Course: Introduction to Geometric Modeling (ECS 178), SQ 2016
Professor: Dr. Bernd Hamann

Project 2: Degree Raising & Reduction of Bézier Curves and Interpolation Techniques
Date due: Tuesday, April 26, 2016

The second project requires the implementation of algorithms related to **Bézier curves** and **interpolation techniques**. Your program(s) must be developed on a workstation (or PC or laptop) using the OpenGL (or a similar) library for drawing curves and surfaces—and for providing user interface techniques (buttons, sliders, etc.). You are allowed to use the OpenGL library, X windows, Motif, or the Forms libraries. A user menu must be provided to **interactively specify all the parameters** required for the various algorithms.

a.) Implement a two-dimensional (2D) version of the **degree-raising and -reduction algorithms** for Bézier curves discussed in class. A user will specify the $(n+1)$ control points of an $n-$th degree Bézier curve, your algorithm generates the control points of the degree-raised (degree-reduced) Bézier curve, and your program displays both the original control polygon/curve and the degree-raised (degree-reduced) polygon/curve.

b.) Implement a routine that performs **Aitken's algorithm** for performing polynomial interpolation of $(n+1)$ 2D points. A user will specify a sequence of 2D points $\mathbf{p}_i$ to be interpolated. You determine a **chord length parametrization** for the curve, compute (at least) 100 points on the polynomial curve, and **display the original points, the polygon defined by these points, and the curve**. A user must be able to move the parameter t in the interval $[t_0, t_n]$, and your program **displays all intermediate line segments produced by the geometrical evaluation procedure** of the interpolation scheme.

c.) Write a program for performing C^1 **piecewise cubic curve interpolation**. A user must be able to input a sequence of 2D points and derivative vectors (direction and magnitude) at these points. Alternatively, the derivative vectors can be computed from the sequence of 2D points by computing central differences of point pairs. Both user-specified and automatically computed derivative vectors must be supported by your program. The curve parametrization should be a chord length parametrization.

A user must be able to change all parameters easily by providing window areas used for parameter display and manipulation. Regarding the manipulation of control points (or points to be interpolated), a user must be able to **pick points**, **change their position**, and **insert/delete points**. If two or more points are within the same "pick" region (an $N \times N$ pixel region), your program should select the point closest to the center of the "pick" region. To move a point in a window use the left and right mouse buttons to change its x- and y-coordinates.

Besides having to demonstrate your program, prepare a short, about one-page "user's manual" explaining how to use your program.

DO NOT REMOVE YOUR PROGRAM!!! YOU WILL BE ABLE TO USE IT FOR FURTHER ASSIGNMENTS.

HAVE FUN !!!