LEC14: Transformation-Part2

Ku-Jin Kim

School of Computer Science & Engineering

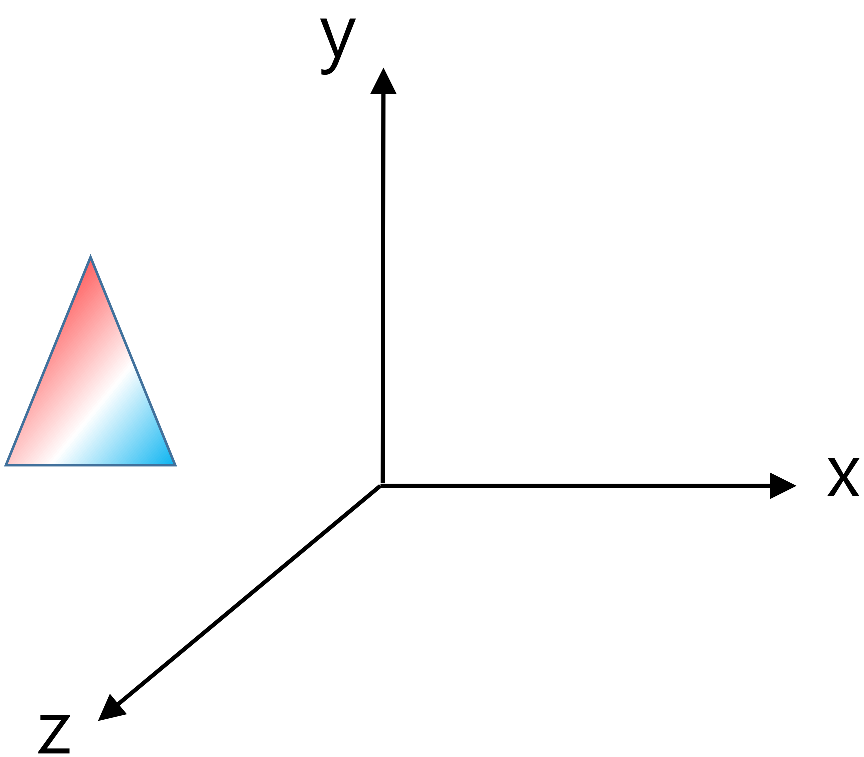
Kyungpook National University

Contents

- Transformations – Part 2
  - Rotation
  - Scale
  - Shear
  - Inverse
- Program Example

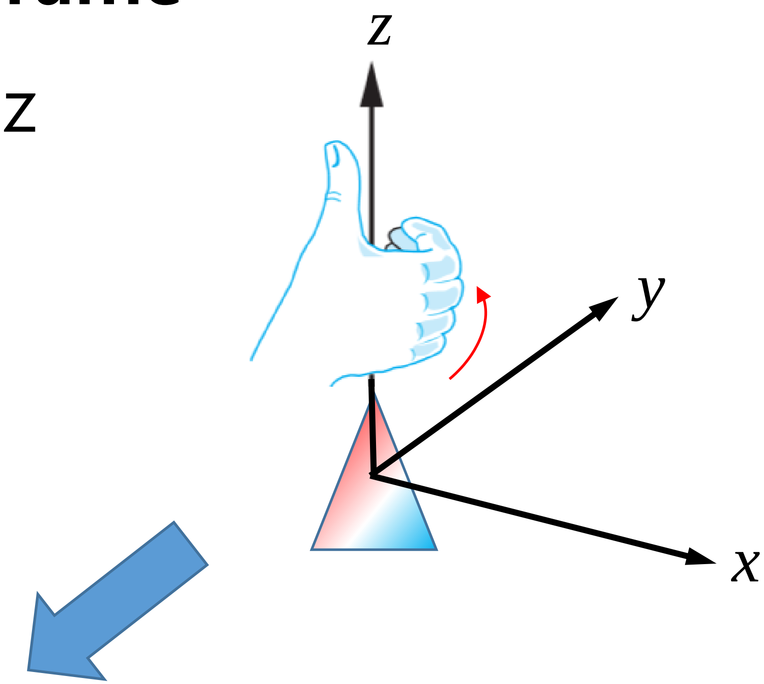
Object Frame & World Frame

- Right-hand rule for x, y, z



World frame

```
gl_Position = utranslate*aPosition;
```

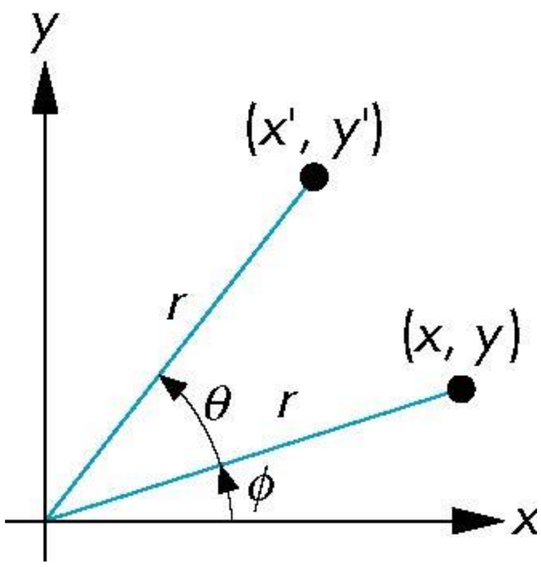


Object frame

```
GLfloat vertices[] =
  { -0.3, -0.3, -0.3, 1.0,
    +0.3, -0.3, -0.3, 1.0,
     ...
  }
```

Rotation

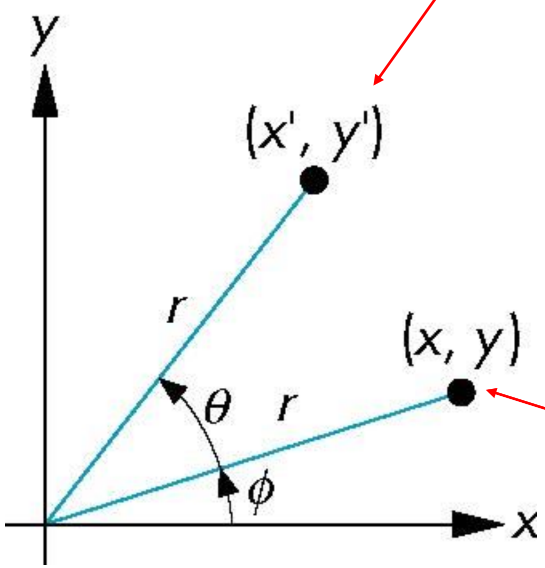
- Consider rotation about z-axis by θ degrees (radian)
  - radius stays the same, rotation angle is θ



Rotation

- Consider rotation about z-axis by θ degrees (radian)
  - radius stays the same, rotation angle is θ

$$x' = r\cos(\phi + \theta)$$
$$y' = r\sin(\phi + \theta)$$



$$x = r\cos\phi$$
$$y = r\sin\phi$$

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$

Greek letters - https://www.wikipedia.org/

Rotation about the z axis

- Rotation about z axis in three dimensions leaves all points with the same z
  - Equivalent to rotation in two dimensions in planes of constant z

$$x' = x \cos \theta - y \sin \theta$$
$$y' = x \sin \theta + y \cos \theta$$
$$z' = z$$

  - or in homogeneous coordinates

$$\mathbf{p}' = \mathbf{R}_{\mathbf{z}}(\theta)\mathbf{p}$$

Rotation Matrix

$$\mathbf{R} = \mathbf{R}_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation about $\mathbf{x}$ and $\mathbf{y}$ axes

- Same argument as for rotation about *z* axis
  - For rotation about x axis, x is unchanged
  - For rotation about y axis, y is unchanged

$$\mathbf{R} = \mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R} = \mathbf{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Program with rotation matrix

```
#include <math.h>
...
GLfloat m[16];
void mydisplay(void) {
        GLuint loc;
        glClearColor(0.7f, 0.7f, 0.7f, 1.0f); // gray
        glClear(GL_COLOR_BUFFER_BIT);

        m[0] = cos(t); m[4] = -sin(t); m[8] = 0.0; m[12] = 0.0;
        m[1] = sin(t); m[5] = cos(t); m[9] = 0.0; m[13] = 0.0;
        m[2] = 0.0; m[6] = 0.0; m[10] = 1.0; m[14] = 0.0;
        m[3] = 0.0; m[7] = 0.0; m[11] = 0.0; m[15] = 1.0;
        loc = glGetUniformLocation(prog, "urotate");
        glUniformMatrix4fv(loc, 1, GL_FALSE, m);
        glDrawElements(GL_TRIANGLES, 12 * 3, GL_UNSIGNED_SHORT, indices);
        glFlush();
        glutSwapBuffers();
}
```

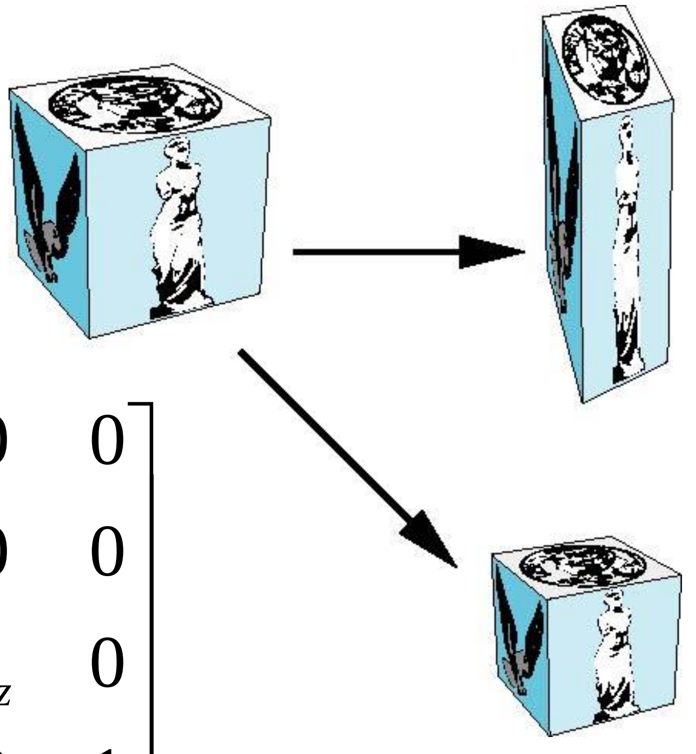
Scaling

- Expand or contract along each axis (fixed point of origin)

$$x' = s_x x$$
$$y' = s_y y$$
$$z' = s_z z$$

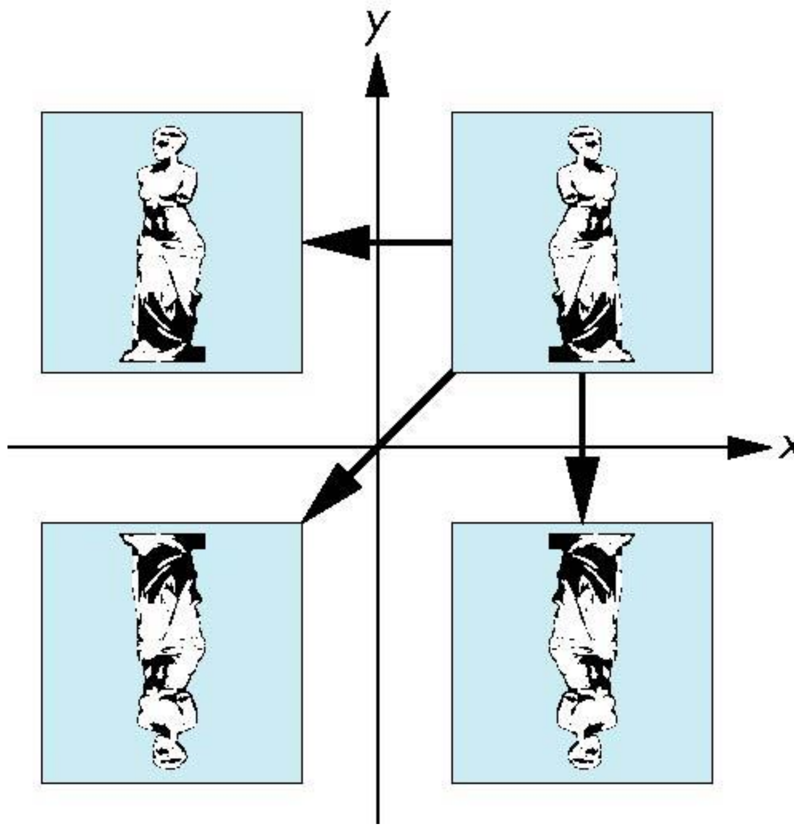
$$\mathbf{p}' = \mathbf{S}\mathbf{p}$$

$$\mathbf{S} = \mathbf{S}(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Reflection

- corresponds to negative scale factors



$s_x = -1 \; s_y = 1$

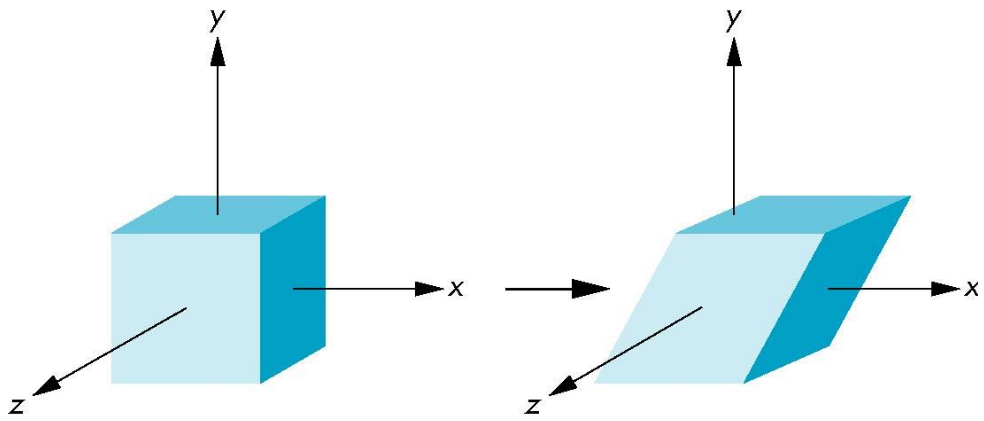
original

$s_x = -1 \; s_y = -1$

$s_x = 1 \; s_y = -1$

Shear

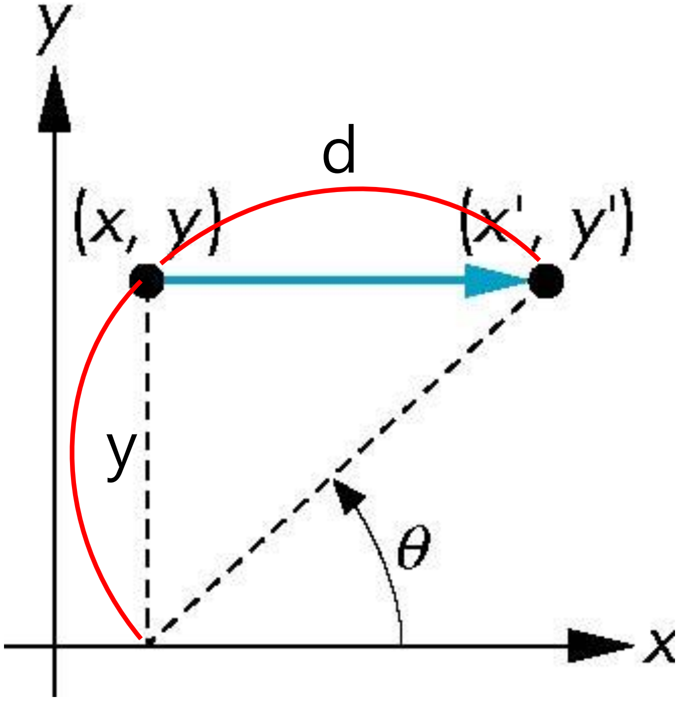
- Equivalent to pulling faces in opposite directions



Shear Matrix

- Consider simple shear along *x* axis

$$x' = x + y \cot\theta, \text{ where } \cot\theta = \tan^{-1}\theta$$
$$y' = y$$
$$z' = z$$



$$\mathbf{H}(\theta) = \begin{bmatrix} 1 & \cot\theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

How can we return the object to the state before transformation?

- For transformation by matrix **M**
  - $\mathbf{p}' = \mathbf{M}\,\mathbf{p} \quad \rightarrow \quad \mathbf{p} = \mathbf{M}^{-1}\mathbf{p}'$
- Do we have to compute $\mathbf{M}^{-1}$?

Inverses

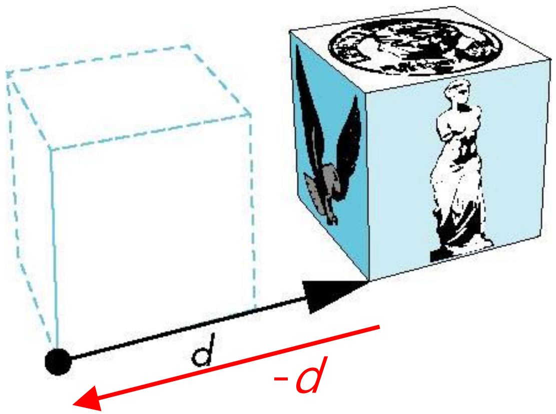
- Although we could compute inverse matrices by general formulas, we can use simple geometric observations
  - Translation: $\mathbf{T}^{-1}(d_x, d_y, d_z) = \mathbf{T}(-d_x, -d_y, -d_z)$
  - Rotation: $\mathbf{R}^{-1}(\theta) = \mathbf{R}(-\theta)$
  - Scaling: $\mathbf{S}^{-1}(s_x, s_y, s_z) = \mathbf{S}(1/s_x, 1/s_y, 1/s_z)$

Inverse of Translation

$\mathbf{p}' = \mathbf{p} + \mathbf{d}$ $\rightarrow$ $\mathbf{p}' = \mathbf{T}(d_x, d_y, d_z)\ \mathbf{p}$

$\mathbf{d} = (\mathbf{d}_x, \mathbf{d}_y, \mathbf{d}_z, 0)^T$

$$\mathrm{T}(d_x, d_y, d_z) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



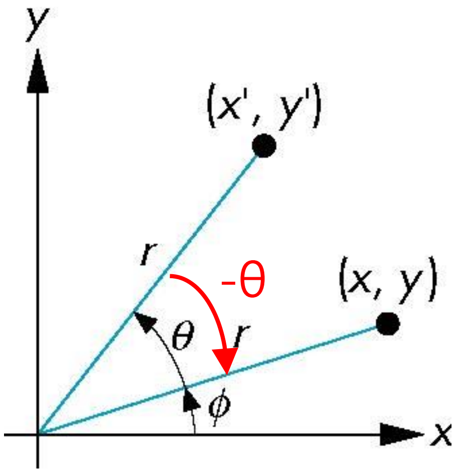
$\mathbf{p} = \mathbf{p}' - \mathbf{d}$ $\rightarrow$ $\mathbf{p} = \mathbf{T}(-d_x, -d_y, -d_z)\ \mathbf{p}'$

- $\mathbf{T}^{-1}(d_x, d_y, d_z) = \mathbf{T}(-d_x, -d_y, -d_z) = \begin{bmatrix} 1 & 0 & 0 & -d_x \\ 0 & 1 & 0 & -d_y \\ 0 & 0 & 1 & -d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Inverse of Rotation

- Rotation: $\mathbf{R}^{-1}(\theta) = \mathbf{R}(-\theta)$
  - Holds for any rotation matrix
  - Since $\cos(-\theta) = \cos(\theta)$ and $\sin(-\theta) = -\sin(\theta)$, $\mathbf{R}^{-1}(\theta) = \mathbf{R}^{T}(\theta)$

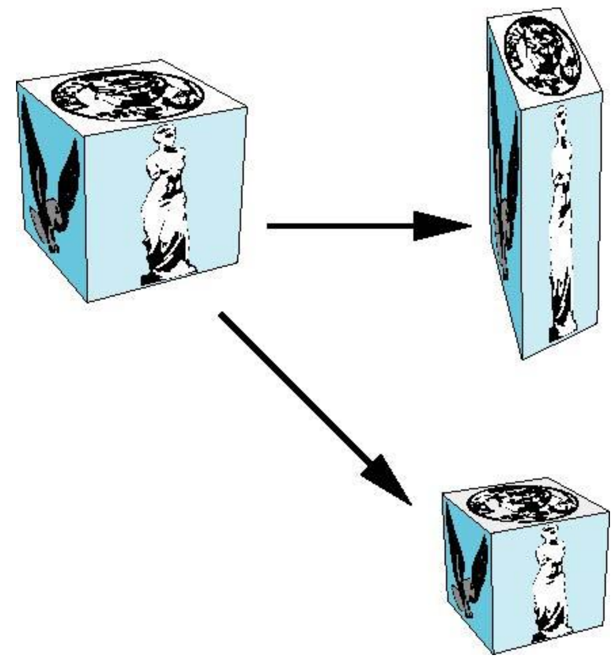
$$\mathbf{R}_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$$\mathbf{R}_z^{-1}(\theta) = \mathbf{R}_z(-\theta) = \begin{bmatrix} \cos(-\theta) & -\sin(-\theta) & 0 & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Inverse of Scaling

$\mathbf{p}' = \mathbf{S}\mathbf{p} \rightarrow \quad \mathbf{p}' = \mathbf{S}(s_x, s_y, s_z)\ \mathbf{p}$



$$\mathbf{S} = \mathbf{S}(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{S}^{-1}(s_x, s_y, s_z) = \mathbf{S}(1/s_x, 1/s_y, 1/s_z) = \begin{bmatrix} 1/s_x & 0 & 0 & 0 \\ 0 & 1/s_y & 0 & 0 \\ 0 & 0 & 1/s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

HW#14 Animating Rotation with Menu Callback (1)

- Due date: This Friday 6:00pm
- Program execution example for HW#14 will be given at the end of this video lecture.
- Implement rotation matrices $R_x(t)$, $R_y(t)$, and $R_z(t)$, where t is increased continuously in myidle().
- You have to use your pyramid shape object that was used in HW#13.
- Use 'glDrawElements' for drawing.

HW#14 Animating Rotation with Menu Callback (2)

- Initially, the object is rotated about x-axis.
- By using menu callback, either 'x-axis', 'y-axis' or 'z-axis' can be chosen.
- Use depth buffer by the followings:
  - glEnable(GL_DEPTH_TEST); // add to myinit()
  - glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  - glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
- Please read the general rule of homework submission in announcement board.