Manual

Running

The program takes one argument, a lua script to display. The program assumes that the lua script is present and correct, and that all objects it points to are also within spec.

Lua

The lua scripts can contain these functions:

- $gr.make_list()$ - Create a list object. The script must return a list object at the end.
- $gr.make_object(obj, density)$ - Create a mesh object. Object is a .obj file inside the Assets directory. Density is the relative density of the object in the scene. Name has been removed because there isn't really a use built into the program for using it.
- $list : add_object(object)$ - Add an object to a list
- $object : translate(x, y, z)$ - Translates the model transform
- $object : rotate(axis, amount)$ - Rotates the model transform
- $object : scale(x, y, z)$ - Scales the model transform
- $object : set_scale_variance(x, y, z)$ - Set the model scale variance
- $object : set_rot_variance(x1, y1, z1, x2, y2, z2)$ - Set the model rotation variance

Provided Scripts

geometryForest.lua

This is the first test script I created. It just uses geometric models from previous projects. I think while the objects aren't trees, it does hit the feel I was going for with the generated objects.

firstForest.lua

This is a forest created using the $Pine_4m$ asset. It captures the feel of a forest well, but the object came with weird material formats that I didn't have time to figure out, so it's nothing special.

VariedForest.lua

This script uses the DownyOak objects. It showcases the main drawback to my program: performance. These are very high poly models, so performance in this script is terrible. Because of the massive performance issues I experienced using these objects I didn't finish integrating them, so they are untextured. If you run the script you can see that the leaves would have also required alpha channel mapping, which is more work for a script that already wasn't turning out well. I instead stopped development and acquired some low poly models.

LowPoly.lua

I think this is one of the better scripts. It's objects don't have textures, but they are low poly because of their stylized design and I think definitely hit the feel I was going for with this project. If I was to be judged solely on one or two scripts, I think I would choose this one and the next one.

I also like how the vertex normals in some of the trees has been altered to give the faces a sort of curved look. Specifically the pine trees seem to curve upwards, and one of the simpler trees just looks like all the leaf faces are curved.

LowPolyTexture.lua and LowPolyComplex.lua

Kind of the same as LowPoly.lua. These are a different set of low poly models, but these have a texture bound to them. The texture in LowPolyTexture.lua is the one that came with the models. It's flat and I think it looks nice. I added a lot of noise to the texture for LowPolyComplex.lua to make it more obvious that the texture is actually being mapped.

Movement

The camera can be controlled with the mouse and keyboard. Holding shift moves twice as fast but doesn't change rotation speed.

- Holding the right mouse button or the C key and moving the mouse rotates the camera.
- Holding the middle mouse button, the X key, or mouse button 4 and moving the mouse will drag the camera. This changes the position along the up/down and left/right vectors.
- Holding any of the W/A/S/D keys moves the camera along the forward/backward and left/right vectors.
- Scrolling the mouse wheel moves the camera forward and backward.

Extra keys

Many extra keys can perform extra functionality like toggling features. Most of the implemented features can be toggled on and off. Here is an explanation of what the extra keys do.

- B - Toggle skybox
- F - Toggle FXAA
- M - Toggle shadows
- N - Show the shadow texture. This draws the shadow depth pass to the screen
- Q - Close the program
- O - Toggle SSAO
- R - Reset the camera position
- +/- - Adjust the FXAA render mode. An explanation of modes is below
- [/] - Adjust the SSAO render mode. An explanation of modes is below and in Implementation.pdf.

FXAA Render Modes

Modes range from 0 to 4.

Render Mode 0 is like toggling FXAA off.

Mode 1 shows the edge detection step. Edges that will be blurred are drawn red.

Mode 2 shows the edge orientation step. Edges are coloured yellow or purple depending on their detected orientation of horizontal or vertical.

Mode 3 shows one step of the searching and blurring algorithm. I find this looks best so it is the default.

Mode 4 lets the algorithm continue to search until it is satisfied. I find this decreases the sharpness of the image a lot.

SSAO Render Modes

Modes range from 0 to 4.

Mode 0 is off.

Mode 1 is draw the depth buffer.

Mode 2 show the noise texture repeating across the screen.

Mode 3 shows the occlusion portion of the image.

Mode 4 is the full calculation.