OpenGL Assignment Documentation

18.04.2021

Gadasu Sreekar
B170739CS

How to run

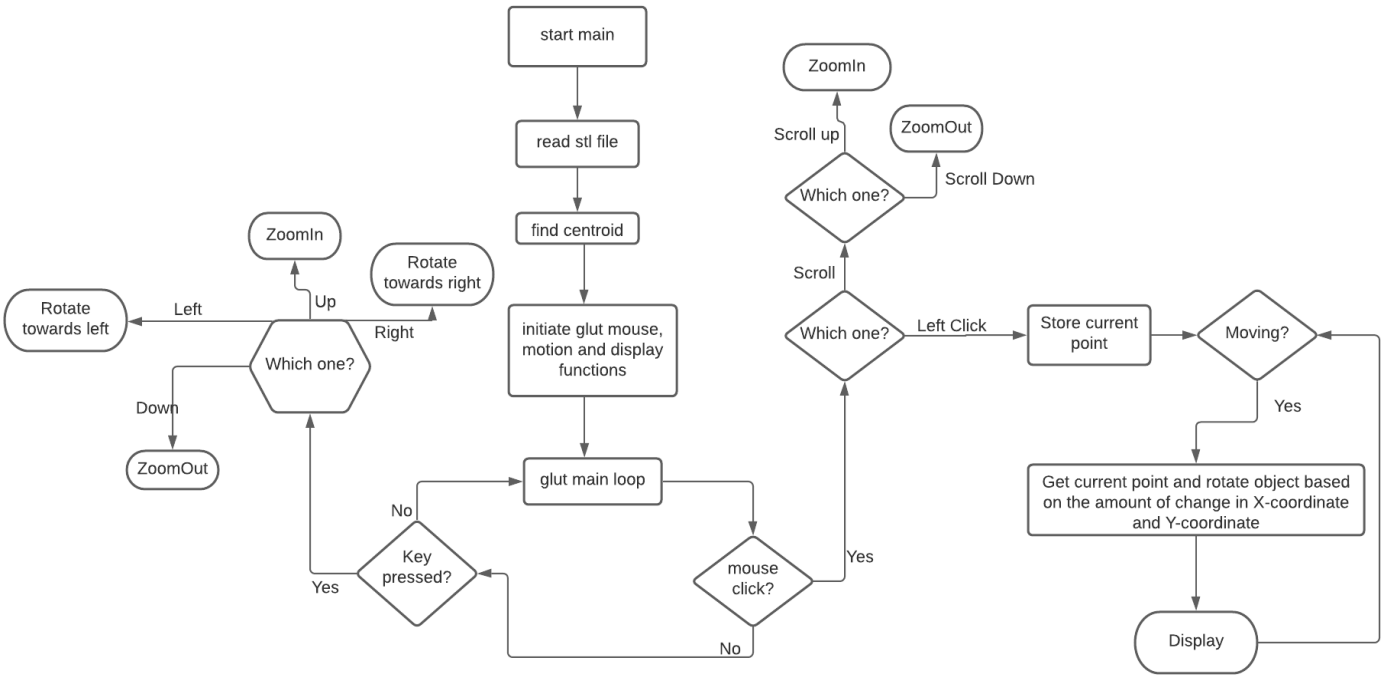
1. Make sure you have code.cpp and the stl files(in .txt format or .stl format) in the same directory.
2. Compile the code using the command:

```
g++ code.cpp -lglut -lGL -lGLEW -lGLU -o OpenGLExample
```

3. Run using the below command:

```
./OpenGLExample <file_name>
```

Functional Flow



Functions and their description

1. `void zoomIn()`

   zoomIn function multiplies the coordinates of each vertex to make a bigger projection when scrolled up.

2. `void zoomOut()`

   zoomOut function diminishes the coordinates of each vertex to make a smaller projection when scrolled down.

3. `void keyPressed(int key,int x,int y)`

   - Users can rotate the object by using keyboard arrow keys.

   - Up arrow key increases the angle of projection with X-axis by a value of 5deg on each stroke.

   - Down arrow key decreases the angle of projection with X-axis by a value of 5deg on each stroke.

   - The left arrow key decreases the angle of projection with Y-axis by a value of 5deg on each stroke.

   - The right arrow key increases the angle of projection with the Y-axis by a value of 5deg on each stroke.

4. `float pow0_1(int a){}, float pow10(int a){}, float gen_vertex(char input[15]){}`

   - These three functions are used for data processing incase of the STL file having scientific notation.

   - Two are for making prior part and post part of number (example. 230e-10) to decimal numbers.

   - Then the gen_vertex is used to generate the float value by combining the both produced above.

5. `void STL_Read(char* s)`

   - Reading the stl files.

   - I ignored the first line as most of the stl files don't have it the same(we will lose some important data but now for us it is not needed).

   - Read data one after the other vertex and append it to the vertices vector.

- Each 6 consecutive numbers in a stl file refers to a vertex, where first 3 are the coordinates and other 3 correspond to the color data.

- Each 3 consecutive vertices make up a triangle and all these triangles make up the object we have to render.

- While reading the vertices we also calculate the centroid of the object around which I restricted the movement(rotation).

- Reading stops when we reach the word that starts with 'e' on the first line of a vertex.

```
6. void mouse( int button, int state, int x, int y )
```

- Mouse function is invoked by glut as we have given 'mouse' to be executed when there is a movement in the mouse.

- Button gives us which button of the mouse it is being pressed.

- **button:**

    -0- left click, 3- scroll up, 4- scroll down.

- **state:**

    - 0- pressed down, 1- released.

- When the user scrolls up, the zoomIn function is called and the projection gets bigger.

- When the user scrolls down, the zoomOut function is called and the projection gets smaller.

- When the user tries to drag:

    - when the left click is pressed down we record the coordinates of the mouse pointer when pressed down.

    - when the user moves the mouse keeping the left click pressed, we track the motion using motion().

```
7. void motion( int x, int y )
```

- Now we have the centroid of the object and the coordinates of the mouse when it is first pressed down.

- And now with this motion tracker we get the current coordinates of the mouse.

- **Method 1:** now our job is to rotate the object based on the angle that is formed by the current point, the started point with the centroid.

- this seems to be right fit for this but when we find the angles it has to be calculated by the cosine rule and that gives a not-defined (nan) when it encounters a 90deg.

- **Method 2**: project the triangle formed by those 3 points on to XZ plane and YZ plane to get the angles formed in each of the planes and transform the projection by that angle.

- this also proved to be wrong as here also there is a chance of getting nan for cosine inverse and the program misbehaves.

- **Method 3:**

- get the difference between x-coordinates and if that is positive increase the transform angle around Y-axis by 1 else decrease by 1.

- get the difference between y-coordinates and if that is positive increase the amount of transformation around X-axis by 1 else decrease by 1.

- and also to smoothen the user interface, if the difference between x-coordinates over dominates that of y-coordinates, make the transformation around x-axis to zero.

- in the same way if the difference between y-coordinates over dominates that of x-coordinates, make the transformation around y-axis to zero.

- **We are using method 3.**

- **NOTE:** after each call of motion function we call glutPostRedisplay() to re-render by calling the display function.

- Also we put a sleep of 5ms to smoothen the user interaction.

```
8. void display()
```

- This is called whenever openGL has to render the output image on screen.

- We draw using the GL primitive 'GL_LINES' to draw all the triangles of each 3 consequent points(a,b,c) as lines ab, bc,ca.

- Give pink colour for drawing.

- Rotate the object projection by theta2 around X-axis and theta1 around Y-axis by using glRotatef() function.

- **NOTE:** The angles theta1 and theta2 are generated in motion() function.

- Translating the projection to the center of the window each time is done by glTranslatef function.

```
9. int main( int argc, char** argv )
```

- Take input file name by command line argument and pass it to STL_Read function.

- Translate the object so that it's centroid moves to origin.

- Calling glutInit().

- Setting window size.

- Creating a window with name "B170739CS".

- Initiating mouse function by using glutMouseFunc.

- Initiating motion capture function by using glutMotionFunc.

- Initiating display function using glutDisplayFunc.

- Start rendering with initRendering function.

- Giving display function as the idle function, this runs display function whenever the program is idle.

- Starting the main loop.

Conclusion

The program runs with either of the extensions (.txt or .stl), please ensure that the object and author information is confined to only one line in the stl file inorder for the program to run properly. And the Standard Triangle Language(STL) should be in ASCII format.