CS 6610: Interactive Computer Graphics Project

On

Ocean Simulation using Projected Grid and Gerstner waves

By Pupul Pradhan

Real-time water rendering has been a very sought-after topic since time immemorial. There are plenty of papers available today explaining the implementation of huge water surface using various techniques. One such paper is the "Real-time water rendering: Introducing the projected grid concept" which is the MS thesis of Claes Johnson. This paper specifically talks about the "projected grid" concept to render a huge water surface. A projected grid is an alternative technique to Level-of-Detail [LOD] which is used commonly when rendering huge surfaces. Gerstner Waves is another technique used most often in ocean simulation. I have attempted to render a huge water surface by combining these 2 techniques in this project.

I was able to incorporate the concept of projected grid to render the plane for the water surface. This required defining a quad is perspective space with coordinates in [-1 to +1] range. Then I used a Projector matrix which is an inverse of viewProjection matrix to get the coordinates in the world space. This matrix is multiplied with a range matrix to control the visible part of the plane. After getting the coordinates in world space, a line-plane intersection technique is used to get the projection of the vertices on a plane (y=0) in world space. Its these vertices in the world space, that I tessellated to create a grid and then applied the height field to generate the waves. For generating the height function, I implemented the concept of Gerstner waves which are basically trochoidal waves to give a realistic simulation of ocean water by replicating the turbulence in the ocean and animating them. I have used 3 Gerstner wave functions to get the turbulent effect and finally computed a height field by adding them up. I applied the height field to the undisplaced coordinates in world space and finally rendered the grid. While rendering the grid I also made use of a normal map to accentuate the wavy effect for ocean. I have also implemented local and global reflection in the form of environment reflection and sunlight reflection. I rendered a rock on the water surface and created a distorted reflection of the rock on the water surface to match the wavy effect. I have mapped an environment map to the water surface to make it look more realistic. I have also added a light source [sun] and reflected the sunlight from the water surface. The sunlight and environment reflections were also distorted according to the normals.

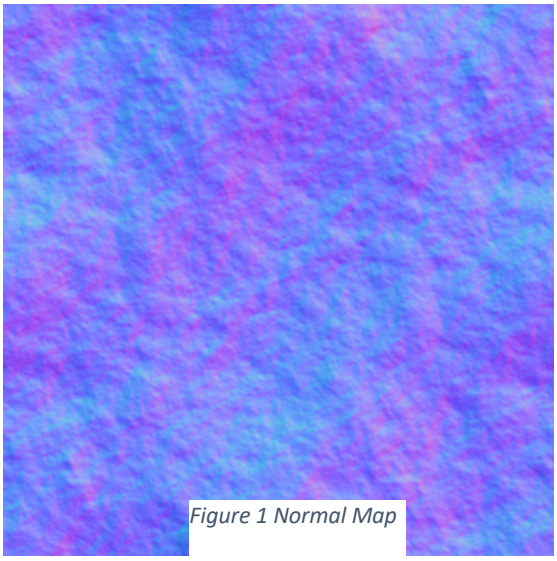


Figure 1 Normal Map

Since we render the water surface using another view matrix of a “Projector”. The projector has restricted movement. It can only have vertical movement and should not go below the camera and should always be pointing down.

This is how it looks from different positions of the projector.

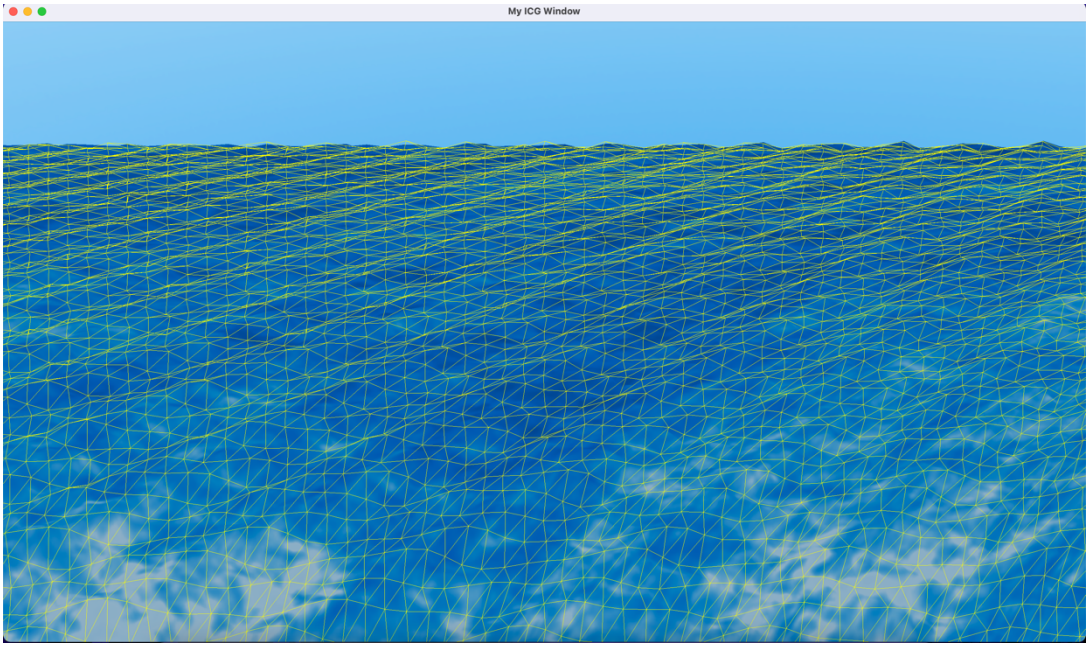


Figure 2 View from Projector height almost same as camera height

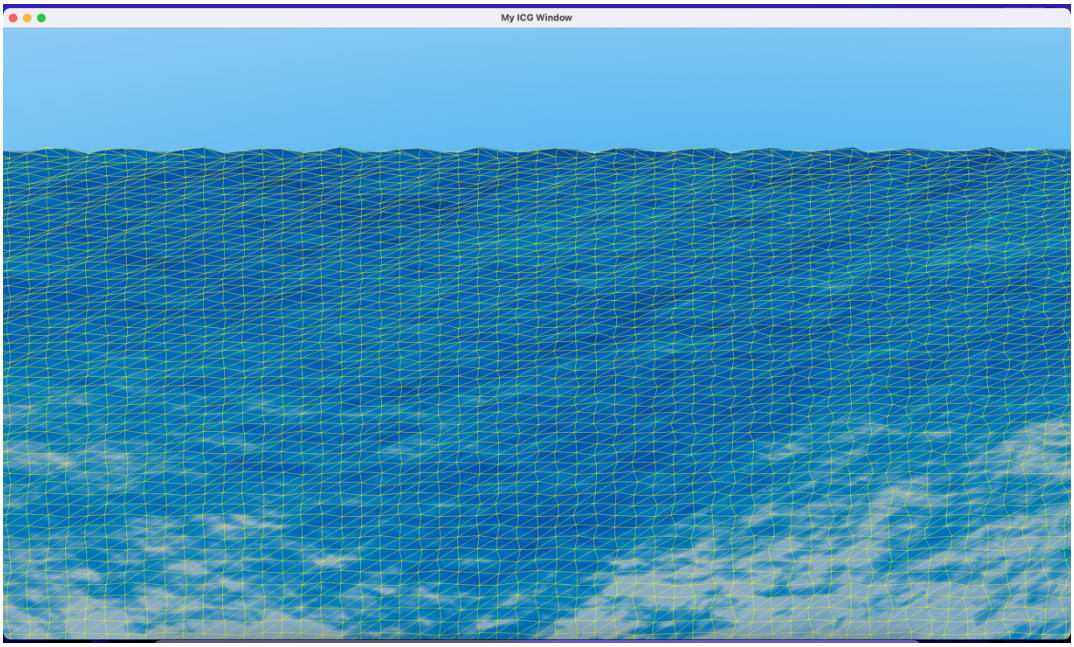
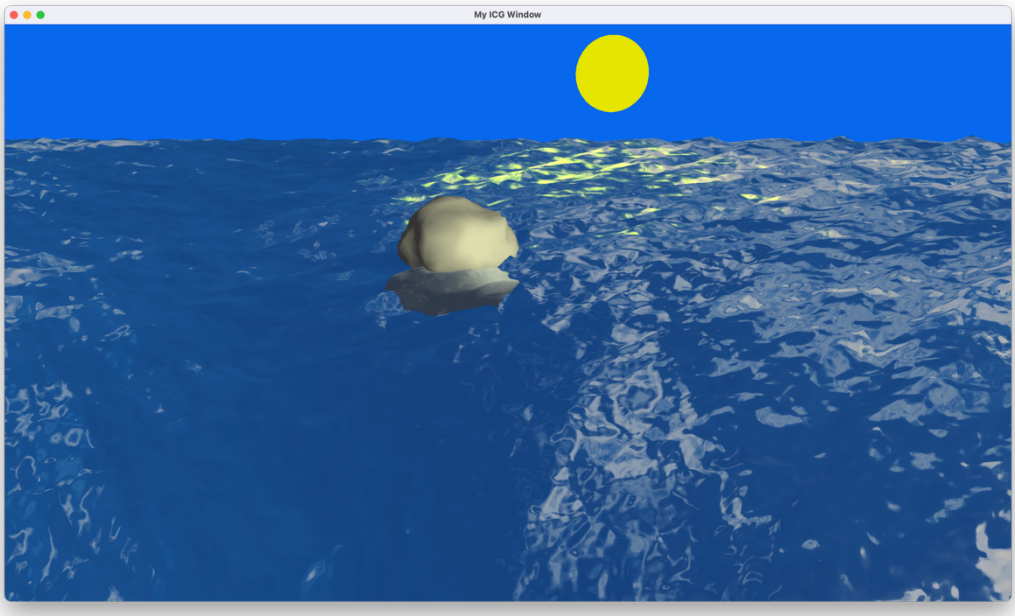
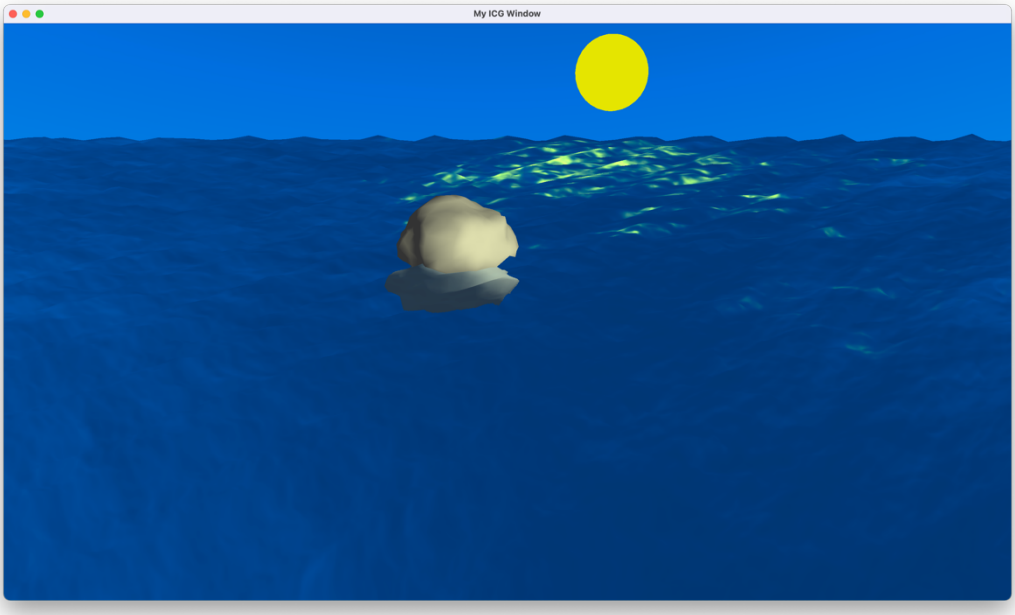
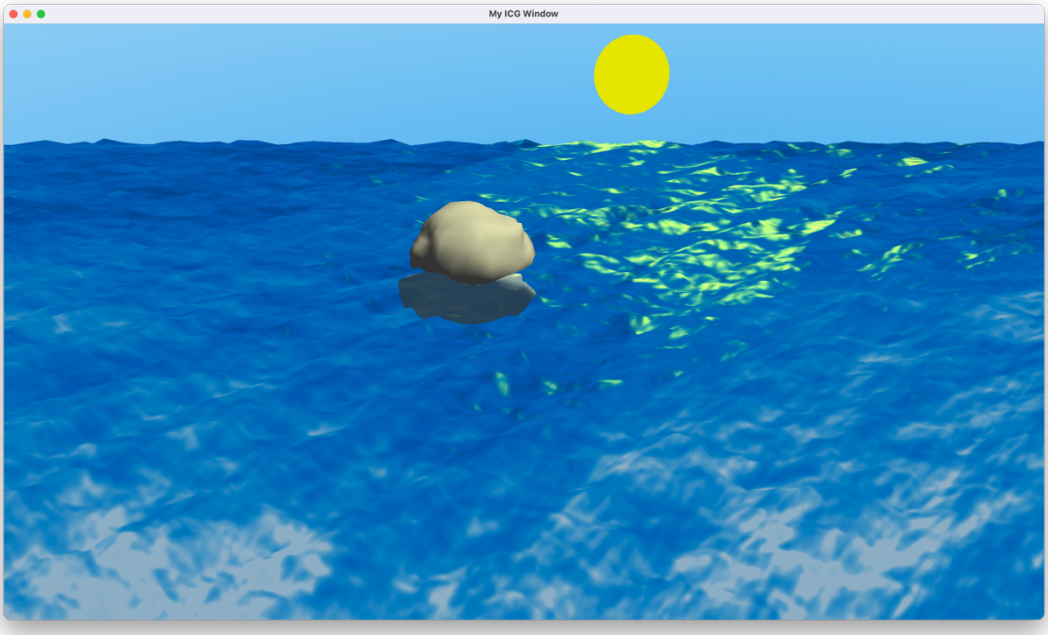


Figure 3 View from elevated Projector height

However, to render objects in the scene I had to restrict the movement of the Projector. I had to use the view matrix of the projector to render all the objects in my scene and hence had to keep the Projector fixed in one position. However, the animation of the waves will still give it a realistic feel. I also didn't have the time implement Fresnel reflectance but have used alpha blending to compensate for the effect to some extent. I did try to play around with different environment maps. This is how it looks with different maps.







The paper had proposed to use Perlin Noise for height field mapping which works great. But I found Gerstner waves to be more visually appealing and hence implemented the animation of water surface using the wave functions.

Requirements to run the project:

I have used MacOS with Clang, Xcode with GLEW and GLFW to run the program. I used OpenGL 4.1 and GLSL 4.1 No external libraries were used.

To compile and run the program, all the obj files, cube maps and the normal map will be required which are available inside the teapot folder of the zip.

References:

[1] Claes, J. (2004). Real-time water rendering-introducing the projected grid concept. Master's thesis.

[2] https://developer.nvidia.com/gpugems/gpugems/part-i-natural-effects/chapter-1-effective-water-simulation-physical-models

YouTube links to Demo:

https://youtu.be/BOaHczmKzdQ

https://youtu.be/po6lzsHGQr4