Assignment 2 - Stuart Douglas

Scene

The scene for my assignment is a small city which is created through loading in multiple complex models from http://www.reinerstilesets.de/. The snow and dragon breath affects are two separate particle effect classes.

The terrain is a slightly modified version of the terrain from the lab which uses the perlin noise, and assigning colours based on the height, colour is calculated using the height to give a gradient effect.

The object loader uses a library called tinyobjloader https://github.com/syoyo/tinyobjloader I am using this library to load in properties from the obj files and I am then parsing through the outputs of the library and build up the vbo to then draw them.

> *I have used a commonly used gingerbread house as the bump mapping was more clear.*

Targets

Things I have covered:

- **Complex object loading**
  - Get normals from obj or image
  - Get texture position from obj
  - Map texture images to complex objects
- **Bump mapping**
  - Use images to calculate normals to create the bump mapping
- **Normal mapping:** not complete, but left in tangent calculating
- **Terrain**: However builds on example from lab
- **Particle effects**: modified from lab examples, fixed issues of affecting other shaders and objects.

Classes

- **DragonBreath:** This is a modified particle effects class which adds additional params at the draw stage to position the starting and rotation of the particle effect.
  - Uses particle shaders
- **Particle_effect**: This is a modified particle effect class which randomly places the start of each particle in the X and Z axis
  - Uses snow shaders
- **obj_loader**: A class which follows the concept of the example but makes use of a library to parse obj files converts.

- **fontObj**: An external class from http://www.opengl-tutorial.org with modification to use SOIL for loading in the font image. This is used to display a message on the scene
- **Terrain:** Worked from the examples to allow for height based colours with a gradient effect

> *I have included notes in the classes which take note of some of the changes I have made to existing classes*

Issues

- **Long loading time:** This is a mixture from the object loader and terrain, the issue with the object loader is that I'm loading in the object and the library gives me all the properties, I am then parsing that to put it into another format. If I had time I would have rewritten this class to make use of the library properties. However this would require changing alot of how I am doing things in other classes aswell.
- **Trying other complex techniques:** This is a small issue where if I wanted to attempt other complex techniques like shadow mapping it would require massively altering how everything is being done.

Controls

I have included a **controls.txt** file which displays all the controls that can be done, it is also printed to the console.

How to interact

Even through the scene has been setup as a static scene, you can move the camera around and change the light position to see the effect of the bump mapping.

> ***Note:***
> *The light source is represented by a desk light model*

> ***Important Note:***
> *Due to the object loader and terrain, it has resulted in a long start up time, please leave up to 1 minute to load (this will depend on your system)*
> *If it has taken longer please restart the app or press enter in the debug window since selecting the debug window can pause the application.*

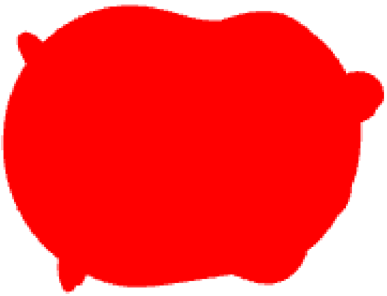
Future improvements

- Move to using PNG images (issue with SOIL)
- Transformations to dragon, the idea would be to have the dragon flying around and randomly blowing out fire.
- Improving the object loader to work from the library (library has little to no documentation or

api docs)
- Day/Night cycle: This has been partially implemented but due to focusing on other aspects I disabled but have left code in.

Overview

Overall I found this assignment alot more fun than the first assignment, I looked into webgl which I found to be really fun and almost identical to c++.



> *One issue I came across with webgl was getting shaders working which as shown in the picture above we can only see the colour.*

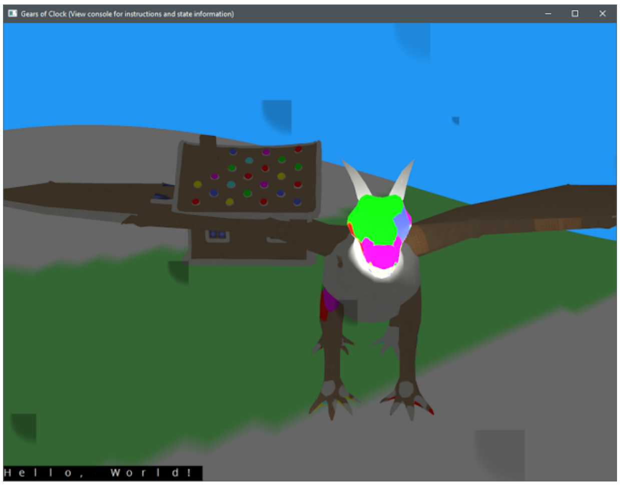
Moving back to c++ I starting working on the object loader which originally started with the example class, I converted to use a library instead since I have having issues with different obj files and the library took the work from me and all I needed to do was convert that data into the format I needed and then store it into the buffers. This allowed me to try different things like getting the normals from the obj instead of generating a default set, and getting the texture coords. This came to another issue which allowed me to learn more about shaders, for example when it came to texture images I was having an issue where the shader was not getting the bump texture, I then discovered that because i was binding them to glTexture0 I need to define this in the shader.

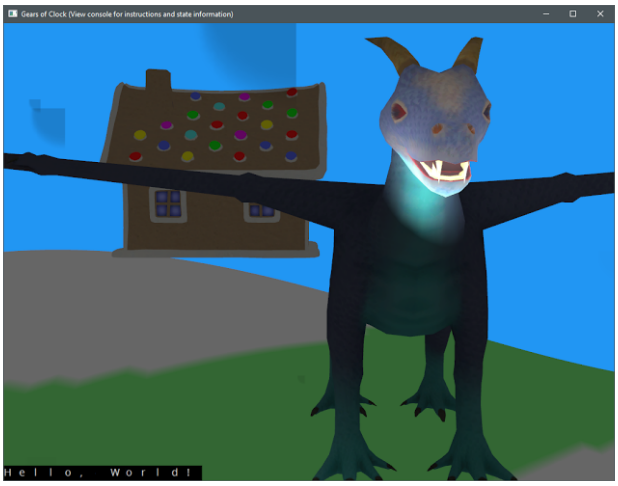
History

Below are images of progress made during the assignment

Texture coords not correct

Textures shared [issue: now fixed]


Gears of Clock (View console for instructions and state information)
Hello, World!

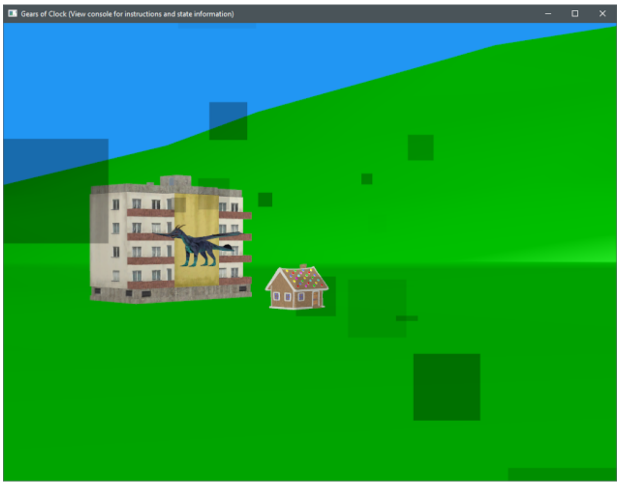

Gears of Clock (View console for instructions and state information)
Hello, World!

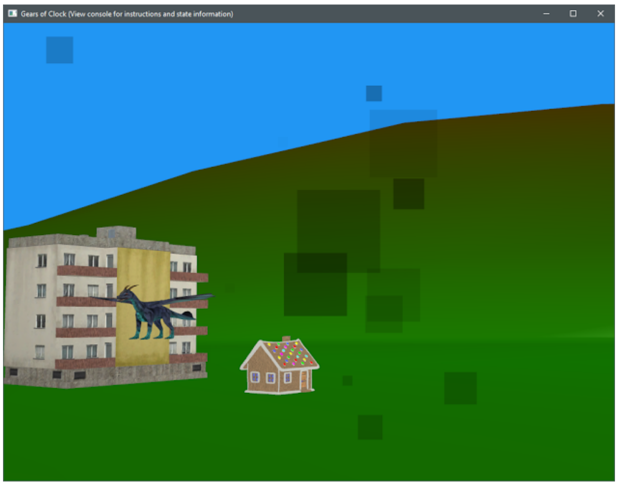
Working on bump mapping

> *example is made with a fixed light*

> *Bump mapping with a movable light*

Adding terrain and height based colours


Gears of Clock (View console for instructions and state information)


Gears of Clock (View console for instructions and state information)