Prof. Dr. Hans-Peter Seidel
Dr. Michael Wand
Nils Hasler, Jens Kerber, Thomas Schultz,
Carsten Stoll, Martin Sunkel

Geometric Modeling

Assignment sheet 4 (Bézier Splines and Uniform B-Splines, due May 30th 2008)

---

(1) Bézier Splines [1+3+2 points]

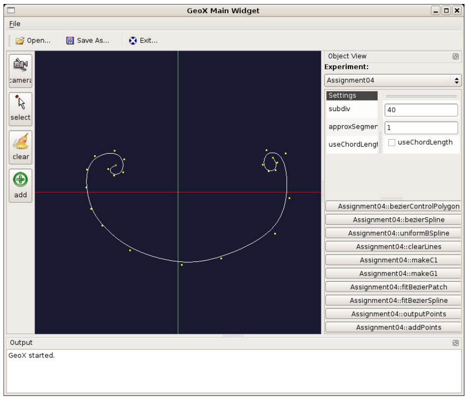
a. Create a new experiment. Use the updated GLGeometryViewer which allows one to add control points interactively. Add a button which draws a control polygon by connecting subsequent user-defined input points. Change the color of those control points that would be interpolated by a cubic Bézier spline to red.
Hint: GeoX uses consecutive point indices, starting from 0, in the order in which points have been added. The number of points is available via the viewer's getNumberOfPoints() method.

b. Add a button which draws the cubic Bézier spline defined by the control polygon in (a). Allow the user to change the number of lines which are used to approximate each curve segment. Please use a different color than for the control polygon.

c. Add a button that changes the control points such that subsequent curve segments are C^1 continuous. Add another button that ensures G^1 continuity, but moves the affected control point as little as possible.

(2) Uniform B-Splines [6 points]

a. Add a button which draws the uniform cubic B-Spline defined by the control polygon in (1a). Again, allow the user to change the number of lines used to approximate the curve. Use duplicate control points such that the first and the last user-defined points are interpolated.



subdiv 40
approxSegmen 1
useChordLengt ☐ useChordLength

GeoX started.

(3) Bézier Splines for Approximation [3+3+2 points]

a. Add a button which finds the control points of a single cubic Bézier curve which approximates the user-defined points in a least squares sense. Draw the resulting curve and its control polygon.
Hint: Assign a unique parameter value t_i to each point $\mathbf{x}_i$ and minimize the squared distances between $\mathbf{x}_i$ and the points $B(t_i)$ on the curve.

b. Add a button which approximates the control points with a C^0 cubic Bézier spline. Allow the user to change the number of spline segments used for the approximation. Output the root mean square (RMS) distance between the approximated points and your curve and observe how it changes with the number of spline segments.

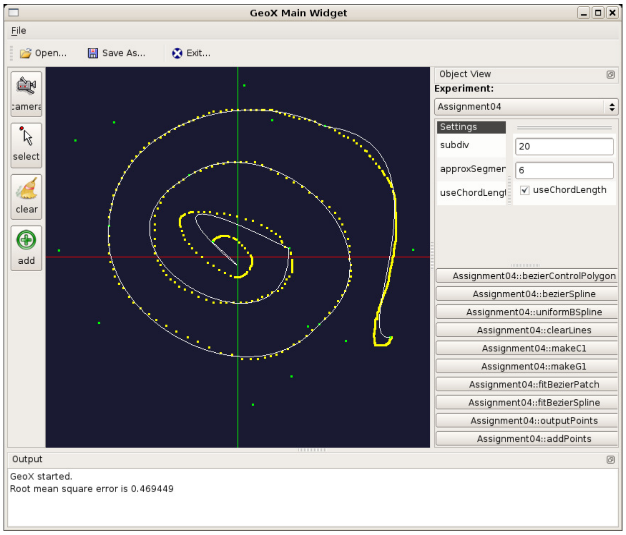
Hint: The RMS distance is defined as $\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} |\mathbf{x}_i - \mathrm{B}(t_i)|^2}$.

c. Imagine you want to capture the movement of the mouse and convert it into a Bézier spline. An example path is given by the point array in points.h. Add a button that loads it into the viewer. Approximate the 484 points with a cubic Bézier spline that uses only 19 control points. If necessary, modify your solution from (b) such that the RMS distance between the curve and the points is below 0.5.

Optional: Try to reduce the RMS as much as you can, but keep the total number of control points constant. The best result will be presented in the lecture.

Hint: You may use the following code to load the sample data:

```
#include "points.h"
for (int i=0; i<benchmarkPointCount; ++i) {
  viewer->addPoint(Point2D(benchmarkPoints[2*i],benchmarkPoints[2*i+1]));
}
```



subdiv 20

approxSegmer 6

useChordLengt ☑ useChordLength

```
GeoX started.
Root mean square error is 0.469449
```