**案例33-圆环面隐线算法**

文档编写：霍波魏

校稿/修订：孔令德

时间2019~2020

联系方式：QQ997796978

**说明：**本套案例由孔令德开发，原版本为Visual C++6.0，配套于孔令德的著作《计算机图形学-基于MFC三维图形开发》一书。孔令德计算机工程研究所的学生霍波魏在学习计算机图形学期间，对本套案例进行了升级并编写了学习文档。现在程序的编写和程序的解释都是基于Windows 10操作系统，使用Microsoft visual studio 2017平台的MFC（中文版）开发。

1. **案例描述：**

本案例利用隐线算法，绘制圆环并进行消隐。

圆环面由一个偏置圆绕Y轴旋转而成，圆环半径表示为r1；环截面半径表示为r2。圆环面片总数为n1×n2，其中n1为环周面片数，n2为环身面片数。

圆环参数：周向网格n1：36；纵向网格n2为：36。

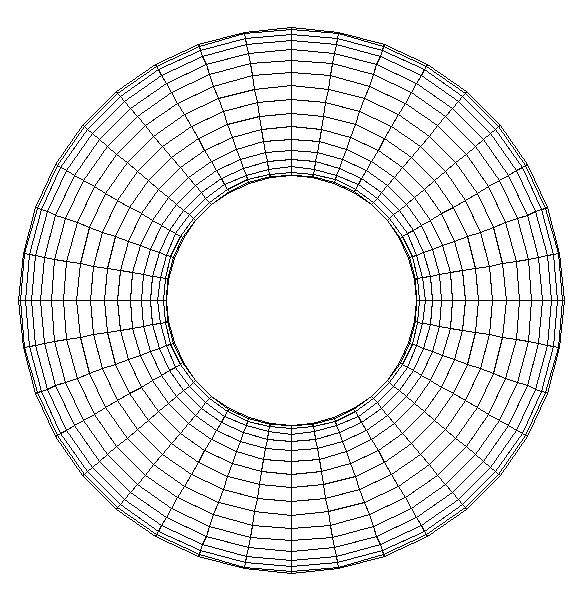
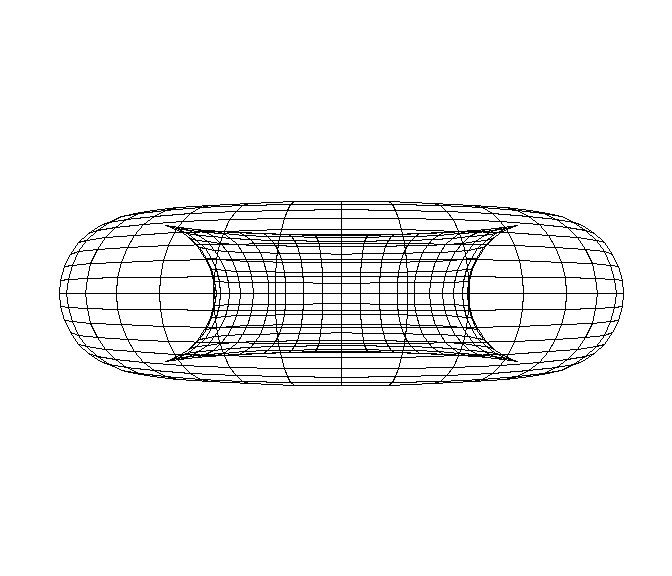


图33-1 主视图 图33-2 侧视图

1. **知识点**

本案例主要讲解的是隐线算法，也称为背面剔除算法，他是解决凸面体消隐问题比较简单的一种方法，背面剔除算法的关键是给出测试其每个表面可见性的判别式，可以根据其外法向量与视向量V（从表面上的一个顶点指向视点）的夹角θ来进行可见性检测。当夹角θ位于之间时，该表面可见，绘制边界线；当夹角θ位于之间时，表面不可见，不绘制边界线。

1. **实现步骤：**
2. 添加基础类与添加绘制圆柱面体的CCircularRing类。
3. 在CCircularRing类中计算顶点坐标、读入面表，绘制图形，透视变换参数初始化、设置视点位置以及透视变换。
4. 在CTestView中添加消息响应函数，在OnDraw中调用DoubleBuffer函数。
5. **主要算法**

1. CCircularRing类：

public:

CCircularRing();

virtual~CCircularRing();

void ReadPoint();//读入点表

void ReadFace();//读入面表

void InitParameter();//参数初始化

void SetViewPoint();//设置视点

CP2i PersPectiveProjection(CP3d P);//透视投影

void Draw(CDC\* pDC);//绘制圆环线框

public:

int nClientWidth;//屏幕客户区宽度

int nClientHeight;//屏幕客户区高度

int nHWidth, nHHeight;//屏幕客户区的半宽和半高

int N1, N2;//N1为周向网格,N2纵向网格

double k[9];

double R, Theta, Phi, d;//视点在用户坐标系中的球坐标

double Alpha, Beta;//x方向旋转α角,y方向旋转β角

CP3d ViewPoint;//视点球坐标位置

CP2i ScreenCoorP;//屏幕坐标系的二维坐标点

CP3d \*V;////点表

CFace \*\*F;//面表

void CCircularRing::ReadPoint()//点表

{

int tAlpha = 10, tBeta = 10;//等分角度

int r1 = 220, r2 = 80;//圆环半径和环截面半径

N1 = 360 / tAlpha, N2 = 360 / tBeta;//面片数量为N1×N2

V = new CP3d[N1 \* N2];//顶点动态数组

for (int i = 0; i < N1; i++)

{

double tAlpha1 = tAlpha \* i \* PI / 180;

for (int j = 0; j < N2; j++)//顶点赋值

{

double tBeta1 = tBeta \* j \* PI / 180;

V[i\*N2 + j].x = (r1 + r2 \* sin(tBeta1)) \* sin(tAlpha1);

V[i\*N2 + j].y = r2 \* cos(tBeta1);

V[i\*N2 + j].z = (r1 + r2 \* sin(tBeta1)) \* cos(tAlpha1);

}

}

}

void CCircularRing::ReadFace()//面表

{

F = new CFace \*[N1];

for (int n = 0; n < N1; n++)

F[n] = new CFace[N2];

for (int i = 0; i < N1; i++)

{

for (int j = 0; j < N2; j++)

{

int tempi = i + 1;

int tempj = j + 1;

if (N2 == tempj) tempj = 0;

if (N1 == tempi) tempi = 0;

F[i][j].SetNum(4);//面的顶点数

F[i][j].vI[0] = i \* N2 + j;//面的顶点索引号

F[i][j].vI[1] = i \* N2 + tempj;

F[i][j].vI[2] = tempi \* N2 + tempj;

F[i][j].vI[3] = tempi \* N2 + j;

}

}

}

void CCircularRing::InitParameter()//透视变换参数初始化

{

k[1] = sin(PI \* Theta / 180);

k[2] = sin(PI \* Phi / 180);

k[3] = cos(PI \* Theta / 180);

k[4] = cos(PI \* Phi / 180);

k[5] = k[2] \* k[3];

k[6] = k[2] \* k[1];

k[7] = k[4] \* k[3];

k[8] = k[4] \* k[1];

}

CP2i CCircularRing::PersPectiveProjection(CP3d P)//透视投影

{

CP3d ViewCoorP;//观察坐标系内的点

ViewCoorP.x = k[3] \* P.x - k[1] \* P.z;//观察坐标系三维坐标

ViewCoorP.y = -k[8] \* P.x + k[2] \* P.y - k[7] \* P.z;

ViewCoorP.z = -k[6] \* P.x - k[4] \* P.y - k[5] \* P.z + R;

ScreenCoorP.x = int(d \* ViewCoorP.x / ViewCoorP.z);//屏幕坐标系二维坐标

ScreenCoorP.y = int(d \* ViewCoorP.y / ViewCoorP.z);

return ScreenCoorP;

}

void CCircularRing::SetViewPoint()//设置视点

{

ViewPoint.x = R \* k[6];

ViewPoint.y = R \* k[4];

ViewPoint.z = R \* k[5];

}

void CCircularRing::Draw(CDC\* pDC)//绘制圆环线框模型

{

CLine \*line = new CLine;

CP2i Point4[4], t4;//圆环顶点数组

for (int i = 0; i < N1; i++)

{

for (int j = 0; j < N2; j++)

{

CVector3 ViewVector(V[F[i][j].vI[0]], ViewPoint);//面的视向量

ViewVector = ViewVector.Normalize();//单位化视向量

F[i][j].SetFaceNormal(V[F[i][j].vI[0]], V[F[i][j].vI[1]], V[F[i][j].vI[2]]);

F[i][j].fNormal.Normalize();//单位化法矢量

if (Dot(ViewVector, F[i][j].fNormal) >= 0)//背面剔除

{

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

{

PersPectiveProjection(V[F[i][j].vI[m]]);

Point4[m] = ScreenCoorP;

}

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

{

if (0 == n)

{

line->MoveTo(pDC, nHWidth + Point4[n].x, nHHeight - Point4[n].y);

t4 = Point4[n];

}

else

line->LineTo(pDC, nHWidth + Point4[n].x, nHHeight - Point4[n].y);

}

line->LineTo(pDC, nHWidth + t4.x, nHHeight - t4.y);//闭合多边形

}

}

}

delete line;

}

2.CTestView类：

public:

void DoubleBuffer(CDC\* pDC);//双缓冲绘图

void DrawObject (CDC\* pDC);

protected:

CTransform3 tran;//变换对象

BOOL bPlay;//动画开关

CCircularRing CirRing;

void CTestView::DoubleBuffer(CDC \* pDC)

{

CRect rect;//定义客户区

GetClientRect(&rect);//获得客户区的大小

cone.nClientWidth = rect.Width();//屏幕客户区宽度

cone.nClientHeight = rect.Height();//屏幕客户区高度

cone.nHWidth = cone.nClientWidth / 2;//屏幕客户区半宽

cone.nHHeight = cone.nClientHeight / 2;//屏幕客户区半高

CDC memDC;

memDC.CreateCompatibleDC(pDC);

CBitmap NewBitmap, \*pOldBitmap;

NewBitmap.CreateCompatibleBitmap(pDC, cone.nClientWidth, cone.nClientHeight);

pOldBitmap = memDC.SelectObject(&NewBitmap);

memDC.FillSolidRect(rect, pDC->GetBkColor());

Draw(&memDC);

pDC->BitBlt(0, 0, cone.nClientWidth, cone.nClientHeight, &memDC, 0, 0, SRCCOPY);

memDC.SelectObject(pOldBitmap);

NewBitmap.DeleteObject();

}

void CTestView::Draw(CDC \* pDC)

{

CirRing.Draw(pDC);

}

1. **实现效果：**

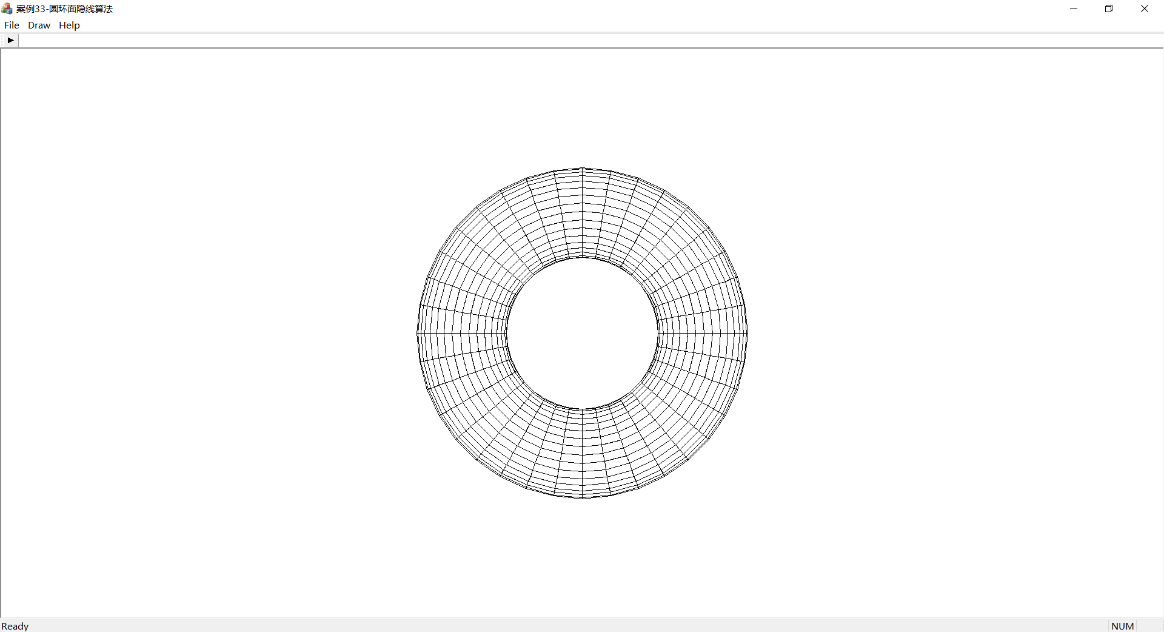


图33-3 圆环面隐线算法效果图