OpenCASCADE Conic to BSpline Curves-Hyperbola

eryar@163.com

Abstract. Rational Bezier Curve can represent conic curves such as circle, ellipse, hyperbola, .etc. But how to convert a conic curve to BSpline curve is still question, i.e. Represent a conic curve in BSpline form. The key point of Hyperbola conversion is to calculate the 2nd pole and its weight factor. The paper focus on the hyperbola convert to the BSpline curves.

Key Words. OpenCASCADE, Convert, Hyperbola, BSplineCurve, Conic Curve

1. Introduction

圆锥截线(Conic 或称为二次曲线)和圆在 CAD/CAM 中有着广泛应用。毫无疑问 NURBS 的一个最大优点就是既能精确表示圆锥截线和圆，也能精确表示自由曲线曲面。这个优点的意义是方便编程，使所有的曲线可以采用统一的数据结构来表示。通过有理的方式可以精确来表示这些二次曲线，那么给定一个二次曲线的相关参数（如圆的圆心和半径等），如何构造出对应的 NURBS 曲线呢？

圆锥截线（Conic curves）是一个平面与一个圆锥相交产生的曲线集合。平面与圆锥相交的角度不同产生不同的截线。如下图所示：

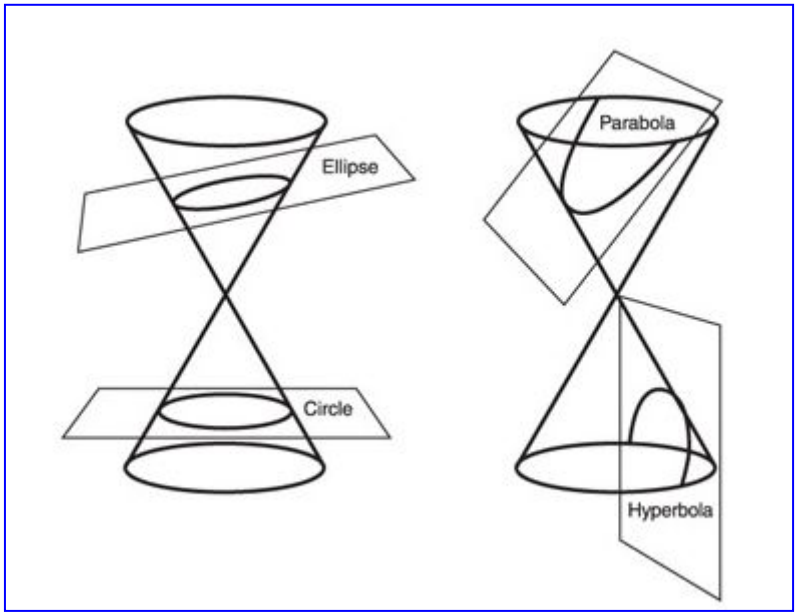

Figure 1.1 Conic Sections

OpenCASCADE 中对应双曲线的隐式方程表示的类是 gp_Hypr/gp_Hypr2d。本文主要介绍 OpenCASCADE 中如何使用包 Convert 将 gp_Parab2d 转换为 NURBS 曲线。

2. Parametric Representations

在 CAD/CAM 的应用中，圆锥截线有两种重要的参数表示形式：有理形式和最大内接面积形式（Rational and maximum inscribed area forms）。表示双曲线的最大内接面积形式，如下所示：

$$\begin{cases} x(u) = achu \\ y(u) = bshu \end{cases} \qquad -\infty < u < \infty$$

其中 chu 和 shu 为双曲函数：

$$shu = \frac{e^{u} - e^{-u}}{2} \qquad chu = \frac{e^{u} + e^{-u}}{2}$$

圆锥截线的有些有理参数表示形式可能是有相当差的参数化，即均匀分布的参数值对应于曲线上分布很不均匀的点。利用线性有理函数对有理曲线进行重新参数化可以改变（因而可能改善）其参数化。

假设 C(u)=(x(u), y(u))是一条在标准位置的圆锥截线的参数表示。现在我们对双曲线给出的参数方程也是上式，它是一个好的参数化：对于任意给定的整数 n 和参数边界 a 与 b，取 n 个等间隔分布的参数：

$$a = u_1, \cdots, u_n = b \qquad u_{i+1} - u_i = const, i = 1,2, \cdots, n-1$$

点 C(u1)，C(u2), ..., C(un)形成曲线上 n-1 边多边形，它的闭合多边形具有最大的内接面积。

3. Conversion Algorithm

将隐式表示的双曲线方程转换为NURBS（有理Bezier是NURBS的特例）曲线需要确定NURBS的以下信息：节点矢量，权因子，次数，控制顶点。

圆锥截线是二次曲线，所以次数为2。根据参数方程的最大内接面积表示法可以求出节点矢量。所以转换的关键是计算控制第二个顶点及其权因子。由有理Bezier曲线的公式得二次有理Bezier曲线弧的表示形式为：

$$C(u)=\frac{(1-u)^2\omega_0 P_0+2u(1-u)\omega_1 P_1+u^2\omega_2 P_2}{(1-u)^2\omega_0+2u(1-u)\omega_1+u^2\omega_2} \qquad u\in[0,1]$$

称k为形状不变因子，公式如下所示：

$$k=\frac{\omega_0\omega_2}{\omega_1^2}$$

可以证明同一组控制顶点选取不同 的权因子，只要形状因子k相等，则由它们决定的二次有理Bezier曲线是同一条曲线段，不同的权因对应不同的参数化，而且可以根据形状不变因子对二次曲线进行分类：

- K=0； 表示退化的二次曲线：一对直线段P0P1和P1P2；
- K∈[0，1]； 表示双曲线；
- K=1； 表示抛物线；
- K∈[1, +∞]； 表示椭圆；
- K=+∞； 表示连接P0和P2的直线段；

习惯上我们选择$\omega_0=\omega_2=1$称为标准参数化。此时只剩下控制顶点P1的权因子ω_1。

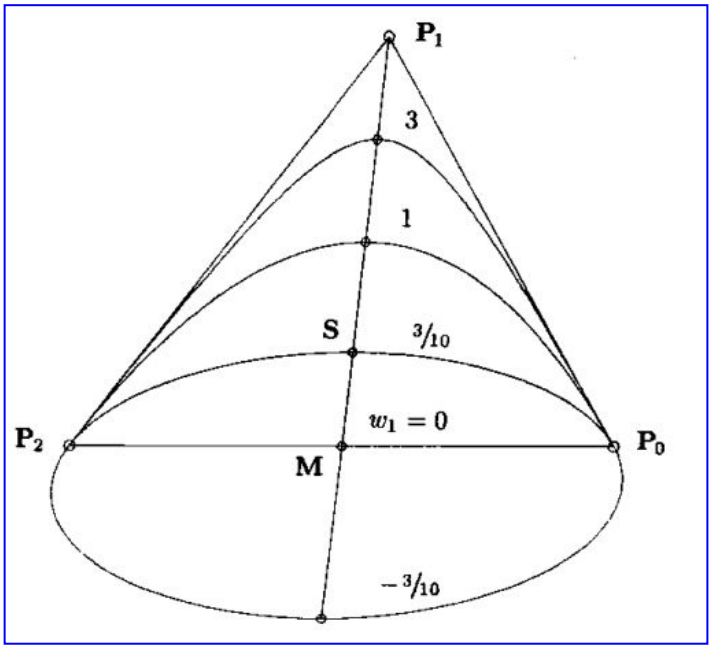

Figure 3.1 不同的权因子ω_1定义的圆锥截线

由二次有理Bezier曲线公式可知，当u=0和u=1时，C(0)=P0, C(1)=P2，即曲线通过特征多边形的首末顶点。由此可确定抛物线的两个控制顶点P0和P2，现在只剩下最后一个

P1 顶点未确定。

由端点处的切矢公式可知，控制多边形通过首末端点且第二个控制顶点 P1 是通过两端点的切线的交点。根据直接线点向式可以列出直线方程来求出交点即 P1 点的坐标。

$$\begin{cases} y-y_0=k_0(x-x_0) \\ y-y_2=k_2(x-x_2) \end{cases} \qquad k=\frac{dy}{dx}=\frac{bchu}{ashu}$$

计算得交点 P1 的坐标如下所示：

$$\begin{cases} x_1=\dfrac{y_2-y_0+k_0x_0-k_2x_2}{k_0-k_2} \\ y_1=\dfrac{k_0y_2-k_2y_0-k_0k_2(x_2-x_0)}{k_0-k_2} \end{cases}$$

根据双曲线的参数方程得：

$$\begin{cases} x_0=achu_0 \\ y_0=bshu_0 \end{cases} \quad \begin{cases} x_2=achu_2 \\ y_2=bshu_2 \end{cases} \quad k_0=\frac{bchu_0}{ashu_0} \quad k_2=\frac{bchu_2}{ashu_2}$$

将上述值代入交点坐标公式得交点 P1 的坐标与参数 u 的关系式为：

$$\begin{cases} x_1=\dfrac{a(shu_2-shu_0)}{sh(u_2-u_0)} \\ y_1=\dfrac{b(chu_2-chu_0)}{sh(u_2-u_0)} \end{cases}$$

根据肩点公式及点 P1，可计算出权因子的 ω_1 值，公式如下所示：

$$S=\frac{1}{1+\omega_1}M+\frac{\omega_1}{1+\omega_1}P_1$$

求得 P1 点对应的权因子 ω_1 的值为：

$$\omega_1=ch\frac{(u_2-u_0)}{2}$$

至此，双曲线的三个控制顶点 P0，P1，P2 都已计算出来了。即双曲线的 NURBS 表示所需的数据都已经得到了。下面看看 OpenCASCADE 中的实现代码。

4. Code Analysis

OpenCASCADE 的 Math 工具集中有个包 Covert 用来将圆锥曲线曲面转换为 NURBS 曲线曲面。其中转换双曲线的类为：Convert_HyperbolaToBSplineCurve，实现代码如下所示：

```
//=======================================================================
//function : Convert_HyperbolaToBSplineCurve
//purpose  :
//=======================================================================
Convert_HyperbolaToBSplineCurve::Convert_HyperbolaToBSplineCurve
  (const gp_Hypr2d&    H ,
   const Standard_Real U1,
   const Standard_Real U2 )

: Convert_ConicToBSplineCurve (MaxNbPoles, MaxNbKnots, TheDegree)
{
  Standard_DomainError_Raise_if( Abs(U2 - U1) < Epsilon(0.),
                "Convert_ParabolaToBSplineCurve");

  Standard_Real UF = Min (U1, U2);
  Standard_Real UL = Max( U1, U2);

  nbPoles = 3;
  nbKnots = 2;
  isperiodic = Standard_False;
  knots->ChangeArray1()(1) = UF;  mults->ChangeArray1()(1) = 3;
  knots->ChangeArray1()(2) = UL;  mults->ChangeArray1()(2) = 3;

  // construction of hyperbola in the reference xOy.
  Standard_Real R = H.MajorRadius();
  Standard_Real r = H.MinorRadius();
  gp_Dir2d Ox = H.Axis().XDirection();
  gp_Dir2d Oy = H.Axis().YDirection();
  Standard_Real S = ( Ox.X() * Oy.Y() - Ox.Y() * Oy.X() > 0.) ?  1 : -1;

  // poles expressed in the reference mark
  // the 2nd pole is at the intersection of 2 tangents to the curve
  // at points P(UF), P(UL)
  // the weight of this pole is equal to : Cosh((UL-UF)/2)
  weights->ChangeArray1()(1) = 1.;
  weights->ChangeArray1()(2) = Cosh((UL-UF)/2);
  weights->ChangeArray1()(3) = 1.;

  Standard_Real delta = Sinh(UL-UF);
  Standard_Real x =     R * ( Sinh(UL) - Sinh(UF)) / delta;
  Standard_Real y = S * r * ( Cosh(UL) - Cosh(UF)) / delta;
  poles->ChangeArray1()(1) = gp_Pnt2d( R * Cosh(UF), S * r * Sinh(UF));
  poles->ChangeArray1()(2) = gp_Pnt2d( x, y);
  poles->ChangeArray1()(3) = gp_Pnt2d( R * Cosh(UL), S * r * Sinh(UL));

  // replace the bspline in the mark of the hyperbola
```

```
  gp_Trsf2d Trsf;
  Trsf.SetTransformation( H.Axis().XAxis(), gp::OX2d());
  poles->ChangeArray1()(1).Transform( Trsf);
  poles->ChangeArray1()(2).Transform( Trsf);
  poles->ChangeArray1()(3).Transform( Trsf);
}
```

由上面的代码可知，先设置曲线次数为 2，再设置节点矢量为[UF，UF，UF，UL，UL，UL]，即首参数 UF 和末参数 UL 的重数皆为 3，由节点矢量可知转换后的 NURBS 曲线为 Bezier 曲线。（抛出异常的提示信息还没改过来，还是抛物线的。）

设置三个控制顶点及其对应的权因子，计算主要涉及到第二个控制顶点 P1 的权因子。

最后根据有理 Bezier 曲线的仿射不变性：对有理 Bezier 曲线进行旋转、平移和缩放变换，其表达式不变，只是控制点发生了改变。新的控制点可以通过对原控制点作变换得到。即要对有理 Bezier 曲线进行仿射变换，只需对其控制点作变换即可。

圆锥截线的转换类的使用是很简单的，且计算都是在构造函数中完成。下面给出一个将双曲线转换为 NURBS 曲线的具体示例来说明其用法。

```
/*
*    Copyright (c) 2014 eryar All Rights Reserved.
*
*        File    : Main.cpp
*        Author  : eryar@163.com
*        Date    : 2014-10-06 20:46
*        Version : 1.0v
*
*    Description : OpenCASCADE conic to BSpline curve-Hyperbola.
*
*      Key words : OpenCascade, Hyperbola, BSpline Curve, Convert
*/

#define WNT
#include <gp_Hypr2d.hxx>
#include <Convert_HyperbolaToBSplineCurve.hxx>

#pragma comment(lib, "TKernel.lib")
#pragma comment(lib, "TKMath.lib")

void DumpConvertorInfo(const Convert_ConicToBSplineCurve &theConvertor)
{
    std::cout << "Degree: " << theConvertor.Degree() << std::endl;

    std::cout << "Poles/Weights: " << std::endl;
    for (Standard_Integer i = 1; i <= theConvertor.NbPoles(); ++i)
    {
        const gp_Pnt2d &aPole = theConvertor.Pole(i);

        std::cout << i << ": " << aPole.X() << ", " << aPole.Y() << " w(" <<
theConvertor.Weight(i) << ")" << std::endl;
    }

    std::cout << "Knots: " << std::endl;
```

```
    for (Standard_Integer j = 1, m = 0; j <= theConvertor.NbKnots(); ++j)
    {
        for (Standard_Integer k = 1; k <= theConvertor.Multiplicity(j); ++k)
        {
            std::cout << ++m << ": " << theConvertor.Knot(j) << std::endl;
        }
    }

    std::cout << "=========================================" << std::endl;
}

void TestHyperbolaConvert(void)
{
    gp_Hypr2d aHyperbola;

    aHyperbola.SetMajorRadius(2.0);
    aHyperbola.SetMinorRadius(1.0);

    Convert_HyperbolaToBSplineCurve aConvertor(aHyperbola, 1.0, M_PI);

    std::cout << "Convert Hyperbola to BSpline Curve: " << std::endl;
    DumpConvertorInfo(aConvertor);
}

int main(int argc, char* argv[])
{
    TestHyperbolaConvert();

    return 0;
}
```

程序输出结果如下所示：

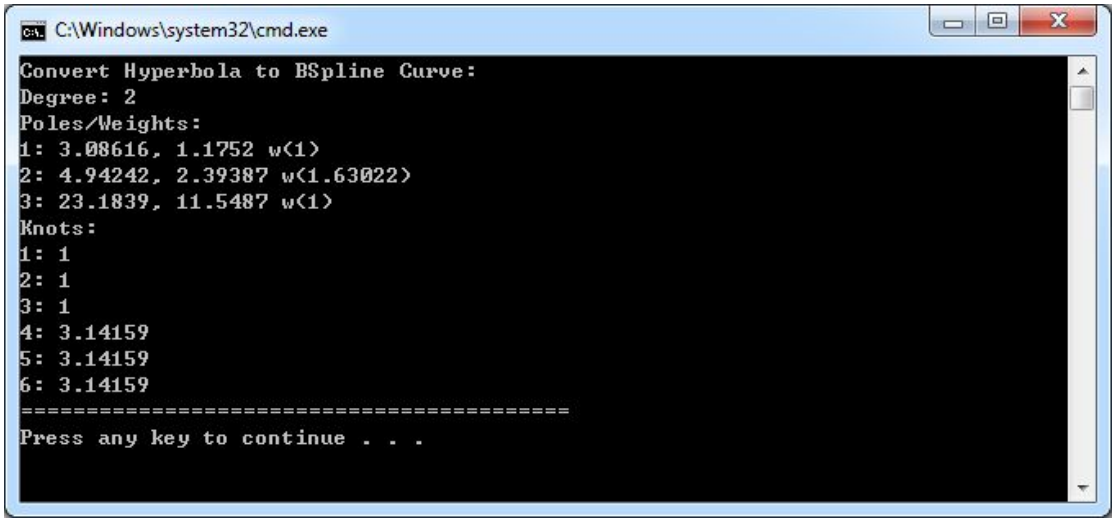

Figure 4.1 Convert Hyperbola to BSpline Curve result

5. Conclusion

NURBS 的一个优势就是统一了曲线曲面的表示方法，即不仅可以表示自由曲线曲面，还可精确表示圆锥曲线曲面。本文详细介绍了 OpenCASCADE 中将双曲线转换为 NURBS 的算法：即根据二次有理 Bezier 曲线的端点性质，求出过两个端点切线的交点来计算出第二个控制顶点 P1 进而计算出对应的权因子。

计算中大量使用到了双曲函数 shx 和 chx 的一些性质，相关公式可参考《数学手册》。

6. References

1. 人民教育出版社中学数学室. 数学第二册（上）. 人民教育出版社. 2000
2. 数学手册编写组. 数学手册. 高等教育出版社. 1979
3. 赵罡，穆国旺，王拉柱译. 非均匀有理 B 样条. 清华大学出版社. 2010
4. 王仁宏，李崇君，朱春钢. 计算几何教程. 科学出版社. 2008