Daily Assignment 20

- *Visualizing rotation about an arbitrary axis:* start from today's practice code, replace the render() function by the one in the next page

- 1. Add **getRotMatFrom(axis, theta)** function
  - *axis* : (unnormalized) rotation axis vector
  - *theta* : rotation angle in degrees
  - Returns the rotation matrix for rotation about *axis* by *theta*
  - **Do not use Rodrigues' rotation formula. You have to use the method described in today's slides from "Let's compute the rotation matrix R"**

- Maybe you'll need
  - The provided normalized() to normalize a vector
  - **a** x **b** (cross product) : np.cross(**a**, **b**)
  - **a** · **b** (inner product) : np.dot(**a**, **b**)
  - To build a matrix using column vectors **a, b, c**: np.column_stack((**a, b, c**))
    - google it for more information
  - Inverse of a matrix **M**: np.linalg.inv(**M**)

Daily Assignment 20

- 2. Add key handling code to change the rotation axis
  - If you **press or repeat** a key, the x, y, z coordinate value of the rotation axis (*gAxis* variable in the code) should be changed as shown in the table:
  - (The rotation axis is already visualized as a white line and initialized to (0,1,0) in the code in the next page)

Key	Transformation
A	Increase x by 0.1
Z	Decrease x by 0.1
S	Increase y by 0.1
X	Decrease y by 0.1
D	Increase z by 0.1
C	Decrease z by 0.1
V	Initialize gAxis with (0,1,0)

```
gAxis = np.array([0.,1.,0.])
def render(ang):
    global gCamAng, gCamHeight
    global gAxis
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT)

    glEnable(GL_DEPTH_TEST)

    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluPerspective(45, 1, 1,10)

    glMatrixMode(GL_MODELVIEW)
    glLoadIdentity()

gluLookAt(5*np.sin(gCamAng),gCamHeight,5*np.cos(gCam
Ang), 0,0,0, 0,1,0)

    drawFrame() # draw global frame

    # draw rotation axis
    glBegin(GL_LINES)
    glColor3ub(255, 255, 255)
    glVertex3fv(np.array([0.,0.,0.]))
    glVertex3fv(gAxis)
    glEnd()

    glEnable(GL_LIGHTING)
    glEnable(GL_LIGHT0)
    glEnable(GL_RESCALE_NORMAL)

    glLightfv(GL_LIGHT0, GL_POSITION, (1.,2.,3.,1.))
    glLightfv(GL_LIGHT0, GL_AMBIENT, (.1,.1,.1,1.))
    glLightfv(GL_LIGHT0, GL_DIFFUSE, (1.,1.,1.,1.))
    glLightfv(GL_LIGHT0, GL_SPECULAR, (1.,1.,1.,1.))
```

```
    # for your answer
    R = getRotMatFrom(gAxis, ang)
    M = np.identity(4)
    M[:3,:3] = R
    glMultMatrixf(M.T)

    # # for debugging - your result should be same
with the result from this glRotate() call
    # glRotatef(ang, gAxis[0], gAxis[1], gAxis[2])

    glScalef(.5,.5,.5)

    # draw cubes
    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE,
(.5,.5,.5,1.))
    drawUnitCube_glDrawArray()

    glTranslatef(1.5,0,0)
    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE,
(1.,0.,0.,1.))
    drawUnitCube_glDrawArray()

    glTranslatef(-1.5,1.5,0)
    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE,
(0.,1.,0.,1.))
    drawUnitCube_glDrawArray()

    glTranslatef(0,-1.5,1.5)
    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE,
(0.,0.,1.,1.))
    drawUnitCube_glDrawArray()

    glDisable(GL_LIGHTING)
```

```
def l2norm(v):
    return np.sqrt(np.dot(v, v))
def normalized(v):
    l = l2norm(v)
    return 1/l * np.array(v)
```

How to Submit

- What you have to submit:
  - Only **one** .py file: ***main.py***
- Write down all your code to ***main.py***
- `> py -3 main.py` or `$ python3 main.py` should show your glfw window.

How to Submit

- Submit your assignment **only through the Assignment (과제) menu of the lecture home** at portal.hanyang.ac.kr.

- **Recommended due date: Today's lecture end time**

- (Hard due date: 23:59 Today)