**Programming Project 5 Report**

Student Name: Robert Johnson

Student UAID: 010796992

**Problem Statement:**

The goal of this project was to work with textures in OpenGL. More specifically working on putting them on 3D objects. We had to upgrade our 2D Tetris objects into 3d, put correct textures on them and be able to rotate/translate them in the 3D space. The inputs needed were to be able to type (I,J,L,O,T,S,Z) which are all of the 7 Tetris pieces. The output should then produce the correct Tetris piece with the correct texture on it and then you can hit the arrow keys or X/Y to rotate or translate the pieces.

**Design:**

On the design portion I decided to just use a handful of global variables, mainly arrays that will be able to store various things like the texture names, textures, (x,y,z) rotation angles and positions for each piece added. Many other variables regarding what was needed. I decided to do this instead of making a struct/class for a piece just simply because it is what I preferred, no other reason, did not think it was that important

I do not think there are any real pros or cons for the choices I used. This project was designed to get us to use specific formulas so almost everyone should have used the same formulas/algorithms for these operations. The only big differences for this program will be what keys they decided to use for their inputs and/or if they decided to make a class/struct for the pieces.

**Implementation:**

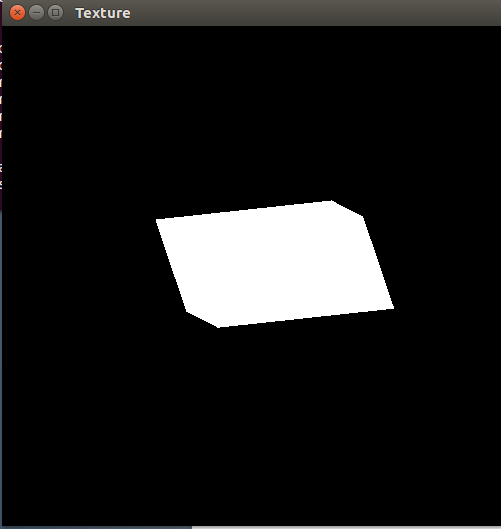
I started with the texture3.cpp sample code as was instructed to do. To extend this code you had to correct multiple global variable arrays to store various things that you would need for each task like textures, rotation angles, positions, etc. Then in the init() function you read in the 7 different textures and store those in your respective data structure.

For task 2 and 3 I brought in some code from tetris2.cpp and adapted it into how the texture3 code worked. Basically instead of calling draw\_square() function you change it to calling block() function 4 different times to create the respective Tetris piece made up of 4 cubes. The code for the keyboard/special callbacks were kept very similar to tetris2.cpp and how other projects used them in the past to add Tetris pieces and rotate/translate the last object created.

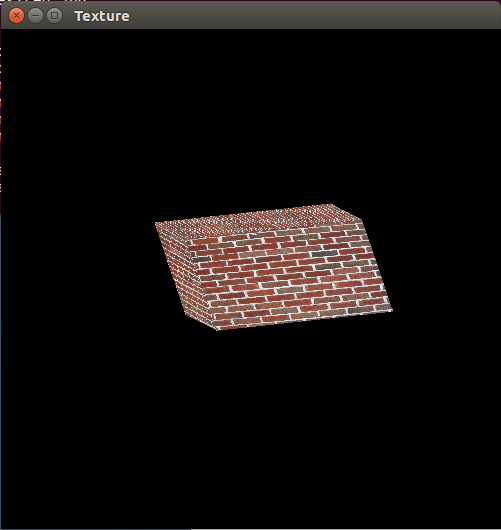
The development timeline for this project was much quicker than previous projects, only took a few hours of working time each day over 2 days.

**Testing:**

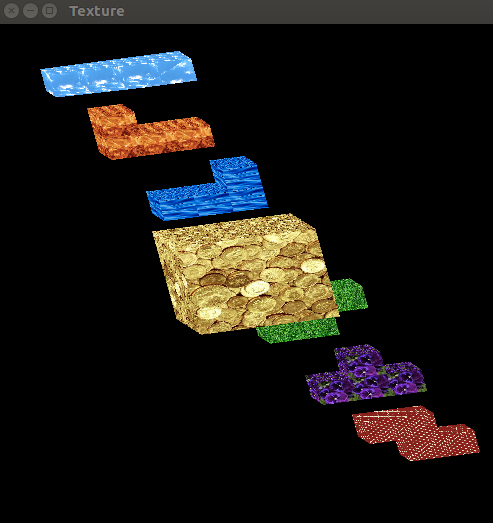
Once again, I used a Virtual Machine running Ubuntu to compile and test my code. The first task we just had to keep the texture3 code and figure out how to read in the 7 different Tetris textures and put them on the initial cube whenever you hit the respective input keys. This first task is actually what I spent the most time stuck on, because I didn’t understand how the init\_texture() function and the glTexImg2d() functions work and how textures really worked in general. At first, I hadn’t setup data structures to store each texture and I didn’t know when/what OpenGL functions to call to put on a new texture. So initially this is what I was getting after I would hit an input key.



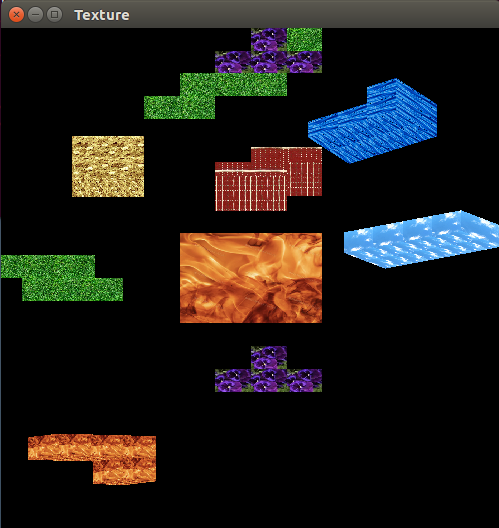
The texture would just turn white no matter which key. After studying the texture3 code and doing some OpenGL research online I understood I needed to setup arrays to store the textures and how/when to call glTex2dImg() to put on a new texture. This was the result.



After finishing task 1, task 2 and 3 was simpler in comparison. Basically just had to implement the tetris2 code into how the texture3 block() code worked which wasn’t hard. This is how my program was working after I finished task 2 and was at least getting the Tetris images to spawn with their correct textures.



Then I just implemented the keyboard/special callback functions from tetris2 and added a little bit onto them. The translating was easy to setup because that was literally given to us with tetris2. The most time consuming part was figuring how to setup the display() function to get the objects to rotate correctly. This was done with a push/pop method. This was the final product, cool little image after I spawned a few different pieces and translated/rotated them around.



**Conclusions:**

Overall, this project was a good success for me. It was a good “balanced” project I would say. Was not too easy but not too hard like the previous two projects. I had to do some research, understand how all the code works together which I enjoyed learning and implementing. Good way to work with OpenGL textures and working in 3D space as well. I completed all the tasks correctly (I think), and it didn’t take that many long hard hours to figure out. Did not even need to ask for help with this one like I did on almost all of the other projects. I honestly do not know if I would do anything differently next time, maybe make a Struct for the pieces if I really needed to. Good project, somewhat challenging and fun, the final product is nice to look at and feels good to literally see what you created.