BUTTERWORTH HEINEMANN

Computer-Aided Design, Vol. 27, No. 1, pp. 48–58, 1995

0010-4485/95 $10.00 + 0.00

Multidimensional curve fitting to unorganized data points by nonlinear minimization

Lian Fang and David C Gossard

Many papers have addressed the problem of fitting curves to data points. However, most of the approaches are subject to a restriction that the data points must be ordered. The paper presents a method for generating a piecewise continuous parametric curve from a set of unordered and error-filled data points. The resulting curve not only provides a good fit to the original data but also possesses good fairness. Excluding the endpoints of the curve, none of the connectivity information needs to be specified, thus eliminating the necessity of an initial parameterization. The standard regularization method for univariate functions is modified for multidimensional parametric functions and results in a nonlinear minimization problem. Successive quadratic programming is applied to find the optimal solution. A physical model is also supplied to facilitate an intuitive understanding of the mathematical background.

Keywords: data interpolation, regularization, nonlinear minimization

Fitting a smooth curve to a set of data points is a general problem arising in many fields. This problem can be stated as 'given a set of data points $\mathbf{P}_i$, $i = 1 \ldots N$, taken from a target curve, reconstruct a curve which approximates the original curve to a satisfactory extent and also possesses visually good appearance'. This kind of problem has many applications in diverse domains, for example the following:

- In reverse engineering in the automobile industry, it is necessary to generate curves from digitized data points that are taken from a clay model in order to blend a surface from the curve network. In such applications, generation of fair curves is essential since the surface's quality is influenced consequently.
- In those curve design systems which are able to create curves from a 'sketch' provided by the user, the input data is assumed to follow a continuous trace of the input device. Unfortunately, a curve created by one continuous sweep motion is generally not satisfactory. Most users like to sketch a curve by moving the mouse along the curve back and forth repeatedly. In such a case, either the order of the data points should be inferred from the original unordered data set or a fitting procedure that is independent of data point ordering should be devised.

Computer-Aided Design Laboratory, Department of Mechanical Engineering, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA
Paper received: 30 April 1993. Revised: 6 August 1993

Reconstructing a curve from a set of data points is an inverse problem which is generally ill posed because the information we have does not uniquely determine the solution. Problems of this kind can be broadly categorized into interpolation problems and approximation (or smoothing) problems in terms of whether the resulting curve is required to pass through all the data points exactly or not, respectively. Generally, when data points are subject to measurement errors, an approximation scheme will be preferable to an interpolation scheme since the latter would generate a curve with many unwanted undulations. Many papers have addressed this problem that are based on the assumption that the data points are ordered. To the authors' knowledge, however, little has been published on the related problem of fitting a curve to unordered data points. This is of particular interest because, when a large number of error-filled data points need to be fitted, the determination of connectivity information of data points becomes a time-consuming task.

This paper reports further research carried out since the preliminary research described in Reference 1. In this paper, a method which simulates the deformation of a perfectly elastic beam under the application of spring forces is presented for reconstructing a smooth curve from a set of unordered and error-filled data points. Curves are represented in parametric form so that the developed method is suitable for both single-valued and multivalued data. The standard regularization method for univariate functions has been modified for multidimensional parametric functions. The resulting nonlinear minimization problem is then solved by successive quadratic programming. None of the connectivity information except the endpoints of the curve need to be specified, thus eliminating the necessity for an initial

parameterization. Both cubic and quintic Hermite polynomials are implemented as the curve basis. Other geometric constraints can be incorporated into the minimization process if necessary. Some technical issues are also addressed in more detail.

Throughout this paper, medium italic lower-case letters, such as f and $f(x)$, are reserved for scalar values or functions, and bold italic lower-case letters, such as $\boldsymbol{w}$ and $\boldsymbol{w}(u)$, for vectors or vector functions. Bold upper-case letters, such as **K**, indicate matrices.

The rest of this paper is organized as follows. In the second section, we review the literature related to parametric curve fitting including diverse parameterization methods, curve fairing and minimal-energy splines. In the third section, we discuss the necessary mathematical background for this problem including the standard regularization method for univariate functions and the modified version for multidimensional parametric functions, the physical interpretation and the minimization scheme. Curve primitives are introduced in the fourth section. Several technical issues are discussed in the fifth section. Finally, we present the results and conclude this paper in the sixth and seventh sections, respectively.

LITERATURE REVIEW

The problem of fitting a parametric curve to a set of data points has been addressed for many years. Most methods are developed on the basis of the assumption that the data points are ordered and methods differ in the way that the parameterization is made. The simplest one is uniform parameterization. Besides the fact that results generated by uniform parameterization are generally unsatisfactory, it has also been proved[2] that, by using uniform parameterization, singularities like corners can possibly occur even though the defining equations appear to make that impossible. A better choice than uniform parameterization is chord length parameterization because the effect of data points distribution is incorporated. Chord length is actually an approximation of the true arc length of the fitted curve. Another alternative is circular arc parameterization in which the arc length is estimated by fitting a circle through each group of three consecutive points[3]. The parameterization can even be improved by calculating the arc length of the fitted curve and doing the fitting iteratively[3,4]. Marin proposed a method in which the 'optimal' parameterization is found by mininizing the integral of the squared second derivative of the curve[5]. Lee suggested a 'centripetal model' parameterization which, in most cases, has better results than chord length parameterization[6]. Hoschek proposed a Newton-like iterative approach of intrinsic parameterization in which the shortest distances between given data points and the approximation curve are minimized[7]. In these methods seeking the 'optimal' parameterization, once the parameterization has been decided, the curve's shape is simply determined by a least square fit (for approximation cases) or by introducing certain boundary conditions (for interpolation cases) to solve a set of linear equations.

Another approach to obtain a curve with good quality is to smooth the curve by a process called fairing. The given ordered points are interpolated first on the basis of a particular parameterization, and data points associated with large discontinuities in the first and second derivatives are adjusted so that a curve with better fairness is achieved. This approach generally requires users to decide which data points should be moved by interactively detecting the 'unpleasant' regions. Readers are pointed to References 8–10 for more details.

In addition to methods finding the 'optimal' parameterization, the minimal-energy spline has been used widely to obtain the 'optimal' shape directly based on a specific parameterization. Data interpolation using minimal energy splines can date back to Schweikert's spline under tension[11] in which the curve's shape is determined by minimizing a function representing the potential energy stored in the curve. The following work differs in the form of energy function to be minimized, to name a few, the v-spline proposed by Nielson[12] and the τ-spline proposed by Hagen[13]. Some other research falling in this category is described in References 14–16. It is interesting to note that the published references for data smoothing using minimal energy splines are fewer than those for data interpolation. We refer to References 17–19 for interested readers. In these references, the function includes not only the curve's potential energy but also an error term measuring the closeness between the approximating curve and the data. This is a more general formulation since the error term will vanish for data interpolation problems.

MATHEMATICAL BACKGROUND

Function to be minimized

A problem is 'ill posed' when the existence, uniqueness and stability of the solutions are not guaranteed without additional constraints. Reconstructing a curve from a set of data points is referred to as an inverse problem which is generally ill posed. In fact, given any set of points on a curve, there are infinitely many curves interpolating or approximating these points. One way to make such a problem well posed is to restrict the class of admissible solutions, and to provide a method for ranking the plausibilities of these solutuions. In this regard, constraints such as the smoothness of the solution have been introduced to act as a 'stabilizing term' in the regularization theory pioneered by Tikhonov and others[20]. This technique reformulates the original ill posed problem into a well posed minimization problem. In Reference 20, the function to be minimized is concerned with univariate solutions which represent single-valued planar curves. For representing curves in higher dimensions, the parametric form is obviously a better choice. The function is hence rewritten for a multidimensional parametric solution as

$$E(\boldsymbol{w}) = \sum_{m=1}^{p} \int_C \alpha_m(u) \left\| \frac{\mathrm{d}^m \boldsymbol{w}(u)}{\mathrm{d}u^m} \right\|^2 \mathrm{d}u + \sum_{i=1}^{N} \lambda_i \| \boldsymbol{P}_i - \boldsymbol{w}(u) \|^2 \quad (1)$$

where $\boldsymbol{w}(u) = (x_1(u), x_2(u), \ldots, x_d(u))$,

$$\frac{\mathrm{d}^m \boldsymbol{w}(u)}{\mathrm{d}u^m} = \left(\frac{\mathrm{d}^m x_1(u)}{\mathrm{d}u^m}, \frac{\mathrm{d}^m x_2(u)}{\mathrm{d}u^m}, \ldots, \frac{\mathrm{d}^m x_d(u)}{\mathrm{d}u^m} \right)$$

$\alpha_m(u)$ are prespecified, nonnegative weighting functions, λ_i are positive weighting factors, and d is the dimension of the curve.

The first term in Equation 1 is a smoothness measurement of the solution. It is also referred to as the stabilizing term in regularization theory. The second term will be referred to as the error term since it measures how close the solution is to the given data points.

There are reasons why it is necessary to modify the function in Equation 1. First of all, calculation of the error term requires a parameterization of the data points which is not available *a priori*. (Consequently, diverse parameterization methods are developed based on the assumption that the data points are ordered). Second, even if the parameterization is known, using $\|\boldsymbol{P}_i - \boldsymbol{w}(u_i)\|$ as a measurement of error is also somewhat inappropriate since it could cause nonzero measurements even though all the points already lie exactly on the curve. Furthermore, for most applications in geometric modelling and aesthetic design, there is no need to use high-order derivatives in the stabilizing term since discontinuity of high-order derivatives is almost unrecognizable by human perception. It can be said that, if the human's visual system can only recognize discontinuities up to the mth order derivatives, only the solutions with continuous derivatives up the the $(m+1)$th order need to be considered. Experiences show that discontinuities of C^0 and C^1 can be easily detected by the human visual system, and C^2 discontinuity can be recognized by experienced users. Hence, derivatives will be taken up to the third order in this paper. Higher order derivatives can be included whenever they are necessary for particular applications at the cost of raising the degree of the polynomial used as the curve basis.

From these considerations, we modified the function as follows:

$$E(\boldsymbol{w}) = \int_C (\alpha\|\boldsymbol{w}'\|^2 + \beta\|\boldsymbol{w}''\|^2 + \gamma\|\boldsymbol{w}'''\|^2)\,\mathrm{d}u + \sum_{i=1}^{N} \lambda_i D(\boldsymbol{P}_i, \boldsymbol{w}(u))^2 \qquad (2)$$

where $\boldsymbol{w}(u) = (x_1(u), x_2(u), \ldots, x_d(u))$,

$$\boldsymbol{w}' = \frac{\mathrm{d}\boldsymbol{w}}{\mathrm{d}u}$$

$$\boldsymbol{w}'' = \frac{\mathrm{d}^2\boldsymbol{w}}{\mathrm{d}u^2}$$

$$\boldsymbol{w}''' = \frac{\mathrm{d}^3\boldsymbol{w}}{\mathrm{d}u^3}$$

and α, β and γ are prespecified, nonnegative weighting functions. $D(\boldsymbol{P}_i, \boldsymbol{w}(u))$ is a distance metric which is defined as the shortest Euclidean distance from $\boldsymbol{P}_i$ to $\boldsymbol{w}(u)$. The evaluation of $D(\boldsymbol{P}_i, \boldsymbol{w}(u))$ would result in a constrained single variable minimization problem which will be discussed later.

Compared with the energy functions used in References 17 and 18, our potential energy of the curve includes not only the squared second derivative but also the first and the third derivatives. By choosing different weighting values, the shapes obtained from the minimization are more capable of fitting the given data points. Another significant difference resides in the formulation of the error term which measures the closeness between the approximating curve and the data. In the above-mentioned references, the error term is taken as the sum of the squared vector differences of the data points and 'nodes' defining the curve. This kind of formulation still requires the data points to be ordered. Defining the closeness measurement as the sum of the shortest Euclidean distances between data points and the curve eliminates the requirement for the data points' connectivity information at the cost of increasing computing time.

Physical analogy

A physical model of the function in Equation 2 is illustrated in *Figure 1*. A perfectly elastic beam is deformed under the application of spring forces which are proportional to the shortest distance between the force sources, the data points, and the deflected shape, the curve. Each spring is assumed to have one end anchored at a data point and the other end slides along the curve so that the shortest distance is maintained. The squared first derivative term in the function stands for the strain energy due to stretching of the beam and the squared second derivative term stands for the strain energy due to bending. The error term corresponds to the strain energy stored in the springs. Since there exists such an analogy between the function to be minimized and the strain energy in material mechanics, we sometimes also refer to it as an 'energy function'.

Although the squared third derivative term in the energy function does not correspond to any physical meaning, it does have some geometric meaning. The magnitude of the second derivative $\|\boldsymbol{w}''\|$ is actually an approximation of the curvature κ when $\|\boldsymbol{w}'\|$ is assumed to be small. Similarly, the third derivative term is treated as a rough estimate of the rate of change of curvature, $\mathrm{d}\kappa/\mathrm{d}s$. Since the integral of the squared magnitude of the derivative of curvature evaluates to zero for circular arcs and straight lines, while minimizing the integral of the squared magnitude of the first and second derivatives makes the curve stretch and bend as little as possible, introducing the third derivative term enables the curve to form a circular arc (approximately) when constraints allow.

Finite element solution and nonlinear minimization

Continuous models are made by approximating the desired minimum energy shape as a superposition of a

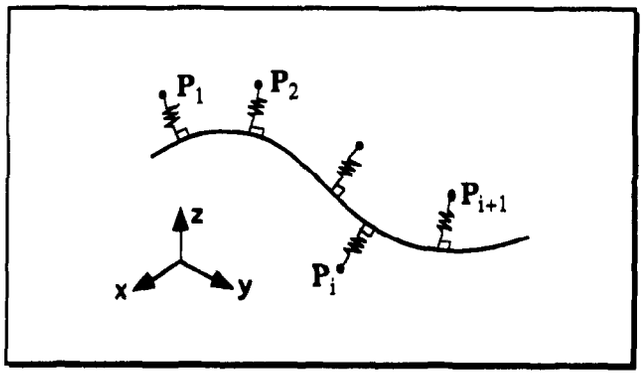


Figure 1 Physical model of function to be minimized
[Note that the springs are attached to the curve by the shortest route.]

finite series of weighted continuous basis functions. This approximation for $w(u)$ is

$$w(u) \approx \sum_{i=1}^{n} q_i \phi_i(u) = \mathbf{Q}^{\mathrm{T}} \Phi \tag{3}$$

where q_i is a set of vectors representing the degrees of freedom of the curve, $\phi_i(u)$ is a set of known functions, n is the number of degrees of freedom, $\mathbf{Q} = [q_1, q_2, \ldots, q_n]^{\mathrm{T}}$ and $\Phi = [\phi_1, \phi_2, \ldots, \phi_n]^{\mathrm{T}}$.

Substituting Equation 3 into Equation 2, we obtain

$$E(\mathbf{Q}) = \mathbf{Q}^{\mathrm{T}}\mathbf{K}\mathbf{Q} + \sum_{i=1}^{N} \lambda_i D_i^2(\mathbf{Q}) \tag{4}$$

where $\mathbf{K} = \mathbf{K}_1 + \mathbf{K}_2 + \mathbf{K}_3$ is called the system matrix or system stiffness matrix owing to the analogy with material mechanics, and

$$\mathbf{K}_1 = \int_c \alpha(\Phi'\Phi'^{\mathrm{T}})\mathrm{d}u$$

$$\mathbf{K}_2 = \int_c \beta(\Phi''\Phi''^{\mathrm{T}})\mathrm{d}u$$

$$\mathbf{K}_3 = \int_c \gamma(\Phi'''\Phi'''^{\mathrm{T}})\mathrm{d}u$$

It can be shown from regularization theory that, if the stabilizing term consists of only one squared derivative of order m, at least m constraints must be applied so that a stable solution can be found. This can be interpreted more easily by the analogy of material mechanics mentioned above. If only stretch energy is included in the stabilizing term, the case is like a beam under axial deformation which requires at least one position constraint. If only bending energy is included in the stabilizing term, a beam under the transverse deformation is simulated which requires at least two position constraints (simply supported beam) or one position and one slope constraint (cantilever beam). If the stabilizing term is a hybrid of the squared derivatives of different orders, the number of constraints should be at least the same as the highest order used in stabilizing term.

There are $n \times d$ variables (n degrees of freedom each of which is d dimensional) involved in this nonlinear problem. Inspired by the fact that the first term is quadratic, successive quadratic programming (SQP) is used to seek the solution. The original objective function is replaced by its local quadratic approximation at the solution estimate and the resulting approximate subproblem is solved. This replacement is repeated and the solution estimate is improved during iterations until a certain convergence criterion is achieved (as illustrated in *Figure 2*). The advantage of using local quadratic approximation is that the solution of the subproblem in each iteration can be easily obtained by solving a linear equation set if we restrict the constraints to be linear combinations of the explicit variables. Thus, solving the original nonlinear constrained minimization problem with $n \times d$ variables is converted into iteratively solving a linear equation set of n degrees of freedom subject to linear constraints. Following is the algorithm using SQP to find the optimal solution:

(1) Calculate the initial estimate $\mathbf{Q}^{(0)}$ by finding the solution which minimizes $E(\mathbf{Q}) = \mathbf{Q}^{\mathrm{T}}\mathbf{K}\mathbf{Q}$ subject to linear geometric constraints.

(2) Formulate the subproblem as minimizing

$$\bar{E}(\mathbf{d}; \mathbf{Q}^{(n)}) = \mathbf{d}^{\mathrm{T}}\mathbf{K}\mathbf{d} + \nabla E(\mathbf{Q}^{(n)})^{\mathrm{T}}\mathbf{d} \qquad \mathbf{d} = \mathbf{Q} - \mathbf{Q}^{(n)} \tag{5}$$

subject to linear geometric constraints.

(3) Solve the subproblem with the reduced transformation method for enforcing the linear geometric constraints, update $\mathbf{Q}^{(n)}$ by $\mathbf{Q}^{(n+1)} = \mathbf{Q}^{(n)} + \mathbf{d}$, and calculate the corresponding value of the energy function.

(4) Carry out a convergence check. If energy function's value converges and the error term is smaller than the prespecified tolerance, stop. Otherwise, go to Step 2.

It is easier to consider the minimization process using SQP as a simulation of the deforming process of an elastic beam under the application of spring forces. Starting with an initial shape which has a large energy function value, in each iteration, springs anchored on the fixed data points apply restoring forces on the curve, and the curve will deform to a more appropriate shape with a smaller energy function value.

CURVE PRIMITIVES

The basis functions of the parametric curve can be either global or local, but local support functions are preferable in the computer-aided design community since they result in a banded system matrix $\mathbf{K}$, thus speeding up the calculation and easing the imposition of complicated geometric constraints. In this work, Hermite polynomials were adopted as the curve basis although Bernstein polynomial and B-splines could also serve as the basis. A piecewise Hermite polynomial curve is determined by its geometric properties at a set of nodes. Parametric values are assigned to each node monotonically. The shape of the curve between two nodes, often called an element, is solely determined by the information at the

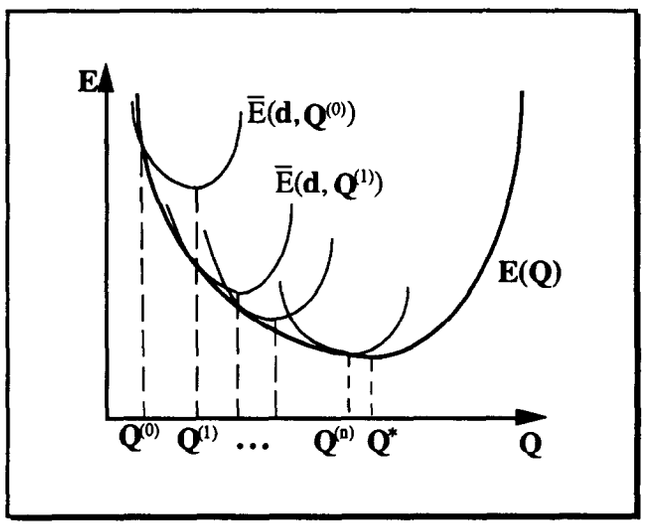


Figure 2 Successive quadratic programming

two nodes. Both cubic and quintic Hermite polynomials were implemented. The shape functions of a cubic/quintic Hermite element are given in *Table 1*.

By using the piecewise Hermite curve, calculation of the system matrix **K** can be simplified into evaluating 4×4 (or 6×6 for a quintic Hermite curve) submatrices of integrals for each curve element and assembling them together properly. When the weighting functions α, β and γ are constant over each element, the evaluation of these element matrices is simply an exercise in calculus and it can be done easily by software such as MACSYMA which performs symbolic mathematical manipulations. The matrices are given in *Table 2*.

SOME TECHNICAL ISSUES

Shortest distance computation

Mathematically, to find the minimum distance between a given point and a piecewise continuous curve, the minimum distances between the given point and each curve element, defined as being between 0 and h, should be calculated first. Then compare them to obtain the global minimum. This will raise the following constrained single variable minimization problem. Minimize

$$G(u) = \|\boldsymbol{P} - \boldsymbol{w}(u)\|^2 \qquad 0 \leqslant u \leqslant h \tag{6}$$

Table 1 Shape functions for cubic and quintic Hermite elements

i	Cubic elements (φ_i^c)	Quintic elements (φ_i^q)
1	$1-3\left(\frac{u}{h}\right)^2+2\left(\frac{u}{h}\right)^3$	$1-10\left(\frac{u}{h}\right)^3+15\left(\frac{u}{h}\right)^4-6\left(\frac{u}{h}\right)^5$
2	$h\left(\frac{u}{h}-2\left(\frac{u}{h}\right)^2+\left(\frac{u}{h}\right)^3\right)$	$h\left(\frac{u}{h}-6\left(\frac{u}{h}\right)^3+8\left(\frac{u}{h}\right)^4-3\left(\frac{u}{h}\right)^5\right)$
3	$3\left(\frac{u}{h}\right)^2-2\left(\frac{u}{h}\right)^3$	$h^2\left(0.5\left(\frac{u}{h}\right)^2-1.5\left(\frac{u}{h}\right)^3+1.5\left(\frac{u}{h}\right)^4-0.5\left(\frac{u}{h}\right)^5\right)$
4	$h\left(-\left(\frac{u}{h}\right)^2+\left(\frac{u}{h}\right)^3\right)$	$10\left(\frac{u}{h}\right)^3-15\left(\frac{u}{h}\right)^4+6\left(\frac{u}{h}\right)^5$
5		$h\left(-4\left(\frac{u}{h}\right)^3+7\left(\frac{u}{h}\right)^4-3\left(\frac{u}{h}\right)^5\right)$
6		$h^2\left(0.5\left(\frac{u}{h}\right)^3-\left(\frac{u}{h}\right)^4+0.5\left(\frac{u}{h}\right)^5\right)$

[h is the element's parametric length.]

Table 2 Element matrices for cubic and quintic Hermite elements

Element matrix	Cubic elements	Quintic elements
$\mathbf{K}_1^e=\int_0^h \Psi'^{\mathrm{T}}\Psi' \mathrm{d}u$	$\frac{1}{30h}\begin{bmatrix} 36 & 3h & -36 & 3h \\ 3h & 4h^2 & -3h & -h^2 \\ -36 & -3h & 36 & -3h \\ 3h & -h^2 & -3h & 4h^2 \end{bmatrix}$	$\frac{1}{1260h}\begin{bmatrix} 1800 & 270h & 15h^2 & -1800 & 270h & -15h^2 \\ & 288h^2 & 21h^3 & -270h & -18h^2 & 6h^3 \\ & & 2h^4 & -15h^2 & -6h^3 & h^4 \\ & \text{symm.} & & 1800 & -270h & 15h^2 \\ & & & & 288h^2 & -21h^3 \\ & & & & & 2h^4 \end{bmatrix}$
$\mathbf{K}_2^e=\int_0^h \Psi''^{\mathrm{T}}\Psi'' \mathrm{d}u$	$\frac{1}{h^3}\begin{bmatrix} 12 & 6h & -12 & 6h \\ 6h & 4h^2 & -6h & 2h^2 \\ -12 & -6h & 12 & -6h \\ 6h & 2h^2 & -6h & 4h^2 \end{bmatrix}$	$\frac{1}{70h^3}\begin{bmatrix} 1200 & 600h & 30h^2 & -1200 & 600h & -30h^2 \\ & 384h^2 & 22h^3 & -600h & 216h^2 & -8h^3 \\ & & 6h^4 & -30h^2 & 8h^3 & h^4 \\ & \text{symm.} & & 1200 & -600h & 30h^2 \\ & & & & 384h^2 & -22h^3 \\ & & & & & 6h^4 \end{bmatrix}$
$\mathbf{K}_3^e=\int_0^h \Psi'''^{\mathrm{T}}\Psi''' \mathrm{d}u$	$\frac{36}{h^5}\begin{bmatrix} 4 & 2h & -4 & 2h \\ 2h & h^2 & -2h & h^2 \\ -4 & -2h & 4 & -2h \\ 2h & h^2 & -2h & h^2 \end{bmatrix}$	$\frac{3}{h^5}\begin{bmatrix} 240 & 120h & 20h^2 & -240 & 120h & -20h^2 \\ & 64h^2 & 12h^3 & -120h & 56h^2 & -8h^3 \\ & & 3h^4 & -20h^2 & 8h^3 & -h^4 \\ & \text{symm.} & & 240 & -120h & 20h^2 \\ & & & & 64h^2 & -12h^3 \\ & & & & & 3h^4 \end{bmatrix}$

$$\Psi=\begin{cases}[\varphi_1^c, \varphi_2^c, \varphi_3^c, \varphi_4^c] & \text{cubic element} \\ [\varphi_1^q, \varphi_2^q, \varphi_3^q, \varphi_4^q, \varphi_5^q, \varphi_6^q] & \text{quintic element.}\end{cases}$$

Using the squared distance rather than the distance itself in the object function enables the square root to disappear without losing any generality.

Generally, a constrained minimization problem is turned into an unconstrained one by using the penalty functions which prescribe a penalty whenever the solution estimate falls in infeasible regions. Fortunately, in our case the inequality constraints posed above are just the upper and lower bounds of the variable. This simplifies the problem significantly. Typically, it can be solved by finding the roots of $g(u)$, the gradient of $G(u)$, selecting those falling in the interval of interest and comparing the corresponding objective function values. There may be none, one or multiple roots in the interval $[0, h]$. Hence, the minimum point might fall in $[0, h]$ or exactly on the endpoint. It should be noted that some point–curve relationships can produce solutions to $g(u)$ that are maximum distances in the interval. Thus all extrema must be examined, and those that are maxima eliminated.

So far this problem seems to be converted into a problem of finding roots of $g(u)$ in the range $[0, h]$. However, this is still not an easy task. Problems still exist, such as those of how many roots exist in this interval and how to bracket these roots so that other algorithms such as the Newton–Raphson method can be applied. Many methods can be used to find roots bracketed by an interval of a general single variable function. However, some of them are not able to identify whether roots exist in the interval, or they fail to extract all the roots in the interval. Since the root finding has to be performed with respect to each curve element for each data point, a fast and robust root-finding algorithm is required.

One of the beauties of using polynomials as the curve basis is that this also gives polynomial functions in Equation 6. For a cubic (quintic) Hermite element, $G(u)$ is a 6th (10th) degree polynomial function. The number of real roots of a polynomial function cannot exceed the function's degree. Another advantage is that a polynomial function represented in the power basis can be converted to the Bernstein basis and all the nice properties of the Bernstein basis can be applied to further locate the roots more efficiently and robustly. The following paragraphs will introduce this method.

The portion of a polynomial of degree M, $g(u)=\sum_{i=0}^{M} a_i u^i$, inside $[0, h]$ can be represented by a set of the Bernstein basis of degree M by using the following identity:

$$t^k=\sum_{i=0}^{M} b_{i,k}B_{i,M}(t) \qquad b_{i,k}=\frac{C_k^i}{C_k^M} \tag{7}$$

where $b_{i,k}=0$ if $i<k$, and then

$$g(u)=\sum_{i=0}^{M} a_i u^i=\sum_{i=0}^{M} \bar{a}_i t^i=\sum_{i=0}^{M} \bar{a}_i \sum_{j=0}^{M} b_{j,i}B_{j,M}(t)$$
$$=\sum_{j=0}^{M} g_j B_{j,M}(t) \tag{8}$$

where $t=u/h$, $\bar{a}_i=a_i h^i$ and $g_j=\sum_{i=0}^{M} \bar{a}_i b_{j,i}$ are the Bézier ordinates.

Once the original polynomial has been transformed into the Bézier form, we can rewrite $g(u)$ as an explicit Bézier curve:

$$\boldsymbol{g}(t)=\begin{pmatrix} t \\ g(t) \end{pmatrix}=\sum_{i=0}^{M} \begin{pmatrix} \frac{i}{M} \\ g_i \end{pmatrix} B_{i,M}(t) \tag{9}$$

Now, finding the roots of a univariate polynomial function $g(u)$ inside the range $[0, h]$ is equivalent to the problem of finding roots of $g(t)$ inside $[0, 1]$. The latter can then be converted into a problem of finding the intersection of the Bézier curve $\boldsymbol{g}(t)$ with the parameter axis. Thanks to the variation diminishing property of Bézier curves, the number of roots falling in $[0, h]$ cannot exceed the number of sign changes of the Bézier ordinates. If all the control points are at one side of the axis, namely, the g_is are all positive or all negative, then there are no roots inside $[0, h]$ and the minimum will occur at either $u=0$ or $u=h$. If the number of sign changes of Bézier ordinates is larger than zero, we can subdivide the curve using the recursive de Casteljau algorithm to locate the subintervals in which the roots exist, and then use the Newton–Raphson method to obtain the roots.

Geometric constraints enforcement

General geometric constraints can be imposed by augmenting the original energy function with Lagrange multipliers. The resulting equations often lose the linearity and complicate the problem. However, if we restrict the class of constraints by considering those only composed of a linear combination of the degrees of freedom of the problem, the constraint imposition can be achieved easily by the reduced transformation technique. The most common linear geometric constraint is to fix some degrees of freedom at known locations. Imposing a linear relationship on some degrees of freedom is another common linear constraint. In the following, we will briefly discuss the reduced transformation technique.

The reduced transformation method is a technique used to enforce linear equality constraints in solving the following quadratic programming (QP) problem. Minimize

$$F=\frac{1}{2}\mathbf{Q}^{\mathrm{T}}\mathbf{K}\mathbf{Q}+\mathbf{F}^{\mathrm{T}}\mathbf{Q} \tag{10}$$

subject to

$$h_k(\mathbf{Q})=0.0;\ k=1, \ldots, m$$

where $\mathbf{Q}$ is a column vector containing n variables, and h_k, $k=1, \ldots, m$, are equality constraints composed of a linear combination of the variables. These linear equality constraints can be written in matrix form as $\mathbf{AQ}=\mathbf{B}$.

Since each linear constraint equation will reduce the problem's dimension by 1, the first step in solving this problem by the reduced transformation technique is to select the independent variables and represent the dependent variables by the selected independent variables. By premultiplying a proper permutation matrix, we can rearrange $\mathbf{Q}$ so that the dependent and independent

variables are separate:

$$\mathbf{AQ}=\mathbf{AP}^{-1}\bar{\mathbf{Q}}=[\mathbf{A}_1\,|\,\mathbf{A}_2]\begin{Bmatrix}\mathbf{Q}_{\text{indep}}\\ \mathbf{Q}_{\text{dep}}\end{Bmatrix}=\mathbf{B} \tag{11}$$

or

$$\mathbf{A}_1\mathbf{Q}_{\text{indep}}+\mathbf{A}_2\mathbf{Q}_{\text{dep}}=\mathbf{B}$$

where $\mathbf{P}$ is a permutation matrix such that $\mathbf{PQ}=\bar{\mathbf{Q}}=[\mathbf{Q}_{\text{indep}}\,|\,\mathbf{Q}_{\text{dep}}]^{\mathrm{T}}$.

$\mathbf{Q}_{\text{indep}}$ is a reduced column vector containing only $(n-m)$ independent variables. $\mathbf{Q}_{\text{dep}}$ is a column vector of dimension m containing dependent variables. The dependent variables can be represented by the independent variables as

$$\mathbf{Q}_{\text{dep}}=\mathbf{A}_2^{-1}\mathbf{B}-\mathbf{A}_2^{-1}\mathbf{A}_1\mathbf{Q}_{\text{indep}}=\mathbf{D}_0+\mathbf{D}_1\mathbf{Q}_{\text{indep}} \tag{12}$$

Now, the original column vector $\mathbf{Q}$ can be represented by the independent variables only as

$$\mathbf{Q}=\mathbf{P}^{-1}\begin{Bmatrix}\mathbf{Q}_{\text{indep}}\\ \mathbf{Q}_{\text{dep}}\end{Bmatrix}=\mathbf{P}^{-1}\left(\begin{bmatrix}\mathbf{I}\\ \mathbf{D}_1\end{bmatrix}\mathbf{Q}_{\text{indep}}+\begin{Bmatrix}\mathbf{Z}\\ \mathbf{D}_0\end{Bmatrix}\right)$$
$$=\mathbf{D}_3\mathbf{Q}_{\text{indep}}+\mathbf{D}_2 \tag{13}$$

in which $\mathbf{I}$ is an identity matrix and $\mathbf{Z}$ is a zero vector. Substituting Equation 13 into the objective function in Equation 10 and taking $\partial F/\partial\mathbf{Q}_{\text{indep}}=0$, we get a reduced set of unconstrained linear equations of number $n-m$:

$$(\mathbf{D}_3^{\mathrm{T}}\mathbf{K}\mathbf{D}_3)\mathbf{Q}_{\text{indep}}=\mathbf{D}_3^{\mathrm{T}}\mathbf{F}-\mathbf{D}_3^{\mathrm{T}}\mathbf{K}\mathbf{D}_2 \tag{14}$$

which automatically enforce the linear constraints while the solution is found which minimizes the quadratic function of the subproblem. The new system matrix $\mathbf{D}_3^{\mathrm{T}}\mathbf{K}\mathbf{D}_3$ retains the symmetry and the positive definiteness of the original matrix $\mathbf{K}$ and usually remains banded. In practice, when the constraints only fix some degrees of freedom at known values, generating the new system matrix can be achieved by deleting some rows and columns from the original system matrix instead of performing the full matrix multiplication twice.

How many elements should be used?

The choice of how many elements to use to reconstruct the curve from a set of error-embedding data is not a trivial problem. While using too few elements will fail to represent the characteristics of the target curve, using too many elements will make the curve follow the noise and possess many unwanted undulations. Speaking more specifically, raising the number of elements will reduce the error between data points and the solution, but will not necessarily mean that the solution is a better one. One thing we can state is that we would prefer using as few elements as possible to represent the resulting curve as long as the error between the data points and the curve is within some prespecified tolerance. In our work, the number of elements starts from one. If the curve's shape obtained after convergence does not satisfy the tolerance requirement, the curve is broken into two elements and the iteration resumes. This process is repeated until the tolerance requirement is satisfied and guarantees that the number of elements used is minimal.

Effect of introducing squared third derivative

The stabilizing term in Equation 2 is actually a general form of the energy functions used in many minimal-

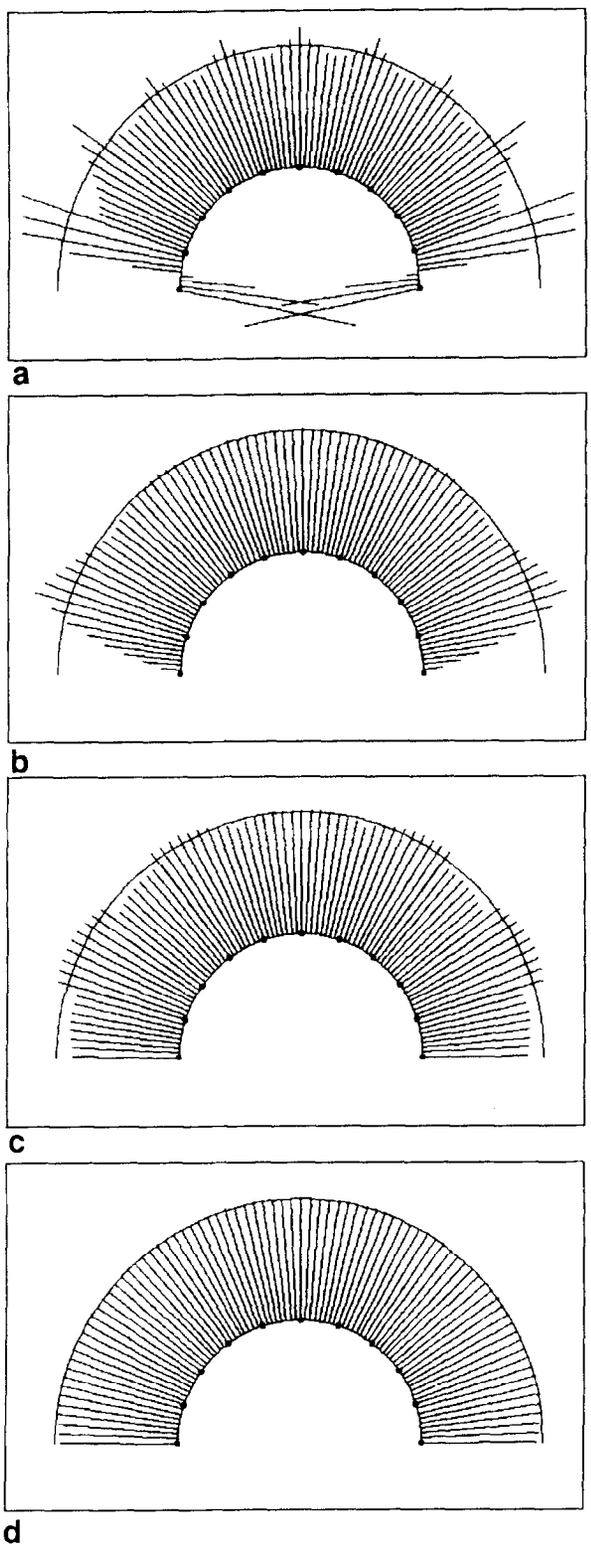


Figure 3 Effect of third derivative term (curves interpolating 11 data points taken from half circle with different weighting values); (a) C^1 cubic Hermite curve with $\alpha_i=1.0$, $\beta_i=\gamma_i=0.0$, (b) C^1 cubic Hermite curve with $\beta_i=1.0$, $\alpha_i=\gamma_i=0.0$, (c) C^1 cubic Hermite curve with $\alpha_i=\beta_i=1.0, \gamma_i=0.1$, (d) C^2 cubic Hermite curve with $\alpha_i=\beta_i=1.0, \gamma_i=0.1$

energy splines. The many different energy functions used for the splines under tension, the (weighted) v-splines, and the (weighted) τ-splines emerge by the weighting functions being properly chosen. While most people are familiar with the effect of minimizing the squared first and second derivatives, the effect of the squared third derivative is relatively unclear. In this section, we will use a simple example to show how the presence of the third derivative term affects the curve's shape.

We fit a set of data points taken from a half circle with radius 1. No error was inserted in the data. Different results based on different weighting values are shown in *Figure 3*. The curves shown are cubic Hermite curves which interpolate the data points assuming the connectivity information is known. Without the curvature plot, these curves have no visible differences. We can see that minimizing only the stretching energy certainly does not give us a pleasant curvature distribution (see *Figure 3a*). When only the bending energy is minimized (see *Figure 3b*), the curvature distribution is much better, but the curvature almost vanishes at the two ends, which is far from what it should be. The curvature plots in *Figures 3c* and *d* reveal that introducing the squared third derivative term makes the curve behave more like a circular arc. The large curvature discontinuity shown in *Figure 3c* is due to the fact that the cubic Hermite curve has C^2 discontinuity at the junctions, and the presence of the γ term tends to minimize the curvature variation of each element, but controversially enlarges the discontinuity at the junctions. This enlarged discontinuity can be eliminated by using a C^2 continuous curve as shown in *Figure 3d*.

When only positional information is available for constraining the curve, minimizing the bending energy only would result in a curve with zero curvature at its ends. Although this is obviously not always acccptable for all applications, the choice of when we should

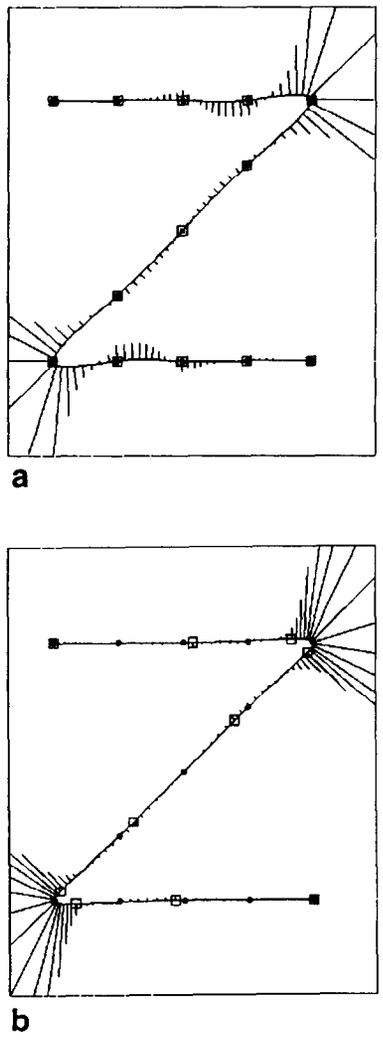


Figure 4 Fitting curves from Z-shaped distributed data points; (a) interpolating curve (12 elements), (b) approximating curve (9 elements)

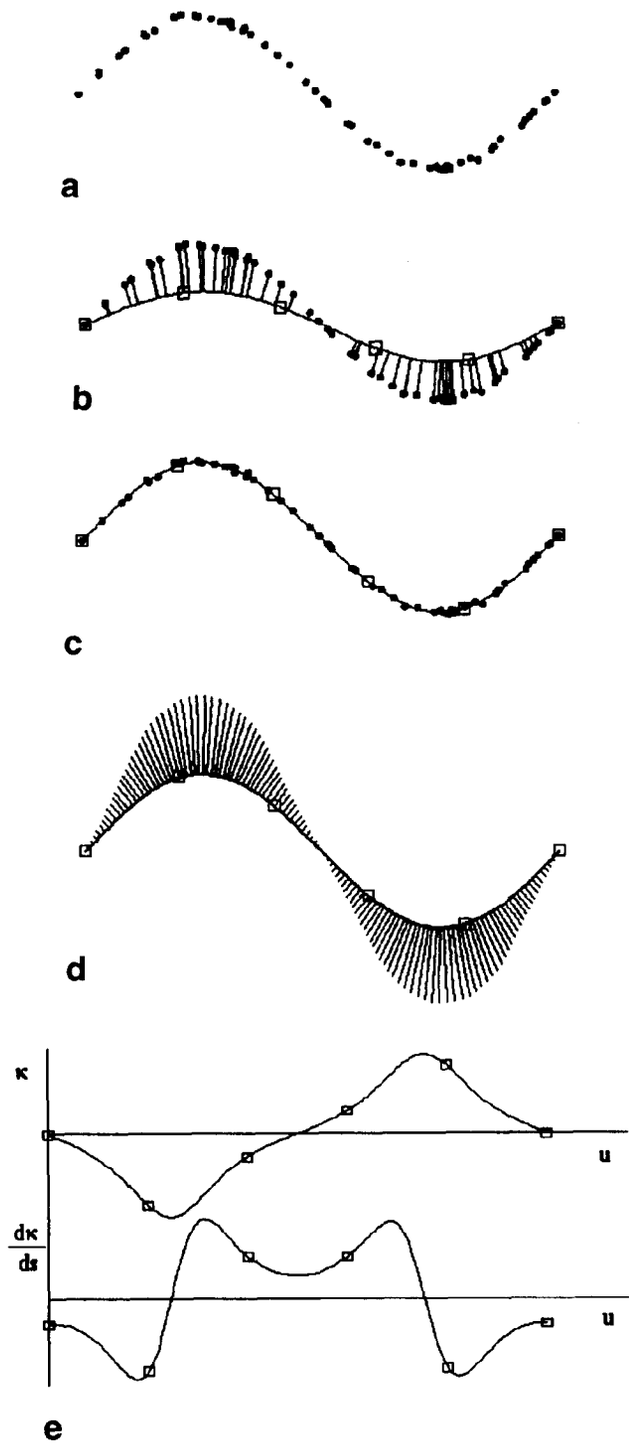


Figure 5 Sine curve (open curve fitted from 60 data points using five quintic Hermite elements); (a) input unordered, error-filled data, (b) deforming curve, (c) final curve, (d) final curve with curvature comb, (e) distribution of curvature and curvature variation

introduce the squared third derivative term to 'regulate' the curvature distribution is still hard to judge from pure positional data and will be left to the user's intention.

EXAMPLES

All of the figures we present here have been generated on a Silicon Graphics 4D GT/70 workstation. Data points are shown as small solid dots and nodes as open squares. All the figures are accompanied by curvature plots (drawn opposite to the direction of the normal vector) to show the fairness of the curve. *Figure 4* shows the interpolation and approximation curves for a set of Z-shaped distributed data points. While the curve shown in *Figure 4a* interpolates all data points, the curve shown in *Figure 4b* only approximates them. The curve is expected to be straight everywhere except at the two corners with large curvature. We can see that, even though fewer curve elements are used in the approximation case, the outwards propagation of the unwanted undulations from the corners diminishes faster in the approximation case and results in better shape.

Some other examples generated from error-filled data points are shown in *Figures 5–7*. Data points were randomly generated from analytic target curves with random perturbations of 5% maximum error. In *Figure 5*, 60 data points were generated randomly from a sine curve with errors in the y direction only. In *Figure 6*, 50 data points were taken from a 4th order B-spline with errors in the normal direction. Data in *Figures 5* and *6* were also used in Reference 1. However, instead of cubic Hermite curves, quintic Hermite curves with fewer elements are used. In *Figure 7*, data points were generated from a curve described in polar coordinates by $r = 1 + 0.25 \sin 3\theta$. Errors were taken in the radial direction only. All examples presented here are at least subject to two position constraints at the two ends of the curve. While the sine curve is constrained at its two ends at different positions, the aerofoil is constrained at its trailing edge. A compatibility equation which equates the tangent vectors at two ends was used to guarantee the continuous closed curve shown in *Figure 7*.

Since this algorithm does not assume connectivity information, the initial estimate in the minimization process is often a straight line connecting the two endpoints specified by the user. This 'universal' initial guess works quite well for most cases, but, for data points

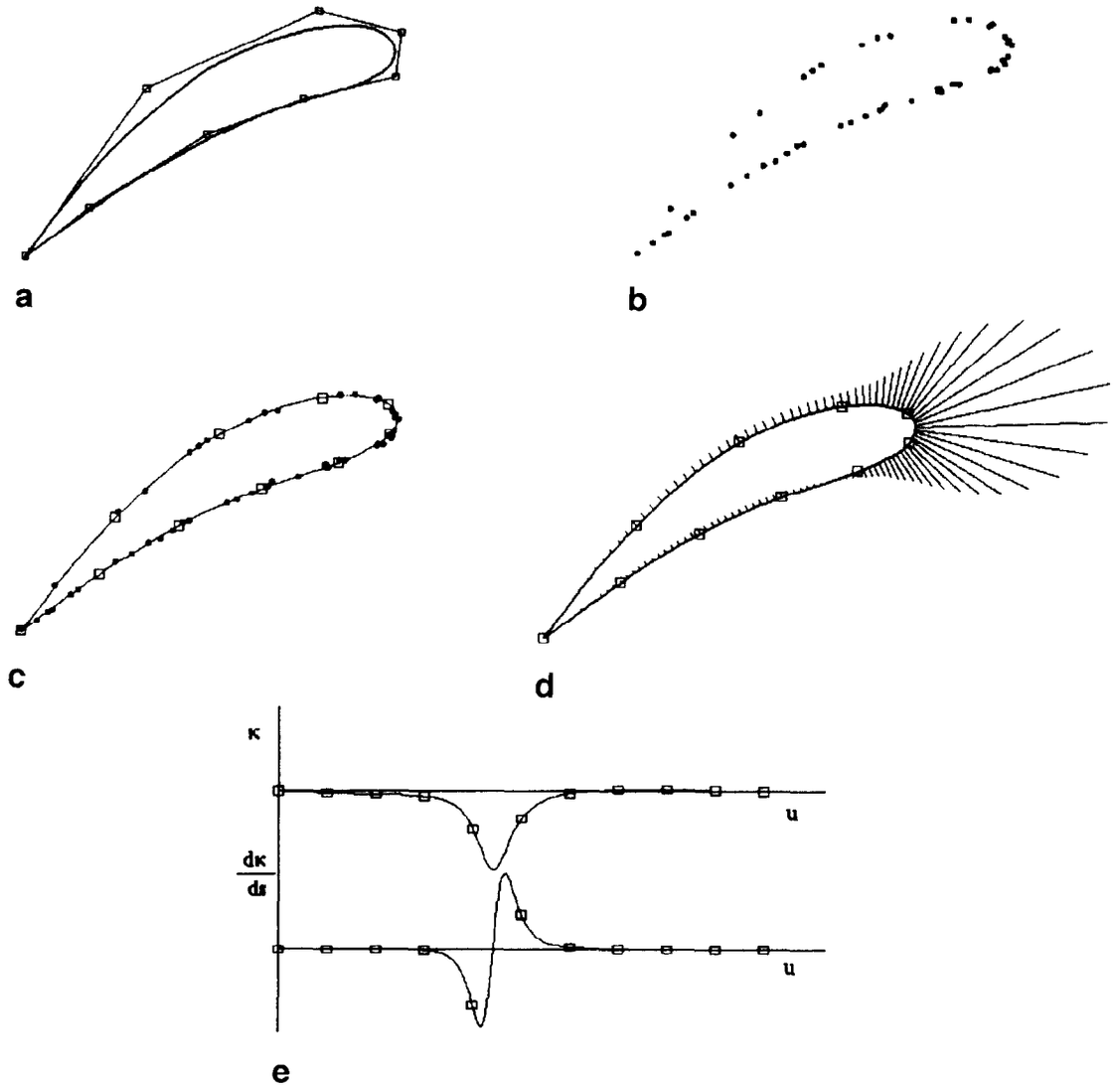


Figure 6 Aerofoil (closed curve with cusp fitted from 50 data points using ten quintic Hermite elements); (a) original curve defined by B-spline, (b) input unordered, error-filled data, (c) final curve, (d) final curve with curvature comb, (e) distribution of curvature and curvature variation [(a) Control points are represented by open squares.]

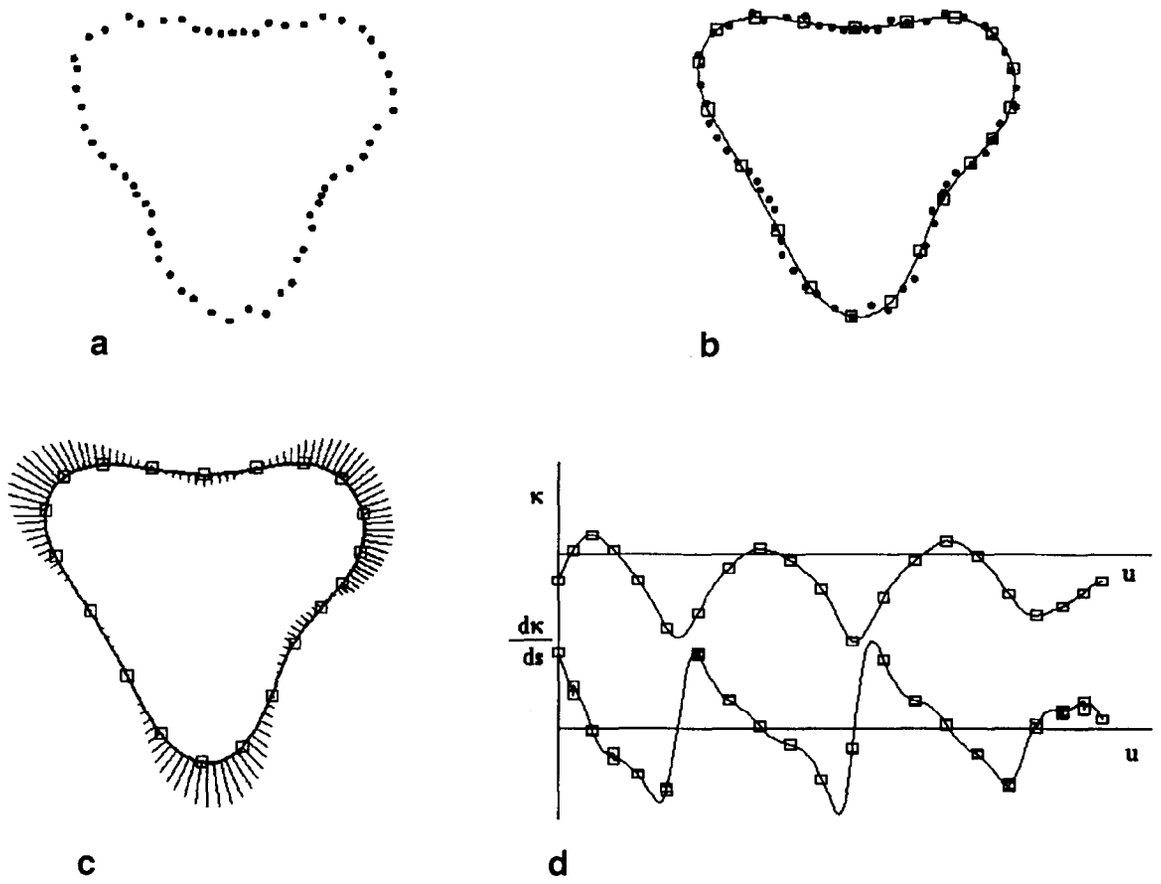


Figure 7 3-leaves rose (continuous closed curve fitted from 60 data points using 20 quintic Hermite elements); (a) input unordered, error-filled data, (b) final curve, (c) final curve with curvature comb, (d) distribution of curvature and curvature variation

taken from a helix-like curve or a self-intersecting curve, it often causes the solution to be trapped by a local minimum and results in a bad fit. If the characteristic of a curve can be captured by some means to generate a good initial estimate, it is believed that a good fit with satisfactory fairness is still achievable. This issue is currently under study.

CONCLUSIONS

We present a method for generating a smooth parametric curve which approximates a set of error-filled, unordered data points. Users only need to specify the endpoints of the curve. The resulting curve generally has very good fairness. Compared to the existing data point interpolation or approximation schemes, this method obviously has the advantage that no connectivity information other than the endpoints is required. The interpolation problem could also be treated as a special case in our method in which the data points to be interpolated are considered as constraints in the minimization process. On the other hand, this method also suffers from some limitations that are mainly due to the assumption of unknown connectivity information. Although only 2D examples are presented, the proposed method can be applied to higher dimensional curves without any modification. Furthermore, this technique can be extended to parametric surface fitting from scattered data points.

ACKNOWLEDGEMENTS

The authors would like to thank the sponsor, the Ford Motor Company, who provided the funds for this research. Comments and suggestions from Professor Hoschek, Professor Lyche and the referees of this paper were much appreciated. The authors are also grateful to Barbara Balents and Kenji Shimada for helpful suggestions.

REFERENCES

1 Fang, L and Gossard, D C 'Fitting 3D curves to unorganized data points using deformable curves' *Proc. 10th Int. Conf. Computer Graphics* (Jun 1992) pp 535–543
2 Epstein, M P 'On the influence of parametrization in parametric interpolation' *SIAM J. Numer. Anal.* Vol 13 No 2 (1976) pp 261–268
3 Earnshaw, J L and Yuille, I M 'A method of fitting parametric equations for curves and surfaces to sets of points defining them approximately' *Comput.-Aided Des.* Vol 3 (1971) pp 19–22
4 de Boor, D *A Practical Guide to Splines* Springer-Verlag, Germany (1978)
5 Marin, S P 'An approach to data parametrization in parametric cubic spline interpolation problems' *Approx. Theor.* Vol 41 (1984) pp 64–86
6 Lee, E T Y 'Choosing nodes in parametric curve interpolation' *Comput. Aided Des.* Vol 21 No 6 (1989) pp 363–370
7 Hoschek, J 'Intrinsic parametrization for approximation' *Comput. Aided Geom. Des.* Vol 5 (1988) pp 27–31

8 Renz, W 'Interactive smoothing of digitized point data' *Comput.-Aided Des.* Vol 14 No 5 (1982) pp 267–269

9 Kjellander, J A P 'Smoothing of cubic parametric splines' *Comput.-Aided Des.* Vol 15 No 3 (1983) pp 175–179

10 Farin, G, Rein, G, Sapidis, N and Worsey, A J 'Fairing cubic B-spline curves' *Comput. Aided Geom. Des.* Vol 4 (1987) pp 91–103

11 Schweikert, D G 'An interpolation curve using a spline in tension' *J. Math. Phys.* Vol 45 (1966) pp 312–317

12 Nielson, G M 'Some piecewise polynomial alternatives to splines under tension' *in* Barnhill, R E and Riesenfeld, R F (Eds.) *Computer Aided Geometric Design* Academic Press (1974)

13 Hagen, H 'Geometric spline curves' *Comput. Aided Geom. Des.* Vol 2 (1985) pp 223–227

14 Foley, T A 'Interpolation with interval and point tension controls using cubic weighted *v*-splines' *ACM Trans. Math. Soft.* Vol 13 (1987) pp 68–96

15 Pottmann, H 'Smooth curves under tension' *Comput.-Aided Des.* Vol 22 No 4 (1990) pp 241–245

16 Celniker, G and Gossard, D C 'Continuous deformable curves and their application to fairing 3D geometry' *Proc. IFIP TC 5/WG 5.2 Working Conf. Geometric Modeling* Rensselaerville, NY, USA (1990) pp 16–21

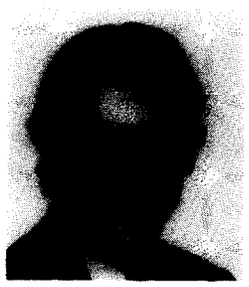
17 Reinsch, C H 'Smoothing by spline functions' *Numerische Math.* Vol 10 (1967) pp 177–183

18 Hosaka, M 'Theory of curve and surface synthesis and their smooth fitting' *Inf. Proc. Jap.* Vol 9 (1969) pp 60–68

19 Nowacki, H, Liu, D and Lu, X 'Mesh fairing GC^1 surface generation method' *in* Strasser, W and Seidel H-P (Eds.) *Theory and Practice of Geometric Modeling* (1988) pp 93–108

20 Tikhonov, S N and Aresnin, V A *Solutions of Ill-Posed Problems* Winston, USA (1977)

Lian Fang received a BS and an MS from National Taiwan University, Taiwan, in 1985 and 1987, respectively. He received a PhD from the Massachusetts Institute of Technology, USA, in 1994. His research interests include geometric modelling, computer graphics and optimization.



David C Gossard received a BSME and an MSME from Purdue University, USA, in 1968 and 1970, respectively. He received a PhD from the Massachusetts Institute of Technology, USA. He is currently a full professor in the Design, Systems and Controls Division of the Mechanical Engineering Department at MIT. In 1982, he received the US Society of Manufacturing Engineers Outstanding Young Engineers of the Year Award.