Project 4

Robert House

Sorry for not submitting this. I will go over what is in this again for the record.

* Cook-Torrance BRDF
* Deferred Rendering
* Single Directional Light (this is intentional, indirect lighting (GI) will come after the quarter ends)
* Composite Texture Materials
  + Diffuse or Color Map (RGBA)
  + Normal Map (RGB) + Specular Map (A)
  + Roughness (R)

In the future, the depth buffer will be stored in the third render target’s B and G channels with the stencil map stored in the A channel. This “byte packing” allows for the use of less render targets and thus decreases the already steep bandwidth requirements of the Deferred Render.

The BRDF is an exact implementation of the Cook-Torrance model that I programmed using the equation found on Wikipedia. Getting better looking material properties would involve experimenting with the roughness map and using a better normal map. Having a physically based lighting model is really nice, as I can make materials that look accurate in any lighting environment.

There are also numerous backend changes that I could not help but implement. This may have been foolhardy, but I really wanted to try to build a decent renderer. I hate hardcoding things, so the back end systems were a requirement for me to accomplish my objectives.

Again, the remaining features are dependent on getting Depth-Stencil working, and I am trying to debug the best I can. If I can’t get it to work, I could emulate this feature in software by outputting the depth values to a render target. This is not ideal as the whole point of Hardware Depth-Stencil is to have hardware support for early-z rejection.

This feels like an essay now, so I will stop. I will try to comment better as the features come online. Most of the code is self-commenting, but descriptors in the interface files would be helpful.

How to Run:

Browse to: **/Project4/Release/Renderer.exe** and run the executable. Visual C++ 2013 Runtime Redistributable is required to run.

