CNG 477 Introduction to Computer Graphics

Spring 2015-2016

Homework #3

The Tangram Puzzle
Due Date: April 17, Sunday, 23:55
(submit via ODTU-Class, -20pts/day late submission penalty)

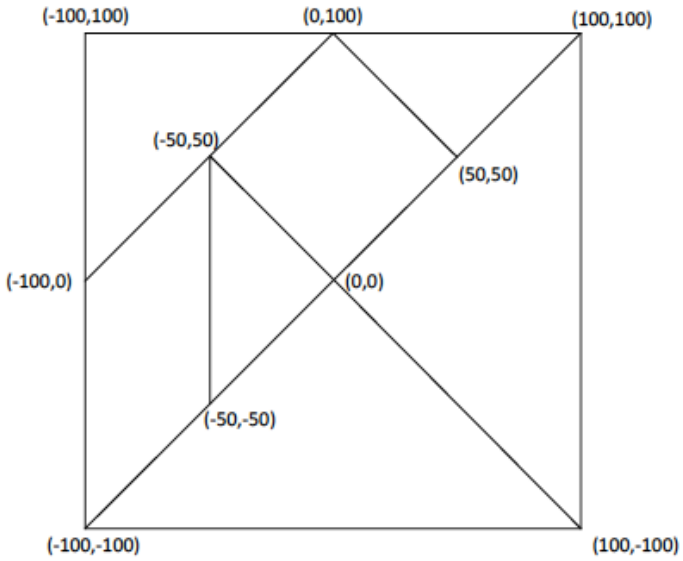
In this assignment, you are going to write an interactive tangram puzzle editing program using the OpenGL library. Tangram is a 2D puzzle in which different shapes are constructed by moving around (using translation and rotation) 7 different pieces cut out of a square. The pieces of the tangram puzzle are shown in the picture below.

In this homework, you are going to write a keyboard based interactive program to let the user interactively select different pieces and move and rotate the selected piece. Below are the detailed specifications:

Specifications:

- The width and height of the GLUT window will be 600 by 600.
- You are going to setup the OpenGL display so that the 2D region between $x \in [-300,300]$ and $y \in [-300,300]$ will be displayed withing the GLUT window. The origion (0,0) will be at the center of the screen.

The 7 puzzle pieces are going to make up a square of size 200 by 200. The square is going to be centered at the origin. The coordinates of the vertices of the puzzle pieces are given below:



- You only need GL_QUADS and GL_TRIANGLES primitives in order to draw all the puzzle pieces.
- You need to provide the coordinates in the picture above to draw the puzzle pieces. You are not allowed to write different coordinates in a glVertex** call in your program. The movement and rotation of the puzzle pieces should be done by changing the MODELVIEW matrix only either by calls to glTranslate* or glRotate* functions or by directly modifying the MODELVIEW matrix using the glLoadMatrix* function.
- The puzzle pieces are going to be selected by using the keys from '1' to '7'. The key '0' will undo the selection. In other words, when the user presses the '0' key, no puzzle piece will be selected. You may number the puzzle pieces any way you want.
- The selected puzzle piece will be shown in green color. The unselected pieces will be in red.
- The user will be able to move the selected puzzle piece up, down, left, and right by pressing the 'w', 's', 'a', and 'd' keys. For each press of the respective key, the piece will move 3 units in the respective direction.
- The user will be able to rotate the selected puzzle piece by pressing the 'r' key. The selected piece will be rotated with respect to its center of mass. You do not need to compute the center of mass exactly; the center of mass can be computed approximately (See the sample executable and the video). For each press of the 'r' key, the piece will rotate 3 degrees in counter clockwise direction.
- When the program is run the first time, the puzzle pieces should be drawn separated from each other. In order to achieve this you may translate different pieces differently. For example the big triangle at the bottom may be translated 7 units down, while the big triangle on the right may be translated 7 units to the right.

Additional Information:

You will get the keys pressed by the user by registering a keyboard callback function using the GLUT function glutKeyboardFunc(<your function name>) function. Your function should have the following prototype

void <your function name> (unsigned char, int, int)

The first parameter gives the key that is pressed as an ASCII character.

After user interaction, the OpenGL display function is not automatically called. OpenGL automatically calls the display function only when it is necessary to redraw the scene, for example, when part of the window disappears and appears on the screen. In order to call your display function manually, you need to call the glutPostRedisplay() function at the end of your keyboard function.

You will move the puzzle pieces by modifying the MODELVIEW matrix using calls to glRotate* or glTranslate* or directly loading a matrix by the glLoadMatrix* function.

The puzzle pieces are independent of each other. In other words, you will have to apply different MODELVIEW matrices before drawing each puzzle piece.

You can get the contents of the current MODELVIEW matrix using the glGetFloatv(GL_MODELVIEW_MATRIX,*matrix*) function. Here, *matrix* is a one dimensional array of 16 floats. For example, the following piece of code gets the current matrix and then loads it again as the MODELVIEW matrix. The code does not have any effect, because overall the MODELVIEW matrix is unchanged after execution of the code. However, used in different parts of the code you can achieve different effects with these two functions.

```
float matrix[16];
glGetFloatv(GL_MODELVIEW_MATRIX,matrix);
glLoadMatrixf(matrix);
```

Bonus:
If you can implement interaction using the mouse instead of the keyboard (for selection and translation of puzzle pieces) you will get a bonus of 20 pts.

Notes:
All the parameters that are not specified in the specifications is up to you. In other words, you are free to decide on anything related to the homework that is not specifically mentioned in above.

Submission:

Submit your solution via ODTU-Class before the deadline.

A sample executable and the video of an example run can be found at the following addresses:

http://www.ceng.metu.edu.tr/~tcan/cng477/hw1.exe
http://www.ceng.metu.edu.tr/~tcan/cng477/hw1.wmv

Note: You need glut32.dll in order to execute hw1.exe