COSC 4370 – Homework 2

Name: Rayyan Rahman

PSID: 1893113

February 28, 2023

1. Problem

   The assignment requires us to use OpenGL's transformation mechanisms to reproduce 3 images as well as create our own image by rendering at least one triangle with direct coordinates and at least one instance of a nested matrix (i.e a glPushMatrix within another glPushMatrix).

2. Method

   It was required that I program 4 different functions to reproduce 3 images and create my own image as well. The assignment required us to use glutSolidTeapot and glutSolidCube on top transformations functions to produce a pyramid of teapots, a circle of teapots and a staircase / histogram. For my own image I designed a sun with a mini diamond in the middle. General idea was to use glTranslatef as well as glRotatef and even glScalef to place the objects in the correct position with the correct dimensions.

3. Implementation

   For creating the circle consisting of pots, I first created an angle and orientation variable with type float. This is because the angle of teapots being plotted from each other as well as the teapot orientation consisted of a decimal values. Since the teapots are being placed

in a circle, I used a for loop to iterate it 11 types since there was 11 teapots in the circle and placed them at the cosine and sine of the angle as well as orientated it based on x – axis values.

For creating the histogram / staircase, I first declared the number of steps I needed which was 20 and created a constant ‘Depth’ and ‘Width’ variable as the height is only being changed and not the width or depth. For the height, I created an array of heights called “histogramData” to make the staircase decrease by 0.25 in height after each iteration. Then iterated through a loop which translated each solid cube based on width for x, and histogramData[i] for y to find the correct position of cube placement. Then I scaled to the correct height and depth to form the staircases.

For creating the teapot pyramid, the "translation" array is used to specify the initial x-coordinate of each row of teapots. The code uses a nested loop to draw the teapots. Within the nested loop, the code uses glPushMatrix() to save the current modelview matrix, glTranslatef() to translate the teapot to the correct position, glRotatef() to rotate the teapot, glutSolidTeapot() to draw the teapot, and glPopMatrix() to restore the modelview matrix. The translation is calculated using the "translation" array to specify the starting position of each row, j*0.3 to add spacing between each teapot in a row, and i*teapotSize/2 to stagger each row. The y-coordinate of each teapot is calculated using (5-i)*0.3 + teapotSize, which places the first row at the top of the triangle and subsequent rows lower. The x-rotation of each teapot is determined by the outer loop index, i, and is calculated using glRotatef(5.0*i, 1, 0, 0), which rotates the teapot around the x-axis by 5*i degrees. Finally, it draw the teapot with the specified size. Once the teapot is drawn,

glPopMatrix() is called to restore the modelview matrix to its previous state before drawing the next teapot.

For the custom image, it draws 12 rays emanating from a central point. For each ray, it first sets the color to yellow and rotates the current matrix by an angle of i * 30 degrees around the Z-axis, where i is the index of the current ray. It then draws a triangle with the center point at (0,0), and two other points at (0.2,-0.2) and (0.2,0.2) relative to the center point. After that, it pushes the current matrix onto the stack and sets the color to black, and then draws a quadrilateral (a square) with vertices at (-0.1,0), (0,0.1), (0.1,0), and (0,-0.1), which serves as the arrowhead of the ray. Finally, it pops the matrix from the stack to restore it to its previous state before moving on to draw the next ray.

4. Results

The output of the file is a graphics window and when clicking on numbers between “1-4” on the keyboard, it will switch to different images.