《计算机图形学原理实践》

实验报告

|  |  |  |
| --- | --- | --- |
| 实验名称 | ： | 制作动态三视图 |
|  |  |  |
| 姓 名 | ： |  |
| 学 号 | ： |  |
| 专业班级 | ： |  |
| 实验时间 | ： |  |

西南科技大学计算机科学与技术学院

**一、实验目的**

1、掌握主视图变换矩阵、俯视图变换图、侧视图变换矩阵

2、掌握斜等测图绘制方法

**二、实验步骤**

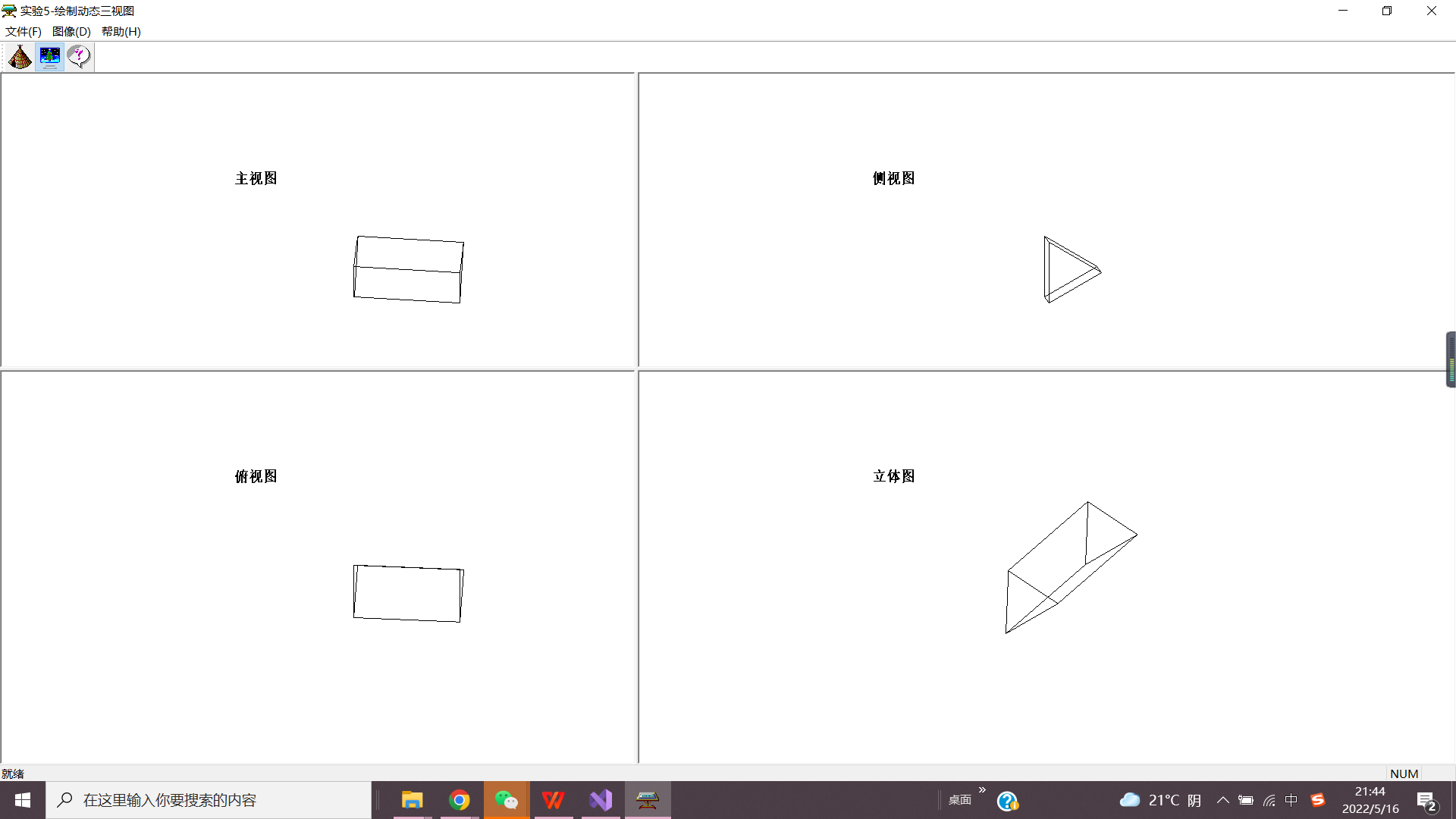
1、将屏幕客户区静态分为4个窗格，左上窗格为00，左下窗格编号为10，右上窗格编号为01，右下窗格编号为11；

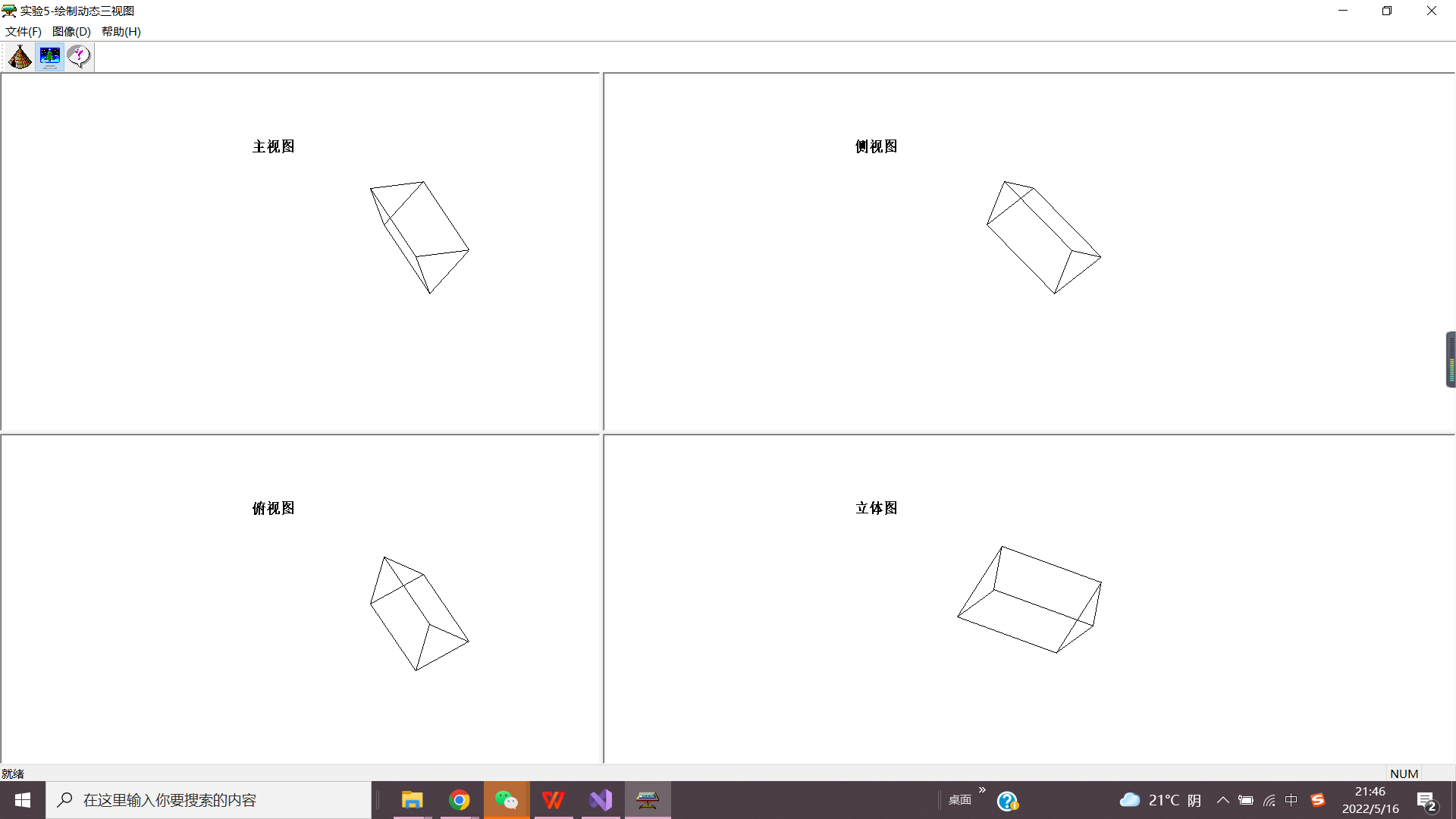
2、读入多面体的定点表和和表面表；

3、使用主视图变换矩阵在00窗口绘制主视图，使用俯视图变换矩阵在10窗口绘制俯视图，使用侧视图变换矩阵在0窗口绘制侧视图；

4、使用斜等测变换矩阵在11窗口绘制斜等测图。

1. **实验结果**





1. **实验总结**

利用书本第六章三维变换与投影的知识，在屏幕中心建立三维坐标系，建立齐次三维点类，通过继承二维点类定义三维点类。定义多个函数，从而达到实验目的。这是第五次实验，比之前四次增加了难度，将屏幕静态切分为了四个视区，再去绘制斜等测图和三视图。过程中遇到了一些困难，出现的动画不符合预期，不过最终在同学和老师帮助下也是顺利完成了，收获良多。

**附录：源代码（关键代码）**

// Transform.h

// Transform.h: interface for the CTransform class.

//

//////////////////////////////////////////////////////////////////////

#if !defined(AFX\_TRANSFORM\_H\_\_C4FA86B2\_0AF5\_409A\_94EB\_B02A1EE13E05\_\_INCLUDED\_)

#define AFX\_TRANSFORM\_H\_\_C4FA86B2\_0AF5\_409A\_94EB\_B02A1EE13E05\_\_INCLUDED\_

#if \_MSC\_VER > 1000

#pragma once

#endif // \_MSC\_VER > 1000

#include "P3.h"

class CTransform//三维变换

{

public:

CTransform();

virtual ~CTransform();

void SetMat(CP3 \*,int);

void Identity();

void Translate(double,double,double);//平移变换矩阵

void Scale(double,double,double);//比例变换矩阵

void Scale(double,double,double,CP3);//相对于任意点的比例变换矩阵

void RotateX(double);//旋转变换矩阵

void RotateX(double,CP3);//相对于任意点的X旋转变换矩阵

void RotateY(double);//旋转变换矩阵

void RotateY(double,CP3);//相对于任意点的Y旋转变换矩阵

void RotateZ(double);//旋转变换矩阵

void RotateZ(double,CP3);//相对于任意点的Z旋转变换矩阵

void ReflectX();//X轴反射变换矩阵

void ReflectY();//Y轴反射变换矩阵

void ReflectZ();//Z轴反射变换矩阵

void ReflectXOY();//XOY面反射变换矩阵

void ReflectYOZ();//YOZ面反射变换矩阵

void ReflectZOX();//ZOX面反射变换矩阵

void ShearX(double,double);//X方向错切变换矩阵

void ShearY(double,double);//Y方向错切变换矩阵

void ShearZ(double,double);//Z方向错切变换矩阵

void MultiMatrix();//矩阵相乘

public:

double T[4][4];

CP3 \*POld;

int num;

};

#endif // !defined(AFX\_TRANSFORM\_H\_\_C4FA86B2\_0AF5\_409A\_94EB\_B02A1EE13E05\_\_INCLUDED\_)

// TransForm.cpp

// TestView.cpp : CTestView 类的实现

//

#include "stdafx.h"

// SHARED\_HANDLERS 可以在实现预览、缩略图和搜索筛选器句柄的

// ATL 项目中进行定义，并允许与该项目共享文档代码。

#ifndef SHARED\_HANDLERS

#include "Test.h"

#endif

#include "MainFrm.h"

#include "TestDoc.h"

#include "TestView.h"

#include "VView.h"//

#include "HView.h"//

#include "WView.h"//

#include "Transform.h"

#include "Vector.h"

#define PI 3.1415926//圆周率

#include "math.h"//数学头文件

#include "math.h"//包含数学头文件

#define PI 3.1415926//PI的宏定义

#define Round(d) int(floor(d+0.5))//四舍五入宏定义

#ifdef \_DEBUG

#define new DEBUG\_NEW

#endif

// CTestView

IMPLEMENT\_DYNCREATE(CTestView, CView)

BEGIN\_MESSAGE\_MAP(CTestView, CView)

// 标准打印命令

ON\_COMMAND(ID\_FILE\_PRINT, &CView::OnFilePrint)

ON\_COMMAND(ID\_FILE\_PRINT\_DIRECT, &CView::OnFilePrint)

ON\_COMMAND(ID\_FILE\_PRINT\_PREVIEW, &CView::OnFilePrintPreview)

// ON\_COMMAND(IDM\_DRAWPIC, &CTestView::OnDrawpic)

// ON\_WM\_LBUTTONDOWN()

// ON\_WM\_LBUTTONUP()

ON\_WM\_KEYDOWN()

// ON\_COMMAND(IDM\_PLAY, &CTestView::OnPlay)

ON\_WM\_TIMER()

// ON\_UPDATE\_COMMAND\_UI(IDM\_PLAY, &CTestView::OnUpdatePlay)

ON\_WM\_ERASEBKGND()

END\_MESSAGE\_MAP()

// CTestView 构造/析构

CTestView::CTestView()

{

}

CTestView::~CTestView()

{

}

BOOL CTestView::PreCreateWindow(CREATESTRUCT& cs)

{

// CREATESTRUCT cs 来修改窗口类或样式

return CView::PreCreateWindow(cs);

}

// CTestView 绘制

void CTestView::OnDraw(CDC\* pDC)

{

CTestDoc\* pDoc = GetDocument();

ASSERT\_VALID(pDoc);

if (!pDoc)

return;

DoubleBuffer(pDC);

}

// CTestView 打印

BOOL CTestView::OnPreparePrinting(CPrintInfo\* pInfo)

{

// 默认准备

return DoPreparePrinting(pInfo);

}

void CTestView::OnBeginPrinting(CDC\* /\*pDC\*/, CPrintInfo\* /\*pInfo\*/)

{

// TODO: 添加额外的打印前进行的初始化过程

}

void CTestView::OnEndPrinting(CDC\* /\*pDC\*/, CPrintInfo\* /\*pInfo\*/)

{

}

// CTestView 诊断

#ifdef \_DEBUG

void CTestView::AssertValid() const

{

CView::AssertValid();

}

void CTestView::Dump(CDumpContext& dc) const

{

CView::Dump(dc);

}

CTestDoc\* CTestView::GetDocument() const // 非调试版本是内联的

{

ASSERT(m\_pDocument->IsKindOf(RUNTIME\_CLASS(CTestDoc)));

return (CTestDoc\*)m\_pDocument;

}

#endif //\_DEBUG

// CTestView 消息处理程序

void CTestView::ReadPoint()//点表

{

int a=80,b=140;//a三角形边长，b棱长

P[0].x=-a/2;P[0].y=0; P[0].z=-b/2;//P0

P[1].x=0; P[1].y=sqrt(3.0)/2\*a;P[1].z=-b/2;//P1

P[2].x=a/2; P[2].y=0; P[2].z=-b/2;//P2

P[3].x=-a/2;P[3].y=0; P[3].z=b/2; //P3

P[4].x=0; P[4].y=sqrt(3.0)/2\*a;P[4].z=b/2; //P4

P[5].x=a/2; P[5].y=0; P[5].z=b/2; //P5

}

void CTestView::ReadFace()//面表

{

//面的顶点数和面的顶点索引

F[0].SetNum(4);F[0].vI[0]=0;F[0].vI[1]=3;F[0].vI[2]=4;F[0].vI[3]=1;

F[1].SetNum(3);F[1].vI[0]=0;F[1].vI[1]=1;F[1].vI[2]=2;

F[2].SetNum(4);F[2].vI[0]=0;F[2].vI[1]=2;F[2].vI[2]=5;F[2].vI[3]=3;

F[3].SetNum(4);F[3].vI[0]=1;F[3].vI[1]=4;F[3].vI[2]=5;F[3].vI[3]=2;

F[4].SetNum(3);F[4].vI[0]=3;F[4].vI[1]=5;F[4].vI[2]=4;

}

void CTestView::Rotate()//旋转变换

{

CTransform trans;

trans.SetMat(P,6);

trans.RotateX(1);

trans.RotateY(1);

}

void CTestView::DoubleBuffer(CDC \*pDC)//双缓冲

{

CRect rect;//定义客户区

GetClientRect(&rect);//获得客户区的大小

CDC memDC[4];

CDC\* pViewDC[4];

CMainFrame \*pFrame=(CMainFrame\*)AfxGetMainWnd();//获得框架指针

CVView \*pVView=(CVView\*)pFrame->m\_wndSplitter.GetPane(0,0);//获得主视图窗格指针

CWView \*pWView=(CWView\*)pFrame->m\_wndSplitter.GetPane(0,1);//获得侧视图窗格指针

CHView \*pHView=(CHView\*)pFrame->m\_wndSplitter.GetPane(1,0);//获得俯视图窗格指针

pViewDC[0]=pDC;//多面体屏幕DC

pViewDC[1]=pVView->GetDC();//主视图屏幕DC

pViewDC[2]=pWView->GetDC();//侧视屏幕DC

pViewDC[3]=pHView->GetDC();//俯视图屏幕DC

CBitmap NewBitmap[4],\*pOldBitmap[4];//内存中承载图像的临时位图

for(int i=0;i<4;i++)//创建内存DC

{

pViewDC[i]->SetMapMode(MM\_ANISOTROPIC);//pDC自定义坐标系

pViewDC[i]->SetWindowExt(rect.Width(),rect.Height());//设置窗口范围

pViewDC[i]->SetViewportExt(rect.Width(),-rect.Height());//x轴水平向右，y轴铅直向上

pViewDC[i]->SetViewportOrg(rect.Width()/2,rect.Height()/2);//屏幕中心为原点

NewBitmap[i].CreateCompatibleBitmap(pViewDC[i],rect.Width(),rect.Height());//创建兼容位图

memDC[i].CreateCompatibleDC(pViewDC[i]);

pOldBitmap[i]=memDC[i].SelectObject(&NewBitmap[i]);

memDC[i].FillSolidRect(&rect,pViewDC[i]->GetBkColor());//按原来背景填充客户区，否则是黑色

memDC[i].SetMapMode(MM\_ANISOTROPIC);//MemDC自定义坐标系

memDC[i].SetWindowExt(rect.Width(),rect.Height());

memDC[i].SetViewportExt(rect.Width(),-rect.Height());

memDC[i].SetViewportOrg(rect.Width()/2,rect.Height()/2);

}

rect.OffsetRect(-rect.Width()/2,-rect.Height()/2);

DrawOblique(&memDC[0]);//绘制斜等测图

DrawVView(&memDC[1]); //绘制主视图

DrawWView(&memDC[2]); //绘制侧视图

DrawHView(&memDC[3]); //绘制俯视图

for(int i=0;i<4;i++)

{

pViewDC[i]->BitBlt(rect.left,rect.top,rect.Width(),rect.Height(),&memDC[i],-rect.Width()/2,-rect.Height()/2,SRCCOPY);//将内存memDC中的位图拷贝到显示pDC中

memDC[i].SelectObject(pOldBitmap[i]);//恢复位图

NewBitmap[i].DeleteObject();//删除位图

ReleaseDC(pViewDC[i]);//释放视图DC

}

}

void CTestView::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)

{

// TODO: 在此添加消息处理程序代码和/或调用默认值

CTransform trans;

trans.SetMat(P,6);

switch(nChar)

{

case VK\_UP:

trans.RotateX(1);//设定步长

break;

case VK\_DOWN:

trans.RotateX(-1);

break;

case VK\_LEFT:

trans.RotateY(-1);

break;

case VK\_RIGHT:

trans.RotateY(1);

break;

default:

break;

}

Invalidate(FALSE);

CView::OnKeyDown(nChar, nRepCnt, nFlags);

}

void CTestView::DrawTriView(CDC \*pDC,CP3 P[])//绘制三视图线框模型

{

for (int nFace = 0;nFace < 5;nFace++)

{

CP3 ScreenP, t;

CLine \*line = new CLine;

for (int nVertex = 0;nVertex < F[nFace].vN;nVertex++)

{

ScreenP = P[F[nFace].vI[nVertex]];

if (0 == nVertex)

{

line->MoveTo(pDC, -ScreenP.z, ScreenP.y);

t = ScreenP;

}

else

{

line->LineTo(pDC, -ScreenP.z, ScreenP.y);

}

}

line->LineTo(pDC, -t.z, t.y);

delete line;

}

}

void CTestView::OnTimer(UINT\_PTR nIDEvent)

{

// TODO: 在此添加消息处理程序代码和/或调用默认值

if(((CMainFrame\*)AfxGetMainWnd())->bPlay)

{

Rotate();

Invalidate(FALSE);;

}

CView::OnTimer(nIDEvent);

}

void CTestView::TOMatrix()//斜等测变换矩阵

{

TO[0][0] = 1; TO[0][1] = 0; TO[0][2] = 0; TO[0][3] = 0;

TO[1][0] = 0; TO[1][1] = 1; TO[1][2] = 0; TO[1][3] = 0;

TO[2][0] = -1 / sqrt(2.0);TO[2][1] = -1 / sqrt(2.0);TO[2][2] = 0;TO[2][3] = 0;

TO[3][0] = 0;TO[3][1] = 0; TO[3][2] = 0; TO[3][3] = 1;

}

void CTestView::TVMatrix()//主视图变换矩阵

{

TV[0][0] = 0;TV[0][1] = 0;TV[0][2] = 0;TV[0][3] = 0;

TV[1][0] = 0;TV[1][1] = 1;TV[1][2] = 0;TV[1][3] = 0;

TV[2][0] = 0;TV[2][1] = 0;TV[2][2] = 1;TV[2][3] = 0;

TV[3][0] = 0;TV[3][1] = 0;TV[3][2] = 0;TV[3][3] = 1;

}

void CTestView::THMatrix()//俯视图变换矩阵

{

TH[0][0] = 0;TH[0][1] = -1;TH[0][2] = 0;TH[0][3] = 0;

TH[1][0] = 0;TH[1][1] = 0;TH[1][2] = 0;TH[1][3] = 0;

TH[2][0] = 0;TH[2][1] = 0;TH[2][2] = 1;TH[2][3] = 0;

TH[3][0] = 0;TH[3][1] = 0;TH[3][2] = 0;TH[3][3] = 1;

}

void CTestView::TWMatrix()//侧视图变换矩阵

{

TW[0][0] = 0;TW[0][1] = 0;TW[0][2] = -1;TW[0][3] = 0;

TW[1][0] = 0;TW[1][1] = 1;TW[1][2] = 0;TW[1][3] = 0;

TW[2][0] = 0;TW[2][1] = 0;TW[2][2] = 0;TW[2][3] = 0;

TW[3][0] = 0;TW[3][1] = 0;TW[3][2] = 0;TW[3][3] = 1;

}

void CTestView::DrawOblique(CDC \*pDC)

{

pDC->TextOut(-230,130,CString("立体图"));

MultiMatrix(TO);

DrawObject(pDC,PNew);

}

void CTestView::DrawObject(CDC \*pDC,CP3 P[])//绘制斜等测图线框模型

{

for(int nFace=0;nFace<5;nFace++)

{

CP2 ScreenP,t;

CLine \*line=new CLine;

for(int nVertex=0;nVertex<F[nFace].vN;nVertex++)//顶点循环

{

ScreenP=P[F[nFace].vI[nVertex]];//斜等测投影

if(0==nVertex)

{

line->MoveTo(pDC,ScreenP.x,ScreenP.y);

t=ScreenP;

}

else

line->LineTo(pDC,ScreenP.x,ScreenP.y);

}

line->LineTo(pDC,t.x,t.y);//闭合多边形

delete line;

}

}

void CTestView::DrawVView(CDC \*pDC)

{

pDC->TextOut(-230,130,CString("主视图"));

MultiMatrix(TV);

DrawTriView(pDC,PNew);

}

void CTestView::DrawHView(CDC \*pDC)

{

pDC->TextOut(-230,130,CString("俯视图"));

MultiMatrix(TH);

DrawTriView(pDC,PNew);

}

void CTestView::DrawWView(CDC \*pDC)

{

pDC->TextOut(-230,130,CString("侧视图"));

MultiMatrix(TW);

DrawTriView(pDC,PNew);

}

void CTestView::MultiMatrix(double T[][4])//两个矩阵相乘

{

int i;

for( i=0;i<6;i++)

{

PNew[i].x=P[i].x\*T[0][0]+P[i].y\*T[1][0]+P[i].z\*T[2][0]+P[i].w\*T[3][0];

PNew[i].y=P[i].x\*T[0][1]+P[i].y\*T[1][1]+P[i].z\*T[2][1]+P[i].w\*T[3][1];

PNew[i].z=P[i].x\*T[0][2]+P[i].y\*T[1][2]+P[i].z\*T[2][2]+P[i].w\*T[3][2];

PNew[i].w=P[i].x\*T[0][3]+P[i].y\*T[1][3]+P[i].z\*T[2][3]+P[i].w\*T[3][3];

}

}

BOOL CTestView::OnEraseBkgnd(CDC\* pDC)

{

// TODO: 在此添加消息处理程序代码和/或调用默认值

return true;

//return CView::OnEraseBkgnd(pDC);

}

void CTestView::OnInitialUpdate()

{

CView::OnInitialUpdate();

// TODO: 在此添加专用代码和/或调用基类

TOMatrix();

TVMatrix();

THMatrix();

TWMatrix();

ReadPoint();

ReadFace();

SetTimer(1,50,NULL);//设置定时器

}

}