Devin White

CS 5610

Interactive Computer Graphics

Project 2-Transformations

- What was implemented:
  - I implemented all of the required functionality for the project. Right clicking and dragging zooms in, left clicking and dragging rotates the object. In addition, orthogonal transformation can be toggled by pressing P. I also think I got the shaders to recompile by pressing F6, but I'm not entirely sure since it's extremely hard to tell it it's doing anything.
  - Second Submission Update:
    - Fixed pot disappearing if too far(increased range)
    - Refactored code to make it cleaner
- What you could not implement.
  - To my knowledge I implemented everything required~~. Though if zooming out too far, the object will vanish "behind" the screen. I'm not entirely sure if this is a bug, or expected, as it does not do it when orthogonal transformation is toggled. In any cause, I couldn't find a fix for it despite trying for a decent amount of time.~~ This was fixed in the second submission by increasing the range on the perspective matrix.
- Additional functionalities beyond project requirements.
  - I implemented the orthogonal transformation and refreshing shaders required for 6610 students. Aside from that, no additional functionalities. Except for the fact that it's green, because green is the best color!
- How to use your implementation.
  - Right click and scroll to zoom in and out
  - Left click and scroll to rotate. Left and right rotates left and right, up and down rotates up and down
  - P switches between orthogonal and normal transforms
  - F6 recompiles the shaders….I think(hard to test)
  - ESC exits the window
- What operating system and compiler you used
  - Operating system: Windows 11
  - Programmed in Visual studio: used VS internal compiler(gl????)
- External libraries and additional requirements to compile your project.
  - Required libraries:
    - FreeGlut
    - Glew
    - CyCodeBase

In addition, FreeGlut.dll and Glew.dll were required to be put in System 32. Libraries were put in folders next to the project along with the associated headers.
In addition, I used:

```
#include <string.h>
#define _USE_MATH_DEFINES
#include <math.h>
```

I included math in order to use M_PI and make a DEG2RAD function that I probably didn't really need to, math defines allows using M_PI which was the main thing.

In addition, I had to input the following line to make it compile correctly, which I have no idea the reason for since the glew libraries and associated files were all in the correct paths that were set up for the project:

```
//I had to put this to make it work...for some reason
#pragma comment(lib, "glew32.lib")
```

Lastly, here are some screenshots of the implementation

Figure 1: A shots of the normal display.

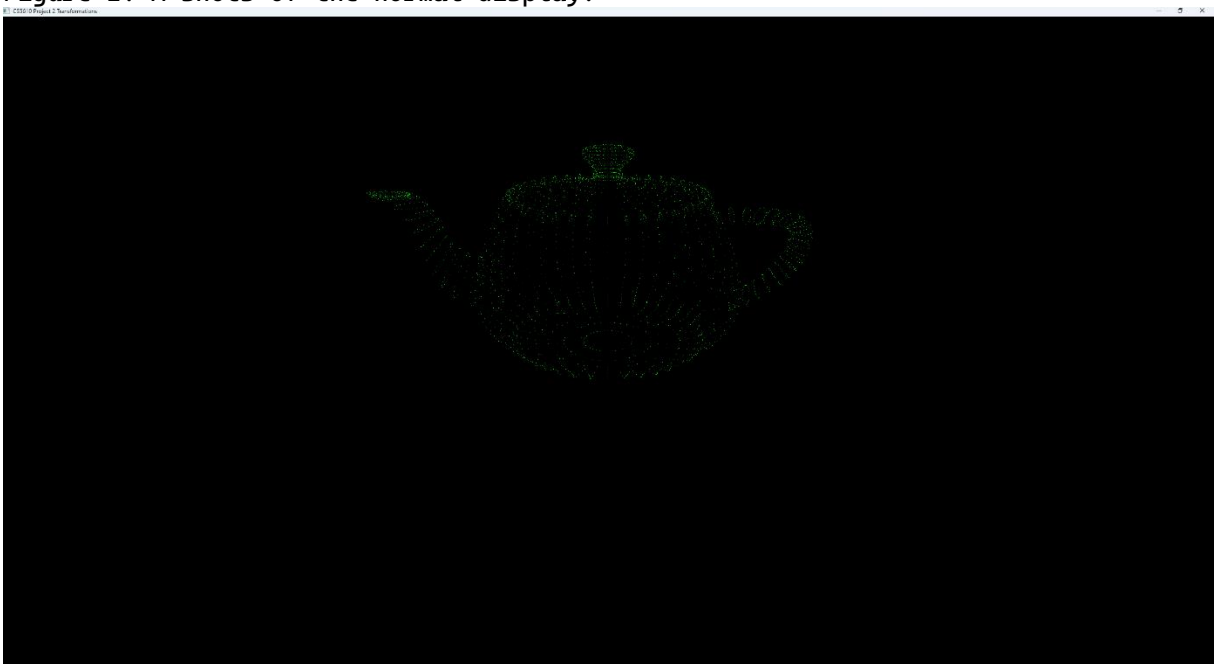


Figure 2: Normal view but zoomed out slightly with right click

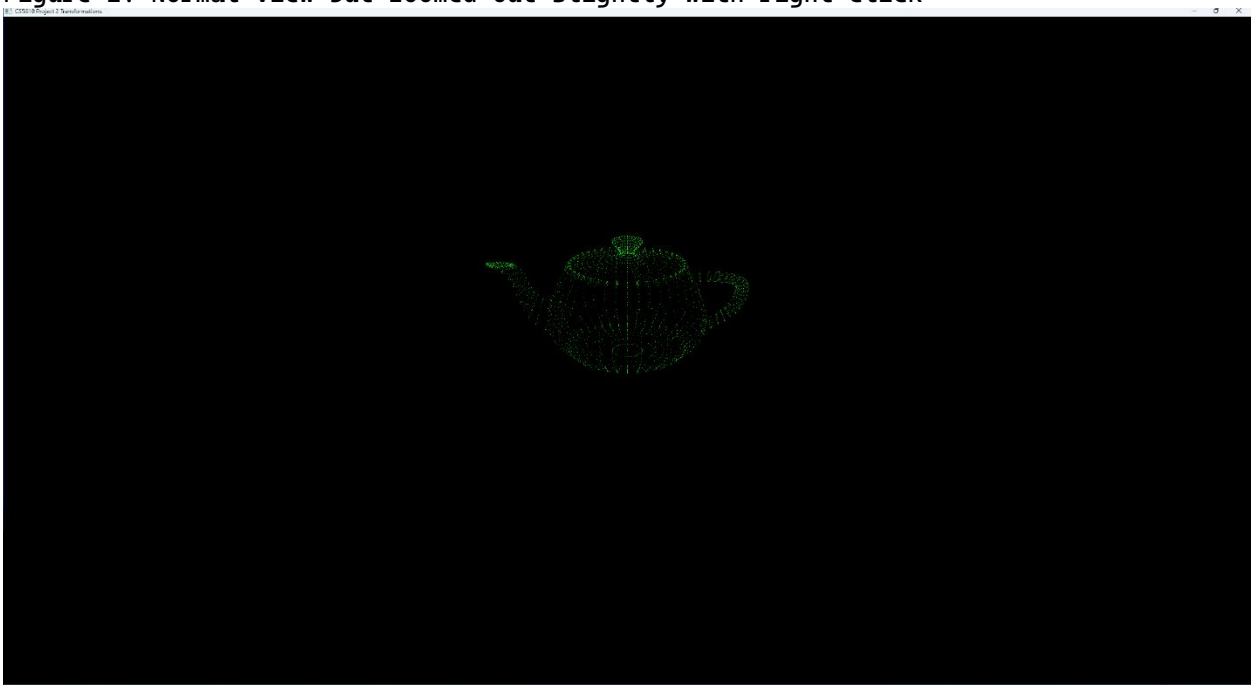


Figure 3: rotating the teapot with the left click and drag

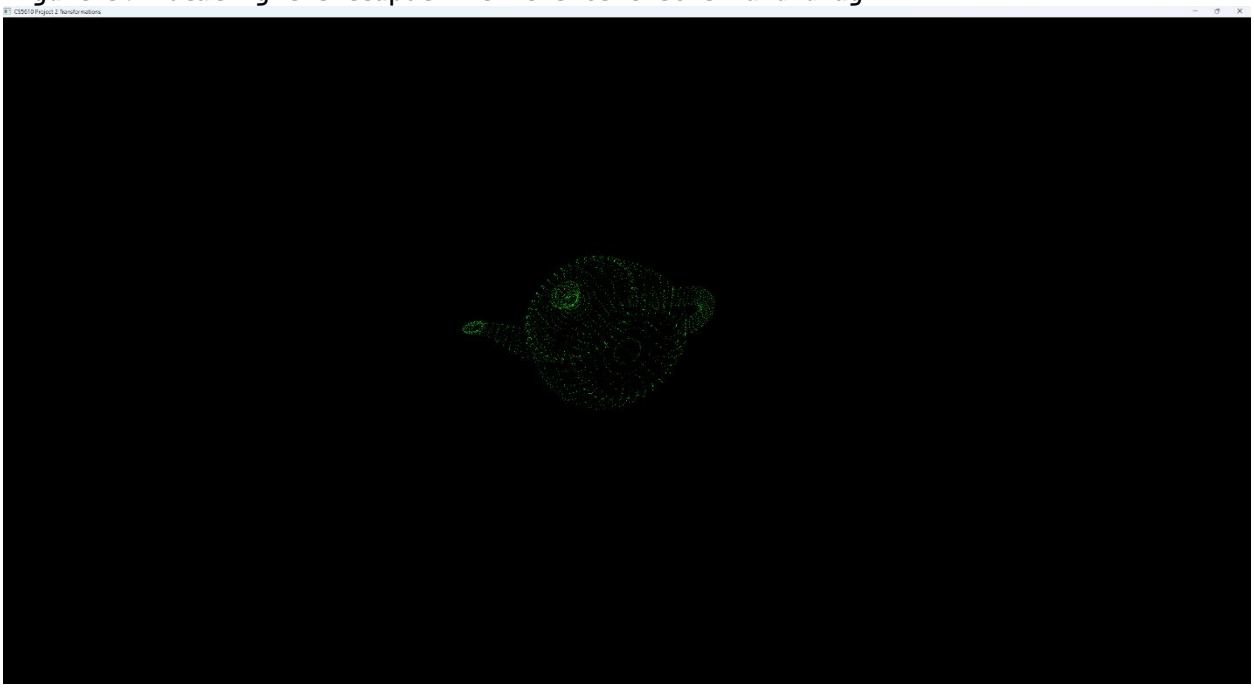
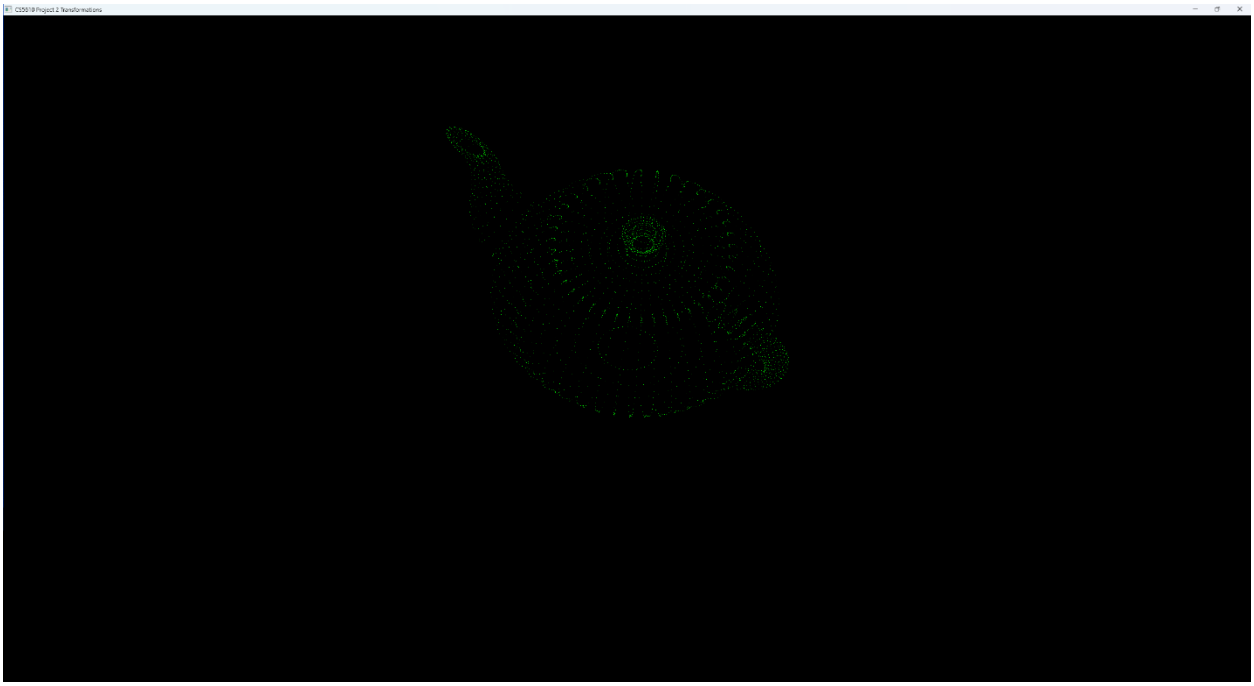


Figure 4: Another rotation with left click and drag



F

Figure 5: Orthogonal transformation

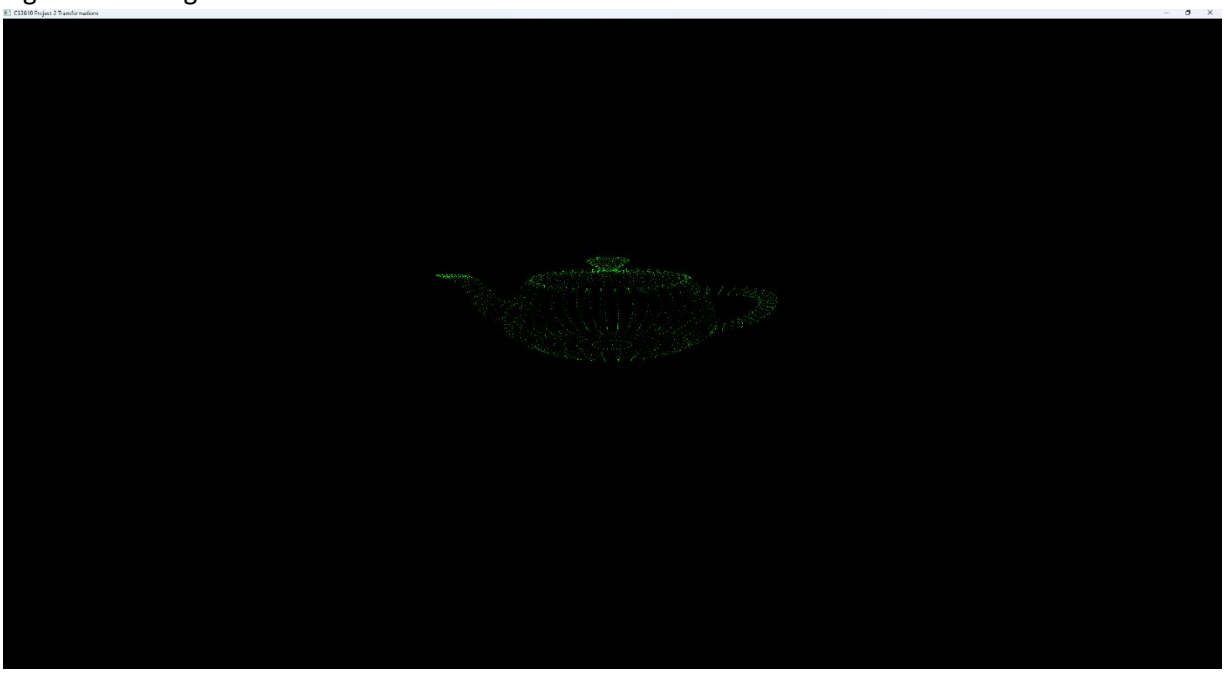


Figure 6. Orthogonal and WAY zoomed out