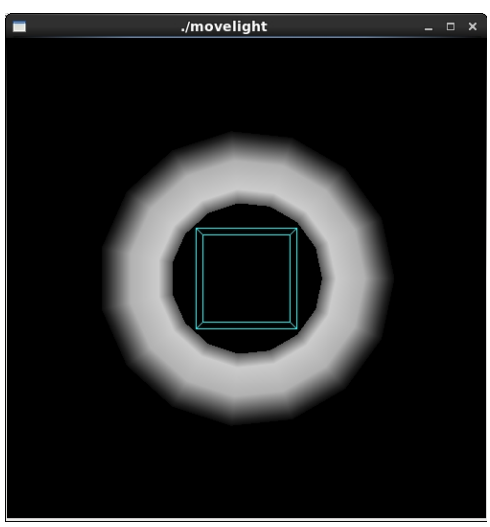
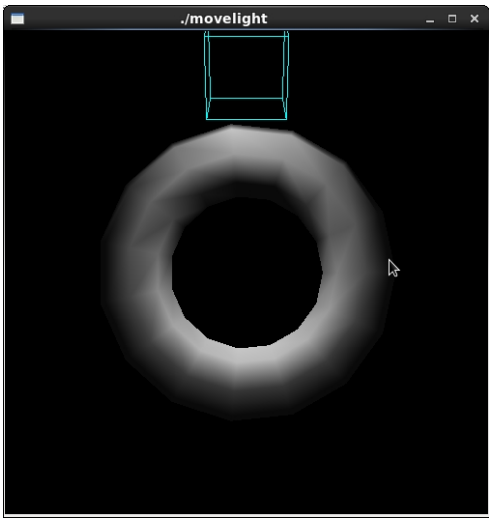
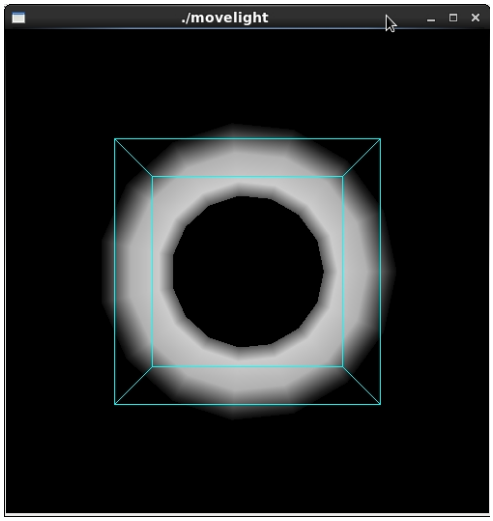
Yazhuo Liu
Lab 16

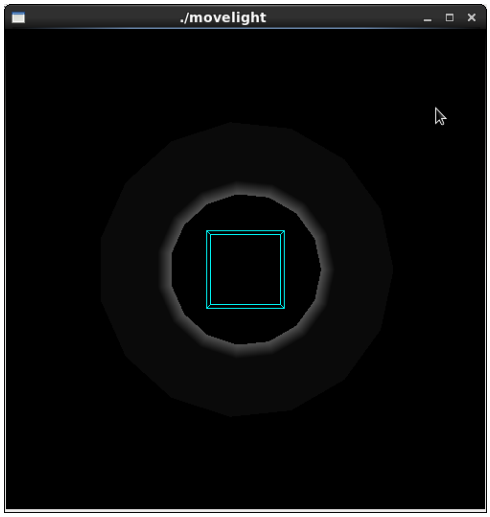
Copy the program movelight.cpp from the lecture notes. Compile and execute it.





Make the light translate past the object instead of rotating around it. Hint: Use glTranslated() rather than the first glRotated() in display(), and choose an appropriate value to use instead of spin.

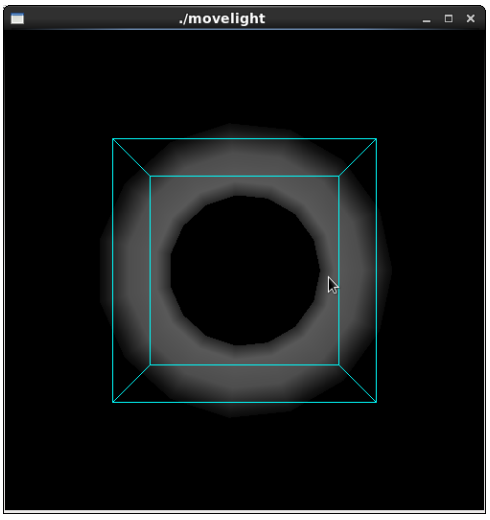


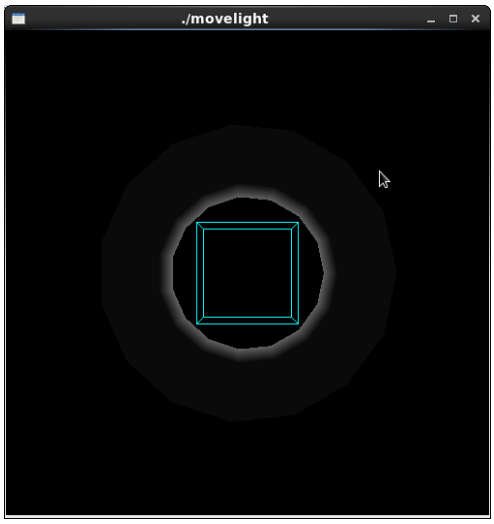



```
void display(void)
{
  GLfloat position[] = { 0.0, 0.0, 1.5, 1.0 };

  glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  glPushMatrix ();
  glTranslatef (0.0, 0.0, -5.0);

  glPushMatrix ();
  //glRotated ((GLdouble) spin, 1.0, 0.0, 0.0);
  glTranslated (0.0, 0.0, (GLdouble) dis);
  glLightfv (GL_LIGHT0, GL_POSITION, position);
```

```
  //glTranslated (0.0, 0.0, 1.5);
  glTranslated (0.0, 0.0, 3.5);
  glDisable (GL_LIGHTING);
  glColor3f (0.0, 1.0, 1.0);
  glutWireCube (0.5);
  glEnable (GL_LIGHTING);
  glPopMatrix ();

  glutSolidTorus (0.275, 0.85, 8, 15);
  glPopMatrix ();
  glFlush ();
}
...
void mouse(int button, int state, int x, int y)
{
  switch (button) {
    case GLUT_LEFT_BUTTON:
      if (state == GLUT_DOWN) {
        //spin = (spin + 30) % 360;
        dis -= 1;
        glutPostRedisplay();
      }
      break;
    default:
      break;
  }
}
```

Change the attenuation so that the light decreases in intensity as it's moved away from the object. Hint: Add calls to glLight*() to set the desired attenuation parameters.

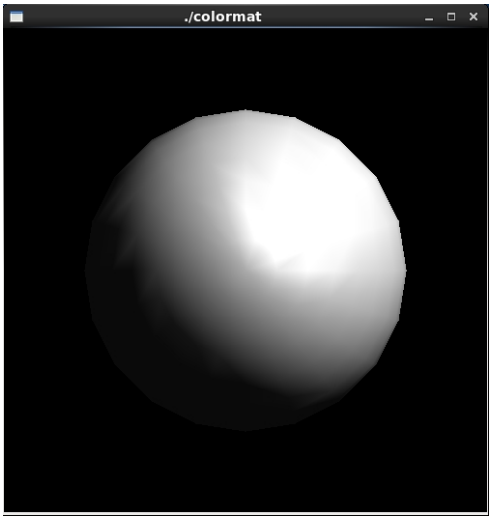


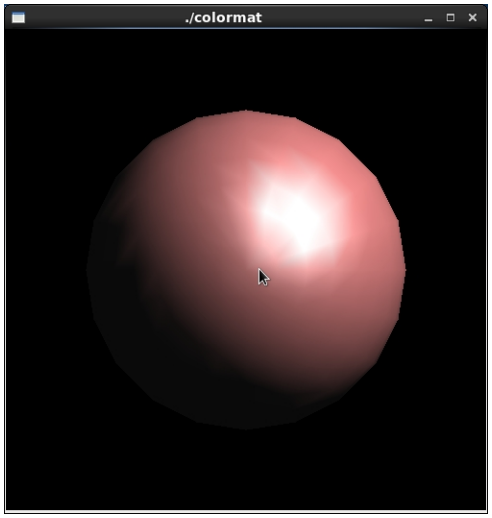



```
void display(void)
{
  GLfloat position[] = { 0.0, 0.0, 1.5, 1.0 };
```

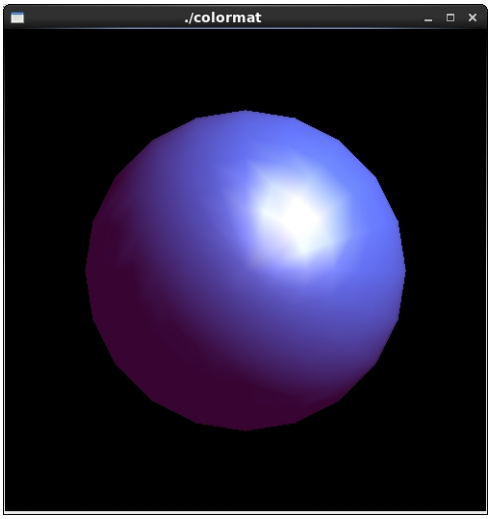
```
  glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  glPushMatrix ();
  glTranslatef (0.0, 0.0, -5.0);

  glPushMatrix ();
  //glRotated ((GLdouble) spin, 1.0, 0.0, 0.0);
  glTranslated (0.0, 0.0, (GLdouble) dis);
  glLightfv (GL_LIGHT0, GL_POSITION, position);
  glLightf(GL_LIGHT0, GL_LINEAR_ATTENUATION, 1.0);

  //glTranslated (0.0, 0.0, 1.5);
  glTranslated (0.0, 0.0, 3.5);
  glDisable (GL_LIGHTING);
  glColor3f (0.0, 1.0, 1.0);
  glutWireCube (0.5);
  glEnable (GL_LIGHTING);
  glPopMatrix ();

  glutSolidTorus (0.275, 0.85, 8, 15);
  glPopMatrix ();
  glFlush ();
}
```

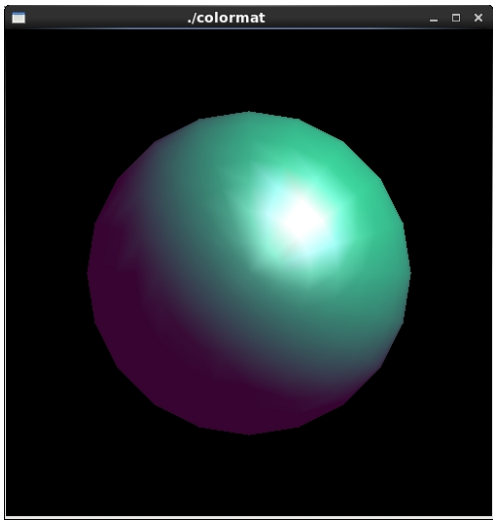
Copy the program colormat.cpp from /pool/u/class/cs420/lighting. Compile and execute it. Click the mouse to see the change in colors.





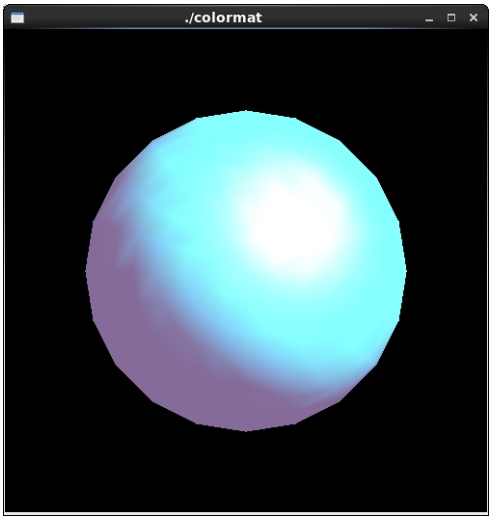
Change the global ambient light in the scene. Hint: Alter the value of the GL_LIGHT_MODEL_AMBIENT parameter.

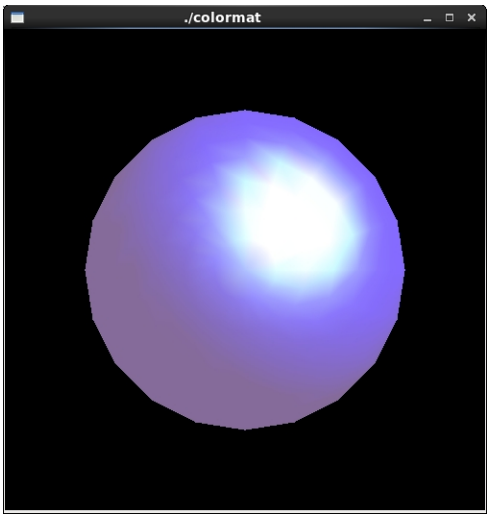




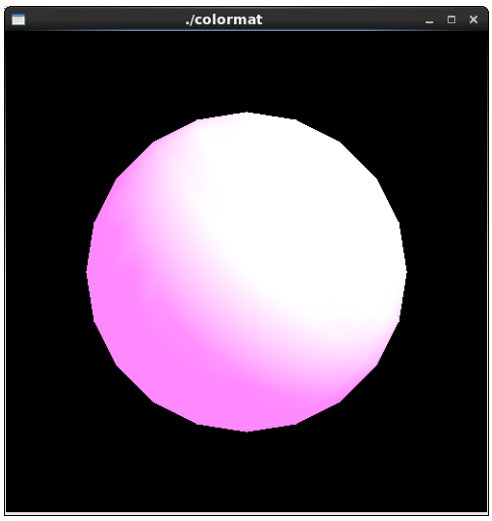

```
void init(void)
{
  GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
  GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };
  GLfloat lmodel_ambient[] = { 1.1, 0.1, 1.0, 1.0 };

  glClearColor (0.0, 0.0, 0.0, 0.0);
  glShadeModel (GL_SMOOTH);
  glEnable(GL_DEPTH_TEST);
  glMaterialfv(GL_FRONT, GL_DIFFUSE, diffuseMaterial);
  glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
  glLightModelfv(GL_LIGHT_MODEL_AMBIENT, lmodel_ambient);
  glMaterialf(GL_FRONT, GL_SHININESS, 25.0);
  glLightfv(GL_LIGHT0, GL_POSITION, light_position);
  glEnable(GL_LIGHTING);
  glEnable(GL_LIGHT0);

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

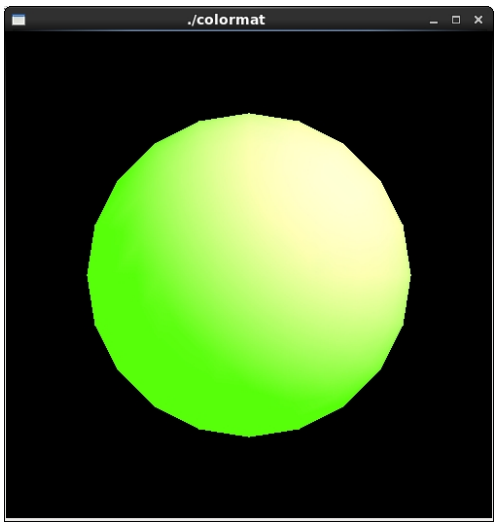
Change the diffuse, ambient, and specular reflection parameters, the shininess exponent, and the emission color. Hint: Use the glMaterial*() command, but avoid making excessive calls.





Remove all the glMaterialfv() calls, and use the more efficient glColorMaterial() calls to achieve the same lighting.






```
void init(void)
{
  GLfloat mat_specular[] = { 1.0, 2.0, 2.0, 1.0 };
  GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };
  GLfloat lmodel_ambient[] = { 0.1, 0.1, 1.0, 1.0 };
  GLfloat mat_emission[] = {0.3, 0.2, 0.2, 0.0};

  glClearColor (0.0, 0.0, 0.0, 0.0);
  glShadeModel (GL_SMOOTH);
  glEnable(GL_DEPTH_TEST);

  glMaterialf(GL_FRONT, GL_SHININESS, 5.0);
  glLightfv(GL_LIGHT0, GL_POSITION, light_position);
  glEnable(GL_LIGHTING);
  glEnable(GL_LIGHT0);

  glColorMaterial(GL_FRONT, GL_AMBIENT_AND_DIFFUSE);
  glColor3f (0.5, 1.0, 2.5);
  glColorMaterial(GL_FRONT, GL_SPECULAR);
  glColor3f (0.0, 2.5, 2.5);
  glColorMaterial(GL_FRONT, GL_EMISSION);
  glColor3f (2.5, 0.5, 1.5);
  glEnable(GL_COLOR_MATERIAL);
}
```

Report:

The last part of the lab is a little bit difficult since it is not in the lecture notes. I went online and found solutions for glColorMaterial(). I think I successfully completed this lab.