

COMP3201 – Computer Graphics

Module 2: Transformations and Scene Creation

2.6 Scene Creation

2.6.1 Introduction

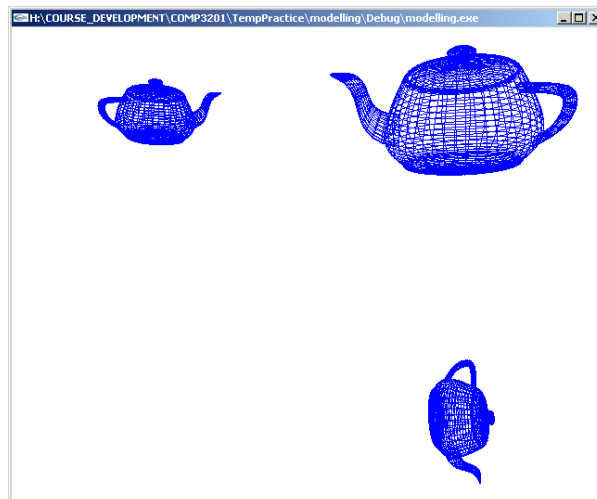
The functions **glPushMatrix** and **glPopMatrix** are used primarily when constructing a scene following a modular design. The following example demonstrates the effectiveness of this approach.

Suppose we wish to construct a robot, which consists of a body, legs, head and arms. The entire robot could be constructed from the one reference point, where each transformation depends on the previous position; but this is not a good approach, as if part of the robot construction had to be altered (for example, the arms had to be positioned lower), then all parts of the robot constructed subsequent to this change would be affected too.

A better approach is to design the robot in modules, and then to move each module to its designated position.

2.6.2 Scene Construction Example

Suppose the following scene is to be created.



We will now construct this scene from two different approaches. The first approach involves the transformations continually building up, while the second approach uses Push/Pop.

Build-up of Transformations

```

GLvoid drawScene(GLvoid)
{
    static GLfloat blue[] = {0.0f, 0.0f, 1.0f};
    static GLdouble teapotSize = 5.0;

    glClear(GL_COLOR_BUFFER_BIT);
    glLoadIdentity();           I

    glColor3fv(blue);

    /* Translate into the viewing volume */
    glTranslatef(0.0f, 0.0f, -75.0f);   T1

    /* Translate to the top left of the window */
    glTranslatef(-20.0f, 20.0f, 0.0f);  T1 T2
    /* Draw the first teapot */
    glutWireTeapot(teapotSize);

    /* Move to the right of the first teapot */
    glTranslatef(40.0f, 0.0f, 0.0f);    T1 T2 T3
    /* Scale up by factor of 2 */
    glScalef(2.0f, 2.0f, 2.0f);         T1 T2 T3 S1
    /* Make the second teapot's spout point to the left, and be upright */
    glRotatef(180.0f, 0.0f, 1.0f, 0.0f); T1 T2 T3 S1 R1
    /* Draw the second teapot */
    glutWireTeapot(teapotSize);

    /* Move to below the second teapot */
    glTranslatef(0.0f, -20.0f, 0.0f);   T1 T2 T3 S1 R1 T4
    /* Scale down by factor 2 */
    glScalef(0.5f, 0.5f, 0.5f);         T1 T2 T3 S1 R1 T4 S2
    /* Make the third teapot's spout point downwards, with its base pointing
    * to the left */
    glRotatef(90.0f, 0.0f, 0.0f, 1.0f);  T1 T2 T3 S1 R1 T4 S2 R2
    glRotatef(180.0f, 0.0f, 1.0f, 0.0f); T1 T2 T3 S1 R1 T4 S2 R2 R3
    /* Draw the third teapot */
    glutWireTeapot(teapotSize);

    glFlush();
}

```

The following transformations were used:

	Build-up of Transformations
T1	T(0.0, 0.0, -75.0);
T2	T(-20.0, 20.0, 0.0);
T3	T(40.0, 0.0, 0.0);
S1	S(2.0, 2.0, 2.0);

R1	R(180.0, y-axis);
T4	T(0.0, -20.0, 0.0);
S2	S(0.5, 0.5, 0.5);
R2	R(90.0, z-axis);
R3	R(180.0, y-axis);

Push/Pop Solution

```
GLvoid drawScene(GLvoid)
```

```
{
    static GLfloat blue[] = {0.0f, 0.0f, 1.0f};
    static GLdouble teapotSize = 5.0;

    glClear(GL_COLOR_BUFFER_BIT);
    glColor3fv(blue);
    glLoadIdentity(); (I)

    /* Translate into the viewing volume */
    glTranslatef(0.0f, 0.0f, -75.0f); (T1)

    glPushMatrix();  $\begin{pmatrix} T1 \\ T1 \end{pmatrix}$ 
        /* Translate to the top left of the window */
        glTranslatef(-20.0f, 20.0f, 0.0f);  $\begin{pmatrix} T1 & T2 \\ T1 \end{pmatrix}$ 
        /* Draw the first teapot */
        glutWireTeapot(teapotSize);
        glPopMatrix(); (T1)

    glPushMatrix();  $\begin{pmatrix} T1 \\ T1 \end{pmatrix}$ 
        /* Move to the right of the first teapot */
        glTranslatef(20.0f, 20.0f, 0.0f);  $\begin{pmatrix} T1 & T3 \\ T1 \end{pmatrix}$ 
        /* Scale up by factor of 2 */
        glScalef(2.0f, 2.0f, 2.0f);  $\begin{pmatrix} T1 & T3 & S1 \\ T1 \end{pmatrix}$ 
        /* Make the second teapot's spout point to the left, and be upright */
        glRotatef(180.0f, 0.0f, 1.0f, 0.0f);  $\begin{pmatrix} T1 & T3 & S1 & R1 \\ T1 \end{pmatrix}$ 
        /* Draw the second teapot */
        glutWireTeapot(teapotSize);
        glPopMatrix(); (T1)

    glPushMatrix();  $\begin{pmatrix} T1 \\ T1 \end{pmatrix}$ 

```

```

/* Move to below the second teapot */
glTranslatef(20.0f, -20.0f, 0.0f);
/* Make the third teapot's spout point downwards, with its base pointing
 * to the left */
glRotatef(-90.0f, 0.0f, 0.0f, 1.0f);
/* Draw the third teapot */
glutWireTeapot(teapotSize);
glPopMatrix();
glFlush();
}

```

$$\begin{pmatrix} T1 & T4 \\ & T1 \end{pmatrix}$$

$$\begin{pmatrix} T1 & T4 & R2 \\ & & T1 \end{pmatrix}$$

(T1)

	Push/Pop Solution
newT1	T(0.0, 0.0, -75.0);
newT2	T(-20.0, 20.0, 0.0);
newT3	T(20.0, 20.0, 0.0); (= T2+T3)
newS1	S(2.0, 2.0, 2.0);
newR1	R(180.0, y-axis);
newT4	T(20.0, -20.0, 0.0); (= T2+T3+2*T4)
newS2	not required
newR2	R(-90.0, z-axis); (= -R2)
newR3	not required

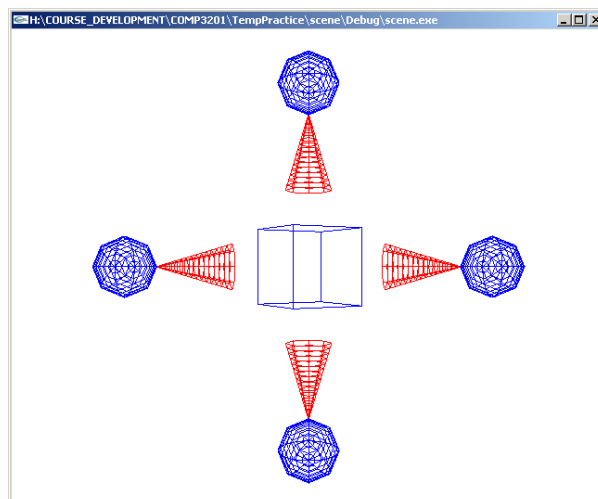
Note how the latter transformations have changed, especially

$$\text{newT4} = T2 + T3 + 2 * T4,$$

as a scale factor must be taken into account.

2.6.3 Scene Construction Exercise

Suppose the following scene is to be created.



The wire cube (size 10.0) is drawn after rotation of 30 degrees around the y-axis. Then the cone (parameters 3.0, 10.0, 8, 15) and sphere (parameters 4.0, 8, 15) are each positioned to the right, left, top and bottom of the cube.

Making use of the Matrix Stack, write down the transformations required to produce this scene. The first part of the scene is as follows:

Action	Matrix Stack
Translate into viewing volume	$T1(0.0, 0.0, -75.0)$
Push Matrix	$\begin{pmatrix} T1 \\ T1 \end{pmatrix}$
Rotate and draw cube	$\begin{pmatrix} T1 R1 \\ T1 \end{pmatrix}$
Pop	(T1)
Push	$\begin{pmatrix} T1 \\ T1 \end{pmatrix}$
Move to right of cube	$\begin{pmatrix} T1 T2 \\ T1 \end{pmatrix}$
Push	$\begin{pmatrix} T1 T2 \\ T1 T2 \\ T1 \end{pmatrix}$
Rotate and draw cone	$\begin{pmatrix} T1 T2 R2 \\ T1 T2 \\ T1 \end{pmatrix}$
Pop	$\begin{pmatrix} T1 T2 \\ T1 \end{pmatrix}$
Move to right of cone and draw sphere	$\begin{pmatrix} T1 T2 T3 \\ T1 \end{pmatrix}$
Pop	(T1)

Now you are back at the cube's position. Continue the process for constructing the rest of the scene. Then try programming the solution – this code will be available on the web site under Sample Programs, for you to check your answer.