Topology and Geometry in OpenCascade-Edge

eryar@163.com

摘要 Abstract：本文简要介绍了几何造型中的边界表示法（BRep），并结合程序说明 OpenCascade 中的边界表示的具体实现，即拓朴与几何的联系。对具有几何信息的拓朴结构顶点（vertex）、边（edge）、面（face）进行了详细说明。本文只对拓朴边数据进行说明，着重介绍了 OpenCascade 中两个种特别的边缝合边（seam edge）和退化边（degenerated edge）。

关键字 Key Words：OpenCascade、BRep、Topology、Edge、Geometry

一、引言 Introduction

边（Edge）是两个邻面（对正则形体而言）或多个邻面（对非正则形体而言）的交集。边有方向，它由起始顶点和终止顶点来界定；边的形状由边的几何信息来表示，可以是直线，也可以是曲线，曲线边可用一系列控制点或型值点来描述，也可以用显示、隐式或参数方程来描述。

边（Edge）是边界表示法中的重要结构，因为边界表示法（BRep）是用形体的边界来描述形体的一种方法。BRep 认为形体是由有限数量的边界表面（平面或曲面）构成，而每个表面又由若干边界边与顶点构成，所有的单元面构成了形体的边界，形体的边界将形体和周围的环境分隔开来。

边界表示法不仅详细记录了构成形体的面、边方程的系数和顶点坐标值的几何信息，而且描述了这些几何元素之间的拓朴信息，即体、面、边、顶点的组成关系等。在保证对形体面的定义确定并且无二义性的前提下，它允许根据形体的拓朴结构、面表示的方便性等因素确定一个面是以一个整体表示，还是以几个部分之和进行表示。

在 OpenCascade 中边包含了一系列的曲线，其结构如下图所示：

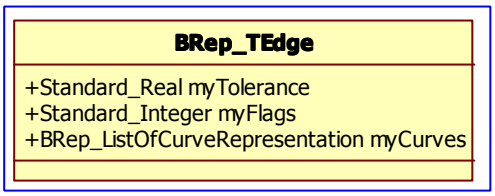

Figure 1.1 BRep_TEdge members

其中，包含一系列的曲线由下面几种：

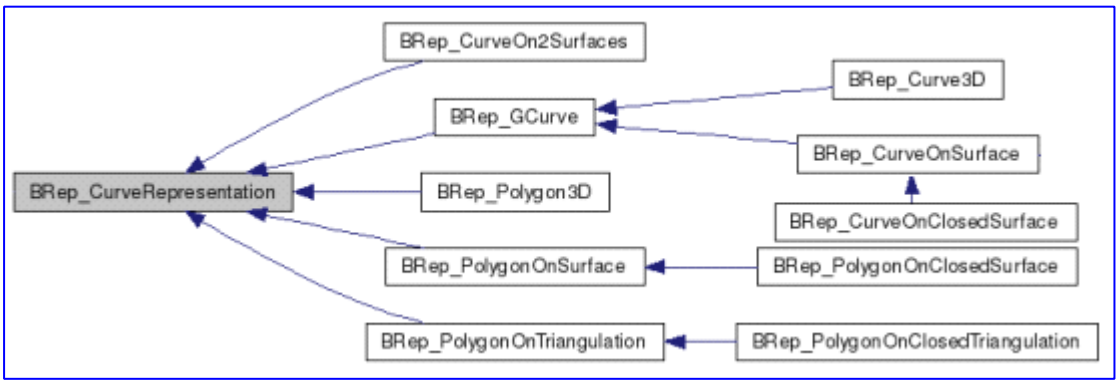

Figure 1.2 BRep_CurveRepresentation class diagram

二、边 Edge

边（edge）是对应于一维对象－曲线的拓朴实体。边可以是面的边界（如长方体的 12 条边之一）；也可以只是一条不属于任何面的“悬空”边（floating edge），想像一下在构建锥形体或扫掠体之前的轮廓线；面的边可以被两个或更多面共享，或者只属于一个面。如下图所示：

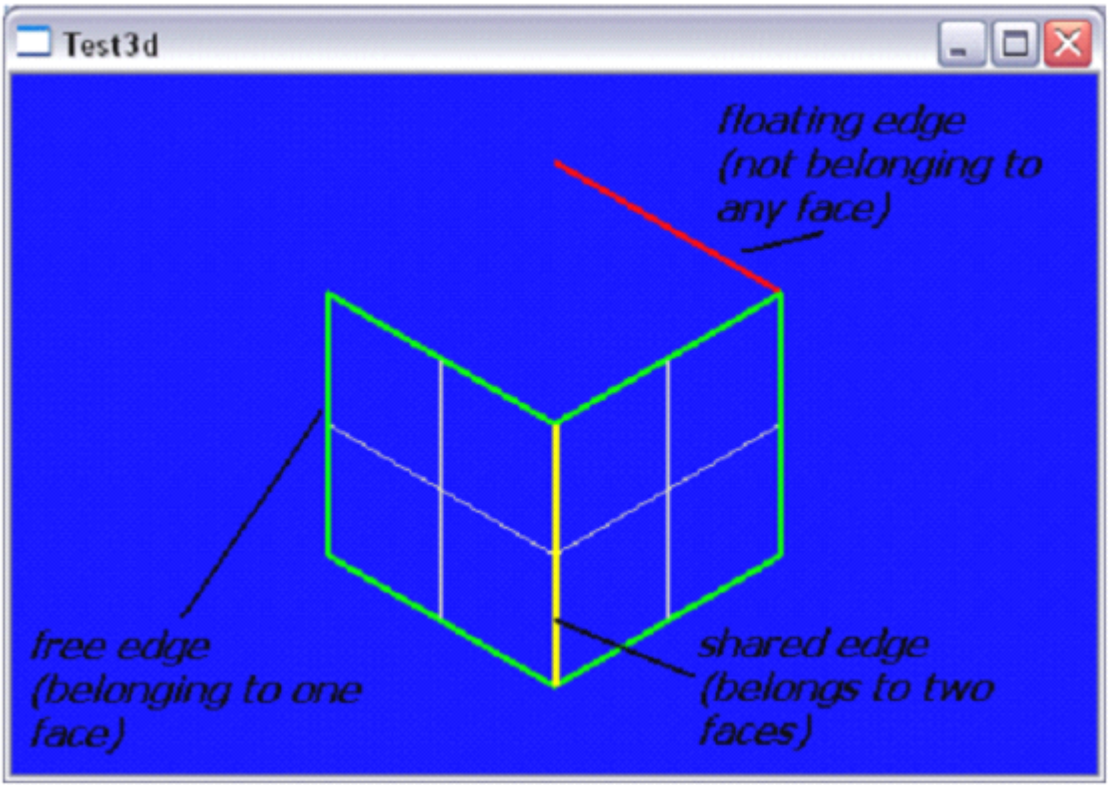

Figure 2.1 Model used to illustrate Edge

在上图中用不同的颜色将不同类型的边区别开来：

- 红色：不属于任何面的悬空边（floating edge）；
- 绿色：只属于一个面的自由边（free edge）；
- 黄色：属于两个或多个面的共享边（shared edge）；

边 Edge 包含几种几何表示：

- 三维空间中的曲线 C（t），由 Geom_Curve 实现。这是边的基本表示方式；
- 曲线 P（t）为二维空间的参数曲线，用来表示属于面的边，通常被称为 pcurves，由类 Geom2d_Curve 实现；
- 多段线（Polygonal）由一组三维点表示，且由类 Poly_Polygon3D 实现。
- 多段线（Polygonal）也可由一组三角剖分面上点的索引来表示，且由类 Poly_PolygonOnTriangulation 实现。

他们的表示都可以使用前面提到的类 BRep_Tool 来获取。例如：

```
Standard_Real aFirst, aLast, aPFirst, aPLast;
Handle(Geom_Curve) aCurve3d = BRep_Tool::Curve (anEdge, aFirst,
aLast);
Handle(Geom2d_Curve) aPCurve = BRep_Tool::CurveOnSurface (anEdge,
aFace, aPFirst, aPLast);
```

边必须有曲面上的参数曲线（pcurves），除了平面以外。边中所有曲线必须一致，即朝向相同。这样边上的点可以使用任意表示方式计算得到，如曲线 C t）可以用[first, last]

区间上的 t 来计算；也可根据 u 在区间[first1, last1]上取得曲面 S1（P1x(u), P1y(u)）上的点 Pi，这里 Pi 是曲面 Si 上的参数曲线 pcurve 上的一点。

1. 边的标志位 Edge Flags

边中的标志位有三种：

```
static const Standard_Integer ParameterMask       = 1;
static const Standard_Integer RangeMask           = 2;
static const Standard_Integer DegeneratedMask     = 4;
```

这里对前两种标志位进行说明：

- RangeMask “same range”：（BRep_Tool::SameRange()）取值区间相同，即几何表示的曲线参数取值区间相同；
- ParameterMask “same parameter”：（BRep_Tool::SameParameter()）参数相同，即当 C（t）＝S1(P1x(t), P1y(t))时，对于同样的参数 t，C（t）和曲面 S1 上的点 P1(t)相同。即任何边上的点都对应参数曲线上相同的参数值。

许多算法假定设置了这两个标志位，因此建议你注意这种情况，一定要设置这些标志位。

2. 边的容差 Tolerance

边的容差（Tolerance）是其三维曲线和其他任何表示方式之间的最大偏差。其几何意义就是以容差为半径沿边的一个包含边的三维曲线及其他任何形式表示的管子。如下图所示：

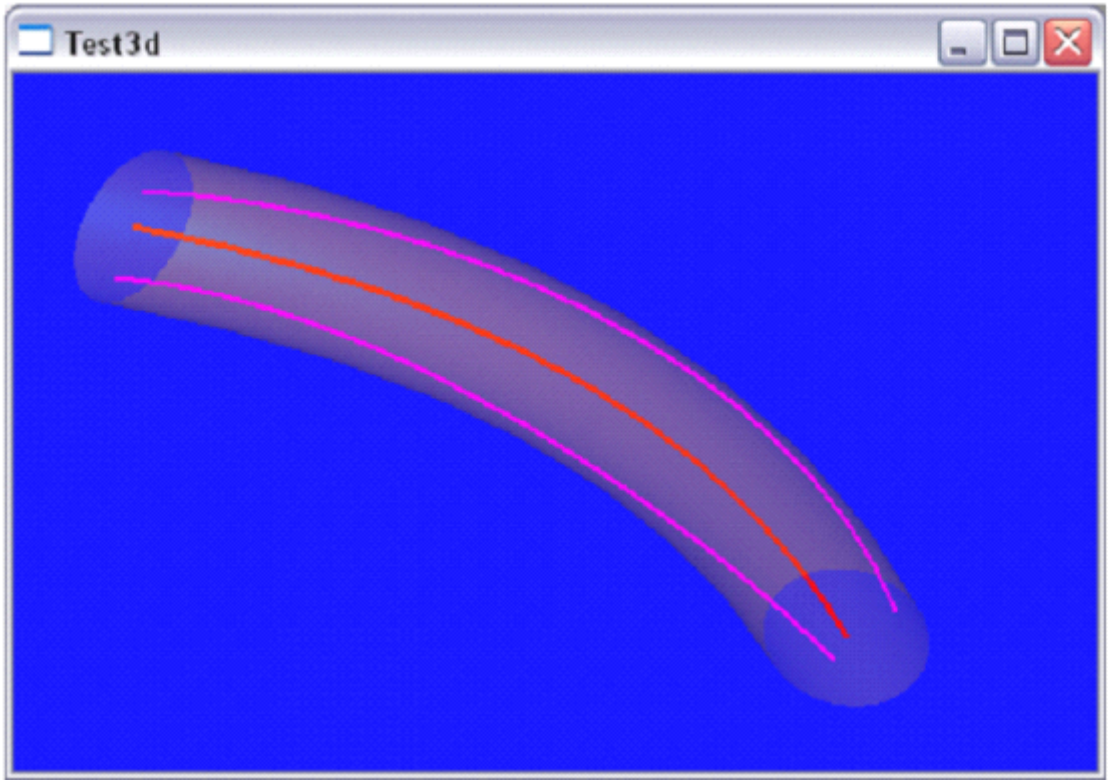

Figure 2.2 Edge Tolerance

3. 特殊类型的边 Special edge types

在 OpenCascade 有两种特别类型的边，他们是：

- 缝合边（seam edge）：即在同一个面上出现两次的边（如：在同一个面上具有 2 个参数曲线）；
- 退化边（degenerated edge）：这种边位于曲面的奇异点处，在三维空间中退化为一个点；

球面中这两种类型的边都有。缝合边位于经线（U iso-lines），参数为 0 和 2*PI。退化边位于南北极点，对应于纬线（V iso-lines），参数为－PI/2 和 PI/2。因为球面的参数方程为：

$$S(u,v)=P+r\cdot\cos(v)\cdot(\cos(u)\cdot D_X+\sin(u)\cdot D_Y)+r\cdot\sin(v)\cdot D_Z,\ (u,v)\in[0,2\cdot\pi)\times[-\pi/2,\pi/2].$$

当参数 u 取 0 和 2*PI 时，球面的参数方程计算如下：

$$S(0,v)=P+r\cdot\cos(v)\cdot(\cos(0)\cdot Dx+\sin(0)\cdot Dy)+r\cdot\sin(v)\cdot Dz$$
$$S(2\pi,v)=P+r\cdot\cos(v)\cdot(\cos(2\pi)\cdot Dx+\sin(2\pi)\cdot Dy)+r\cdot\sin(v)\cdot Dz$$
$$S(0,v)=P+r\cdot\cos(v)\cdot Dx+r\cdot\sin(v)\cdot Dz \qquad v\in[-\pi/2,\pi/2]$$
$$S(2\pi,v)=P+r\cdot\cos(v)\cdot Dx+r\cdot\sin(v)\cdot Dz \qquad v\in[-\pi/2,\pi/2]$$

从计算结果可以看出，缝合边是位于 Dx 和 Dz 所在平面上的半圆弧。

当参数 v 取－PI/2 和 PI/2 时，球面的参数方程计算如下：

$$S(u,\pm\frac{\pi}{2})=P+r\cdot\cos(\pm\frac{\pi}{2})\cdot(\cos(u)\cdot Dx+\sin(u)\cdot Dy)+r\cdot\sin(\pm\frac{\pi}{2})\cdot Dz$$
$$S(u,\pm\frac{\pi}{2})=P\pm r\cdot Dz$$

从计算结果可以看出，曲面上的两个边分别退化为两个点。即 v 取-PI/2 和 PI/2 时球面的两个退化边分别位于南北极点上。如下图所示：

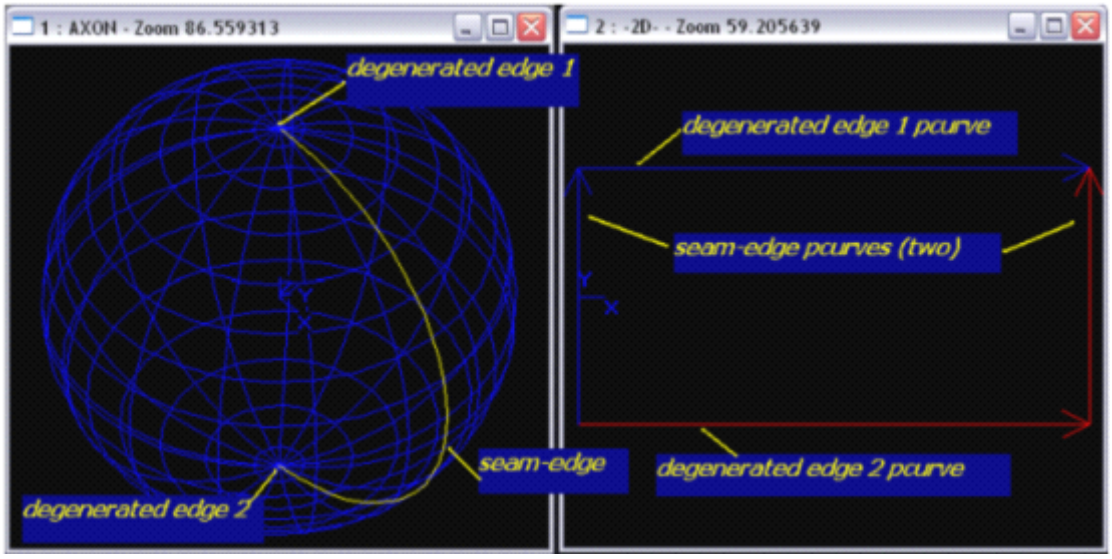

Figure 2.3 seam edge and degenerated edge of sphere

另外例子：环形体（torus）、圆柱体（cylinder）、圆锥体（cone）。环形体 torus 有两条缝合边（seam edge），对应于它的参数空间的边界；圆柱体（cylinder）有一条缝合边（seam edge）。圆锥（cone）顶点处为退化边（degenerated edge）。

检查边是否是缝化边或退化边，可以使用函数 BRep_Tool::IsClosed() 和 BRep_Tool::Degenerated()。

4. 边的朝向

边的朝向为正向（forward edge orientation）意味着边的逻辑方向与曲线的方向相同。反向（reversed）意味着逻辑方向与曲线方向相反。所以，缝合边（seam-edge）在一个面中总是有两个朝向：一个反向(reversed)，一个正向（forward）。

三、示例程序 Example Code

以边界表示 BRep 表示的球面为例，说明其边的类型。程序代码如下所示：

```
/*
*    Copyright (c) 2013 eryar All Rights Reserved.
*
*        File    : Main.cpp
*        Author  : eryar@163.com
*        Date    : 2013-08-24 16:11
*        Version : 1.0v
*
*    Description : Demonstrate seam edge and degenerated edge of sphere.
*
*/

#include <iostream>

// OpenCascade Library.
#define WNT
#include <TopoDS.hxx>
#include <TopoDS_Edge.hxx>
#include <TopExp_Explorer.hxx>
#include <BRepPrimAPI_MakeSphere.hxx>

#pragma comment(lib, "TKernel.lib")
#pragma comment(lib, "TKMath.lib")
#pragma comment(lib, "TKBRep.lib")
#pragma comment(lib, "TKPrim.lib")
#pragma comment(lib, "TKTopAlgo.lib")

/**
* @breif Dump orientation types.
*        Orientation definitaion:
*        enum TopAbs_Orientation {

*            TopAbs_FORWARD,

*            TopAbs_REVERSED,

*            TopAbs_INTERNAL,

*            TopAbs_EXTERNAL

*        };
*/
std::string dumpOrientation(const TopAbs_Orientation& orient)
{
    std::string strType;

    switch (orient)
    {
    case TopAbs_FORWARD:
        strType = "TopAbs_FORWARD";
        break;

    case TopAbs_REVERSED:
        strType = "TopAbs_REVERSED";
        break;

    case TopAbs_INTERNAL:
```

```
        strType = "TopAbs_INTERNAL";
        break;

    case TopAbs_EXTERNAL:
        strType = "TopAbs_EXTERNAL";
        break;
    }

    return strType;
}

/**
* @breif Dump edge information.
*/
void processEdge(const TopoDS_Edge& edge, const TopoDS_Face& face)
{
    Standard_Real dTolerance = BRep_Tool::Tolerance(edge);

    Standard_Boolean bIsGeometric = BRep_Tool::IsGeometric(edge);
    Standard_Boolean bIsSameParameter = BRep_Tool::SameParameter(edge);
    Standard_Boolean bIsSameRange = BRep_Tool::SameRange(edge);
    Standard_Boolean bIsDegenerated = BRep_Tool::Degenerated(edge);
    Standard_Boolean bIsClosed = BRep_Tool::IsClosed(edge, face);

    TopAbs_Orientation nOrientation = edge.Orientation();

    // Dump edge info.
    std::cout<<"====== Edge Info ======="<<std::endl;
    std::cout<<"Tolerance: "<<dTolerance<<std::endl;
    std::cout<<"Orientation: "<<dumpOrientation(nOrientation)<<std::endl;
    std::cout<<"Geometric: "<<(bIsGeometric?"True":"False")<<std::endl;
    std::cout<<"Same Parameter: "<<(bIsSameParameter?"True":"False")<<std::endl;
    std::cout<<"Same Range: "<<(bIsSameRange? "True":"False")<<std::endl;
    std::cout<<"Degenerated edge: "<<(bIsDegenerated?"True":"False")<<std::endl;
    std::cout<<"Seam edge: "<<(bIsClosed? "True":"False")<<std::endl;

    // Dump vertex of the edge.
    for (TopExp_Explorer vertexItr(edge, TopAbs_VERTEX);
        vertexItr.More();
        vertexItr.Next())
    {
        const TopoDS_Vertex& aVertex = TopoDS::Vertex(vertexItr.Current());
        gp_Pnt pnt = BRep_Tool::Pnt(aVertex);

        std::cout<<"Vertex: ("<<pnt.X()<<", "<<pnt.Y()<<", "<<pnt.Z()<<")"<<std::endl;
    }
}

int main(void)
{
    Standard_Integer nSphereFaceCount = 0;
    Standard_Integer nSphereEdgeCount = 0;

    TopoDS_Shape sphere = BRepPrimAPI_MakeSphere(1.0);

    for (TopExp_Explorer faceItr(sphere, TopAbs_FACE);
        faceItr.More();
        faceItr.Next())
    {
```

```
        const TopoDS_Face& aFace = TopoDS::Face(faceItr.Current());

        ++nSphereFaceCount;

        for (TopExp_Explorer edgeItr(aFace, TopAbs_EDGE);
            edgeItr.More();
            edgeItr.Next())
        {
            const TopoDS_Edge& aEdge = TopoDS::Edge(edgeItr.Current());

            processEdge(aEdge, aFace);

            ++nSphereEdgeCount;
        }
    }

    std::cout<<"Sphere face count: "<<nSphereFaceCount<<std::endl;
    std::cout<<"Sphere edge count: "<<nSphereEdgeCount<<std::endl;

    return 0;
}
```

程序运行结果如下所示：

```
====== Edge Info =======
Tolerance: 1e-007
Orientation: TopAbs_REVERSED
Geometric: True
Same Parameter: True
Same Range: True
Degenerated edge: True
Seam edge: False
Vertex: (6.12323e-017, -1.49976e-032, 1)
Vertex: (6.12323e-017, -1.49976e-032, 1)
====== Edge Info =======
Tolerance: 1e-007
Orientation: TopAbs_FORWARD
Geometric: True
Same Parameter: True
Same Range: True
Degenerated edge: False
Seam edge: True
Vertex: (6.12323e-017, -1.49976e-032, 1)
Vertex: (6.12323e-017, -1.49976e-032, -1)
====== Edge Info =======
Tolerance: 1e-007
Orientation: TopAbs_FORWARD
Geometric: True
Same Parameter: True
Same Range: True
Degenerated edge: True
Seam edge: False
```

```
Vertex: (6.12323e-017, -1.49976e-032, -1)
Vertex: (6.12323e-017, -1.49976e-032, -1)
====== Edge Info =======
Tolerance: 1e-007
Orientation: TopAbs_REVERSED
Geometric: True
Same Parameter: True
Same Range: True
Degenerated edge: False
Seam edge: True
Vertex: (6.12323e-017, -1.49976e-032, 1)
Vertex: (6.12323e-017, -1.49976e-032, -1)
Sphere face count: 1
Sphere edge count: 4
Press any key to continue . . .
```

从运行结果可以看，当球的边为退化边时，边的两个顶点的坐标值相同。退化边位于球的南北极点上。缝合边为连接两个退化边的曲线。

根据遍历顺序，

第一条边为退化边（degenerated edge），其朝向为反向（reversed）；

第二条边为缝合边（seam-edge），其朝向为正向（forward）；

第三条边为退化边，其朝向为正向（forward）；

第四条边为缝合边，其朝向为反向（reversed）。

由上可见，缝合边有两个朝向，一个正向一个反向。

四、结论 Conclusion

对与几何相关的拓朴边（edge）的类的属性数据进行详细说明。并结合程序代码详细说明边的标志位（myFlags）属性的意义，从参数方程出发，理解缝合边（seam-edge）和退化边（degenerated edge），即标志位中 DegeneratedMask 的意义。

五、参考资料

1. Roman Lygin, OpenCascade notes, opencascade.blogspot.com
2. 孙家广等. 计算机图形学. 清华大学出版社
3. OpenCascade source code.