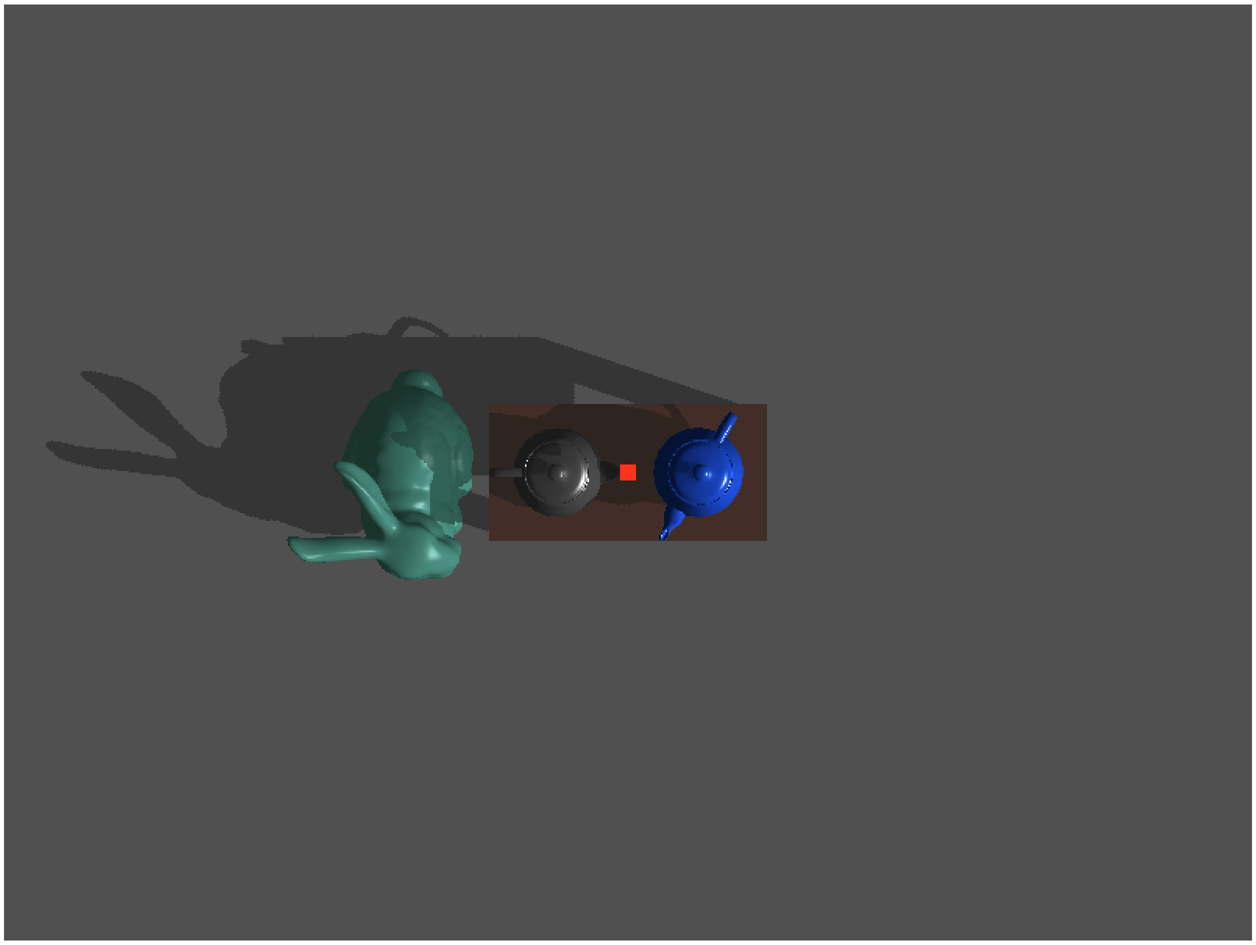


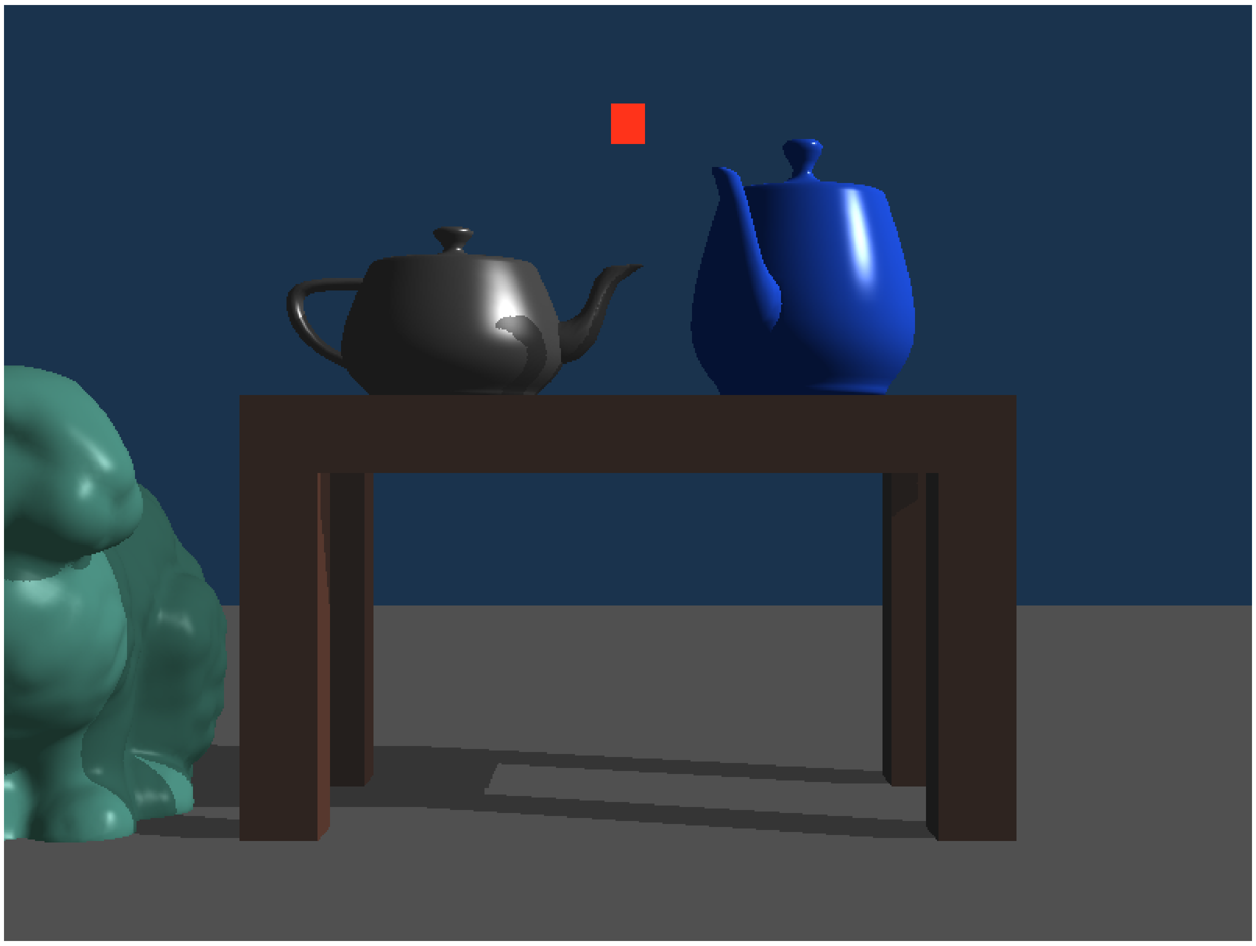
Explanation of Implementation:

In the beginning, we built a new shader type called “DepthShader” to store the information on shadow mapping. Then, we compile our shader in “Scene.inl” for later use. Next, we build separate fragment and vertex shader files for shadow mapping. Since we only need depth information, the implementation for both shader files is simple. Later, we build the depth map texture and its corresponding frame buffer in the scene.cpp file. To distinguish the two passes, we build separate draw functions for each pass and update the light space matrix accordingly. In the end, we communicate with the shader through a uniform command line before the draw to ensure that we are connected with the shader files we just built.

Result / Further Improvement:







Since we are using distant light as a light source, we turn off the red bulb since it may affect the representation of the shadow. Besides, we added the object floor to the scene because we wanted to display the shadow of the bunny and table. To verify the correctness of our shadow, we compare the direction of the distant light and the corresponding shadow area for each object. After observing the shadow from different angles of perspective, our implementation is mostly correct. However, the shadows of the table legs are a bit off. We can either try Stratified Poisson Sampling of shadow map and perspective shadow map algorithm to make our shadow more accurate if we have more time.

Reference:

1: https://en.wikipedia.org/wiki/Shadow_mapping

2: https://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/

3: https://cseweb.ucsd.edu/~alchern/teaching/cse167_fa22/project-shadowmap.pdf

4: https://cseweb.ucsd.edu/~alchern/teaching/cse167_fa22/5-2Texture.pdf