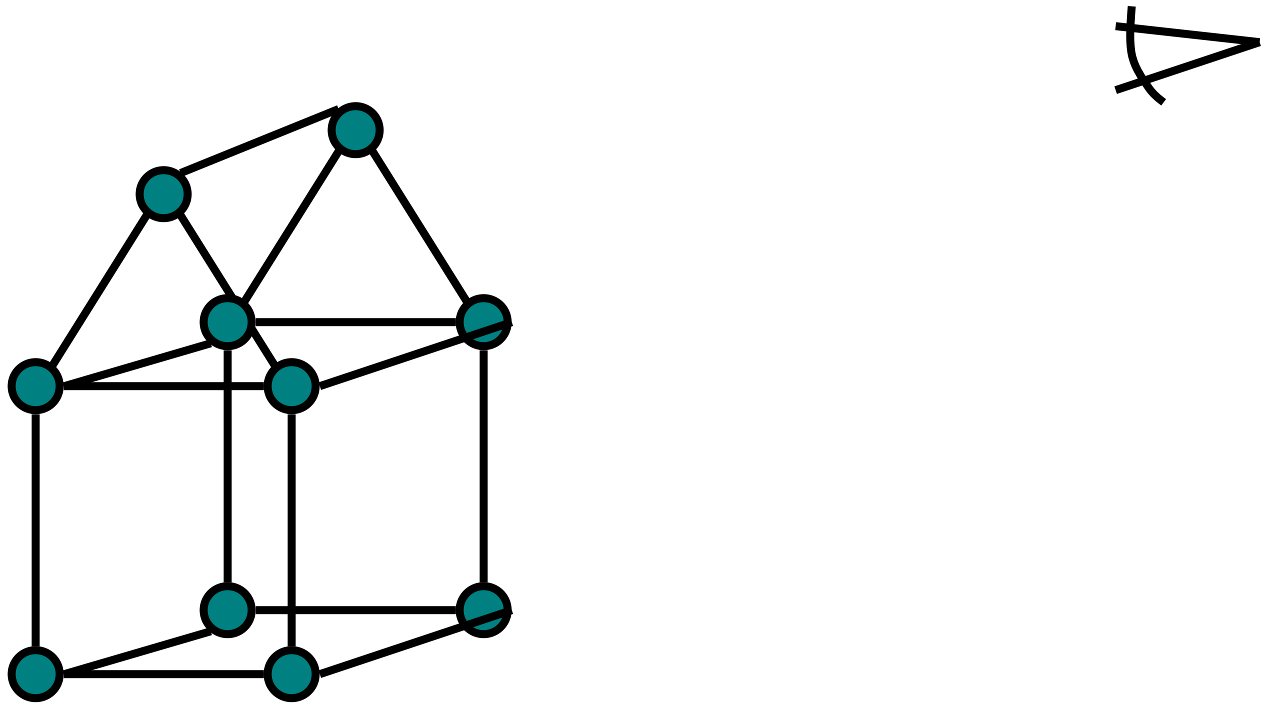
OpenGL简介

计算机图形学——原理、方法及应用（第4版），潘云鹤、童若锋、耿卫东、唐敏、童欣，高等教育出版社，2022。

OpenGL

- 用于显示的多平台图形库，也有少量造型功能，150多个函数
- 速度快
- 公开标准
- 前身为 SGI’s IRIS GL

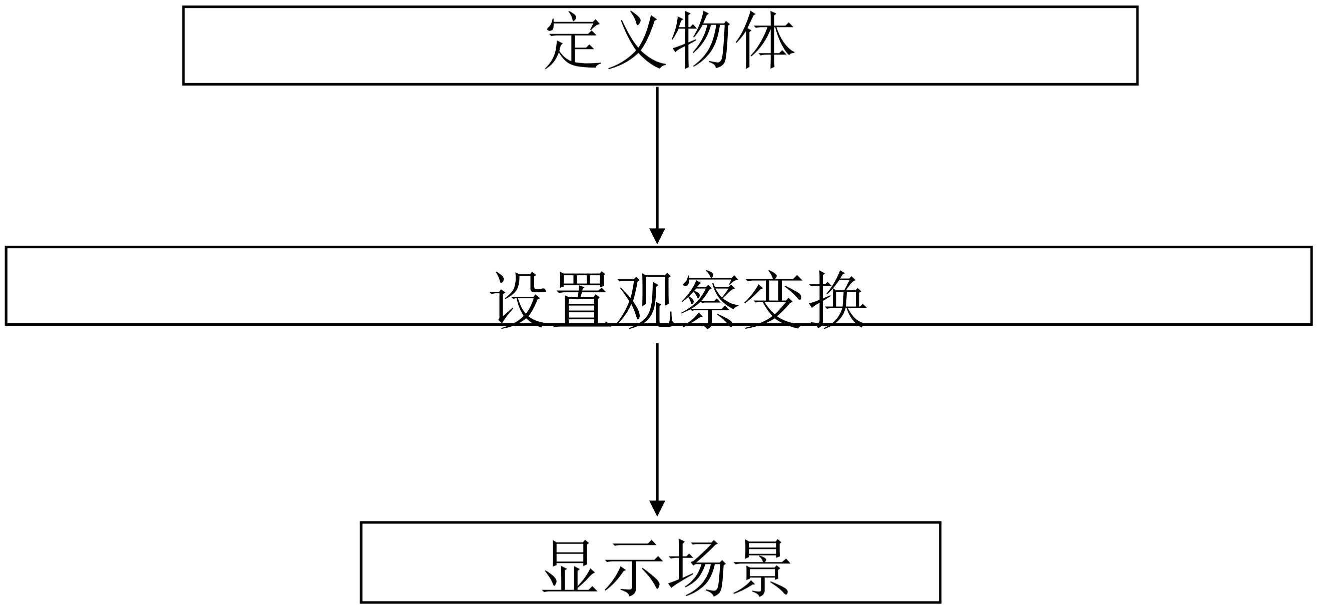
Given 3D data, generate 2D view



OpenGL功能

- 可以定义物体的形状、材质和光源属性
- 在三维空间中放置图形及相机
- 显示真实感图形
- 除了基本的GL库外，还提供附加的GLU/GLUT/GLUI等用于窗口等功能

OpenGL中的三位基本过程



代码示例

```
int main(int argc, char **argv)
{
glutInit(&argc, argv);
 glutInitDisplayMode ( GLUT_SINGLE | GLUT_RGB
| GLUT_DEPTH);

 glutInitWindowPosition(100,100);
 glutInitWindowSize(300,300);
 glutCreateWindow ("square");

 glClearColor(0.0, 0.0, 0.0, 0.0);
 glMatrixMode(GL_PROJECTION);
 glLoadIdentity();
 glOrtho(0.0, 10.0, 0.0, 10.0, -1.0, 1.0);

 glutDisplayFunc(display);
 glutMainLoop();
 return 0;
}
```

```
void display(void)
{
 glClear( GL_COLOR_BUFFER_BIT);
 glColor3f(0.0, 1.0, 0.0);
 glBegin(GL_POLYGON);
  glVertex3f(2.0, 4.0, 0.0);
  glVertex3f(8.0, 4.0, 0.0);
  glVertex3f(8.0, 6.0, 0.0);
  glVertex3f(2.0, 6.0, 0.0);
 glEnd();
 glFlush();
}
```

OpenGL 基本元素

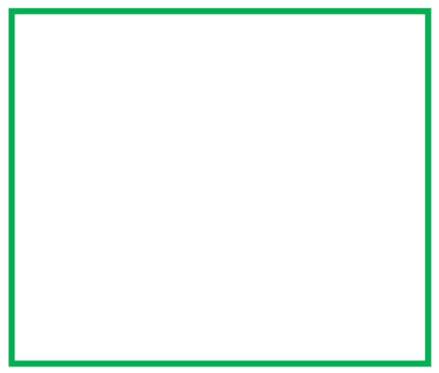
- GL_POINTS
- GL_LINES
- GL_LINE_STRIP
- GL_LINE_LOOP

- GL_TRIANGLES
- GL_QUADS
- GL_POLYGON
- GL_TRIANGLE_STRIP
- GL_TRIANLE_FAN
- GL_QUAD_STRIP

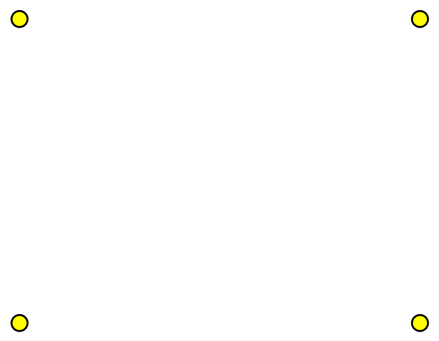
GL_POLYGON and GL_TRIANGLE are the only ones in common usage; valid OpenGL polygons are closed, convex, co-planar and non-intersecting, which is always true for triangles!

实例

```
glBegin(GL_POLYGON);
    glVertex2i(0,0);
    glVertex2i(0,1);
    glVertex2i(1,1);
    glVertex2i(1,0);
glEnd() ;
```

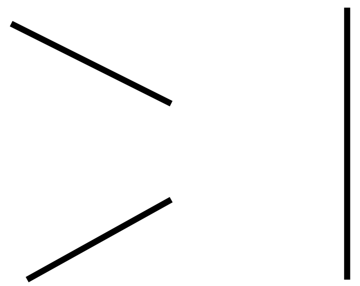


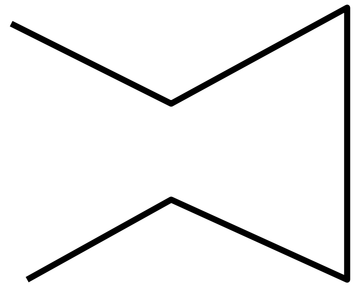
```
glBegin(GL_POINTS);
    glVertex2i(0,0);
    glVertex2i(0,1);
    glVertex2i(1,1);
    glVertex2i(1,0);
glEnd() ;
```

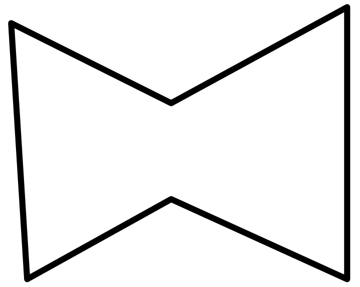


实例

```
GLfloat list[6][2] ;
glBegin(GL_LINES)
    for (int i = 0 ; i < 6 ;i++)
        glVertex2v(list[i]);
glEnd() ;
glBegin(GL_LINE_STRIP)
    for (int i = 0 ; i < 6 ;i++)
        glVertex2v(list[i]);
glEnd() ;
glBegin(GL_LINE_LOOP)
    for (int i = 0 ; i < 6 ;i++)
        glVertex2v(list[i]);
glEnd() ;
```

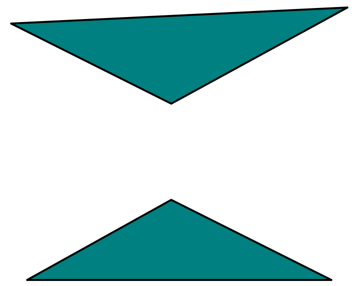


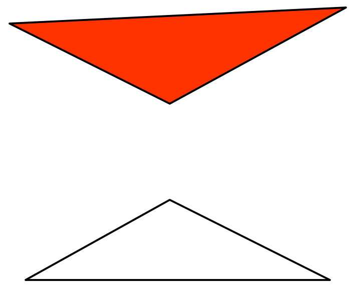




实例

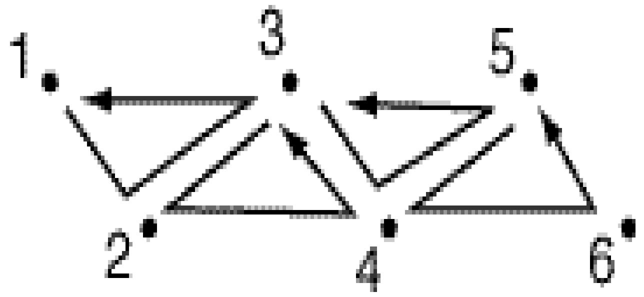
```
GLfloat list[6][2] ;

glColor3f(0.0, 1.0, 0.0);
glBegin(GL_TRIANGLES)
    for (int i = 0 ; i < 6 ;i++)
        glVertex2v(list[i]);
glEnd() ;

glBegin(GL_TRIANGLES)
    glColor3f(1.0, 0.0, 0.0);
    for ( i = 0 ; i < 3 ;i++)
        glVertex2v(list[i]);
    glColor3f(1.0, 1.0, 1.0);
    for ( i = 3 ; i < 6 ;i++)
        glVertex2v(list[i]);
glEnd() ;
```



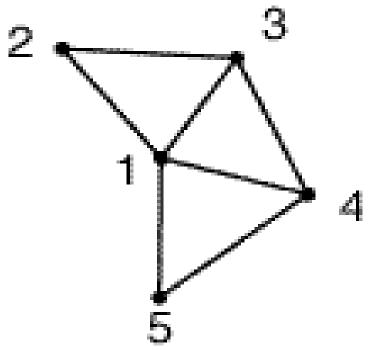


实例

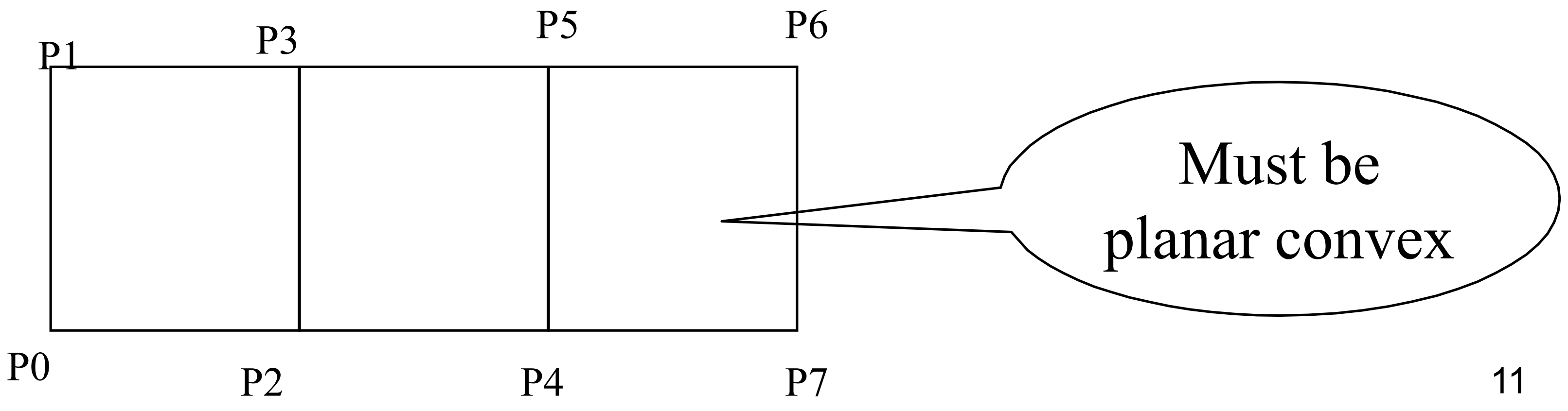
GL_TRIANGLE_STRIP



GL_TRIANGLE_FAN



GL_QUAD_STRIP



指令体系

- Gl库中的所有函数的命名以gl开头

  Ex.: glVertex3f(0.0, 1.0, 1.0);
- 常量的都大写

  Ex.: GL_COLOR_BUFFER_BIT
- 数据类型以GL开头

  Ex.: GLfloat onevertex[3];
- 大多数函数的最后两个字符代表参数的数据类型和个数

  Ex.: glVertex3f(…) => 3 GLfloat arguments

glVertex

- 所有体元由顶点定义

```
glVertex2f( x, y );
glVertex3f( x, y, z );
glVertex4f( x, y, z, w );
glVertex3fv( a );       // with a[0], a[1], a[2]
```

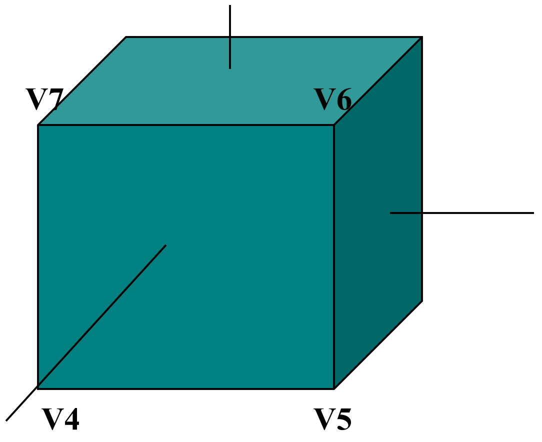
由顶点构建物体

- 设定图元模式, 并在一个 glBegin / glEnd 组中定义顶点

```
glBegin( GL_POLYGON );
    glVertex3f( 1.0, 2.0, 0.0 );
    glVertex3f( 0.0, 0.0, 0.0 );
    glVertex3f( 3.0, 0.0, 0.0 );
    glVertex3f( 3.0, 2.0, 0.0 );
glEnd();
```

实例

```
void drawOneCubeface(size)
{
  static Glfloat v[8][3];
   v[0][0] = v[3][0] = v[4][0] = v[7][0] = -size/2.0;
   v[1][0] = v[2][0] = v[5][0] = v[6][0] = size/2.0;
   v[0][1] = v[1][1] = v[4][1] = v[5][1] = -size/2.0;
   v[2][1] = v[3][1] = v[6][1] = v[7][1] = size/2.0;
   v[0][2] = v[1][2] = v[2][2] = v[3][2] = -size/2.0;
   v[4][2] = v[5][2] = v[6][2] = v[7][2] = size/2.0;
  glBegin( GL_POLYGON);
    glVertex3fv(v[0]);
    glVertex2fv(v[1]);
    glVertex2fv(v[2]);
    glVertex2fv(v[3]);
  glEnd();
 }
```



颜色

- OpenGL 由[0.0, 1.0]间的RGB 组定义
- 如：将背景定位黑色:

  glClearColor(0.0, 0.0, 0.0); // black color

  glClear(GL_COLOR_BUFFER_BIT);
- 将物体定为白色

  glColor3f(1.0, 1.0, 1.0);   // white
color

例子

```
glBegin( GL_POLYGON );
   glColor3f( 1.0, 1.0, 0.0 );
   glVertex3f( 0.0, 0.0, 0.0 );
   glColor3f( 0.0, 1.0, 1.0 );
   glVertex3f( 5.0, 0.0, 0.0 );
   glColor3f( 1.0, 0.0, 1.0 );
   glVertex3f( 0.0, 5.0, 0.0 );
glEnd();
```

多边形显示模式

- glPolygonMode(GLenum face, GLenum mode);
  - Faces: GL_FRONT, GL_BACK, GL_FRONT_AND_BACK
  - Modes: GL_FILL, GL_LINE, GL_POINT
  - By default, both the front and back face are drawn filled
- glFrontFace(GLenum mode);
  - Mode is either GL_CCW (default) or GL_CW
- glCullFace(Glenum mode);
  - Mode is either GL_FRONT, GL_BACK, GL_FRONT_AND_BACK;
- You must enable and disable culling with glEnable(GL_CULL_FACE) or glDisable(GL_CULL_FACE);

OpenGL 程序编译

- 调用GLUT需要：

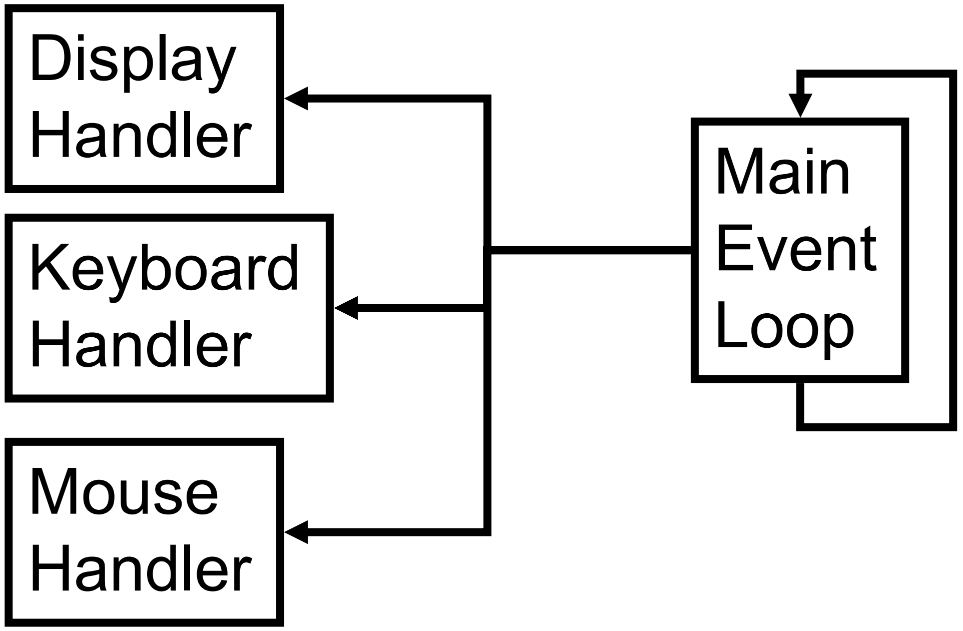
  #include <GL/glut.h>

  确保glut.lib (or glut32.lib) 在编译库目录中
- 参考 *OpenGL Game Programming* 或线上教程.

其他物体

- GLU 包含 柱、锥和 NURBS调用
- GLUT 球和长方体

事件驱动的编程


Display Handler
Keyboard Handler
Mouse Handler
Main Event Loop

GLUT 实例

Displaying a square

```
int main (int argc,  char *argv[])
{
  glutInit(&argc, argv);
  glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE);
  int windowHandle
        = glutCreateWindow("Simple GLUT App");

  glutDisplayFunc(redraw);

  glutMainLoop();

  return 0;
}
```

Display Callback

Called when window is redrawn

```
void redraw()
{
  glClear(GL_COLOR_BUFFER_BIT);
  glBegin(GL_QUADS);
  glColor3f(1, 0, 0);
    glVertex3f(-0.5, 0.5, 0.5);
    glVertex3f(0.5, 0.5, 0.5);
    glVertex3f(0.5, -0.5, 0.5);
    glVertex3f(-0.5, -0.5, 0.5);
  glEnd(); // GL_QUADS
  glutSwapBuffers();
}
```

More GLUT

Additional GLUT functions

```
glutPositionWindow(int x,int y);
glutReshapeWindow(int w, int h);
```

Additional callback functions

```
glutReshapeFunction(reshape);
glutMouseFunction(mousebutton);
glutMotionFunction(motion);
glutKeyboardFunction(keyboardCB);
glutSpecialFunction(special);
glutIdleFunction(animate);
```

Reshape Callback

Called when the window is resized

```
void reshape(int w, int h)
{
  glViewport(0.0,0.0,w,h);

  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  glOrtho(0.0,w,0.0,h, -1.0, 1.0);

  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
}
```

Mouse Callbacks

Called when the mouse button is pressed

```
void mousebutton(int button, int state, int x, int y)
{
  if (button==GLUT_LEFT_BUTTON && state==GLUT_DOWN)
  {
    rx = x; ry = winHeight - y;
  }
}
```

Called when the mouse is moved with button down

```
void motion(int x, int y)
{
  rx = x; ry = winHeight - y;
}
```

Keyboard Callbacks

Called when a button is pressed

```
void keyboardCB(unsigned char key, int x, int y)
{
  switch(key)
  { case 'a': cout<<"a Pressed"<<endl; break; }
}
```

Called when a special button is pressed

```
void special(int key, int x, int y)
{
  switch(key)
  { case GLUT_F1_KEY:
      cout<<“F1 Pressed"<<endl; break; }
}
```

OpenGL – GLUT 实例

```
#include <gl/glut.h>
#include <stdlib.h>

static GLfloat spin = 0.0;

void init( void )
{
	glClearColor( 0.0, 0.0, 0.0, 0.0 );
	glShadeModel( GL_FLAT );
}

void display( void )
{
	glClear( GL_COLOR_BUFFER_BIT );
	glPushMatrix();
	glRotatef( spin, 0.0, 0.0, 1.0 );
	glColor3f( 1.0, 1.0, 1.0 );
	glRectf( -25.0, -25.0, 25.0, 25.0 );
	glPopMatrix();
	glutSwapBuffers();
}
```

OpenGL – GLUT 实例

```
void spinDisplay( void )
{
    spin += 2.0;
    if( spin > 360.0 )
        spin -= 360.0;
    glutPostRedisplay();
}

void reshape( int w, int h )
{
    glViewport( 0, 0, (GLsizei) w, (GLsizei) h );
    glMatrixMode( GL_PROJECTION );
    glLoadIdentity();
    glOrtho( -50.0, 50.0, -50.0, 50.0, -1.0, 1.0 );
    glMatrixMode( GL_MODELVIEW );
    glLoadIdentity();
}
```

OpenGL – GLUT 实例

```
void mouse( int button, int state, int x, int y )
{
    switch( button )
    {
        case GLUT_LEFT_BUTTON:
            if( state == GLUT_DOWN )
                glutIdleFunc( spinDisplay );
            break;
        case GLUT_RIGHT_BUTTON:
            if( state == GLUT_DOWN )
                glutIdleFunc( NULL );
            break;
        default:    break;
    }
}
```

OpenGL – GLUT Example

```
int main( int argc, char ** argv )
{
    glutInit( &argc, argv );
    glutInitDisplayMode( GLUT_DOUBLE | GLUT_RGB );
    glutInitWindowSize( 250, 250 );
    glutInitWindowPosition( 100, 100 );
    glutCreateWindow( argv[ 0 ] );

    init();
    glutDisplayFunc( display );
    glutReshapeFunc( reshape );
    glutMouseFunc( mouse );
    glutMainLoop();
    return 0;
}
```

网络资源

WWW.OpenGL.org

nehe.gamedev.net

http://www.xmission.com/~nate/glut.html

Thank You