NURBS EDITING SUPPORT

Detailed Description

1. insertion knot
De Boor's algorithm is a generalization of de Casteljau's algorithm

http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/spline/B-spline/de-Boor.html
http://web.mit.edu/hyperbook/Patrikalakis-Maekawa-Cho/node18.html

2. smooth curves
Dual-Loop Algorithm
The first stage aims to generate an estimate of NURBS curve to approximate the given points with
a relatively large tolerance; and predetermine the number of control points as a reference for the
NURBS tool path in the fine fitting
The second stage provides a refined fitting to the tool path, where the objective is to calculate a
NURBS curve representing the cutter locations with a higher accuracy. In the second stage, the
roughly fitted NURBS curve is used to improve the approximation extent.

http://www.scientific.net/AMR.97-101.2477

3 Trimmed to untrimmed curves

Convert a NURBS surface with trimming curve into an approximately equivalent TSpline
with no trimming curve
http://www.tsplines.com/technology/WTN.pdf

4. Contour refinement

A point is on the border of a hole if atleast one of the eight neighbours dont have the same state as it.
Search the contour point that is closest to the point of move, in the parametric plane.
The point obtained will be on the border of the contour and the resulting trimmed surface will have correct shape.
Alternative algo: Compute each point of the hole and connect it to the surface.

http://www.tsplines.com/technology/WTN.pdf

5.Surface Intersection Algorithm

The intersection of two surfaces can be complicated in general, with a
number of closed loops and self-intersections (singularities).
Standard methods include Subdivision Methods,Lattice Evaluation,Analytic Methods,Marching Methods
But there exists an algorithm better than all of these. I shall be using the following algorithm:

http://www.paritycomputing.com/projects/AcmClassification/tr1.1/237751.pdf

Given combinations of NURBS surfaces, the algorithm
decomposes them into Be´zier patches. It uses spatial techniques such as
bounding-box tests and linear programming to reduce the number of
pairwise intersections. Given a pair of Be´zier and/or algebraic surfaces, it
finds a starting point on each component of the intersection curve, decomposes

the domain, marches along the curve choosing appropriate step sizes
to prevent component jumping, and finds the singularities and all branches
at each singularity.

6.Project curves on surface

Several algorithms have been developed which solve these problems. Some depend on special properties of the objects to be projected onto, such as Mortenson (1985), who essentially finds the root of a polynomial using a Newton-Raphson method. Limaien and Trochu (1995) compute the orthogonal projection of a point onto parametric curves and surfaces by constructing an auxiliary function and finding its zeros. Hartmann (1999) proposes a first Preprint submitted to Elsevier Science 14 February 2005order algorithm for foot point computation by using a normalform (again, an auxiliary function) and its first derivatives.

Piegl and Tiller (2001) provide an algorithm for point projection on NURBS surfaces by decomposing a NURBS surface into quadrilaterals, projecting the test point onto the closest quadrilateral, and then recover the parameter from the closest quadrilateral. Ma and Hewitt (2000) present a practical algorithm for computing a good initial value for the Newton-Raphson method.
Apparently there are two key issues in the projection and inversion problems:
— computing a good initial value;
— and using a Newton-type or other iteration to improve the solution.

I will be using the following second order algo for Orthogonal projection onto a curve and Orthogonal projection onto a surface

http://www.geometrie.tugraz.at/wallner/sproj.pdf

8. Project surfaces on plane
I will use the following link for implementing spherical and cylindrical projections on flat surface

http://paulbourke.net/geometry/transformationprojection/

9. Intersection of triangles and planes/ two triangles

http://geomalgorithms.com/a06-_intersect-2.html

PROJECT TIMELINE

1st/2nd/3rd week
Develop a basic understanding of the editing operations (knot insertion etc)
Going through the source code
Discussing with mentor and othe developers

4th week
Start with easy editing operations for curves:
1. Cutting curves
2. Add points
3. Extend
4. Insertion knot

5th week
Start with moderately difficult editing operations for curves:
1. Smooth curve
2. Curve fillet
3. Align curves

6th week
1. Detach curves
2. Open/close curves
3. Rebuild curves
4. Tests and documentation

7th week
Implement logic for easy editing operations for surfaces:
1. Attach surfaces
2. Attach surfaces without movement
3. Align surfaces

8th week
Implement logic for:
1. Open/close surfaces
2. Detach surfaces
3. Project curve on surface
4. Project surface on plane

9th week
Implement logic for:
1. Intersect surfaces
2. Trim tool
3. Insert isoparm

10th week
Implement logic for:
1. Extend surface
2. Offset surface
3. Tests and Documentation

11th week
Implement logic for:
1. Circular fillet
2. Freeform fillet
3. Fillet blend

12th week
1. Fix bugs
2. Improve performance

13th week
Pencils down
1. Code clean up
2. Documentation (wiki pages)

14th week
1. Final evaluation
2. Submit code to Google