Nonlinear Spline Modeling with Curvature-Driven Curve Deformations

Shin Yoshizawa (1031157)
Department of Computer Software
supervised by Prof. A. G. Belyaev

February 2, 1999

Abstract

The paper proposes a new method to design nonlinear splines on a plane. The method is based on consideration of curve evolution processes which have desired nonlinear splines as limit curves. The processes are constructed as curve deformations along the curve normals with the speed equal to functions of the curvature and curvature derivatives.

The method is applied to model the wooden splines (the curvature is a linear function of arc length) and mechanical splines (a mechanical spline minimizes the bending energy, the integral of the curvature squared).

Instead of considering smooth curves, we deal with their polygonal approximations. The notions of the curvature, curvature derivatives, tangent and normal vectors of a polygonal curve are discussed.

KEYWORDS: Nonlinear splines, Curve evolution, Curvature, Wooden splines, Mechanical splines, Polygonal approximations

1 Introduction

The paper is devoted to modeling nonlinear splines on a plane. Nonlinear splines are curves which do not depend linearly on boundary conditions, control points, etc. Such splines are important for various applications in shape analysis and modeling [6], [7], [8], computer graphics, computer vision [9], [10] because of their remarkable smoothing properties. The following two examples of nonlinear splines are considered: the wooden splines and mechanical splines.

The wooden splines are curves whose curvatures are represented by linear functions of their arc lengths. Wooden splines are often used for road design purposes [8]. The wooden splines are also known as the Cornu spirals, Euler splines and clothoids. The curvature of a wooden spline satisfies the relation

$$\frac{d^2k(s)}{ds^2} = 0 \quad \Longrightarrow \quad k(s) = \alpha s + \beta,$$

where α, β are constants. Here $k(s)$ is the curvature of the spline, s is the arc length.

The mechanical splines are curves minimizing the bending energy $\int k^2(s)\ ds$. The mechanical splines were first considered by Euler [5] and are often used in shape analysis and modeling [6], [7], [8] and computer vision [10]. The mechanical splines are a particular case of the elastica curves [3], [4] which minimize the bending energy

$$\int (\alpha k^2(s) + \beta) ds.$$

Here $k(s)$ is the curvature of the curve, α, β are constants.

The paper proposes a general method to design the wooden, mechanical, and other nonlinear splines and a numerical implementation of the method. The method is based on consideration a family of curves C evolving according to the nonlinear partial differential equation

$$C_t = F\vec{N},$$

where C_t is the time partial derivative of a family of smooth curves, $\vec{N}$ is the normals, F is a smooth function (call the speed functions). This differential equation corresponds to the curve evolution processes which have desired nonlinear splines as limit curves. The processes are constructed as curve deformations along the curve normals with the speed equal to a function of the curvature and curvature derivatives [1], [2]. We call such curve evolutions the curvature-driven curve deformations (see Section 2).

The paper is organized as follows. In Section 2 we introduce our method and consider speed functions associated with wooden, mechanical and other splines. In Section 3 we present a discrete implementation of the method. Instead of considering smooth curves we deal with their polygonal approximations. The curvature, tangent and normal vectors are approximated via the Lagrange interpolation method. The derivatives of the curvature is approximated by the central difference approximation. In Section 4 the computer simulation results will be shown.

2 Idea of Design:

Curvature-Driven Curve Deformations

The curvature-driven curve deformations model is based on consideration of the curve evolution where a curve evolves along a family of smooth curves $C(p,t)$ passing through two given points. Here p parameterizes the curve, t parameterizes the family and p is independent of t. Let us assume that this family evolves according to the following nonlinear partial differential equation

$$\frac{\partial C(p,t)}{\partial t} = F(p,t)\vec{N}(p,t),\ C(p,0) = C_0(p), \qquad (1)$$

where $\vec{N}(p,t)$ is the normal vector of the curve, $C_0(p)$ is an initial curve, $F(p,t)$ is a smooth function. Suppose the evolving curve

- satisfies the prescribed boundary conditions at the end points (for example, the curvature values or tangent directions), and
- converges to a limit curve $C(p,\infty)$, $t \longrightarrow \infty$.

Since the evolution equation (1) does not change the limit curve $C(p,\infty)$, then $\underline{F(p,t) \equiv 0}$ along $C(p,\infty)$.

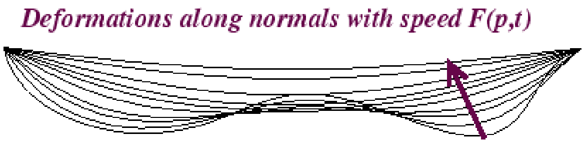


Fig. 1: *Visualization of Curve Evolutions*

The speed functions $F(p,t)$ for some limit curves are represented in the following table, where k is the curvature of the curve, k_s and k_{ss} are the first and second derivatives of the curvature with respect to the arc length s, α and β are constants.

limit curves	speed function $F(p,t)$
straight lines	k
wooden splines	$-k_{ss}$
mechanical splines	$-(2k_{ss}+k^3)$
elastica curves	$-\alpha(2k_{ss}+k^3)+\beta k$

3 Implementation of Method

In our numerical implementation we approximate smooth curves by polygonal curves.

Let us add the tangent component $G(p,t)\vec{T}(p,t)$ to the equation (1) to achieve the uniform distribution of vertices, where $\vec{T}(p,t)$ is the tangent and $G(p,t)$ is a smooth function. The $G(p,t)$ function is defined to affect only reparametrization of smooth curve, therefore that modifying is not inconsistent with discussions of section 2. In discrete implementation, the vertices shift along their curve by the tangent component. If the distribution of polygonal vertices is uniform, then $G(p,t)$ is 0.

The new evolution equation is defined by

$$\frac{\partial C(p,t)}{\partial t} = F(p,t)\vec{N}(p,t) + G(p,t)\vec{T}(p,t). \qquad (2)$$

3.1 Evolution Equation

The evolution equation (2) is implemented by the forward Euler-Cauchy Method

$$\frac{\partial C(p,t)}{\partial t} \approx \frac{C(p,t+\epsilon) - C(p,t)}{\epsilon},$$

where ϵ is small value (called the step-size). The smooth curve evolutions are approximated by the polygonal curve evolutions, then the discrete curve evolution equation is implemented by

$$C_{i,j+1} = C_{i,j} + \epsilon(F_{i,j}\vec{N}_{i,j} + G_{i,j}\vec{T}_{i,j}),$$

where $C_{i,j}$ is the polygonal curve, $\vec{N}_{i,j}$ is the discrete normal vector, $\vec{T}_{i,j}$ is the discrete tangent vector, $F_{i,j}$ and $G_{i,j}$ are the discrete speed functions, i indices the curve, j indices the family.

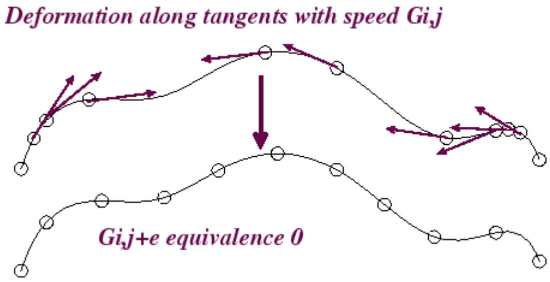
3.2 G(p,t) function

The uniform distribution of vertices is important in our numerical implementations, because the step-size of evolution can take the larger value within the effective range.

The $G(p,t)$ function $\approx G_{i,j}$ at $C_{i,j}$ is implemented by the equation

$$G(p,t) \approx G_{i,j} = (d_{i+1,j} - d_{i,j})/(d_{i+1,j} + d_{i,j}),$$

where $d_{i,j}$ is the distance between $C_{i-1,j}$ and $C_{i,j}$.



3.3 Derivatives of Curve

The Lagrange three points difference methods are

$$\frac{\partial C(p,t)}{\partial p} \approx \frac{\partial C_{i,j}}{\partial p} = \frac{C_{i+1,j} - C_{i-1,j}}{2h},$$
$$\frac{\partial^2 C(p,t)}{\partial p^2} \approx \frac{\partial^2 C_{i,j}}{\partial p^2} = \frac{C_{i+1,j} - 2C_{i,j} + C_{i-1,j}}{h^2},$$

where $\frac{\partial C_{i,j}}{\partial p}$ and $\frac{\partial^2 C_{i,j}}{\partial p^2}$ are the first and second derivatives of curve at $C_{i,j}$ with respect to the p, h is small value.

It means the curvature, tangent and normal vectors were approximated via the Lagrange interpolation method which interpolation curve is performed at each three successive points, satisfying the conditions: $C(h,t) = C_{i-1,j}$, $C(2h,t) = C_{i,j}$, $C(3h,t) = C_{i+1,j}$.

3.4 Curvature

For a regular curve, its signed curvature $k \approx k_{i,j}$ at $C_{i,j} = (x_{i,j}, y_{i,j})$ is given by equation

$$k_{i,j} = \frac{\dot{x}_{i,j}\ddot{y}_{i,j} - \dot{y}_{i,j}\ddot{x}_{i,j}}{(\dot{x}_{i,j}^2 + \dot{y}_{i,j}^2)^{\frac{3}{2}}},$$

where $\dot{x}_{i,j}$, $\ddot{x}_{i,j}$, $\dot{y}_{i,j}$ and $\ddot{y}_{i,j}$ are first and second derivatives of $x_{i,j}$ and $y_{i,j}$. These derivatives were approximated in Section 3.3.

3.5 Tangent and Normal Vector

The tangent $\vec{T}_{i,j}$ and normal $\vec{N}_{i,j}$ vector at $C_{i,j}$ are defined by equations

$$\vec{T}(p,t) \approx \vec{T}_{i,j} = \frac{\frac{\partial C_{i,j}}{\partial p}}{|\frac{\partial C_{i,j}}{\partial p}|}, \quad \vec{N}(p,t) \approx \vec{N}_{i,j} = (\vec{T}_{i,j})^{\perp},$$

where $|v|$ is the absolute value of v, $^{\perp}$ means $\frac{\pi}{2}$-rotation.

3.6 Analysis of discrete approximations of the curvature, tangents and normals

Fig. 2 demonstrates that our implementation provides with a good approximation of the curvature tangents and normals.

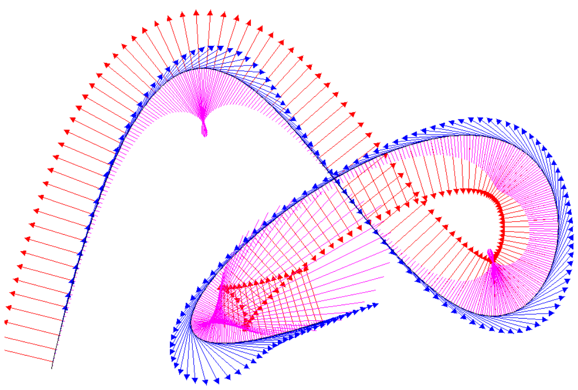


Fig. 2: *Intrinsic curve properties: two arrows represent the unit normals and tangents; other small segments represent the curvature size and, depending on their normal direction, the sign of the interpolation curve.*

Several methods were considered for computing the curvature, tangent and normal vectors: parametric spline interpolations [11], osculating circle interpolation, and the curvature defined by the tangent angle divided by the arc length [3]. These parametric spline interpolations are the Lagrange (three degree passing through each three adjacent points), B-spline (three and five degree passing through all points) and Hermite (three and five degree passing through all points) interpolations for computing the curve derivatives. Our numerical experiments show that the Lagrange interpolation provides the best results to approximate the curvature, tangent and normal vectors of polygonal curves.

3.7 Derivatives of Curvature

The derivatives of the curvature are approximated by the central difference approximation at each three successive points.

- The first derivative of curvature $k_s \approx \frac{\partial k_{i,j}}{\partial s}$ at $C_{i,j}$ is approximated by equation

$$\frac{\partial k_{i,j}}{\partial s} = \frac{k_{i+1,j} - k_{i,j}}{d_{i+1,j}} + \frac{k_{i,j} - k_{i-1,j}}{d_{i,j}} - \frac{k_{i+1,j} - k_{i-1,j}}{d_{i+1,j} + d_{i,j}}.$$

- The second derivative of curvature $k_{ss} \approx \frac{\partial^2 k_{i,j}}{\partial s^2}$ at $C_{i,j}$ is approximated by equation

$$\frac{\partial^2 k_{i,j}}{\partial s^2} = \frac{2k_{i+1,j}}{d_{i+1,j}(d_{i,j} + d_{i+1,j})} - \frac{2k_{i,j}}{d_{i,j}d_{i+1,j}} + \frac{2k_{i-1,j}}{d_{i,j}(d_{i,j} + d_{i+1,j})}.$$

Here $d_{i,j}$ is the distance between $C_{i-1,j}$ and $C_{i,j}$.

3.8 Techniques for stability

The following techniques were used to achieve stability of the considered curve evolutions.

Let us introduce two thresholds (*High* and *Low*) to which the inserting and eliminating points are decided for the uniform distribution of vertices. If the distance between two successive points is greater than *High*, additional points are inserted. If the distance between two successive points are less than *Low*, one point is removed.

If the distance between $C_{i-1,j}$ and $C_{i+1,j}$ is much smaller than the absolute value of the speed functions, then during the evolution $C_{i,j}$ may oscillate between the points $C_{i-1,j}$ and $C_{i+1,j}$ (see Fig. 3) and $C(p,t)$ does not converge as $t \to \infty$. To avoid oscillations, it is better to decrease ϵ.

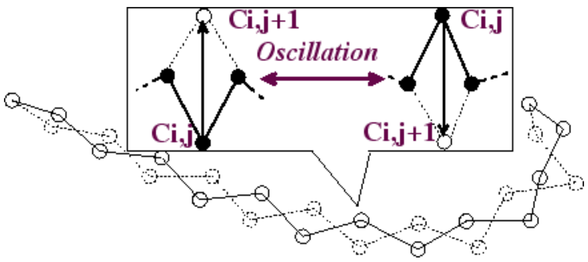


Fig. 3: *Oscillation of Vertices*

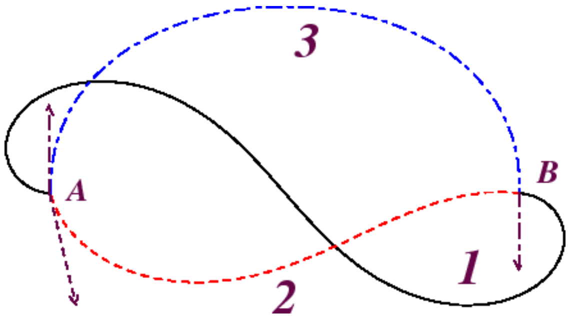
If one distance is much smaller than two others, the curvature becomes extremely large. It is so because the evolving polygonal curve is approximated via three successive points (see Section 3.3). To avoid such extreme values, we set the thresholds (*High* and *Low*) to be closer to each other.

4 Computer Simulation Results

This section contains samples of numerical experiments with curve evolution processes which have desired nonlinear splines as limit curves: wooden splines and mechanical splines. The initial polygonal curves are chosen as straight segments. The evolutions were continued until the speed function becomes close to zero. $High$ takes 0.3 and Low takes 0.1 ($h = 0.2$), ϵ takes $\frac{Min(d_{i,j}|i=1,2,...,N-1)^3}{10.0N}$ where $Min(A)$ returns the minimum value of a set A, N is number of vertices, $d_{i,j}|i = 0, 1, 2, ..., N-1$ are set of distances between $C_{i-1,j}$ and $C_{i,j}$ ($i = 1, 2, ..., N-1$).

4.1 Sample Evolutions for

Wooden Splines: $F(p,t) = -k_{ss}$

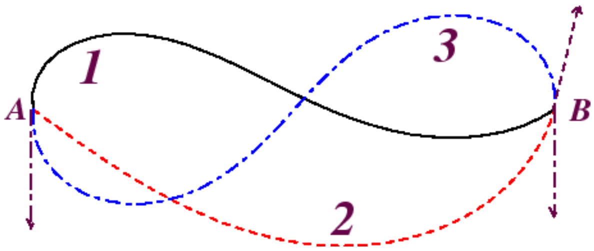


No.	Boundary Conditions
1	$k(A,t) = -1.5$, $k(B,t) = 1.5$
2	$T(A,t) = (0.26, -0.97)$, $k(B,t) = -0.5$
3	$T(A,t) = (0, 1)$, $T(B,t) = (0, -1)$

In the case when the speed function is given by $F(p,t) = -k_{ss}$, the evolving curves converge to wooden splines.

4.2 Sample Evolutions for

Mechanical Splines: $F(p,t) = -(2k_{ss} + k^3)$



No.	Boundary Conditions
1	$k(A,t) = -1.0$, $k(B,t) = 0.5$
2	$k(A,t) = 0$, $T(B,t) = (0.28, 0.96)$
3	$T(A,t) = (0, -1)$, $T(B,t) = (0, -1)$

In the case where the speed function is given by $F(p,t) = -(2k_{ss}+k^3)$, the evolving curves converge to mechanical splines.

Through simulation, the shapes of the limit curves (wooden and mechanical splines) depend on the boundary conditions.

5 Conclusion

This paper presented a new method to design nonlinear splines on a plane and an implementation of its method. The method is based on consideration of curve evolution processes which have desired nonlinear splines as limit curves. The wooden and mechanical splines were constructed by this method. The current implementation of the method does not work fast, but the method seems to be very promising for nonlinear spline design and for a variety of shape modeling applications. In the future, better numerical implementations of the method and extensions of the method to 3D curve and surface designs will be developed.

Acknowledgment

I would like to thank Prof. A. G. Belyaev for discussions and advice during the course of this work.

References

[1] A. G. Belyaev, E. V. Anoshkina and S. Yoshizawa, “Nonlinear Spline Generation with Curve Evolutions Driven by Curvature”, *Shape Modeling International ’99*, Aizu-Wakamatsu, Japan, March. 1999.

[2] S. Yoshizawa, “Nonlinear Spline Modeling with Curvature-Driven Deformations”, *Human and Computer ’98*, 1st International Conference, Aizu-Wakamatsu, Japan, September 1998.

[3] A. M. Bruckstein, R. J. Holt and A. N. Netravali, “Discrete Elastica”, *Discrete Geometry for Computer Imagery*, 6th International Workshop, Lyon, France, Nov. 1996, pp. 59–72.

[4] G. Brunnett and J. Wendt, “A univariate method for plane elastic curves”, *Computer Aided Geometric Design*, Vol. 14, pp. 273–292, 1997.

[5] L. Euler, Additamentum “De Curvis Elasticis”, *Methodus Inveniendi Lineas Curvas Maximi Minimive Proprietate Gaudentes*, Lausanne, 1744.

[6] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide*, 2nd edition, 1990.

[7] M. Hosaka, *Modeling of Curves and Surfaces in CAD/CAM*, Springer-Verlag, 1996.

[8] J. Hoschek and D. Lasser, *Fundamentals of Computer Aided Geometric Design*, Springer-Verlag, A. K. Peters, Ltd. 1993.

[9] B. K. P Horn, “The curve of least energy”, *ACM Trans. on Math. Software*, Vol. 9, pp. 441–460, 1983.

[10] D. Mumford, “Elastica and computer vision”, in C. L. Bajaj, editor, *Algebraic Geometry and its Applications*, Springer-Verlag, 1994, pp. 491–506.

[11] K. Sugihara, *Mathematics of Graphics*, Kyoritsu, 2nd edition, 1996.