Computer Graphics, Lab Assignment 5

Handed out: April 13, 2020

Due: 23:59, April 17, 2020 (NO SCORE for late submissions!) – extended due to no lab on April 15

- *Only accept answers submitted via git push to this course project for you at https://hconnect.hanyang.ac.kr (<Year>_<Course no.>_<Class code>/<Year>_<Course no.>_<Student ID>.git).*
- *Place your files under the directory structure <Assignment name>/<Problem no.>/<your file> just like the following example.*

```
+ 2020_ITE0000_2019000001
  + LabAssignment2/
    + 1/
      - 1.py
    + 2/
      - 2.py
    + 3/
      - 3.py
```

- *The submission time is determined not when the commit is made but when the git push is made.*

1. Write down a Python program to draw a transformed triangle and its local frame in a 3D space.
   A. Set the window title to **your student ID** and the window size to (480,480).
   B. Use the following drawFrame() and drawTriangle() to draw the frame and triangle:

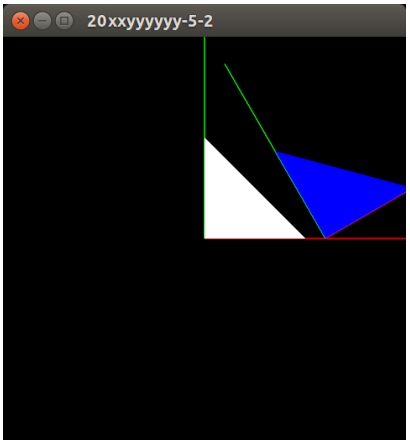
```
def drawFrame():
    glBegin(GL_LINES)
    glColor3ub(255, 0, 0)
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([1.,0.]))
    glColor3ub(0, 255, 0)
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([0.,1.]))
    glEnd()

def drawTriangle():
    glBegin(GL_TRIANGLES)
    glVertex2fv(np.array([0.,.5]))
    glVertex2fv(np.array([0.,0.]))
    glVertex2fv(np.array([.5,0.]))
    glEnd()
```

C. First draw an untransformed white triangle and a global frame.

D. Then draw a transformed blue triangle and its local frame. The triangle should be first rotated by 30 degrees and then translated by (0.6, 0, 0) w.r.t. the global frame.

E. Expected result:

i. 

F. Files to submit: A Python source file (Name the file whatever you want (in English). Extension should be .py)

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

A. Set the window title to **your student ID** 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. Files to submit: A Python source file (Name the file whatever you want (in English). Extension should be .py)