Simple Programming with OpenGL

In Week 9 tutorial, you have already seen an OpenGL program. You should make sure that you understand the geometry of the viewing system and the modelling system. You should also know that now we don’t specify colors like (255,0,0), but we specify as (1.0,0,0), for red color. Consult your classmates or your instructor if necessary.

However, this program is very simple. It does not include lighting and material properties. And the construction of the object is done by clumsy construction of polygons.

In this handout, you’ll see some more examples, which involves rendering of animated lighting and objects with material properties.

1. Drawing with OpenGL Auxiliary Library Functions

In previous example, the cube was drawn using clumsy “glBegin() .. glVertex3f .. glVertex3f .. glEnd()” instructions. Indeed we can use more advanced functions in the OpenGL Auxiliary Library to draw the cube.

```c
void DoPainting(CDC *pDC, int WhichCallInCurrentCycle)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); //clear with background color
    glPushMatrix();
    glTranslatef(0.0, 0.0, -5.5); //define a translation
    glRotatef(gObjectRotationX, 1.0, 0.0, 0.0);  //define a rotation transformation
    glRotatef(gObjectRotationY, 0.0, 1.0, 0.0);  // with rotation angle and vector
    glRotatef(gObjectRotationZ, 0.0, 0.0, 1.0);  // of rotation (eg. 1,0,0 = x-axis)
    gObjectRotationX += 1.0;  //prepare for bigger rotation in next picture
    gObjectRotationY += 10.0;
    gObjectRotationZ += 5.0;
    glColor3f(1.0,0,1.0);  //specify a pink color
    auxSolidCube(1.0);  //draw a solid cube
    glPopMatrix();
}
```

In the OpenGL Auxiliary Library, there are also other advanced functions for drawing some primitive shapes:

```c
void auxWireSphere(GLdouble);
void auxSolidSphere(GLdouble);
void auxWireCube(GLdouble);
void auxSolidCube(GLdouble);
void auxWireBox(GLdouble, GLdouble, GLdouble);
void auxSolidBox(GLdouble, GLdouble, GLdouble);
void auxWireTorus(GLdouble, GLdouble);
void auxSolidTorus(GLdouble, GLdouble);
void auxWireCylinder(GLdouble, GLdouble);
void auxSolidCylinder(GLdouble, GLdouble);
void auxWireIcosahedron(GLdouble);
void auxSolidIcosahedron(GLdouble);
void auxWireOctahedron(GLdouble);
void auxSolidOctahedron(GLdouble);
void auxWireTetrahedron(GLdouble);
void auxSolidTetrahedron(GLdouble);
void auxWireDodecahedron(GLdouble);
void auxSolidDodecahedron(GLdouble);
void auxWireCone(GLdouble, GLdouble);
void auxSolidCone(GLdouble, GLdouble);
```

1. Drawing with OpenGL Auxiliary Library Functions
2. Drawing with Lighting Effects

However, we still not applied lighting effects, so the rendered image is not realistic. To apply lighting effects, see the following example:

```cpp
void DoPainting(CDC *pDC,int WhichCallInCurrentCycle)
{
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);  //clear with background color

  glEnable(GL_LIGHTING); //enable the lighting system
  glEnable(GL_LIGHT0);  //enable a light source. Other sources are GL_LIGHT1,...,GL_LIGHT7

  glEnable(GL_COLOR_MATERIAL);  //enable the object material coloring
  glColorMaterial(GL_FRONT,GL_DIFFUSE);  //the following specified material color is for diffuse reflection

  glPushMatrix();

  glTranslatef(0.0, 0.0, -5.5); //define a translation
  glRotatef(gObjectRotationX, 1.0, 0.0, 0.0); //define a rotation transformation
  glRotatef(gObjectRotationY, 0.0, 1.0, 0.0);  // with rotation angle and vector
  glRotatef(gObjectRotationZ, 0.0, 0.0, 1.0);  // of rotation (eg. 1,0,0 = x-axis)

  gObjectRotationX += 1.0;  //prepare for bigger rotation in next picture
  gObjectRotationY += 10.0;
  gObjectRotationZ += 5.0;

  glColor3f(1.0,0,1.0); //specify a pink color
  auxSolidTorus(0.4,0.9); //draw a solid torus

  glPopMatrix();
}
```

3. Defining Specific Lighting Behaviour

However, the default light setting for GL_LIGHT0 may not be satisfactory, so we have to specify our own design:

```cpp
void DoPainting(CDC *pDC,int WhichCallInCurrentCycle)
{
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);  //clear with background color

  glEnable(GL_LIGHTING); //enable the lighting system

  {   glEnable(GL_LIGHT0); //enable a light source. Other sources are GL_LIGHT1,...,GL_LIGHT7

      GLfloat DiffuseLight_color[] = {1.0,1.0,1.0,1.0}; //Specify the color of light
      glLightfv(GL_LIGHT0,GL_DIFFUSE,DiffuseLight_color);

      GLfloat light_position[] = {-1.0,1.0,-4.0,1.0}; //Specify the position of light source
      glLightfv(GL_LIGHT0, GL_POSITION, light_position);
  }

  glEnable(GL_COLOR_MATERIAL);
  glColorMaterial(GL_FRONT,GL_DIFFUSE);

  glPushMatrix();

  glTranslatef(0.0, 0.0, -5.5); //define a translation
  glRotatef(gObjectRotationX, 1.0, 0.0, 0.0); //define a rotation transformation
  glRotatef(gObjectRotationY, 0.0, 1.0, 0.0); // with rotation angle and vector

  ....

  glColor3f(1.0,0,1.0);  //specify a pink color
  auxSolidTorus(0.4,0.9); //draw a solid torus

  glPopMatrix();
}
```
4. Add One More Light Source

We can add variation by changing the existing light source to red color and adding the 2nd blue color light source.

```c
void DoPainting(CDC *pDC, int WhichCallInCurrentCycle)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); //clear with background color

    glEnable(GL_LIGHTING); //enable the lighting system
    {
        glEnable(GL_LIGHT0); //enable a light source. Other sources are GL_LIGHT1,...,GL_LIGHT7
        GLfloat DiffuseLight_color[] = {1.0,0.0,0.0,1.0}; //Specify the color of light
        glLightfv(GL_LIGHT0,GL_DIFFUSE,DiffuseLight_color);

        GLfloat light_position[] = {-1.0,1.0,-4.0,1.0}; //Specify the position of light source
        glLightfv(GL_LIGHT0,GL_POSITION,light_position);
    }
    {
        glEnable(GL_LIGHT1); //enable 2nd light source
        GLfloat DiffuseLight_color[] = {0.0,0.0,1.0,1.0}; //Specify the color of light
        glLightfv(GL_LIGHT1,GL_DIFFUSE,DiffuseLight_color);

        GLfloat light_position[] = {1.0,1.0,-4.0,1.0}; //Specify the position of light source
        glLightfv(GL_LIGHT1,GL_POSITION,light_position);
    }

    glEnable(GL_COLOR_MATERIAL);
    glColorMaterial(GL_FRONT,GL_DIFFUSE);

    glPushMatrix();
    glTranslatef(0.0, 0.0, -5.5); //define a translation
    glRotatef(gObjectRotationX, 1.0, 0.0, 0.0); //define a rotation transformation
    glRotatef(gObjectRotationY, 0.0, 1.0, 0.0); //   with rotation angle and vector
    ...
    glColor3f(1.0,0,1.0); //specify a pink color
    auxSolidTorus(0.4,0.9); //draw a solid torus
    glPopMatrix();
}
```

5. Add one more object

The following example simply adds one more sphere, but which does not move.

```c
void DoPainting(CDC *pDC, int WhichCallInCurrentCycle)
{
    ...
    glPushMatrix();
    glTranslatef(-1.0, 0.0, -5.5); //define a translation
    ...
    glColor3f(1.0,0,1.0);
    auxSolidTorus(0.4,0.9); //draw a solid torus
    glPopMatrix();

    glPushMatrix();
    glTranslatef(1.0, 0.0, -5.5); //define a translation
    glColor3f(1.0,0,1.0);
    auxSolidSphere(0.4); //draw a solid sphere
    glPopMatrix();
}
```
6. Animating Position of Light Source, Sizes of objects

Special effect can be produced by animating the position of light sources, scaling the size of objects, and the coloring of light or objects etc. The following example animate the position of 2nd light source, and keep on changing the scale of the solid sphere, as well as occasionally changing its color.

```c
void DoPainting(CDC *pDC,int WhichCallInCurrentCycle)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);  //clear with background color
    glEnable(GL_LIGHTING);  //enable the lighting system

    glEnable(GL_LIGHT0); //enable a light source.  Other sources are GL_LIGHT1,...,GL_LIGHT7
    GLfloat DiffuseLight_color[] = {1.0,0.0,0.0,1.0}; //Specify the color of light
    glLightfv(GL_LIGHT0,GL_DIFFUSE,DiffuseLight_color);
    GLfloat light_position[] = {-1.0,1.0,-4.0,1.0}; //Specify the position of light source
    glLightfv(GL_LIGHT0,GL_POSITION,light_position);

    glEnable(GL_LIGHT1); //enable 2nd light source
    GLfloat DiffuseLight_color[] = {0.0,0.0,1.0,1.0}; //Specify the color of light
    glLightfv(GL_LIGHT1,GL_DIFFUSE,DiffuseLight_color);
    GLfloat light_position1[] = {1.0,1.0,-4.0,1.0}; //Specify 1st position of light source
    GLfloat light_position2[] = {-1.0,1.0,-4.0,1.0}; //Specify 2nd position of light source
    if ((WhichCallInCurrentCycle%2)==0)
    {   glLightfv(GL_LIGHT1,GL_POSITION,light_position1);
    } else
    {   glLightfv(GL_LIGHT1,GL_POSITION,light_position2);
    }

    glEnable(GL_COLOR_MATERIAL);
    glColorMaterial(GL_FRONT,GL_DIFFUSE);
    glPushMatrix();
    glTranslatef(-1.0, 0.0, -5.5); //define a translation
    glRotatef(gObjectRotationX, 1.0, 0.0, 0.0); //define a rotation transformation
    glRotatef(gObjectRotationY, 0.0, 1.0, 0.0); //   with rotation angle and vector
    glRotatef(gObjectRotationZ, 0.0, 0.0, 1.0); //   of rotation (eg. 1,0,0 = x-axis)
    gObjectRotationX += 1.0;  //prepare for bigger rotation in next picture
    gObjectRotationY += 10.0;
    gObjectRotationZ += 5.0;
    glColor3f(1.0,0,1.0);
    auxSolidTorus(0.4,0.9); //draw a solid torus
    glPopMatrix();

    glPushMatrix();
    glTranslatef(1.0, 0.0, -5.5); //define a translation
    if ((WhichCallInCurrentCycle%2)==0) //occasionally scale down the sphere
    {   glScalef(0.9,0.9,0.9);
    }
    if ((WhichCallInCurrentCycle==7 || WhichCallInCurrentCycle>10) //the sphere is changed to red only for the 7th picture
    {   glColor3f(1.0,0,0);
    } else
    {   glColor3f(1.0,0,1.0); //specify a pink color
    }
    auxSolidSphere(0.4); //draw a solid sphere
    glPopMatrix();
}
```
7. Different reflections

So far we have been dealing with only diffuse reflection. We can add different kinds of reflections. Let's start from a simple example, which we have a static sphere.

A. Only diffuse reflection

```c
void DoPainting(CDC *pDC, int WhichCallInCurrentCycle)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); //clear with background color
    glEnable(GL_LIGHTING); //enable the lighting system
    {
        glEnable(GL_LIGHT0); //enable a light source. Other sources are GL_LIGHT1,...,GL_LIGHT7
        GLfloat DiffuseLight_color[] = {0.8,0.8,0.8,1.0}; //the diffuse color component of the light
        glLightfv(GL_LIGHT0, GL_DIFFUSE, DiffuseLight_color); //set the light's diffuse color component
        GLfloat light_position[] = {-3.5,0.5,0.0,1.0}; //the position of light source
        glLightfv(GL_LIGHT0, GL_POSITION, light_position); //set the position of light source
    }
    glEnable(GL_COLOR_MATERIAL); //enable the object material coloring
    glPushMatrix(); //define an object to be displayed.
    glTranslatef(0.0, 0.0, -5.5); //define a translation
    glColorMaterial(GL_FRONT, GL_DIFFUSE); //the following glColor3f sets diffuse reflection property
    glColor3f(1.0,0,1.0); //specify a pink color: it can reflect red and blue light
    auxSolidSphere(1.0); //draw a solid cube
    glPopMatrix();
}
```

B. Diffuse + ambient reflection

```c
void DoPainting(CDC *pDC, int WhichCallInCurrentCycle)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); //clear with background color
    glEnable(GL_LIGHTING); //enable the lighting system
    {
        glEnable(GL_LIGHT0); //enable a light source. Other sources are GL_LIGHT1,...,GL_LIGHT7
        GLfloat DiffuseLight_color[] = {0.8,0.8,0.8,1.0}; //the diffuse color component of the light
        GLfloat AmbientLight_color[] = {0.2,0.2,0.2,1.0}; //the ambient color component of the light
        glLightfv(GL_LIGHT0, GL_DIFFUSE, DiffuseLight_color); //set the light's diffuse color component
        glLightfv(GL_LIGHT0, GL_AMBIENT, AmbientLight_color); //set the light's ambient color component
        GLfloat light_position[] = {-3.5,0.5,0.0,1.0}; //the position of light source
        glLightfv(GL_LIGHT0, GL_POSITION, light_position); //set the position of light source
    }
    glEnable(GL_COLOR_MATERIAL); //enable the object material coloring
    glPushMatrix(); //define an object to be displayed.
    glTranslatef(0.0, 0.0, -5.5); //define a translation
    glColorMaterial(GL_FRONT, GL_DIFFUSE); //the following glColor3f sets diffuse reflection property
    glColor3f(1.0,0,1.0); //specify a pink color: it can reflect red and blue light
    auxSolidSphere(1.0); //draw a solid cube
    glPopMatrix();
}
```
C. Diffuse + ambient + specular reflection

```c
void DoPainting(CDC *pDC, int WhichCallInCurrentCycle)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); //clear with background color
    glEnable(GL_LIGHTING); //enable the lighting system
    {
        glEnable(GL_LIGHT0); //enable a light source. Other sources are GL_LIGHT1....GL_LIGHT7
        GLfloat DiffuseLight_color[] = {0.8,0.8,0.8,1.0};   //the diffuse color component of the light
        GLfloat AmbientLight_color[] = {0.2,0.2,0.2,1.0};  //the ambient color component of the light
        GLfloat SpecularLight_color[] = {0.5,0.5,0.5,1.0};  //the specular color component of the light

        glLightfv(GL_LIGHT0, GL_DIFFUSE, DiffuseLight_color); //set the light's diffuse color component
        glLightfv(GL_LIGHT0, GL_AMBIENT, AmbientLight_color); //set the light's ambient color component
        glLightfv(GL_LIGHT0, GL_SPECULAR, SpecularLight_color); //set the light's specular color component

        GLfloat light_position[] = {-3.5,0.5,0.0,1.0};    //the position of light source
        glLightfv(GL_LIGHT0, GL_POSITION, light_position);  //set the position of light source
    }
    glEnable(GL_COLOR_MATERIAL); //enable the object material coloring
    glPushMatrix(); //define an object to be displayed.
    glTranslatef(0.0, 0.0, -5.5); //define a translation
    glColorMaterial(GL_FRONT, GL_DIFFUSE); //the following glColor3f sets diffuse reflection property
    glColor3f(1.0,0,1.0); //specify a pink color: it can reflect red and blue light
    glMaterialf(GL_FRONT, GL_SHININESS, 4.0); //define how shiny the object is (valid value: 0.0 - 128.0)
    glMaterialf(GL_FRONT, GL_SPECULAR, 1.0); //the following glColor3f sets specular reflection property
    glColor3f(1.0,1.0,1.0); //specify a white color - it can reflect all r,g,b light
    auxSolidSphere(1.0); //draw a solid cube
    glPopMatrix();
}
```