접선 지정법을 이용한 대화형 G^1 스플라인

주 우 석[†] 박 경 희[††] 이 희 승[††]

요 약

스플라인 함수는 결과적으로 생성되는 커브모양에 의해 캐드 상에서 물체의 외형을 설계할 수 있게 하는 가장 기본적인 도구이고, 따라서 새롭고 효율적인 커브 모양의 개발은 산업 설계 분야 전반에 직접적인 영향을 줄 수 있다. 본 논문은 더욱 다양하고 자유로운 커브 모양을 만들기위하여 시각적으로 연속인 부류의 스플라인 함수를 그리기 위한 도구를 설계하고 구현한다. 이 부류의 스플라인은 기존 스플라인에 비해 자유자재의 다양한 모양을 생성함과 동시에 일반적인 카디널 스플라인이 갖는 보간성을 아울러 갖고 있다는 점이다. 본 논문의 가장 큰 중요성은 매개변수의 스칼라 수치값을 제시하던 기존 G^1 커브 구현방식에서 벗어나, 사용자가 시각적인 벡터를 사용하여 접선의 모양을 지정케하고 그 결과를 커브 생성을 위한 매개변수로 변형시킬 수 있는 공식을 유도하고 구현한 점이다. 생성된 결과 커브 자체가 원래 지정된 접선에 충실하게 되므로, 캐드 사용자의 입장에서는 설계하고자하는 커브의 개형을 시각화 시킨 접선의 모양에만 치중할 수 있는, 단순한 인터페이스가 가능하게 된다. 따라서, 설계자의 입장에서는 본 논문에 구현된 스플라인 도구를 사용한다면 마우스와 같은 간단한 입력장비만으로도 다양하고 효과적인 커브 모양을 생산할 수 있다.

Interactive G^1 Splines with Tangent Specification Method

Wou Seok Jou,[†] Kgung Hee Park[††] and Hee Seung Lee[††]

ABSTRACT

Spline curve scheme is the most valuable tool for the CAD of industrial products. Hence, the development of a new, effective curve scheme can have immediate impact on the current design industries. This paper develops and implements a new methodology for the implementation of the visually continuous class of splines which can produce a more flexible and diverse curve shapes. This class of splines has advantages over existing splines in that it can accommodate wider range of shapes while maintaining the interpolatory property of the ordinary cardinal splines. Most importantly, we avoid using the previous method of implementing G^1 curves, where users must specify scalar values for the control of curve shapes. We derive and implement an easy-to-use transformation between the user-specified graphical tangent vectors and the actual parameters for the resulting curve. Since the resulting curve shape reflects original tangential direction faithfully, CAD users can simply represent approximate curve shapes with proper tangents. Consequently, a simple user interface device such as a mouse can effectively produce a various spline curves using the proposed spline tool.

1. INTRODUCTION

The exploration of the use of the parametric curves and surfaces can be viewed as the origin of computer aided geometric design. As a result, numerous curve schemes have been widely studied and put into practical use until today. These parametric curves, collectively called the splines, find their use in such CAD areas as the design of car-body, aircraft, and general industrial products since any curved surfaces are represented and ma-

† 정 회 원 : 명지대학교 컴퓨터공학과 조교수
†† 정 회 원 : 명지대학교 컴퓨터공학과 석사과정
논문접수 : 1994년 7월 22일, 심사완료 : 1994년 11월 5일

nipulated in the form of the splines during the design steps. Moreover, modern animation techniques used in movie industries require an extensive use of the spline curves for the generation of motion path and inbetween frames [1] during animation. Development of a new, effective curve scheme can have immediate impact on various industrial design processes.

Splines can be classified into two categories. Interpolating splines exactly hit the control points given by designers during design processes. It is this passing-throughness of this class of splines that makes the interpolating splines to be used whenever exact control of the curve behavior is required. Cardinal or Catmull-Rom splines belong to this class of splines. Approximating splines, on the other hand, do not necessarily pass through given control points. The control points in this sense, take the role of controlling the curve behavior only remotely. The splines in this category are B zier curves[6, 15], B-splines, and-splines. However splines in this class have advantage over the interpolating splines in that they are smoother than the interpolating splines. Mathematically, this means that the approximating splines have the continuity of the second order derivatives (C2 continuous) while the interpolating splines have that of the first order derivatives (C1 continuous) on each and every control point [2, 3].

While satisfying the property of the passing-throughness of the interpolating splines, G^1 splines have much more flexibility in controlling the curve shapes than the interpolating splines. Since the finding of this class of splines [4, 5, 6], the splines in this class are termed "geometrically continuous" or "visually continuous." Most interestingly, the splines show the same smoothness as the corresponding C1 curves. They apparently look like having the property of continuity in the first-order derivatives, although actually they do not. In fact, the left and right derivatives of G^1 splines share a common direction and they may differ only in magnitudes. By allowing the tangential magnitudes to differ widely, we can freely control the curve shapes in the the G^1 splines than the corresponding C1 splines.

The property of the G^1 continuity is investigated in [7] and the application of this type of continuity to the approximating splines appears in [4, 8]. The only endeavor to produce a G^1 interpolating splines is in [9]. In this approach, the fact that the basis functions of the cardinal spline [10] can be expressed in terms of a Lagrange polynomials [3, 11] is used and subsequently, the Bezier control vertices are formed geometrically for the produced piecewise Lagrange curves. Although the existence of such control vertices is given without proof in the approach, by using such algorithms as recursive subdivision [4] or de Casteljau's algorithm [6], these control vertices then lead to the B zier curves, which in this case corresponds to the G^1 cardinal splines. Nevertheless, this approach is purely algorithmic, "owing to the algebraic complexity" [9] involved. Because of this complexity, G^1 splines could hardly find their practical use up to now. A mathematical derivation of an explicit closed form of the G^1 splines is in [12] and part of them is repeated in this paper only for a reference. By using this mathematical equation form of the G^1 splines, algorithmic complexity of the above approach is avoided, and thus an easy framework for the implementation of G^1 splines can be established.

In this paper, we further increase the applicability of the developed equation by presenting an interactive tool for the generation of G^1 curves ,which will facilitate the implementation of G^1 splines in a general CAD environments. Using the proposed tool, the CAD designers can easily acquire desired curve shapes with the aid of a simple interface device such as a mouse.

Most importantly, the tool assumes no a priori knowledge on the concept of splines from the designers' viewpoint This is in direct contrast with the previous mehods for the control of the curve shapes, where the designers have to specify proper numerical values depending on the degree of bias and tension they intend. The resulting curve shape reflecting the inputted numerical values was difficult to guess before the actual drawing of the curves. In other words, the designers' imagination must be mobilized to predict the curve shapes and it must be translated into numerical scalar values in the previous curve control schemes. Therefore, they have to repeat the process until the desired curve shape appears.

On the contrary, the process is reversed in our G^1 tool. An imaginary curve drawn by the designers will control the shape parameters of the resulting curve. Proposed interface algorithm directly interprets the visual input and translates its requirements into proper parameters of the corresponding G^1 curve.

2. G^1 SPLINES IN A CLOSED FORM

Linear interpolation of a pair of control points P_1, P_2 yields a line segment in a normalized parametric form of

$$f(a)=(1-a)P_1 + aP_2 \quad \cdots\cdots\cdots\cdots\cdots \text{Eq.1}$$

where a is a parameter in the range $(0 \leq a \leq 1)$. Given two such line segments f(a) and g(b), each representing line segments P_1P_2 and P_2P_3 respectively, the linear blending of the two basis functions produces a quadratic C1 spline m(s) such that

$$m(s) = (1-s)f(a)+sg(b)$$

$$= [\, s^2 \; s \; 1 \,] \begin{bmatrix} 2 & -4 & 2 \\ -3 & 4 & -1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} \quad \cdots\cdots \text{Eq.2}$$

This curve interpolates the three control points P_1, P_2 and P_3 while the normalized parameter s changes from zero to one inclusively. Notice that this equation is valid only when the value of the parameter corresponding to the point P_2 is set to 0.5, half the parametric interval. If we let the parameter value corresponding to the point P_2 to vary by assigning a new parameter u, we have the following curve equation with more flexibility in controlling the curve shape by the new parameter.

$$m(s)=[\, s^2 \; s \; 1 \,] \begin{bmatrix} \frac{1}{u_1} & -\frac{1}{u}-\frac{1}{1-u} & \frac{1}{1-u} \\ -1-\frac{1}{u} & \frac{1}{u}+\frac{1}{1-u} & 1-\frac{1}{1-u} \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} \quad \cdots\cdots \quad \text{Eq.3}$$

With one more repetition of linear blending of two such quadratics, we get a cubic interpolating spline curve p(s) such that

$$p(s) = [\, s^3\ s^2\ s\ 1\,] A\ [\, P_1\ P_2\ P_3\ P_4 \,]^T$$

$$A = \begin{bmatrix} -u+2-\frac{1}{u} & -1+\frac{1}{u}+v & u+\frac{1}{v-1} & -v-1+\frac{1}{1-v} \\ 2u-4+\frac{2}{u} & 2-\frac{2}{u}-v & -2u-1+\frac{1}{1-v} & v+1-\frac{1}{1-v} \\ -u+2-\frac{1}{u} & -2+\frac{1}{u} & u & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \qquad \text{...... Eq.4}$$

In this equation, u and v represent the parameter values of the corresponding control points P_2 and P_3, respectively. Moreover, this equation represents a curve ranging from the point P_2 to P_3 while the parameter s changes from zero to one inclusively. To draw the curve p(s) between P_2P_3, the points P_1, P_2, P_3, and P_4 are required and the points P_2, P_3, P_4, and P_5 are required to draw the curve q(s) between the points P_3 and P_4. Furthermore, to draw the curve p(s), the parameter u1 of the point P_2 and the parameter v1 of the point P3 have to be known. Likewise, when the curve q(s) is to be drawn, the parameter of the point P_3 has to be u2, and the parameter of the point P4 has to be v2. If we set u2 to be equal to v1, then we are able to get a G^1 spline. That is, if we maintain the same parameter value on a given control point while evaluating the curve left to it and the curve right to it, resulting piecewise combination of the curves becomes visually continuous.

In the curves represented by the Eq. 4, the direction of the left tangent is the same as the right tangent, and the tangents differ only in the tangential magnitude. In other words, the left tangent is a scalar multiple of the right tangent satisfying the necessary and sufficient conditions for a spline to be visually continuous.

3. TANGENT SPECIFICATION METHOD AND INTERACTIVE G^1 TOOL

How do we set the u, v values of the above G^1 splines? If we let the designers to specify the scalar quantities corresponding to them, then they will be left with the responsibility of guessing the curve shape. By trying some test values, they may find some approximate parameter values for the desired curve shape. In this case, the designers must understand the relationship between the parameters and the resulting curve shapes, including such concepts as the bias and tension of the splines. Instead of the previous method where the Eq. 4 is used in its naive form, we derive its vector form and exploit the vector property in our G^1 tool.

For our new tangent specification method, we differentiate the spline expressed in Eq. 4 for a pair of splines centered at a given point with respect to parameter s. Furthermore, we rearrange resulting matrix equation in terms of point subtraction, namely Pi-Pj. Resulting tangential relationship in the splines can be written as

$$D_L = \frac{k^2}{1-k}(P_4-P_3)+(1-k)(P_3-P_2)\ ,$$

$$D_R = k(P_4-P_3)+\frac{(1-k)^2}{k}(P_3-P_2). \quad \text{.........Eq.5}$$

where DL is the left tangent and DR is the right tangent on a given control point, and k (= u2 = v1) is an arbitrary number in the range 0<k<1. Since the right tangent is still a scalar multiple of the left tangent, this equation still satisfies the G^1 property.

One of the most distinguishing features of our derivation is that now it can accommodate a vector notation which enables us to use more intuitive graphical user interface. More specifically, from the given control points, vectors (P3-P2) and (P4-P3) can readily be calculated. The only term remaining to be determined from Eq.5 is the shape control parameter k, and this value must be determined such that the constraints given in Eq.5. are satisfied. Here we can see that the k value depends not only on the magnitude of vectors but also on the left and right tangent which can be given directly from a user specification.

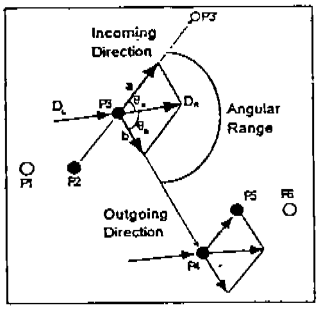
In Eq.5 we find that the right tangent DR is a function of the parameter k, and the chord lengths of line segments involved. Therefore, once the chord lengths and the right tangent are known, we can readily derive the corresponding parameter value of k in a straightforward manner. Namely, with a, b being the decomposed components of the right tangent DR to the direction of the vector P2P3 and P3P4 respectively.

$$k=\frac{1}{1+\sqrt{\frac{a}{b}\left|\frac{P_4-P_3}{P_3-P_2}\right|}} \quad \cdots\cdots \quad \text{Eq.6}$$

Moreover we can further refine Eq.6 through use of a vector calculation. Since the Eq.5 states the vector relationship and not the algebraic sum, care must be taken with regards to the addition of terms. Upon geometrical analysis of Eq.5, we can see that the right tangent DR forms a diagonal in a parallogram with edges a and b as in (Fig. 1). Using trigonometric functions, the angular relationship is expressed as with a and b being the angles formed by the right tangent and the vector P_2P_3 and P3P4 respectively.

$$k=\frac{1}{1+\sqrt{\frac{\sin\theta_a}{\sin\theta_b}\left|\frac{P_4-P_3}{P_3-P_2}\right|}} \quad \cdots\cdots \quad \text{Eq.7}$$

This is shown in (Fig. 1) where we have five control vertices P1 through P_5, and the incoming direction is defined to be the direction of vector P_2P_2 while the outgoing direction is defined to be the direction of P_3P_4. Here, the right tangent is denoted by DR and the decomposition of the vector DR into the corresponding incoming direction and outgoing direction is denoted by the vector a, 버 and b. Notice that the desired tangent must lie in the range of angle formed by the line segments P_3'P_3P_4. There are two boundary cases to be avoided here. If the tangent is parallel to the line P_3P_4, then the sin b component of the right tangent becomes zero, thus letting k approach infinity. On the other hand, if the tangent is parallel to the line segment P_2P_3, sin a becomes zero making it impossible to evaluate the left tangent.



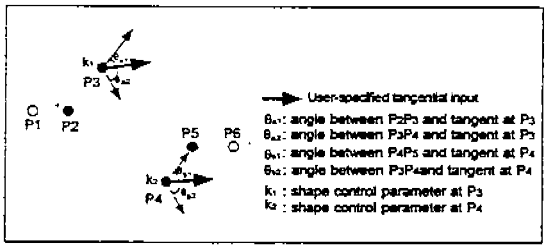
(Fig. 1) Decomposition of right tangent

As a result of this derivation, we can conclude that if a designer gives an approximate curve shape through tangential directions, we can interpret the tangent vectors into proper k values and use the k values to draw the corresponding spline curves satisfying the G^1 tangential restrictions. More specifically, given the tangential directions, we measure the angles formed by the tangent and the incoming and outgoing direction, and then use the information together with the chord length information between control points to calculate proper k.

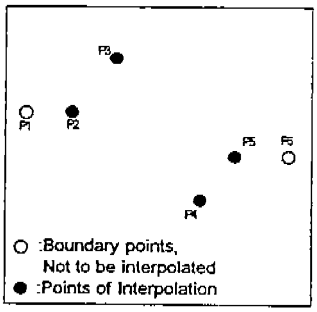
The designers are required to specify the direction of right tangent simply by dragging a mouse from each control points. This tangential information is then passed to our program that interprets the tangential angle into a proper k value and thus a proper interpolating curve which exactly reflects originally specified tangential direction. To further facilitate visual perception, our G_1 tool allows a circle to be drawn around the endpoint of tangent and the numerical values of the relevant parameters are automatically displayed as a reference.

4. RESULT

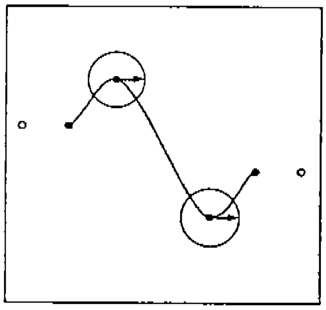
Prototype G^1 tool is programmed in "C" language running on IBM PC 486 for the arbitrary two dimensional sample data points shown in (Fig. 2). Curves in those intervals adjacent to the boundary points P_1, P_5 are not drawn as with the case of ordinary cubic splines. (Fig. 3) shows the G^1 tool interpretation of the user input specified by an arrow indicating the required curve shape. The tangent pair is decomposed into four angular parameters θa1, θa2, θb1, and θb2 by our G^1 tool. Now these parameter values are substituted into Eq. 7 to obtain the corresponding curve control parameter k1, and k2. Finally, these k values are substituted into Eq. 4 for the u, v values to obtain the desired curve shapes by our G^1 tool. Experimental results of our G^1 tool employing the tangent specification method are shown in (Fig. 4) through (Fig. 8), where a family of curves are drawn for the given sample points and the user-specified tangential directions. For the selected control points, a circle is drawn to prompt for the designation of a tangential direction, and this direction is specified as an arrow inside the circle. ⟨Table 1⟩ shows the intermediate parameter values used in our G^1


k1
P3
P1
P2
P5
P6
k2
P4
User-specified tangential input
θa1: angle between P2P3 and tangent at P3
θa2: angle between P3P4 and tangent at P3
θb1: angle between P4P5 and tangent at P4
θb2: angle between P3P4 and tangent at P4
k1 : shape control parameter at P3
k2 : shape control parameter at P4

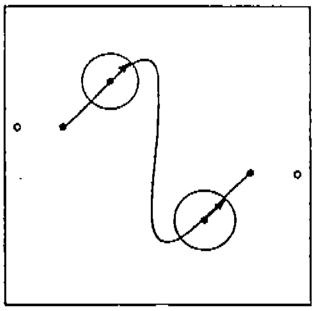
(Fig. 3) Interpretation of user input by Tangential Specification Method

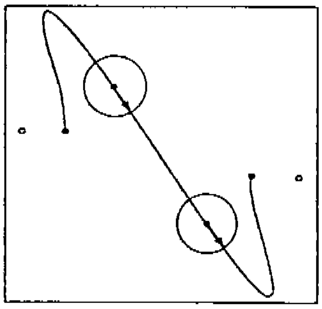

P3
P1
P2
P5
P6
P4
○ :Boundary points, Not to be interpolated
● :Points of Interpolation

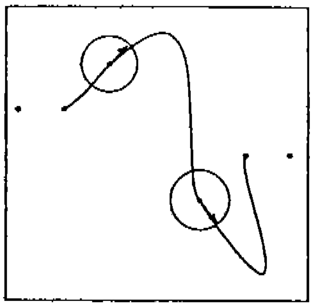
(Fig. 2) Position of sample data points

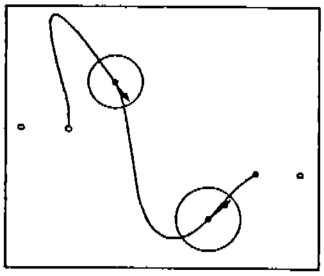


(Fig. 4) Horizontal direction at both points


(Fig. 5) Incomming direction at 1st, out going direction at 2nd point


(Fig. 6) Outgoing direction at 1st, incomming direction at 2nd point


(Fig. 7) Incomming direction at both points


(Fig. 8) Out going direction at both points

tool to produce the examplary curves upon user specification of tangential directions. Here the notation θa1 corresponds to the angle formed by the right tangent and the incoming vector at the point P_3 while θa2 corresponds to the angle formed by the right tangent and the outgoing vector at the same point P_3. At the point P_4, θb1 andθb2 are defined the same way. The notation k_1 and k_2 corresponds to the shape control parameters at the points P_3 and P_4 respectively.

〈Table 1〉 Parameter values showing the intermediate result of the G¹ tool

Figure	θa1	θb1	k1	θa2	θb2	θk2
(Fig.2)	45°	56.3°	0.40689	56.3°	45°	0.59311
(Fig.3)	1.2°	100°	0.81263	100°	1.2°	0.18737
(Fig.4)	100°	1.4°	0.09059	1.4°	100°	0.90941
(Fig.5)	1.2°	100°	0.81263	1.4°	99°	0.90953
(Fig.6)	99°	1.6°	0.09612	99°	2.3°	0.24169

Notice that the specified tangential direction immediately affects the final curve shape thus validating the fact that designation of tangents reflects final G¹ curve shapes faithfully. The length or the magnitude of the right tangent has actually no effect in our G¹ curves. Also note that the curve shape in each interval is determined first, by the geometrical spacing of sample data points and second, by the tangential directions specified at the both ends of the interval. In other words, a pair of user-specified tangents determines the behavior of the curve in that interval.

The terms bias and tension [7, 13, 14] usually represent the behavior of curves. With a high tension value, the curve between the pair of control points becomes closer to a line. The bias is used usually in conjunction with a curve behavior near the control points. If the curve is biased to incoming direction, the curve follows the direction of incoming vector when passing through the control points. If the curve is biased to the outgoing direction, the curve prepares its direction well before it hits the control points. When the curve actually passes through the control point, it has the tangential direction of outgoing vector. In the proposed G^1 tool, the bias and tension can together be controlled through the proper combination of tangential directions at both endpoints of a given interval.

For the test datas in (Fig. 4) through (Fig. 8), chord length ratio of P_2P_3 to P_3P_4 is set to1 : 2.5. (Fig. 4) shows the curve resulting when the right tangents at P_3 and P_4 are specified to be horizontal. The curve looks quite faithful to the original specification of the tangential directions at both ends. (Fig. 5) shows a variation in the tangential directions. At P_3 the bias is given to follow the incoming direction while the bias at P_4 is given to the outgoing direction. As a result, the curve maintains its incoming direction for a while right after passing the point P_3 while it tries to adjust itself before reaching the point P_4. Consequently, the curve undergoes two large inflection inbetween the control points. Tension can be controlled in this way. (Fig. 6) is exactly the opposite case of the (Fig. 5). Here the direction of the incoming and outgoing tangents is reversed and the result is almost linear between the control points achieving a highly tensioned curve.

In (Fig. 7), both the points of interest have directional tangents of the incoming directions. As a result, the curve exhibits a reflex only near the point P3, while in (Fig. 8), the reflexion is shifted to the area near the point P4. Notice that the modification of the tangent at each control point affects only the curves shapes adjacent to the control point in our G^1 tool. This property satisfies the required tight locality of the splines [5, 6, 16], which is one of the most beneficial aspect of the piecewise splines.

Consequently, we can conclude that if the users can approximately visualize the desired curve shapes, they can visualize the tangential direction at each control point. As the experimental curves demonstrate, the resulting curves produced by our G^1 tool can faithfully reflect the original user's intention of the curve behaviors.

5. SUMMARY AND CONCLUSION

In this paper, we presented an interactive G^1 tool based on our mathematical derivation of the G^1 splines. The vector relationship used for the derivation of the G^1 splines could easily led us to calculate the transformation between user-specified tangents and the corresponding parameter values at the control points. By exploiting this fact, a simple user interface has been implemented and tested for the various curve control schemes. The experimental curves state that the intuitive tangents specified by the users can directly be mapped into the expected curve shapes. The biggest advantage of this tool over conventional curve control tools is that the designers can stick to their visual perception and that they can ignore the mathematical details. A space curve scheme in three-dimensional space can readily be extended to the three-dimensional surface generation scheme using such techniques as the Coons patch or bilinear transformation. Extension of the G^1 tool to the three-dimensional surface generation is undergoing in our laboratory.

References

[1] Thalmann, N. M., and Thalmann, D., Computer Animation : Theory and Practice, Springer-Verlag, Tokyo, pp. 43-67, 1985.

[2] Rogers, D. F., and Adams, J. A., Mathematical Elements for Computer Graphics, McGraw-Hill, New York, pp. 250-289, 1976.

[3] Goldman, R. N., and Micchelli, C. A., "Algebraic Aspects of Geometric Continuity," Mathematical Methods in Computer Aided Geometric Design, edited by Lyche, T., and Schumaker, L. L., Academic Press, San Diego, pp. 313-332, 1989.

[4] Bartels, R. H., Beatty, J. C., and Barsky, B. A., An Introduction to Splines for Use in Computer Graphics and Geometric Modeling, Morgan Kaufmann Publisher, Los Altos, California, pp. 65-180, 1987.

[5] Barsky, B. A., and Beatty, J. C., "Local Control of Bias and Tension in Beta splines," Computer Graphics, Vol. 17, No. 3, pp. 193-218, 1988.

[6] Schweikert, D. C., "An Interpolation Curve Using a Spline in Tension," Journal of Mathematical Physics, Vol. 45, pp. 312-317, 1966.

[7] Manning, J. R., "Continuity Conditions for Spline Curves," The Computer Journal, Vol. 17, No. 2, pp. 181-186, 1974.

[8] Farin, G., "Visually C2 Cubic Splines," Computer Aided Design, Vol. 14, No. 3, pp. 137-139, 1982.

[9] Derose, T. D., and Barsky, B. A., "Geometric Continuity, Shape Parameters, and Geometric Constructions for Catmull-Rom Splines," ACM Transactions on Graphics, Vol. 7, No. 1, pp. 1-41, 1988.

[10] Foley, T.A., "Local Control of Interval Tension Using Weighted Spline," Computer Aided Geometric Design, Vol. 3, pp. 281-294, 1986.

[11] Watt, A., "3D Computer Graphics," pp. 163-224, Addison Wesley, University of Sheffield, England, 1993.

[12] Jou, W., "On Control of Motion Path and Speed in a Spline-based Animation," Ph.D Dissertation, University of Florida, 1991.

[13] Smith, A. R., "Spline Tutorial Notes," ACM SIGGRAPH 88 Course Notes : Introduction to Computer Animation, pp. 131-142, 1988.

[14] Kochanek, D. H. U., and Bartels, R. H., "Interpolating Splines with Local Tension, Continuity, and Bias Control," Computer Graphics, Vol. 18, No. 3, pp. 33-41, 1984.

[15] Kim Hyeock Jin, Kim Ha-Jin, Kwon Yonghoon, "Construction of Triplicated Piecewise B zier Cubic-curve as PC-graphics Tool," Journal of the Korea Information Science Society, Vol. 20, No. 2, pp. 225-232, 1993.

[16] Kim Ha-Jin, Lee Kyung-Hwan, Yoon Kyung-Hyung, "On the Some Piecewise Cubic Interpolating Polynomials," Jou-

rnal of the Korea Information Science Society, Vol. 10, No. 3, pp. 55-62, 1983.

주 우 석

1976년~83년 서울대학교 공과대학 전자공학과(학사)
1983년~85년 한국 IBM, 데이콤 정보통신연구소
1985년~87년 University of Florida, 컴퓨터공학(석사)
1987년~91년 University of Florida, 컴퓨터공학(박사)
1992년~현재 명지대학교 컴퓨터공학과 조교수
관심분야 : 컴퓨터그래픽스, 알고리즘, 데이타베이스

이 희 승

1990년~94년 명지대학교 공과대학 전자계산학과(학사)
1994년~현재 명지대학교 공과대학 컴퓨터공학과(석사과정)
관심분야 : 컴퓨터그래픽스, 알고리즘, 컴퓨터통신

박 경 희

1990년~94년 명지대학교 공과대학 전자계산학과 (학사)
1994년~현재 명지대학교 공과대학 컴퓨터공학과(석사과정)
관심분야 : 컴퓨터그래픽스, 알고리즘, 컴퓨터구조