Computer Graphics, Lab Assignment 6

Handed out: April 6, 2021

Due: 23:59, April 6, 2021 (NO SCORE for late submissions!)

- *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.*

```
+ 2021_ITE0000_2019000001
  + LabAssignment6/
    + 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 your own myLookAt() and myFrustum() functions (of the following form) that behaves exactly same as gluLookAt() and glFrustum().

   A.
   ```
   def myLookAt(eye, at, up): # eye, at, up are 1D numpy array of length 3
   def myFrustum(left, right, bottom, top, near, far):
   ```

   B. Set the window title to **your student ID** and the window size to (480,480).

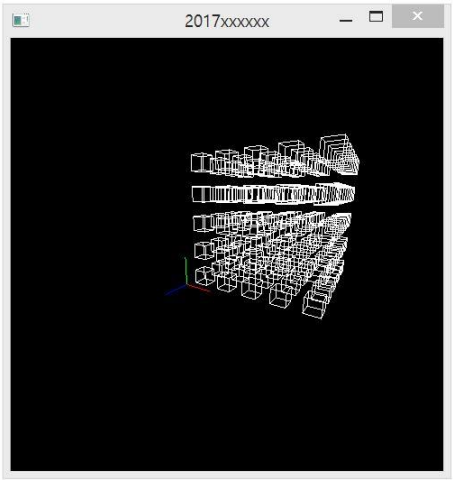
   C. Code skeleton

```
def render():
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glEnable(GL_DEPTH_TEST)
    glPolygonMode( GL_FRONT_AND_BACK, GL_LINE )
    glLoadIdentity()

    myFrustum(-1,1, -1,1, 1,10)
    myLookAt(np.array([5,3,5]), np.array([1,1,-1]), np.array([0,1,0]))

    # Above two lines must behave exactly same as the below two lines

    #glFrustum(-1,1, -1,1, 1,10)
    #gluLookAt(5,3,5, 1,1,-1, 0,1,0)

    drawFrame()

    glColor3ub(255, 255, 255)
    drawCubeArray()

def myFrustum(left, right, bottom, top, near, far):
    # implement here

def myLookAt(eye, at, up):
    # implement here
```

D. Find code for drawFrame(), drawCubeArray() from *6-Viewing & Projection2 & mesh* slides.

E. DO NOT use gluLookAt() inside myLookAt() and glFrustum() inside myFrustum()!

F. Your program should render the following scene:

i. 

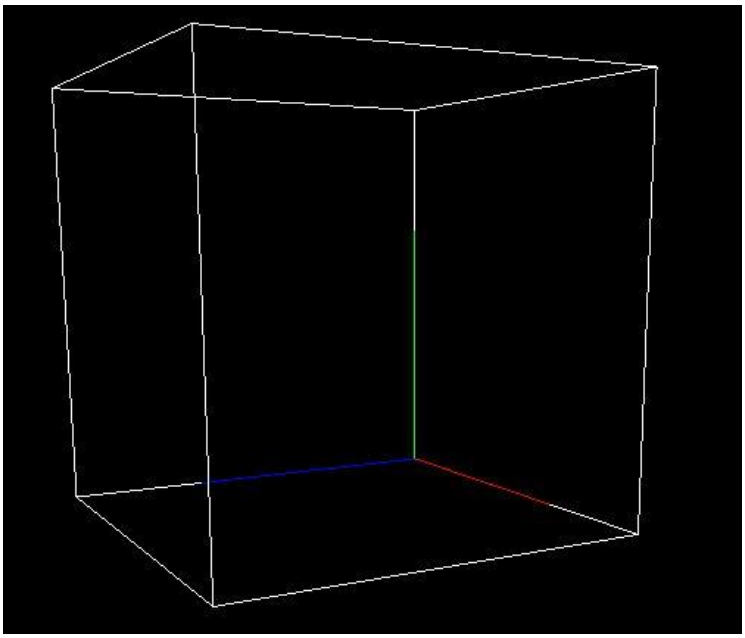
G. Hint:

1. To implement myLookAt(), see lecture slide *5-reference-viewing*. To implement myFrustum(), see lab slide *6-Viewing & Projection2, mesh* .
2. l2 norm of **v** : ||**v**|| = np.sqrt(np.dot(**v**, **v**))
3. **a** x **b** (cross product) : np.cross(**a**, **b**)

4. **a** · **b** (inner product) : np.dot(**a**, **b**) or **a**@**b**
5. Use glMultMatrixf() to multiply your projection matrix and viewing matrix to the current transformation matrix.

H. 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 following cube (정육면체) by using indexed squares representation and glDrawElements().

A. 

B. Length of each line is 1.5

C. Start from the code in *6-Viewing & Projection2 & mesh* slides. Make sure camera manipulation shortcuts '1', '3', '2', 'w' work. (Don't need to care about initial view angle)

D. Set the window title to **your student ID** and the window size to (480,480).

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