Computer Graphics, Lab Assignment 4

Handed out: March 27, 2019

Recommended due: 15:00, March 27, 2019

Hard due: 23:59, March 27, 2019 **(NO SCORE for late submissions!)**

Submit your assignment only through the page of this course at learn.hanyang.ac.kr.

1. Write down a Python program to draw a transformed triangle in a 2D space.

   A. Set the window title to **[studentID]-[assignment#]-[prob#]** and the window size to (480,480).

   B. Draw a triangle using render() function below (DO NOT modify it!).

```
def render(T):
    glClear(GL_COLOR_BUFFER_BIT)
    glLoadIdentity()
    # draw cooridnate
    glBegin(GL_LINES)
    glColor3ub(255, 0, 0)
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([1.,0.]))
    glColor3ub(0, 255, 0)
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([0.,1.]))
    glEnd()
    # draw triangle
    glBegin(GL_TRIANGLES)
    glColor3ub(255, 255, 255)
    glVertex2fv( (T @ np.array([.0,.5,1.]))[:-1] )
    glVertex2fv( (T @ np.array([.0,.0,1.]))[:-1] )
    glVertex2fv( (T @ np.array([.5,.0,1.]))[:-1] )
    glEnd()
```

   C. If you press or repeat a key, the triangle should be transformed as shown in the Table:

Key	Transformation
Q	Translate by -0.1 in x direction **w.r.t global coordinate**
E	Translate by 0.1 in x direction **w.r.t global coordinate**
A	Rotate by 10 degrees counterclockwise **w.r.t local coordinate**
D	Rotate by 10 degrees clockwise **w.r.t local coordinate**
1	Reset the triangle with identity matrix

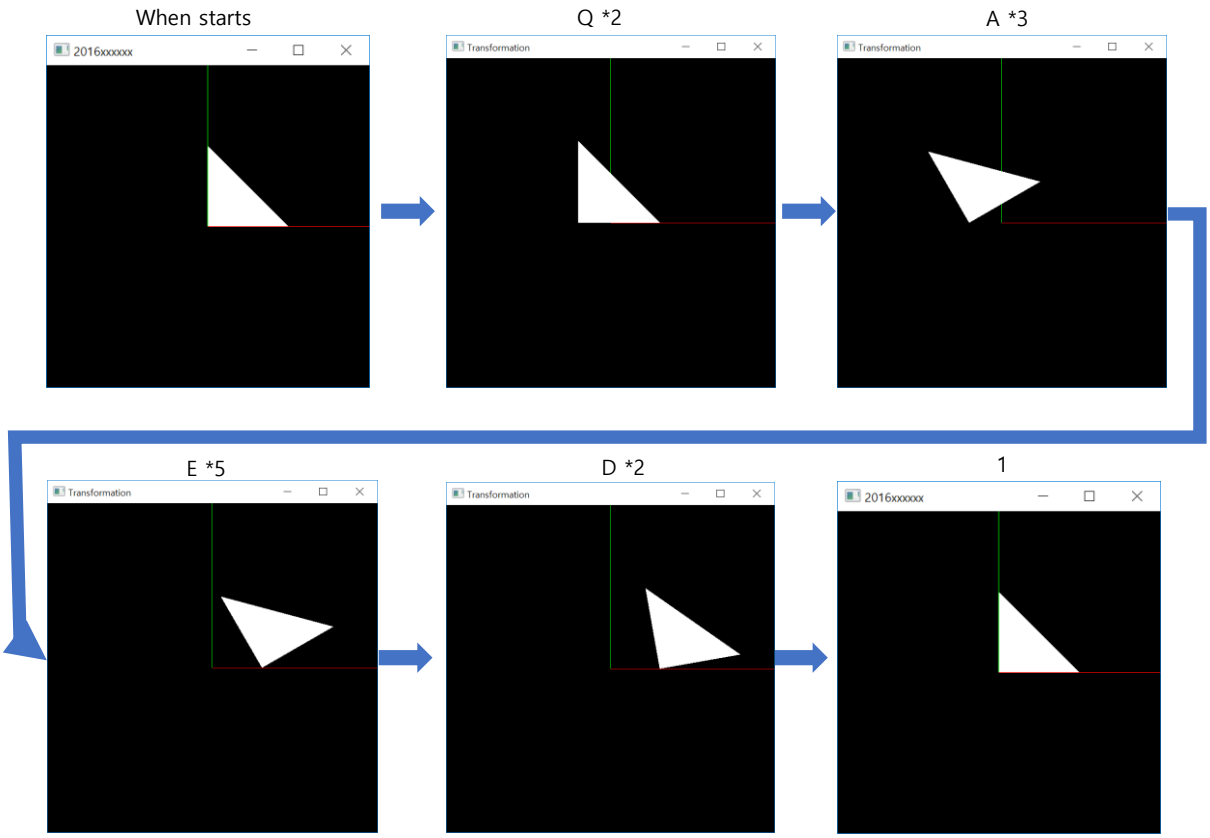
   D. Transformations should be accumulated (composed with previous one) unless you press '1'.

i. You'll need a global variable to store current accumulated transformation.

E. Do not use OpenGL transformation functions.

F. Submit a single .py file - **[studentID]-[assignment#]-[prob#].py**

G. Expected result:



2. Write down a Python program to draw a transformed triangle in a 3D space.

A. Set the window title to **[studentID]-[assignment#]-[prob#]** and the window size to (480,480).

B. Use the following code snippet:

```
gCamAng = 0
gComposedM = np.identity(4)

def render(M, camAng):
    # enable depth test (we'll see details later)
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glEnable(GL_DEPTH_TEST)

    glLoadIdentity()

    # use orthogonal projection (we'll see details later)
    glOrtho(-1,1, -1,1, -1,1)

    # rotate "camera" position to see this 3D space better (we'll see
details later)
    gluLookAt(.1*np.sin(camAng),.1, .1*np.cos(camAng), 0,0,0, 0,1,0)

     # draw coordinate: x in red, y in green, z in blue
    glBegin(GL_LINES)
    glColor3ub(255, 0, 0)
    glVertex3fv(np.array([0.,0.,0.]))
    glVertex3fv(np.array([1.,0.,0.]))
    glColor3ub(0, 255, 0)
    glVertex3fv(np.array([0.,0.,0.]))
    glVertex3fv(np.array([0.,1.,0.]))
    glColor3ub(0, 0, 255)
    glVertex3fv(np.array([0.,0.,0]))
    glVertex3fv(np.array([0.,0.,1.]))
    glEnd()

    # draw triangle
    glBegin(GL_TRIANGLES)
    glColor3ub(255, 255, 255)
    glVertex3fv((M @ np.array([.0,.5,0.,1.]))[:-1])
    glVertex3fv((M @ np.array([.0,.0,0.,1.]))[:-1])
    glVertex3fv((M @ np.array([.5,.0,0.,1.]))[:-1])
    glEnd()

def key_callback(window, key, scancode, action, mods):
    global gCamAng, gComposedM
    if action==glfw.PRESS or action==glfw.REPEAT:

        if key==glfw.KEY_1:
            gCamAng += np.radians(-10)
        elif key==glfw.KEY_3:
            gCamAng += np.radians(10)
```

C. If you press or repeat a key, the triangle should be transformed as shown in the Table. Note that key 1 and 3 are already implemented in the above code snippet.

Key	Transformation
Q	Translate by -0.1 in x direction **w.r.t global coordinate**
E	Translate by 0.1 in x direction **w.r.t global coordinate**
A	Rotate about y axis by -10 degrees **w.r.t local coordinate**
D	Rotate about y axis by +10 degrees **w.r.t local coordinate**
W	Rotate about x axis by -10 degrees **w.r.t local coordinate**
S	Rotate about x axis by +10 degrees **w.r.t local coordinate**

1	Rotate camera -10 degree
3	Rotate camera 10 degree

D. Transformations should be accumulated (composed with previous one).

   i. You'll need two global variables to store current accumulated transformation and current camera angle.

E. Do not use OpenGL transformation functions.

F. Submit a single .py file - **[studentID]-[assignment#]-[prob#].py**