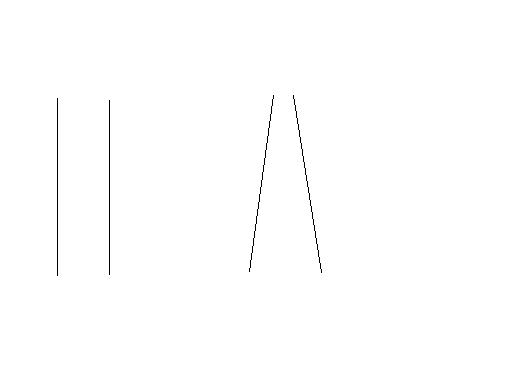
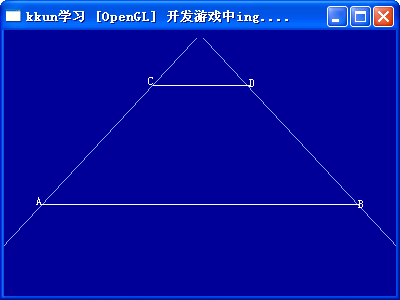
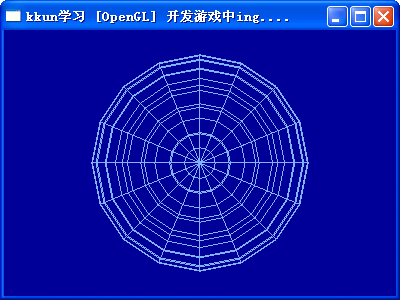
函数原型  
gluLookAt(GLdoble eyex,GLdouble eyey,GLdouble eyez,GLdouble centerx,GLdouble centery,GLdouble centerz,GLdouble upx,GLdouble upy,GLdouble upz);  
  
gluPerspective(GLdouble fovy,GLdouble aspect,GLdouble zNear,GLdouble zFar)  
  
一个一个来,首先得设置gluPerspective,来看看它的参数都表示什么意思  
fovy,这个最难理解,我的理解是,眼睛睁开的角度,即,视角的大小,如果设置为0,相当你闭上眼睛了,所以什么也看不到,如果为180,那么可以认为你的视界很广阔,  
aspect,这个好理解,就是实际窗口的纵横比,即x/y  
zNear,这个呢,表示你近处,的裁面,  
zFar表示远处的裁面,  
  
如果还没有理解就继续看,  
我们知道,远处的东西看起来要小一些,近处的东西看起来会大一些,这就是透视原理  
如下图所示  




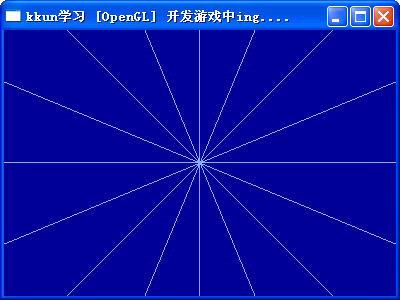
假设那两条线表示公路,理论上讲,它们的两条边是平行的,  
但现实情况中,它们在远方(可以无限远)总要相交于一点,  
实际线段AB的长度=CD的长度,只是在此例中使用了透视角,故会有如上的效果,是不是很接近现实的情况?  
  
结合我们刚才这两个函数  
zNear,眼睛距离近处的距离,假设为10米远,请不要设置为负值,OpenGl就傻了,不知道怎么算了,  
zFar表示远处的裁面,假设为1000米远,  
就是这两个参数的意义了,  
  
再解释下那个"眼睛睁开的角度"是什么意思,  
首先假设我们现在距离物体有50个单位距离远的位置,  
在眼睛睁开角度设置为45时,请看大屏幕:



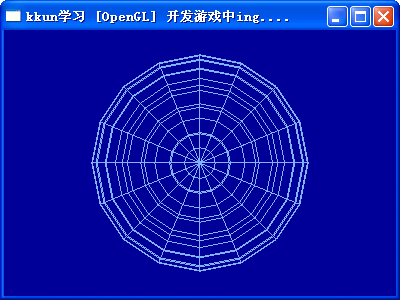
我们可以看到,在远处一个球,,很好玩哈,  
现在我们将眼睛再张开点看,将"眼睛睁开的角度"设置为178  
(180度表示平角,那时候我们将什么也看不到,眼睛睁太大了,眼大无神)



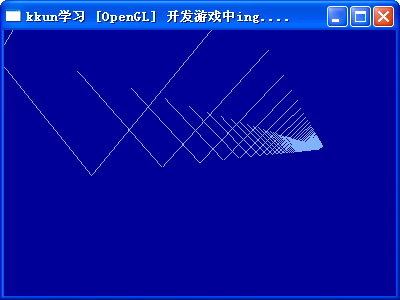
我们只看到一个点,,,,,,,,,,,,,,,,,,,,,,,,,,,  
因为我们看的范围太大了,这个球本身大小没有改变,但是它在我们的"视界"内太小了,  
  
  
反之,我们将眼睛闭小些,改为1度看看会出现什么情况呢?

  
  
在我们距离该物体3000距离远,"眼睛睁开的角度"为1时,我们似乎走进了这个球内,这个是不是类似于相机的焦距?  
  
当我们将"透视角"设置为0时,我们相当于闭上双眼,这个世界清静了,  
  
我们什么也看不到,,,,,,,,,

现在来看gluLookAt(GLdoble eyex,GLdouble eyey,GLdouble eyez,GLdouble centerx,GLdouble centery,GLdouble centerz,GLdouble upx,GLdouble upy,GLdouble upz);  
  
它共接受三对坐标,  
分别为eye,center,up  
故名思义,eye表示我们眼睛在"世界坐标系"中的位置,  
center表示眼睛"看"的那个点的坐标,  
最后那个up坐标表示观察者本身的方向,如果将观察点比喻成我们的眼睛,那么这个up则表示我们是正立还是倒立异或某一个角度在看,所看的影像大不相同,故此时需要指明我们现在正立,那么X,Z轴为0,Y轴为正即可,通常将其设置为1,只要表示一个向上的向量(方向)即可  
球是画在世界坐标系的原点上的,即O(0,0,0)坐标上,我们的眼睛位于观察点A(0,0,100),Z轴向屏幕里看去的方向为负,屏幕外我们的位置,Z轴为正值,其实很好理解,即我们距离原点的距离,设置100,将观察到如下图所示的影像



如果我们向前或向后移动,则相应的图像会变大或变小,这里其实就是运用了透视原理,近处的物体大,远处的物体小,实际物体的大小是不变的,  
  
同理改变center坐标(眼睛看去的那个点,可简单理解为视线的终点)也会影响球的大小,同样可以认为是改变了物体与观察点的距离所致,  
  
最后那个up坐标表示观察者本身的方向,如果将观察点比喻成我们的眼睛,那么这个up则表示我们是正立还是倒立异或某一个角度在看,所看的影像大不相同, 故此时需要指明我们现在正立,那么X,Z轴为0,Y轴为正即可,通常将其设置为1,只要表示一个向上的向量(方向)即可,我们指定0.1f或 0.00001f异或1000.0f,效果是一样的,只要能表示方向即可,  
  
  
  
  
以上理解了之后,来做一个测试  
透视图不变,最远处仍为3000,近处为0.1  
  
*gluPerspective                            // 设置透视图  
        (45,                            // 透视角设置为 45 度,在Y方向上以角度为单位的视野  
        (GLfloat)x/(GLfloat)y,    // 窗口的宽与高比  
        0.1f,                                // 视野透视深度:近点1.0f  
        3000.0f                            // 视野透视深度:始点0.1f远点1000.0f  
        );*  
  
将我们的观察点置于A(0,10,0),  
将观察位置(视线终点)坐标置于(0,0,0)  
然后在原点开始绘图,画一个V字形,并将Z轴的值从-1000递增加到+1000,增量为10,  
代码如下  
 *glColor3f(0.5f, 0.7f, 1.0f);  
  
    glBegin(GL\_LINES);  
        for(int i=-1000;i<=1000;i+=10)  
        {  
            glVertex3f(0,0,i);  
            glVertex3f(10,10,i);  
  
            glVertex3f(0,0,i);  
            glVertex3f(-10,10,i);  
        }  
    glEnd();*  
F5运行效果如下图

  
  
上图证实了我们的推测

//---------------------------------------------  
    //生成网络  
    glColor3f(0.5f, 0.7f, 1.0f);  
    int x=(int)(40\*2);  
      
    glBegin(GL\_LINES);  
            for(int i=-x;i<=x;i+=4)  
            {  
                glVertex3i(-x,0,i);  
                glVertex3i(x,0,i);  
  
                glVertex3i(i,0,x);  
                glVertex3i(i,0,-x);  
            }  
    glEnd();  
  
//生成球体  
    GLUquadricObj \* pObj;  
    pObj = gluNewQuadric();  
    gluQuadricDrawStyle(pObj,GLU\_LINE);  
    gluQuadricNormals(pObj,GLU\_SMOOTH);  
    gluSphere(pObj,16,16,16);  
  
QT范例源代码  
glwidget.h  
#ifndef glwidget\_H\_  
#define glwidget\_H\_  
#include <QtGui/QtGui>  
#include <QtOpenGL/QtOpenGL>  
  
class GLWidget : public QGLWidget   
{  
    Q\_OBJECT  
public:  
    GLWidget();  
protected:  
    void initializeGL();  
    void paintGL();  
    void resizeGL(int width,int height);  
  
    void mousePressEvent(QMouseEvent \*ev);  
    void mouseMoveEvent(QMouseEvent \*ev);  
    void mouseDoubleClickEvent(QMouseEvent \*ev);  
    void wheelEvent(QWheelEvent \*ev);  
private:  
    QPoint lastPos;  
    GLfloat eyeX,eyeY,eyeZ;  
};  
#endif  
  
glwidget.cpp  
#include "glwidget.h"  
  
GLWidget::GLWidget()  
:QGLWidget()  
{  
    setGeometry(300,300,600,480);  
    setWindowTitle(tr("glulookat test"));  
}  
  
void GLWidget::initializeGL()  
{  
    glShadeModel(GL\_SMOOTH);  
    glClearColor(0.5,0.5,0.5,0.5);  
    glClearDepth(1.0);  
    glEnable(GL\_DEPTH\_TEST);  
    glEnable(GL\_LEQUAL);  
    glHint(GL\_PERSPECTIVE\_CORRECTION\_HINT,GL\_NICEST);  
      
    eyeX = 0.0;  
    eyeY = 80.0;  
    eyeZ = 0.0;  
}  
  
void GLWidget::paintGL()  
{  
    glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);  
    glLoadIdentity();  
    //gluLookAt(9.0,0.0,10.0,0.0,0.0,-10.0,0.0,1.0,0.0);  
    //to look at the second lines  
    gluLookAt(eyeX,eyeY,eyeZ,0.0,0.0,0.0,1.0,0.0,0.0);  
  
    //glTranslatef(0.0,0.0,-10.0);  
    /\*glBegin(GL\_TRIANGLES);  
        glColor3f(1.0,0.0,0.0);  
        glVertex3f(0.0,1.0,0.0);  
        glVertex3f(-1.0,0.0,0.0);  
        glVertex3f(1.0,0.0,0.0);  
    glEnd();\*/  
  
    glColor3f(0.5f,0.7f,1.0f);  
    //glBegin(GL\_LINES);  
    //    for(int i = -1000;i <= 1000;i+=10)  
    //    {  
    //        glVertex3f(0.0,0.0,i);  
    //        glVertex3f(10.0,10.0,i);  
    //        glVertex3f(0.0,0.0,i);  
    //        glVertex3f(-10.0,10.0,i);  
    //    }  
    //glEnd();  
    int x = (int)(40\*2);  
    glBegin(GL\_LINES);  
        for(int i = -x ;i <= x ; i+=4 )  
        {  
            glVertex3i(-x,0,i);  
            glVertex3i(x,0,i);  
  
            glVertex3i(i,0,x);  
            glVertex3i(i,0,-x);  
        }  
    glEnd();  
    GLUquadricObj \*pObj;  
    pObj = gluNewQuadric();  
    gluQuadricDrawStyle(pObj,GLU\_LINE);  
    gluQuadricNormals(pObj,GLU\_SMOOTH);  
    gluSphere(pObj,16,16,16);  
}  
  
void GLWidget::resizeGL(int width,int height)  
{  
    if(height == 0)  
        height = 1;  
    glViewport(0,0,width,height);  
    glMatrixMode(GL\_PROJECTION);  
    glLoadIdentity();  
    gluPerspective(45.0,(GLfloat)width/(GLfloat)height,0.1,3000.0);  
    glMatrixMode(GL\_MODELVIEW);  
    glLoadIdentity();  
      
  
}  
  
void GLWidget::mousePressEvent(QMouseEvent \*ev)  
{  
    if(ev->buttons() & Qt::LeftButton)  
    {  
        lastPos = ev->pos();  
    }  
}  
  
void GLWidget::mouseMoveEvent(QMouseEvent \*ev)  
{  
    if(ev->buttons() & Qt::LeftButton)  
    {  
        QPoint pt = ev->pos() - lastPos;  
        if(eyeY >= 3000.0 && pt.y() > 0)  
        {  
            return ;  
        }  
        if(eyeY <= 1.0 && pt.y() < 0)  
            return ;  
        eyeY += pt.y();  
        updateGL();  
    }      
}  
  
void GLWidget::mouseDoubleClickEvent(QMouseEvent \*ev)  
{  
    QString str = QString("X:%1-Y:%2-Z:%3").arg(eyeX).arg(eyeY).arg(eyeZ);  
    QMessageBox::information(this,str,str);  
}  
  
void GLWidget::wheelEvent(QWheelEvent \*ev)  
{  
    QString str = QString("delta: %1").arg(ev->delta());  
    //QMessageBox::information(this,str,str);  
}  
  
运行结果：

