　　最近碰到region的难题：让Region实入库，我的想法是把它炸开后转化为闭合Pline。

那么现在的问题转变为Region炸开后的实体能否转化为Pline？答案是肯定的。但是对于复杂的实体就需要牺牲点什么了，这就是你要付出的代价。下面就看看如何实现了，其中会提到一些收集的知识点和代码。

我 仔细研究了网上贴出的代码，感觉都是一些无用的代码（对于我转pline，代码还是有价值的）。因为Line和Arc转化pline谁都会的，可是 region炸开后不止有Line和Arc，还有Spline和Ellipse。大家的难题不就是Spline和Ellipse如何转化成pline吗？ 我的看法－直接转换是不能；我们只能舍弃精度，把他们离散后生成pline线了。离散的单位越小精度就越高。

……

es = pRegion->explode(pELlipseArray);

……

AcDbVoidPtrArray ptArray;

for (int h = 0; h < pELlipseArray.length(); h++)

{

AcDbCurve \*pEntitytemp = (AcDbCurve \*)pELlipseArray.at(h);

CString str = pEntitytemp->isA()->name();

if (str.CompareNoCase(\_T("AcDbLine")) == 0)

{

AcDbPolyline \*Pline = new AcDbPolyline(2);

AcGePoint3d ptStart,ptEnd;

pEntitytemp->getStartPoint(ptStart);

pEntitytemp->getEndPoint(ptEnd);

Pline->addVertexAt(0, AcGePoint2d(ptStart.x, ptStart.y), 0);

Pline->addVertexAt(1, AcGePoint2d(ptEnd.x, ptEnd.y), 0);

ptArray.append(Pline);

}

else if (str.CompareNoCase(\_T("AcDbEllipse")) == 0)

{

AcDbEllipse \*pEllipse = (AcDbEllipse\*)pELlipseArray.at(h);

double dParam1=0,dParam2=0;

pEllipse->getStartParam(dParam1);

pEllipse->getEndParam(dParam2);

double dLen1 = 0;

pEllipse->getDistAtParam(dParam1,dLen1);

double dLen2 = 0;

pEllipse->getDistAtParam(dParam2,dLen2);

double dLength = dLen2 - dLen1;

double dStep = min(10,dLength/100);

AcGePoint3dArray ptArr;

AcGePoint3d pt;

pEllipse->getStartPoint(pt);

ptArr.append(pt);

double dLen = 0;

while(dLen < dLength)

{

pEllipse->getPointAtDist(dLen, pt);

ptArr.append(pt);

dLen += dStep;

}

pEllipse->getEndPoint(pt);

ptArr.append(pt);

AcDbPolyline \*Pline = new AcDbPolyline(ptArr.length());

for (int i = 0; i < ptArr.length(); i ++)

{

Pline->addVertexAt(i, AcGePoint2d(ptArr.at(i).x, ptArr.at(i).y), 0);

}

ptArray.append(Pline);

}

else if (str.CompareNoCase(\_T("AcDbSpline")) == 0)

{

AcDbSpline \*pSpline = (AcDbSpline\*)pELlipseArray.at(h);

double dParam1=0,dParam2=0;

pSpline->getStartParam(dParam1);

pSpline->getEndParam(dParam2);

double dLen1 = 0;

pSpline->getDistAtParam(dParam1,dLen1);

double dLen2 = 0;

pSpline->getDistAtParam(dParam2,dLen2);

double dLength = dLen2 - dLen1;

double dStep = min(2,dLength/100);

AcGePoint3dArray ptArr;

AcGePoint3d pt;

pSpline->getStartPoint(pt);

ptArr.append(pt);

double dLen = 0;

while(dLen < dLength)

{

pSpline->getPointAtDist(dLen, pt);

ptArr.append(pt);

dLen += dStep;

}

pSpline->getEndPoint(pt);

ptArr.append(pt);

AcDbPolyline \*Pline = new AcDbPolyline(ptArr.length());

for (int i = 0; i < ptArr.length(); i ++)

{

Pline->addVertexAt(i, AcGePoint2d(ptArr.at(i).x, ptArr.at(i).y), 0);

}

ptArray.append(Pline);

}

else if (str.CompareNoCase(\_T("AcDbArc")) == 0)

{

AcDbArc \*pArc=(AcDbArc \*)pEntitytemp;

AcGeCircArc2d \*pGArc=new AcGeCircArc2d;

pGArc->setCenter(AcGePoint2d(pArc->center().x, pArc->center().y));

pGArc->setRadius(pArc->radius());

pGArc->setAngles(pArc->startAngle(), pArc->endAngle());

double bulge = 0.0;

double ang = 0.25 \* (pGArc->endAng() - pGArc->startAng());

bulge = tan(ang);

if(pGArc->isClockWise())

{

bulge = -bulge;

}

AcDbPolyline \*Pline = new AcDbPolyline(2);

AcGePoint3d ptStart,ptEnd;

pArc->getStartPoint(ptStart);

pArc->getEndPoint(ptEnd);

Pline->addVertexAt(0, AcGePoint2d(ptStart.x, ptStart.y), bulge);

Pline->addVertexAt(1, AcGePoint2d(ptEnd.x, ptEnd.y), 0);

ptArray.append(Pline);

}

}

AcGePoint3d ptStart,ptEnd;

AcDbPolyline\* pPolyline;

pPolyline = (AcDbPolyline\*)ptArray.at(0);

pPolyline->getStartPoint(ptStart);

pPolyline->getEndPoint(ptEnd);

int num = pPolyline->numVerts();

int Number = -1;

int startORend = 0;

while (ptArray.length() > 1)

{

double Dist = 999999999;

for (int i = 1; i < ptArray.length(); i ++)

{

AcDbPolyline \*pPolyline1;

AcGePoint3d ptStart1, ptEnd1;

pPolyline1 = (AcDbPolyline\*)ptArray.at(i);

pPolyline1->getStartPoint(ptStart1);

pPolyline1->getEndPoint(ptEnd1);

if (Dist > ptEnd.distanceTo(ptStart1))

{

Dist = ptEnd.distanceTo(ptStart1);

Number = i;

startORend = 1;

}

if (Dist >  ptEnd.distanceTo(ptEnd1))

{

Dist = ptEnd.distanceTo(ptEnd1);

Number = i;

startORend = 2;

}

}

AcDbPolyline \*pPolyline1 = (AcDbPolyline\*)ptArray.at(Number);

if (startORend == 1)

{

for (int j = 0; j < pPolyline1->numVerts(); j ++)

{

AcGePoint2d pttemp;

double bulge, WidthStart, WidthEnd;

pPolyline1->getPointAt(j, pttemp);

pPolyline1->getBulgeAt(j, bulge);

pPolyline1->getWidthsAt(j,WidthStart, WidthEnd);

pPolyline->addVertexAt(num, pttemp, bulge, WidthStart, WidthEnd);

num++;

ptEnd.x = pttemp.x;

ptEnd.y = pttemp.y;

ptEnd.z = 0;

}

ptArray.removeAt(Number);

}

if (startORend == 2)

{

for (int j =  pPolyline1->numVerts() - 1; j >= 0; j --)

{

AcGePoint2d pttemp;

double bulge, WidthStart, WidthEnd;

pPolyline1->getPointAt(j, pttemp);

pPolyline1->getBulgeAt(j, bulge);

pPolyline1->getWidthsAt(j, WidthEnd, WidthStart);

pPolyline->addVertexAt(num, pttemp, bulge, WidthStart, WidthEnd);

num++;

ptEnd.x = pttemp.x;

ptEnd.y = pttemp.y;

ptEnd.z = 0;

}

ptArray.removeAt(Number);

}

}

if (!pPolyline->isClosed())

{

pPolyline->setClosed(true);

}

先来看看大侠们的代码吧：

某 大侠的思路：region炸开后形成一组Line、Arc而不是PolyLines, 这个时候想得到region的边界PolyLine。在AutoCAD 交互式环境下，可以炸开后，把新实体调用PEDIT，链接成polyline（注：此方法不能处理Spline和Ellipse）；或者使用 AcDbPolyline类的方法连结一起。即，调用 AcDbEntity::explode方法，得到的AcDbVoidPtrArray 数组，这个数组里面的是炸开后的实体，你自己把这些实体，使用AcDbPolyline类的相关方法，链接到一起后，再添加到数据库中，这个能保证唯一。 或者先把数组里面的实体添加到数据库，然后调用PEDIT命令去编辑他们。添加后的实体，你自己可以把他们变成选择集交给PEDIT去做。

使 用了explore命令后，问题来了: AcDbVoidPtrArray 数组是按序存放的（也许是碰巧，否则还需排序），这样就可以新建一AcDbPolyLine然后逐点addVertexAt()，arc还需计算 bulge.（但环状Region该有两条PolyLine）

通用的DB--->GE的转换：

//转换AcDbCurve到AcGeCurve3d

Acad::ErrorStatus XdDbUtils::convertDbCurveToGeCurve(AcDbCurve \*pDbCurve,AcGeCurve3d \*&pGeCurve)

{

pGeCurve=NULL;

if (pDbCurve->isKindOf(AcDbLine::desc()))

{

AcDbLine \*pL=(AcDbLine \*)pDbCurve;

AcGeLineSeg3d \*pGL=new AcGeLineSeg3d;

pGL->set(pL->startPoint(),pL->endPoint());

pGeCurve=(AcGeCurve3d \*)pGL;

}

else if (pDbCurve->isKindOf(AcDbArc::desc()))

{

AcDbArc \*pArc=(AcDbArc \*)pDbCurve;

double ans,ane;

ans=pArc->startAngle();

ane=pArc->endAngle();

AcGeCircArc3d \*pGArc=new AcGeCircArc3d;

pGArc->setCenter(pArc->center());

pGArc->setRadius(pArc->radius());

pGArc->setAngles(ans,ane);

pGeCurve=(AcGeCurve3d \*)pGArc;

}

else if (pDbCurve->isKindOf(AcDbCircle::desc()))

{

AcDbCircle \*pCir=(AcDbCircle \*)pDbCurve;

AcGeCircArc3d \* pGCir=new AcGeCircArc3d;

pGCir->setCenter(pCir->center());

pGCir->setRadius(pCir->radius());

pGeCurve=(AcGeCurve3d \*)pGCir;

}

else if (pDbCurve->isKindOf(AcDbEllipse::desc()))

{

AcDbEllipse \*pEli=(AcDbEllipse \*)pDbCurve;

AcGePoint3d pt1,center=pEli->center();

AcGeEllipArc3d \*pGEli=new AcGeEllipArc3d;

pGEli->setCenter(center);

pGEli->setAxes(pEli->majorAxis(),pEli->minorAxis());

pEli->getClosestPointTo(center,pt1,Adesk::kTrue);

pGEli->setMajorRadius(pt1.distanceTo(center)/pEli->radiusRatio());

pGEli->setMinorRadius(pt1.distanceTo(center));

double endang=pEli->endAngle(),startang=pEli->startAngle();

if (startang>endang){

endang+=2\*PI;

}

pGEli->setAngles(endang,startang);

pGeCurve=(AcGeCurve3d \*)pGEli;

}

else if (pDbCurve->isKindOf(AcDbSpline::desc()))

{

AcDbSpline \*pSL=(AcDbSpline \*)pDbCurve;

if (!pSL)

return Acad::eNotImplemented;

if (pSL->isNull()==Adesk::kTrue)

return Acad::eNotImplemented;

int degree;

Adesk::Boolean rational;

Adesk::Boolean closed;

Adesk::Boolean periodic;

AcGePoint3dArray controlPoints;

AcGeDoubleArray knots;

AcGeDoubleArray weights;

double controlPtTol;

double knotTol;

AcGeTol tol;

Acad::ErrorStatus es;

es=pSL->getNurbsData(degree,rational,closed,periodic,controlPoints,knots,weights,

controlPtTol,knotTol);

if (es!=Acad::eOk)

return Acad::eNotImplemented;

if (rational==Adesk::kTrue)

{

AcGeNurbCurve3d \*pNurb=new AcGeNurbCurve3d(degree,knots,controlPoints,weights,periodic);

if (closed==Adesk::kTrue)

pNurb->makeClosed();

if (pSL->hasFitData()==Adesk::kTrue)

{

AcGePoint3dArray fitPoints;

double fitTolerance;

Adesk::Boolean tangentsExist;

AcGeVector3d startTangent;

AcGeVector3d endTangent;

pSL->getFitData(fitPoints,degree,fitTolerance,tangentsExist,startTangent,endTangent);

tol.setEqualPoint(fitTolerance);

if (tangentsExist==Adesk::kTrue)

pNurb->setFitData(fitPoints,startTangent,endTangent,tol);

else

pNurb->setFitData(degree,fitPoints,tol);

}

pGeCurve=(AcGeCurve3d \*)pNurb;

}

else

{

AcGeNurbCurve3d \*pNurb=new AcGeNurbCurve3d(degree,knots,controlPoints,periodic);

if (closed==Adesk::kTrue)

pNurb->makeClosed();

if (pSL->hasFitData()==Adesk::kTrue)

{

AcGePoint3dArray fitPoints;

double fitTolerance;

Adesk::Boolean tangentsExist;

AcGeVector3d startTangent;

AcGeVector3d endTangent;

pSL->getFitData(fitPoints,degree,fitTolerance,tangentsExist,startTangent,endTangent);

tol.setEqualPoint(fitTolerance);

if (tangentsExist==Adesk::kTrue)

pNurb->setFitData(fitPoints,startTangent,endTangent,tol);

else

pNurb->setFitData(degree,fitPoints,tol);

}

pGeCurve=(AcGeCurve3d \*)pNurb;

}

}

else if ((pDbCurve->isKindOf(AcDb2dPolyline::desc()))||

(pDbCurve->isKindOf(AcDbPolyline::desc())))

{

int type=0;

AcDbPolyline \*pPoly;

if (pDbCurve->isKindOf(AcDb2dPolyline::desc()))

{

AcDb2dPolyline \*p2L=(AcDb2dPolyline \*)pDbCurve;

XdDbUtils::Poly2dToLWPoly(p2L,pPoly);

type=1;

}

else

pPoly=(AcDbPolyline \*)pDbCurve;

XdDbUtils::convertPolylineToGeCurve(pPoly,pGeCurve);

if (type)

delete pPoly;

}

return (pGeCurve)?Acad::eOk:Acad::eNotImplemented;

}

<!--[if !supportLineBreakNewLine]-->

<!--[endif]-->

看看几何库的：AcGeCompositeCurve2d ，AcGeCompositeCurve3d 类，专门处理大量几何实体，得到首尾相连的几何实体的。（收获不少）

AcGeCurve2d转换到AcDbPolyline的代码，其中有AcGeCompositeCurve2d，AcGeCompositeCurve3d几何实体的转换。

//转换AcGeCurve2d到AcDbPolyline

Acad::ErrorStatus XdDbUtils::convertGeCurveToPolyline(AcGeCurve2d\* pCurve, AcDbPolyline\*& pResultPoly)

{

AcGeVoidPointerArray resultCurves;

AcGeCompositeCurve2d\* pResultCurve;

AcGeCurve2d\* pThisCurve;

AcGeCircArc2d\* pArc;

AcGeLineSeg2d\* pLine;

AcGePoint2d endPt;

int nCurves;

double bulge, ang;

if(pCurve->isKindOf(AcGe::kCompositeCrv2d))

{

pResultCurve = (AcGeCompositeCurve2d\*)pCurve;

pResultCurve->getCurveList(resultCurves );

}

else

{

resultCurves.append(pCurve);

}

nCurves = resultCurves.length();

pResultPoly = new AcDbPolyline(nCurves);

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

{

pThisCurve = (AcGeCurve2d\*)(resultCurves[i]);

if(pThisCurve->isKindOf(AcGe::kCircArc2d))

{

pArc = (AcGeCircArc2d\*)pThisCurve;

bulge = 0.0;

ang = 0.25 \* (pArc->endAng() - pArc->startAng());

bulge = tan(ang);

if(pArc->isClockWise())

{

bulge = -bulge;

}

pResultPoly->addVertexAt(i, pArc->startPoint(), bulge

);

}

else if(pThisCurve->isKindOf( AcGe::kLineSeg2d))

{

pLine = (AcGeLineSeg2d\*)pThisCurve;

pResultPoly->addVertexAt(i, pLine->startPoint(), 0 );

}

}// for

if(pThisCurve->hasEndPoint(endPt))

{

pResultPoly->addVertexAt(i, endPt, 0);

}

pResultPoly->setClosed(pCurve->isClosed());

return Acad::eOk;

}

AcDbRegion 炸开后是Curves 包括(splines, lines, arcs, circles),

可通过AcGeCompositeCurve2d来获得PolyLineAcDbline => AcGeLineSeg2d

AcDbArc => AcGeCircArc2d

AcDbCircle => AcGeCircArc2d

请问如何转换AcDbSpline到对应的AcGe类?

AcDbSpline => AcGeSplineEnt3d/AcGeSplineEnt2d不能直接使用

AcGeSplineEnt3d,2d是个抽象几何实体基类，本身不能实例对象，它下面还有派生类，用来实例具体的对象：AcGeNurbCurve3d，AcGeNurbCurve2d

参考这个

// convert AcDbLine to AcGeLineSeg3d

AcGeLineSeg3d\* LineDb2GE(AcDbLine\* pDbLine)

{

return(new AcGeLineSeg3d(pDbLine->startPoint(), pDbLine->endPoint()));

}

// convert AcDbArc to AcGeCircArc3d

AcGeCircArc3d\* ArcDb2Ge( AcDbArc\* pDbArc)

{

return(new AcGeCircArc3d(

pDbArc->center(),

pDbArc->normal(),

pDbArc->normal().perpVector(),

pDbArc->radius(),

pDbArc->startAngle(),

pDbArc->endAngle()));

}

// convert AcDbCircle to AcGeCircArc3d

AcGeCircArc3d\* CircleDb2Ge(AcDbCircle\* pDbCircle)

{

return(new AcGeCircArc3d(

pDbCircle->center(),

pDbCircle->normal(),

pDbCircle->radius()));

}

// convert AcDbSpline to AcGeNurbCurve3d

AcGeNurbCurve3d\* SplineDb2Ge(AcDbSpline\* pDbSpline)

{

AcGeNurbCurve3d\* pGeSpline;

AcGePoint3dArray fitPoints;

int degree;

double fitTolerance;

Adesk::Boolean tangentsExist;

AcGeVector3d startTangent, endTangent;

AcGeTol tol;

Adesk::Boolean rational, closed, periodic;

AcGePoint3dArray controlPoints;

AcGeDoubleArray knots, weights;

double controlPtTol, knotTol;

if (pDbSpline->hasFitData()) {

pDbSpline->getFitData(fitPoints, degree, fitTolerance,

tangentsExist,startTangent, endTangent);

tol.setEqualPoint(fitTolerance);

pGeSpline=new AcGeNurbCurve3d(fitPoints, startTangent,

endTangent, tangentsExist, tangentsExist,tol);

}else{

pDbSpline->getNurbsData(degree, rational, closed, periodic,

controlPoints, knots, weights, controlPtTol, knotTol);

pGeSpline=new AcGeNurbCurve3d(degree, knots, controlPoints,

weights, periodic);

if (closed==Adesk::kTrue)

pGeSpline->makeClosed();

};

return(pGeSpline);

}

// convert AcDbEllipse to AcGeEllipArc3d

AcGeEllipArc3d\* EllipseDb2Ge(AcDbEllipse\* pDbEllise)

{

return(new AcGeEllipArc3d(

pDbEllise->center(),

pDbEllise->majorAxis(),

pDbEllise->minorAxis(),

pDbEllise->majorAxis().length(),

pDbEllise->minorAxis().length(),

pDbEllise->startAngle(),

pDbEllise->endAngle()));

}

//转换AcGeCompositeCurve3d到AcDbPolyline, This routine only called by GetRegionBoundaryPolyline

AcDbPolyline\* convertGeCurveToPolyline(AcGeCompositeCurve3d\* pCurve)

{

AcGeVoidPointerArray resultCurves;

AcDbPolyline\* pResultPolyline;

AcGeCurve3d\* pThisCurve;

AcGeCircArc3d\* pArc;

AcGeLineSeg3d\* pLine;

AcGePoint3d startPt,endPt;

int nCurves,i,j;

double bulge, ang;

pCurve->getCurveList(resultCurves );

bool bCannotConvert=false;

nCurves = resultCurves.length();

for(i=0;i< nCurves;i++){

pThisCurve = (AcGeCurve3d\*)(resultCurves[i]);

if (pThisCurve->isKindOf(AcGe::kSplineEnt3d) || pThisCurve->isKindOf(AcGe::kEllipArc3d)){

bCannotConvert=true;

break;

}

};

if (bCannotConvert) {

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

delete (AcGeCurve3d\*)(resultCurves[i]);

acedPrompt("\nCon't Convert to Polyline.");

return(NULL);

};

AcGeIntArray isArcs;

AcGePoint3dArray Vertexes;//存放每一线段的起点和终点

AcGeDoubleArray bulges;

for(i=0;i< nCurves;i++){

pThisCurve = (AcGeCurve3d\*)(resultCurves[i]);

if(pThisCurve->isKindOf(AcGe::kCircArc3d)){

pArc = (AcGeCircArc3d\*)pThisCurve;

isArcs.append(1);

Vertexes.append(pArc->startPoint());

Vertexes.append(pArc->endPoint());

ang = 0.25 \* (pArc->endAng() - pArc->startAng());

bulge = tan(ang);

bulges.append(bulge);

}

else if(pThisCurve->isKindOf( AcGe::kLineSeg3d)){

pLine = (AcGeLineSeg3d\*)pThisCurve;

isArcs.append(0);

Vertexes.append(pLine->startPoint());

Vertexes.append(pLine->endPoint());

bulges.append(0.0);

}

//else nothing, This routine only called by GetRegionBoundaryPolyline

delete pThisCurve;//Ge对象不再有用，删掉

}

j=-1;

for (i=0;i< nCurves;i++){

if (isArcs[i]==0){//找到第一条直线

j=i;

break;

}

}

pResultPolyline = new AcDbPolyline(nCurves);

bool bClockWise=false;

if (j==-1) {//polyline全部由arc构成

if (Vertexes[0]==Vertexes[3]) bClockWise=true;

if (bClockWise){

for(i=0;nCurves;i++)

pResultPolyline->addVertexAt(i, AcGePoint2d(Vertexes[2\*i+1].x,Vertexes[2\*i+1].y), -bulges[i]);

}else{

for(i=0;nCurves;i++)

pResultPolyline->addVertexAt(i, AcGePoint2d(Vertexes[2\*i].x,Vertexes[2\*i].y), bulges[i]);

}

}else{

for(i=j+1;i< nCurves;i++){

if ((isArcs[i]==1)&&(Vertexes[2\*i]!=Vertexes[2\*i-1])){

//当前圆弧的起点不等于上一线段的终点

startPt=Vertexes[2\*i+1];

endPt=Vertexes[2\*i];

Vertexes[2\*i+1]=endPt;

Vertexes[2\*i]=startPt;

bulges[i]=-bulges[i];

}

};

for(i=j-1;i >=0;i--){

if ((isArcs[i]==1)&&(Vertexes[2\*i+1]!=Vertexes[2\*(i+1)])){

//当前圆弧的终点不等于下一线段的起点

startPt=Vertexes[2\*i+1];

endPt=Vertexes[2\*i];

Vertexes[2\*i+1]=endPt;

Vertexes[2\*i]=startPt;

bulges[i]=-bulges[i];

}

};

for(i=0;nCurves;i++)

pResultPolyline->addVertexAt(i, AcGePoint2d(Vertexes[2\*i].x,Vertexes[2\*i].y), bulges[i]);

}

pResultPolyline->close();

return(pResultPolyline);

}

//获取Region的边界PolyLines/Circles/Ellipses/Splines, 返回环的数目

int GetRegionBoundaryPolyline(AcDbRegion \*pRegion, AcDbVoidPtrArray\*& pPolylines)

{

AcDbVoidPtrArray subEntityArray;

AcGeVoidPointerArray tmpGeCurves;

int i, count=0;;

if (pRegion->explode(subEntityArray)!=Acad::eOk){

for (i=0;i< subEntityArray.length();i++)

delete (AcDbObject\*)subEntityArray[i];//To XDSoft: 需要手工删除

return 0;

}

AcDbCurve\* pDbCurve;

for (i=0;i< subEntityArray.length();i++){

pDbCurve=(AcDbCurve\*)subEntityArray[i];

if (pDbCurve->isClosed()){

//this curve(Circle/Spline/Ellipse) is closed, then return the boundary(Circle/Spline/Ellipse)

pPolylines->append(pDbCurve);

count++;

}else{

if(pDbCurve->isKindOf(AcDbLine::desc()))

tmpGeCurves.append(LineDb2GE(AcDbLine::cast(pDbCurve)));

else if(pDbCurve->isKindOf(AcDbArc::desc()))

tmpGeCurves.append(ArcDb2Ge(AcDbArc::cast(pDbCurve)));

else if(pDbCurve->isKindOf(AcDbSpline::desc()))

tmpGeCurves.append(AcDbSpline::cast(pDbCurve));

else if(pDbCurve->isKindOf(AcDbEllipse::desc()))

tmpGeCurves.append(AcDbEllipse::cast(pDbCurve));

//else I don't know

delete pDbCurve;

}

};

AcGeIntArray isOwnerOfCurves;

AcGeCompositeCurve3d\* pGeCompositeCurve;

AcDbPolyline \*pPloyline;

while (tmpGeCurves.length()>0){

isOwnerOfCurves.setLogicalLength(0);

for (i=0;i< tmpGeCurves.length();i++)

isOwnerOfCurves.append(1);

pGeCompositeCurve=new AcGeCompositeCurve3d(tmpGeCurves,isOwnerOfCurves);

if (pGeCompositeCurve==NULL){

for (i=0;i< tmpGeCurves.length();i++)

delete (AcGeCurve3d\*)tmpGeCurves[i];

count=-count;//负数表示有部分边界被获得，但出错

break;

}

pPloyline=convertGeCurveToPolyline(pGeCompositeCurve);

delete pGeCompositeCurve;//连带删除tmpGecurve

if (pPloyline!=NULL){

pPolylines->append(pPloyline);

count++;

}

}

return(count);

}