# Computer Graphics

**Prof. Jibum Kim**

**Department of Computer Science & Engineering**

**Incheon National University**

- 여러 개의 물체에 대한 변환

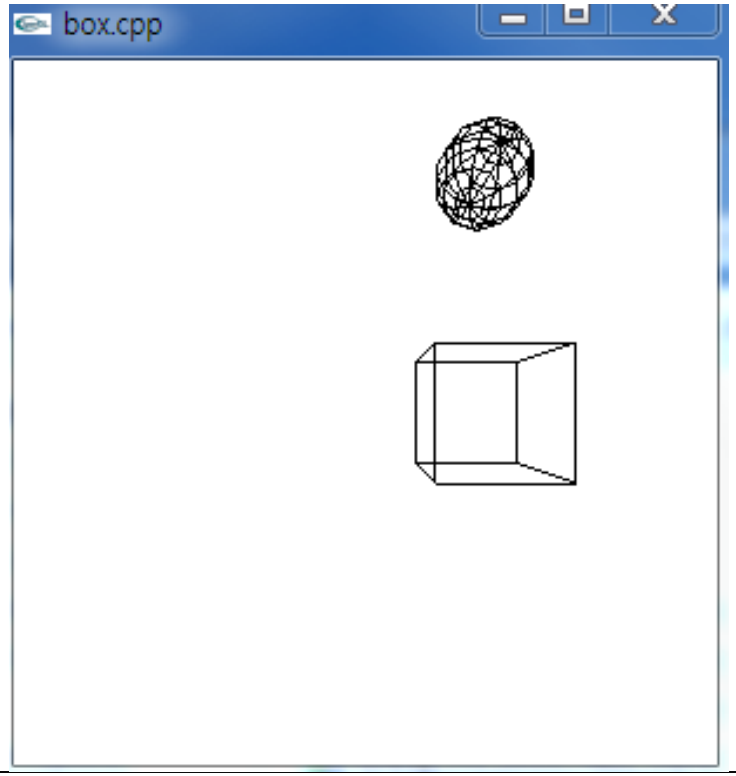
- 지금까지는 하나의 물체에 대하여 변환, 특히 affine 변환이 이루어지는 것에 대하여 알아보았다
- 여러 개의 물체에 대하여 변환이 이루어지는 경우 변환이 각 물체에 어떻게 적용되는지 살펴보자

다음의 예는 3D cube와 sphere 두 물체에 대하여 복합 변환이 일어나는 경우이다. CTM을 이용하여 각 물체에 어떤 변환들이 적용되는지 살펴보자

- glMatrixmode(GL_MODEL_VIEW);  // 4x4 modelview 행렬
- glLoadIdentity();  CTM= I (단위 행렬)
- glTranslatef(0.0, 0.0, -15.0); // T1,  CTM= I ·T1
- glTranslatef(5 , 0 , 0);  // T2,  CTM=T1·T2
- glutWireCube(5.0);  // V1,  T1·(T2 · V1)
- glTranslatef( 0, 10 , 0);  // T3  CTM=T1·T2·T3
- glutWireSphere( 2.0, 10, 8); //V2  T1·T2 · (T3 · V2)

- 1. sphere : T1·(T2 · V1)
- 2. cube : T1·T2 · (T3 · V2)
- 즉, 처음 두 변환은 Cube와 Sphere 공통으로 적용

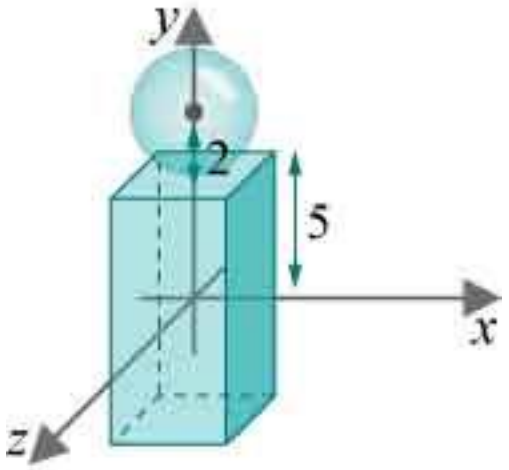
- **https://www.dropbox.com/s/002196scs6bgp17/multiple.txt?dl=0**



- **변환의 분리**

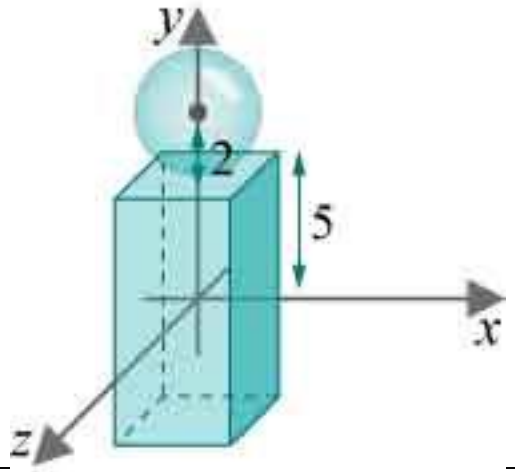
- 3D glut object와 복합 변환을 이용하여 3D sphere와 3D cube 2개의 물체를 아래 그림과 같이 위치시키고자 한다
- 주의: glut object는 최초에 (0, 0, 0)에 중심이 위치한다
- 1. 반지름이 2인 sphere를 +y축으로 7 만큼 translate

  (glutWireSphere(2.0, 8, 8) 사용)
- 2. 한 변의 길이가 5인 cube를 y축 방향으로 2배 scaling
- (glutWireCube(5.0) 사용)

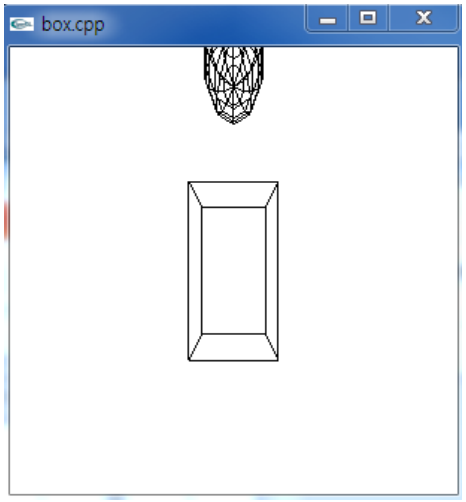


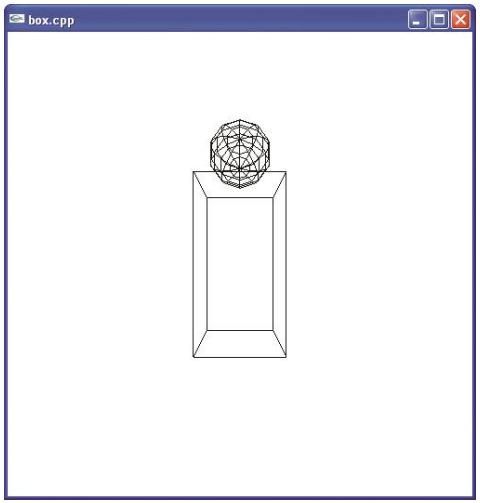
다음 OpenGL 코드의 문제점은 무엇일까?

```
glTranslatef(0.0, 0.0, -15.0);     // CTM = T1
glScalef(1.0, 2.0, 1.0);           // CTM = T1·S
glutWireCube(5.0);                 //  T1·S·v1  (cube)
glTranslatef(0.0, 7.0, 0.0);       //  CTM = T1·S·T2
glutWireSphere(2.0, 10, 8);        //   T1·S·T2·v2 (sphere)
```



- 문제점
- 앞의 코드는 glScalef(1.0, 2.0, 1.0);
- 가 두 object에 모두 적용되기 때문에 변환이 분리가 되지 않아서 아래 왼쪽 그림과 같이 원하지 않는 결과를 얻게 된다



```
#include <GL/glut.h>
void myInit(void)
{
   glClearColor(1.0,1.0,1.0,0.0);      // the background color is white
   glColor3f(1.0f, 0.0f, 0.0f);        // the drawing color is black
            glMatrixMode(GL_PROJECTION);
            glLoadIdentity();
            glFrustum(-5.0, 5.0, -5.0, 5.0, 5.0, 100.0);
}
void Display(void)
{
  glClear(GL_COLOR_BUFFER_BIT);
  glColor3f(0.0, 0.0, 0.0);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  glTranslatef(0.0, 0.0, -15.0);
  glScalef(1.0, 2.0, 1.0);
  glutWireCube(5.0);
  glTranslatef(0.0, 7.0, 0.0);
  glutWireSphere(2.0, 10, 8);
  glFlush();
}

int main(){
  glutInitWindowSize(400,400);
  glutInitWindowPosition(100,100);
  glutCreateWindow("OpenGL Hello World!");
  glClearColor(1.0, 1.0, 1.0, 0.0);
```

- 그렇다면 각 object에 서로 다른 (독립적인) 변환을 주는 방법이 있을까?

- OpenGL에서는 CTM의 stack을 제공하여 각 object에 서로 다른 복합 변환을 수행하는 것을 제공 한다

- **Matrix stack, Modelview matrix stack**

- **CTM stack (Matrix stack) 구조 의 필요성:**
- **1. 앞의 예제와 같이 각 물체별로 변환을 분리시키고 싶다.**
- **즉, 서로 독립적인 변환을 수행하고 싶다**
- **2. CTM은 새로운 변환이 적용되면 계속 변화한다. 이전에 사용한 CTM을 저장 (백업) 하였다가 다시 재 사용할 수는 없을까?**

- **이를 해결하기 위하여 stack 구조를 사용하여 위와 같은 목표를 얻고자 한다**

- **CTM matrix stack**

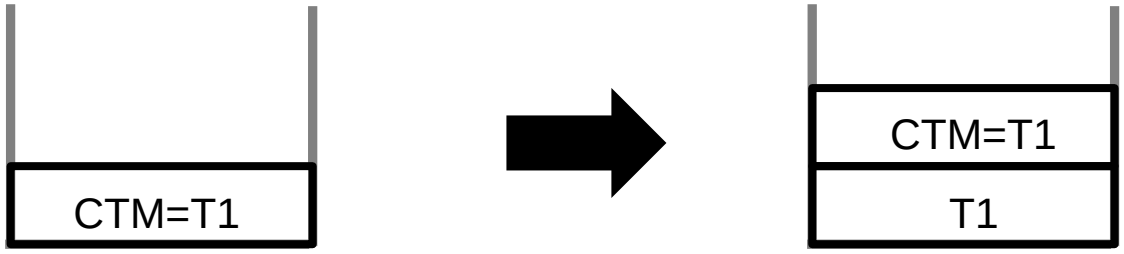
- **이는 CTM을  관리하기 위한 자료 구조 stack 이다.**

  **여기서 실제 CTM은 stack의 가장 위에 있는 행렬이다. 물체에는 이 CTM에 해당하는 변환만 적용된다**

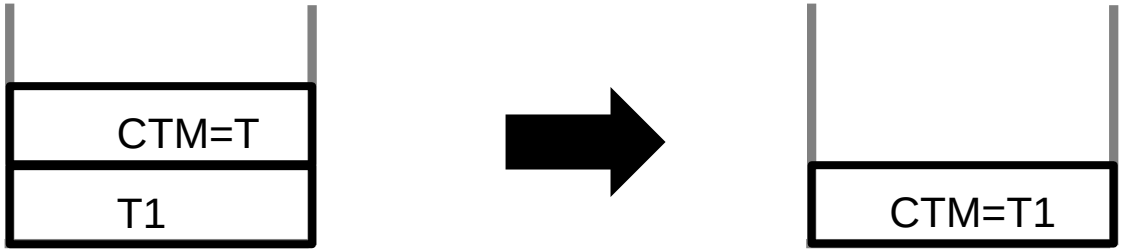
- **matrix stack에는 2가지 명령어를 사용한다 (PUSH, POP)**

**matrix stack에는 2가지 명령어를 사용한다**

- **push: CTM의 복사본을 stack의 가장 위에 놓고 이것이 CTM이 된다 (glPushMatrix();) , Q) 왜 push가 필요할까?**




- **pop: stack의 가장 위에 있는 행렬을 지운다 (glPopMatrix();)**



아래의 코드에 대하여 CTM matrix stack이 어떻게 동작하는지 예를 통해 살펴보자

1. glTranslatef(0.0, 0.0, -15.0); //T1

2. glPushMatrix();

3. glScalef(1.0, 2.0, 1.0); // T2

glutWireCube(5.0); // V1

4. glPopMatrix();

5. glTranslatef(0.0, 7.0, 0.0); // T3

glutWireSphere(2.0, 10, 8); // V2

| 변환 단계 | CTM |
|---|---|
| 1 | T1 |
| 2 | T1 |
| 3 | T1·T2 |
| 물체 v1 | T1·T2 |
| 4 | T1 |
| 5 | T1·T3 |
| 물체 v2 | T1·T3 |

- Matrix stack을 사용시 stack의 가장 위에있는 CTM에 해당하는 변환만 물체에 적용된다
- 이를 통해서 여러 물체에 서로 다른 변환을 독립적으로 적용시킬 수도 있다

CTM stack 구조를 사용하여 각 물체에 적용된 변환을 정리하면 다음과 같다

cube : T1·T2· v1

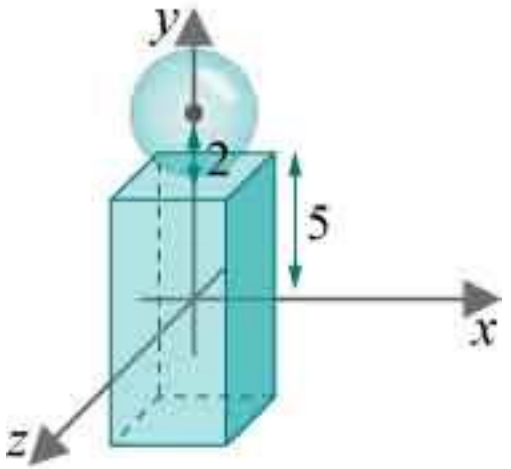
sphere : T1·T3· v2

1. 변환 T1은 두 물체 모두에 적용되었다 (이 경우 frustum안 넣기 위한 translation으로 두 glut object 모두 필요하므로 변환을 분리하지는 않음)

2. 변환 T2는 cube (v1)에만 적용되었다 (scaling은 cube에만 적용)

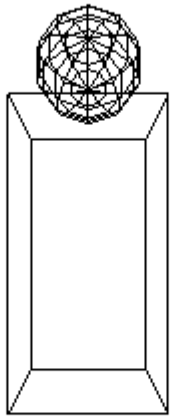
3. 변환 T3는 sphere (v2)에만 적용되었다 (y축 translation은 sphere에만 적용)

```
glTranslatef(0.0, 0.0, -15.0); //T1
glPushMatrix();
glScalef(1.0, 2.0, 1.0);          // T2
glutWireCube(5.0);                // V1
glPopMatrix();
glTranslatef(0.0, 7.0, 0.0);     // T3
glutWireSphere(2.0, 10, 8);  // V2
```



- **matrix stack 구조를 이용하여 변환을 분리시켜 본 예**

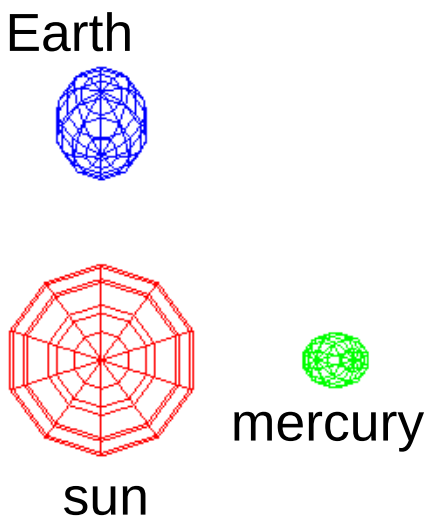
```
void Display(void)
{
  glClear(GL_COLOR_BUFFER_BIT);
  glColor3f(0.0, 0.0, 0.0);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  glTranslatef(0.0, 0.0, -15.0);
  glPushMatrix();
  glScalef(1.0, 2.0, 1.0);
  glutWireCube(5.0);
  glPopMatrix();
  glTranslatef(0.0, 7.0, 0.0);
  glutWireSphere(2.0, 10, 8);
  glFlush();
}
```



- **Q) 각 물체별로 완전히 독립적인 변환을 주기 위하여 물체 별로 glPushMatrix();와 glPopMatrix(); 를 여러 번 사용하면 어떨까?**

- **Matrix stack 구조를 사용한 태양계 만들기**

- **Solar system: Draw this figure using glutWireSphere(4.0, 10, 8);**




- **공통적으로: glTranslatef(0,0,-15) // frustum안으로 이동**
- **1. 태양**
- **2. Mercury: glScalef(0.3,0.3,0.3)후에 glTranslatef(5,0,0)**
- **3. Earth: glScalef(0.5, 0.5, 0.5)후에 glTranslatef(0,10,0)**
- **4. …**