README

MacCoun – Adv Graphics Final Project

My program makes several attempts at simulating water. Clicking the change shader button will show these various attempts.

The first attempt I took was to use the Sombrero function to remap texture coordinates across the face of a quad.

The ripple shader:

1. Took in a water texture for its base color
2. Computed an environment map on top of it from the surrounding texture cube,
3. Remapped the texture coordinates to a space of [-1,1], applied a standard cosine wave equation as a function of time to remap the points distance from center, and then pulled the corresponding value for this texel from the water texture. This shader produces a nice ripple effect.

The noisy-wave shader:

1. For this shader I used a 3D simplex algorithm to generate a noise map. A 3D texture was used so that the third component could be used for time in order to animate the surface.
2. This noise shader was rendered to a FBO and then passed to a normal map shader. In the normal map shader I computed the normal by taking a difference for the x and y vals of the texel (respectively) giving a gradient and applying a constant z-vector to normalize these. This map was also rendered to an FBO, and passed on to a color mapper
3. The color mapper took one component from the normal map and treated it as a texture look up point from a standard water texture. Like the ripple shader, this distorts the texture in the flow of the given noise.
4. Note that I used the normal shader because I was originally going to compute a more specific bump map (which I do later, see below). I tried to do a more accurate normal computation, using a cross product, but the results produced were too stark in contrast to look good. Without the normal shader, the waves were not quite as pronounced, so I simply left it in.

Bump map shader:

1. Here again I used the texture generated from the simplex noise FBO which was then passed in to the same normal mapping. Because I had simple normals since I was only rendering a quad, I did all computations of the light and eyeVector in the vertex shader (as opposed to transforming into a tangent space in the frag shader).
2. Ambient, specular, and diffuse lighting is then applied.
3. Additionally, an environment map was also computed and blended in with the final color.

Given more time, I would like to fix the my environment mapping (which seems to fail only on large scales), add refraction and depth to the water, and produce a more accurate coloration of the water. At the very least, these shaders provide the stepping stone for the actual wave generation and animation of the surface, and clearly some additional parameters need to be added to create better realism.