Untitled Composition

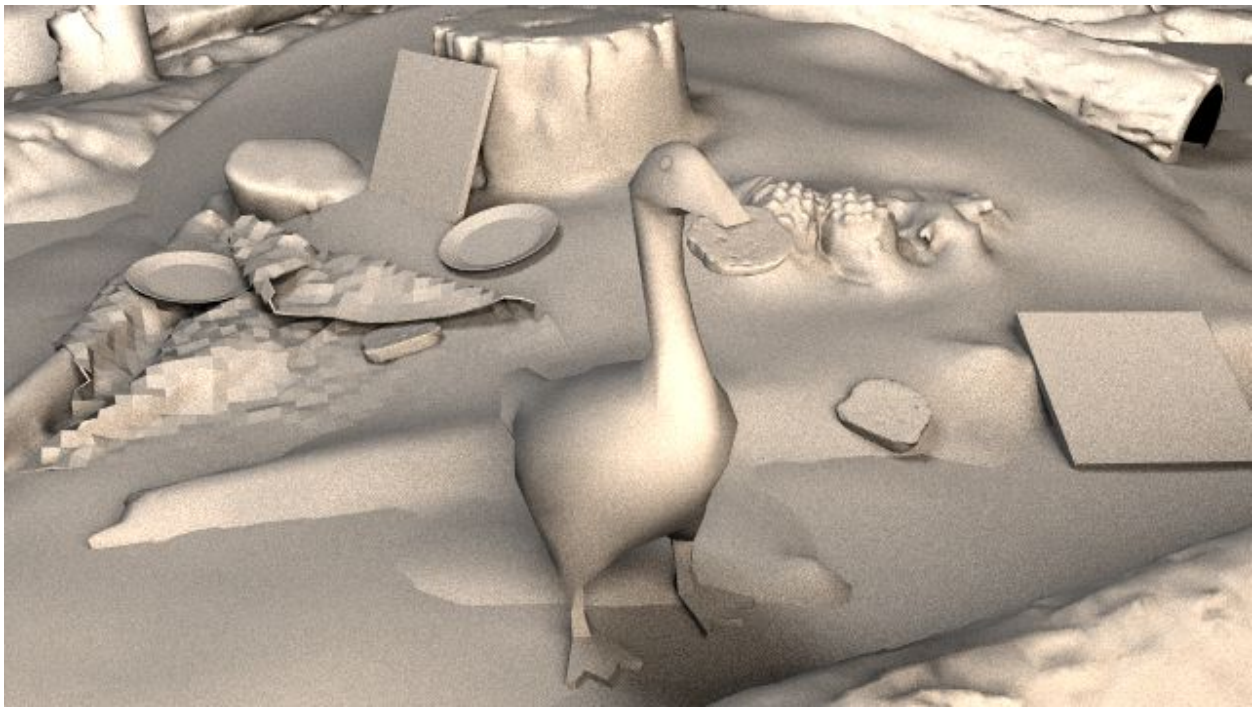
Name: Kevin Chen
SUNet ID: kvchen

Scene

This scene is inspired by the fantastic Untitled Goose Game by House House. In this game you play as a horrible goose terrorizing the inhabitants of a local village. As a fan of the game, I attempted to capture the spirit of this game in a realistic, path-traced way.

I wanted to retain the low-poly but recognizable goose while using other high-poly and detailed objects in the scene. Mixing the low and high polygon objects was challenging but ultimately resolved through two things: depth-of-field blur and soft lighting. The depth-of-field cuts through the sharp textures and focuses attention on the goose, while the soft lighting helps to ground the goose model in the rest of the scene.

Variations

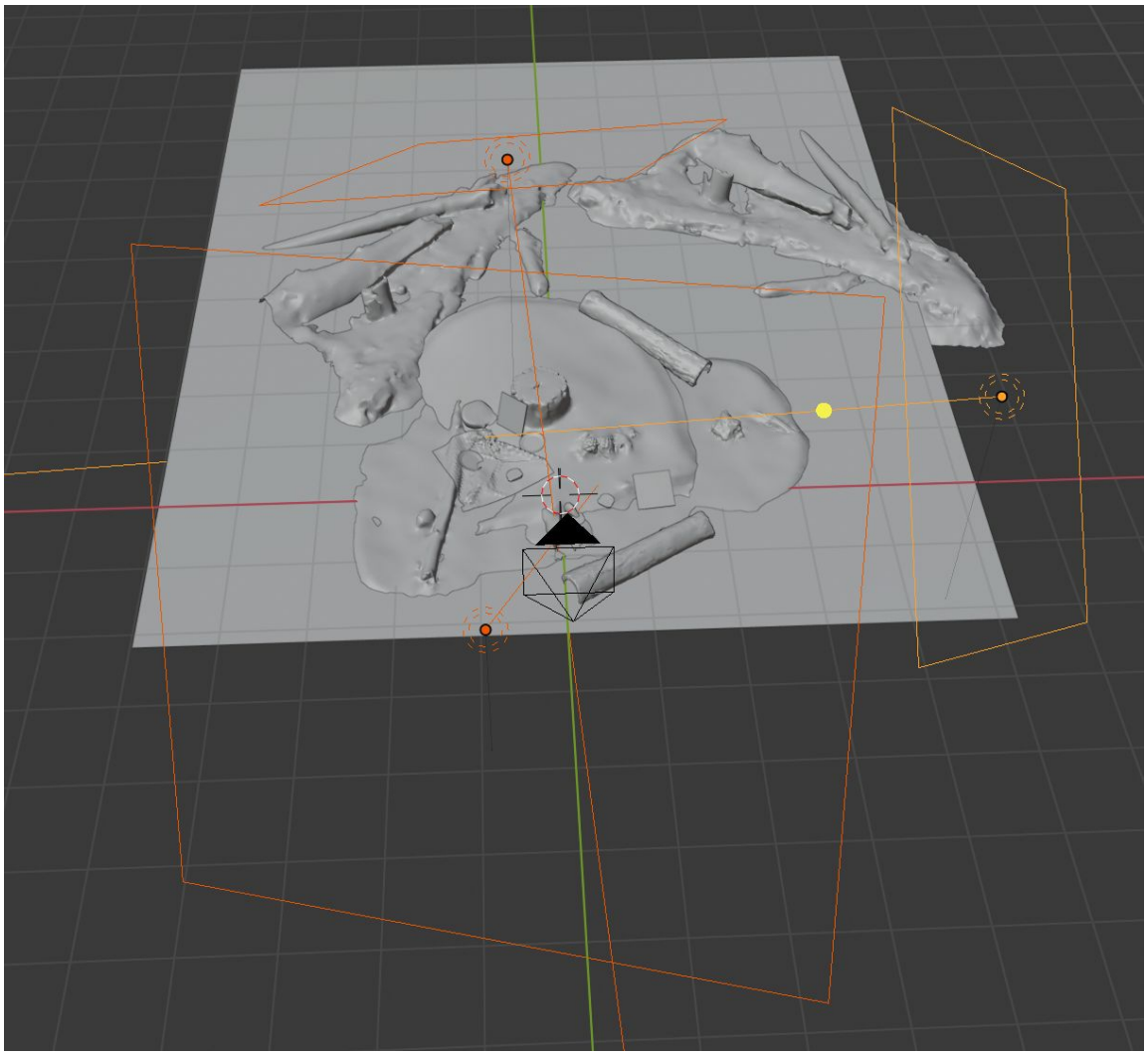


Models, Materials, and Textures

- **Picnic blanket, dishes, signs:** Created and textured myself.
- **Goose:** Downloaded from Thingiverse, textured myself.
- **Foliage, tree stumps, logs, moss, bread:** Models and textures downloaded from Quixel.

Lighting

The scene uses three lights. The light on the right provides a sharp angle that brings out details from all the objects with normal maps. The other two lights serve to illuminate the scene with sunlight, providing soft shadows and lighting.



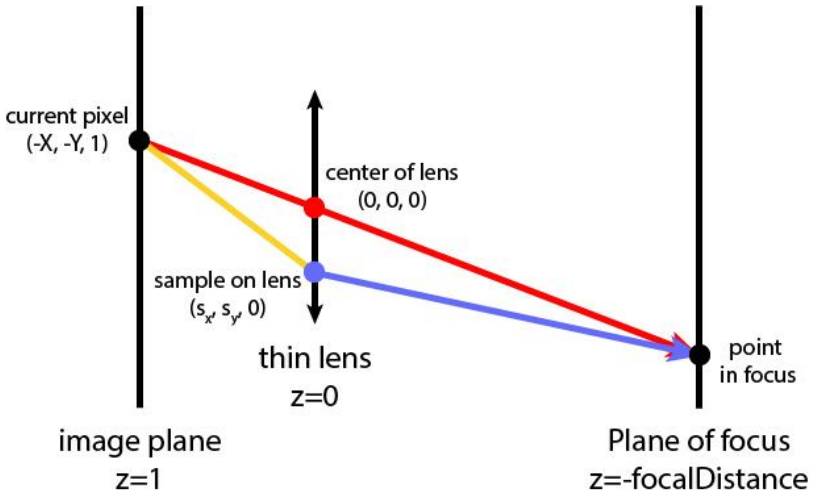
Technical Contributions

Depth of Field

I originally intended to create a tilt-shift scene with a very narrow depth-of-field. Even though I decided to not use tilt-shift for the final scene, it still uses the depth-of-field effect I implemented.

By default, the perspective camera is a pinhole camera with an infinitely small aperture. To add depth-of-field, each outgoing ray has an origin sampled uniformly from a circle around the center of the aperture. The target position is modified to take into account the focal distance, and the ray is traced from the sampled origin to the modified target position.

The following diagram from Berkeley's CS184 class illustrates this modification:



Normal Maps

To improve the quality of the materials, I added support for normal maps to the material framework. To compute the normal, we simply import the texture with **assimp** and read from the texture with a pair of UV coordinates (same as reading from the diffuse texture). We then convert the texture vector into a world-space vector. In **Material::ComputeBRDF**, we use the texture normal instead of the base intersection normal.

The normal mapped lighting is particularly noticeable in the pine needles on the floor and the log in the foreground.

Multithreading

This one was simple: in RayTracer, simply add an OpenMP pragma to parallelize the nested for loops. This yields a sublinear speedup on the 4-core machine I used for rendering.

Photon Mapping

I also implemented the gathering phase for photon mapping following Jensen's A Practical Guide to Global Illumination using Photon Maps. This entailed modifying **PhotonMappingRenderer** to gather the photons surrounding an intersection point and use them to compute a material's outgoing radiance.

I also modified **AreaLight** to emit photon rays. This consisted of selecting a random point on the area light surface as the ray origin, then using cosine-weighted sampling to pick a ray direction.

I ultimately decided not to use photon mapping for the final set of renders due to greatly increased render times. The indirect lighting was also subtle and ultimately didn't contribute much to the scene.

Modified Files

- **PerspectiveCamera.h/cpp:** Added depth-of-field.
- **BlinnPhongMaterial.h/cpp, Material.cpp:** Added support for importing and using normal maps.
- **RayTracer.cpp:** Added OpenMP pragma to multithread main pixel loop.
- **SceneObject.h/cpp:** Added convenience methods for setting Blender coordinates and rotations.
- **PhotonMappingRenderer.h/cpp:** Implemented indirect lighting using photon mapping.
- **AreaLight.h/cpp:** Added code for generating photon rays.

Revisions

The scene went through a number of iterations before I settled on the forest closeup. Some of the assets carried over into the final scene,

and even the final scene itself went through some major revisions. Here were some earlier versions: