COSC 4370: Interactive Computer Graphics

Instructor: Prof. Zhigang Deng

Name: NHAT NGUYEN

UID: 1652765

HW2 REPORT

Assignment problem:

Create several 3D scenes with OpenGL.

Sub-problems:

- Install Glut/freeglut to OpenGL and get it to work.
- Read, understand, and figure out how to run the given code and get an output.
- Find out which function to use and how they work.
- Find the pattern of the object in the scene for easy for loops.
- Implement the code into the pattern to get the correct output.

First thought:

Glut is kind of not hard to install but doesn't leave too much room for adjustment, I must follow the instruction exactly to get it to work. Try out the functions and my newfound OpenGL skills from class. Review some geometry to implement on problem #1, incremental steps for #2, nested for loop for #3 and brainstorm some ideas for #4.

Process:

Problem #1:

A circle immediately came up at first glance at the image. Unit circle was used to minimize the work. The teapot was too big and need to be zoom out to see the whole image. It was scaled down for convenient. This is also implemented in problem #2 and #3. The generating function was then put inside a for loop with i range from 1 to 10. x and y is calculated using x = r * cos(angle_a * i) and y = r * sin(angle_a * i) as in image 1.

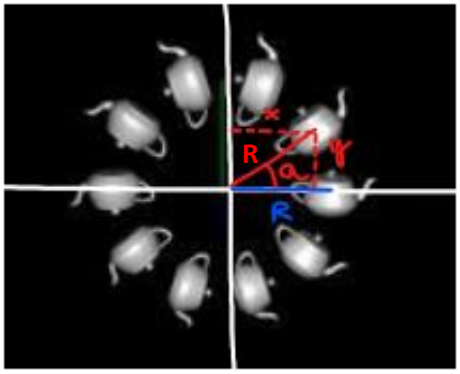


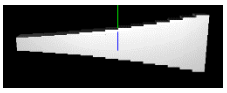
image 1

Then as i increase, the angle increase generating all the other teapots.

Note: This is basically first problem done, but I went further and make i loop for 360/angle_a times, which let the user customize how many teapots they want to see base on the angle.

Conclude problem #1, noticed trigonometry functions such as sin, cos, etc. take angle in rad while glRotatef() take angle in degree.

Problem #2:

First look of the image give us a for loop with x move base on i, y scale base on i and using glutSolidCube() function. After implementing the loop, a double stair  was achieved, because the glScalef() function scale the shape anker at the center of the shape. Now, each step must be move up half of its height to flatten the bottom stair. After a lot of trial and error, tweaking and adjusting, I got the bottom stair to flat out.

In the process of making the stair, I noticed the height of each step of the stair does not increase by a constant, but by an increasing number. Trying to achieve this generate some weird y function.

Note: I also went a step beyond and make the number of steps adjustable so the user can choose how many steps they want. But after testing, the stair became more and more warped as the number of steps increase.

Problem #3:

A nested for loop with one loop to move the teapot down and one to generate them horizontally immediately came to mind. The method to calculate x and y using cos and sin from problem #1 was reused, the angle this time was 225° or 5π/4. The previous x and y were used to calculate the next x and y. The inside for loop increment x by j * offset which also calculate using cos to move the shape horizontally, y stay the same. Do this 6 times will get us the desired scene.

After further inspection, this method is very not efficient, as the next level can easily be generated from moving the first pot down and to the left, without having to do multiplications. But I won't change it because efficiency does not matter at 6 levels.

Note: Same as problem #1 and #2, I also let the number of levels be modifiable. The user can choose how many levels of teapot to be generated.

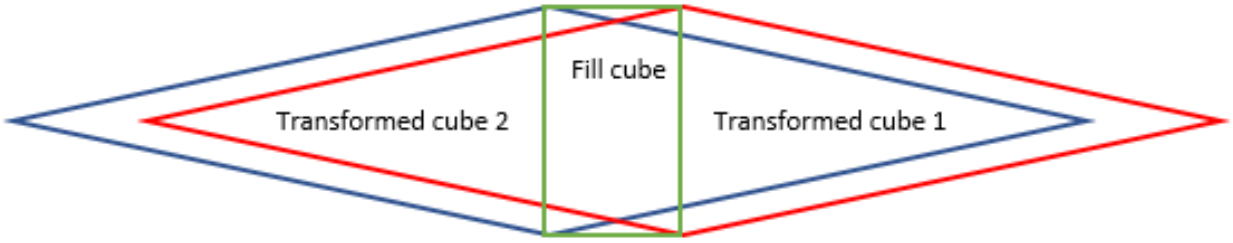
Problem #4:

After a session of brainstorming, a lot of ideas came to mind, but a sword and Triforce was selected because it's easiest to make and looks the coolest.

The blade was generated first by expanding a cube along y axis, then turn it 45^{o} inside one push and pop matrix nested inside another push pop matrix by scaling it down along z axis to flatten it.

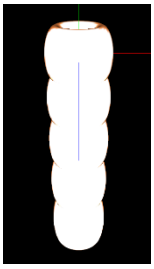
The tip of the blade was also generated through similar procedure, but with a solid Octahedron shape.

The guard compose of 2 transformed cubes, one middle cube (or hyperrectangle) to bridge the gap for the embedded crystals



The crystals are 2 solid Octahedron flatten out.

The handle is 5 mildly vertically stretched tori (plural of torus, as in the function glutSolidTorus()). Generated using a for loop, reduce diameter at the end.



The handle, comprised of tori

The pommel is a normal dodecahedron at the end of the handle.

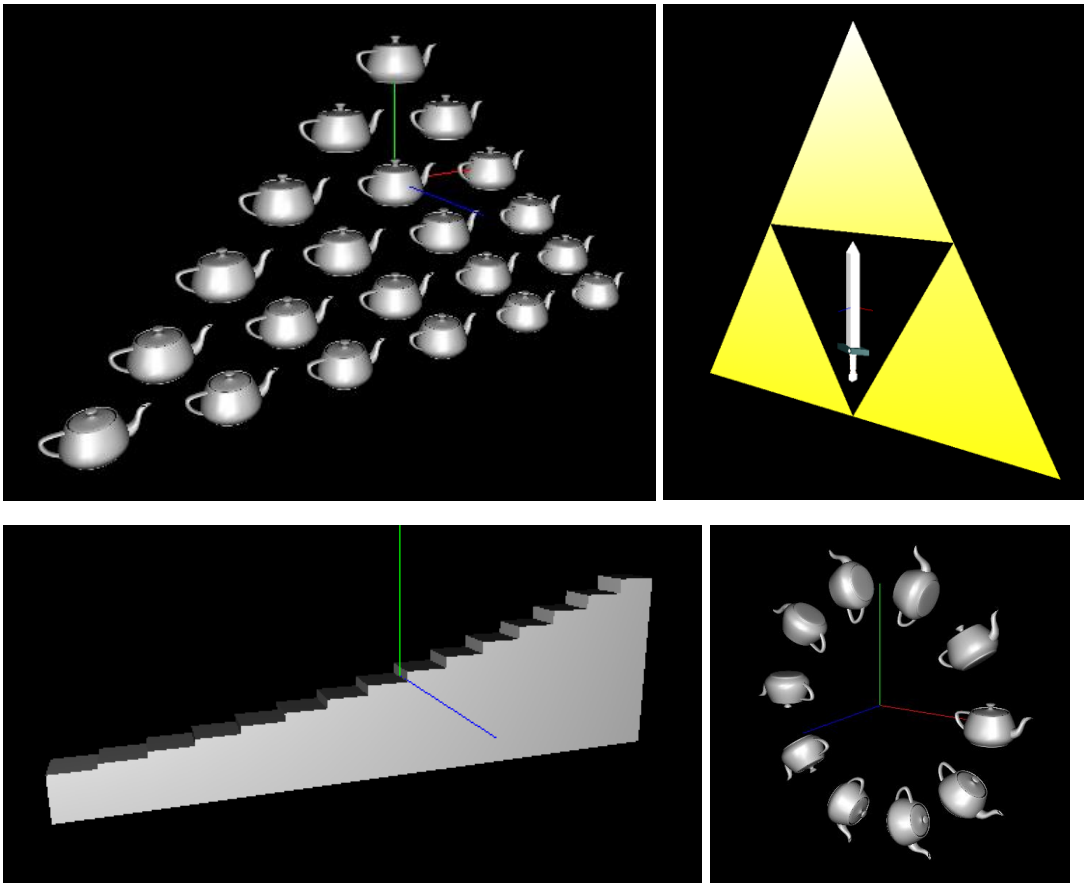
The Triforce, or the 3 triangles generate by feeding in their coordinates directly.

Note: An attempt to color the sword was committed but was not successful due to the lack of knowledge. I also notice the 5 tori made 4 grooves, in which fit 4 fingers. So, in theory, this is a quite a practical sword, an interesting detail.

Method using:

A lot of geometry, trigonometry, critical thinking, trial and error, and imagination.

Result:



Conclusion:

After completing this assignment, I came out some experience in OpenGL. I am somewhat comfortable in generating images and shape using OpenGL commands. Furthermore, I have a chance to review most of my geometry and trigonometry knowledge. Finally, the freestyle part was fun.